Learning Representations in Reinforcement Learning

Jacob Rafati

Ph.D. Candidate

Electrical Engineering and Computer Science

Computational Cognitive Neuroscience Lab

`http://rafati.net`

Ph.D. Advisor: Dr. David C. Noelle

University of California, Merced

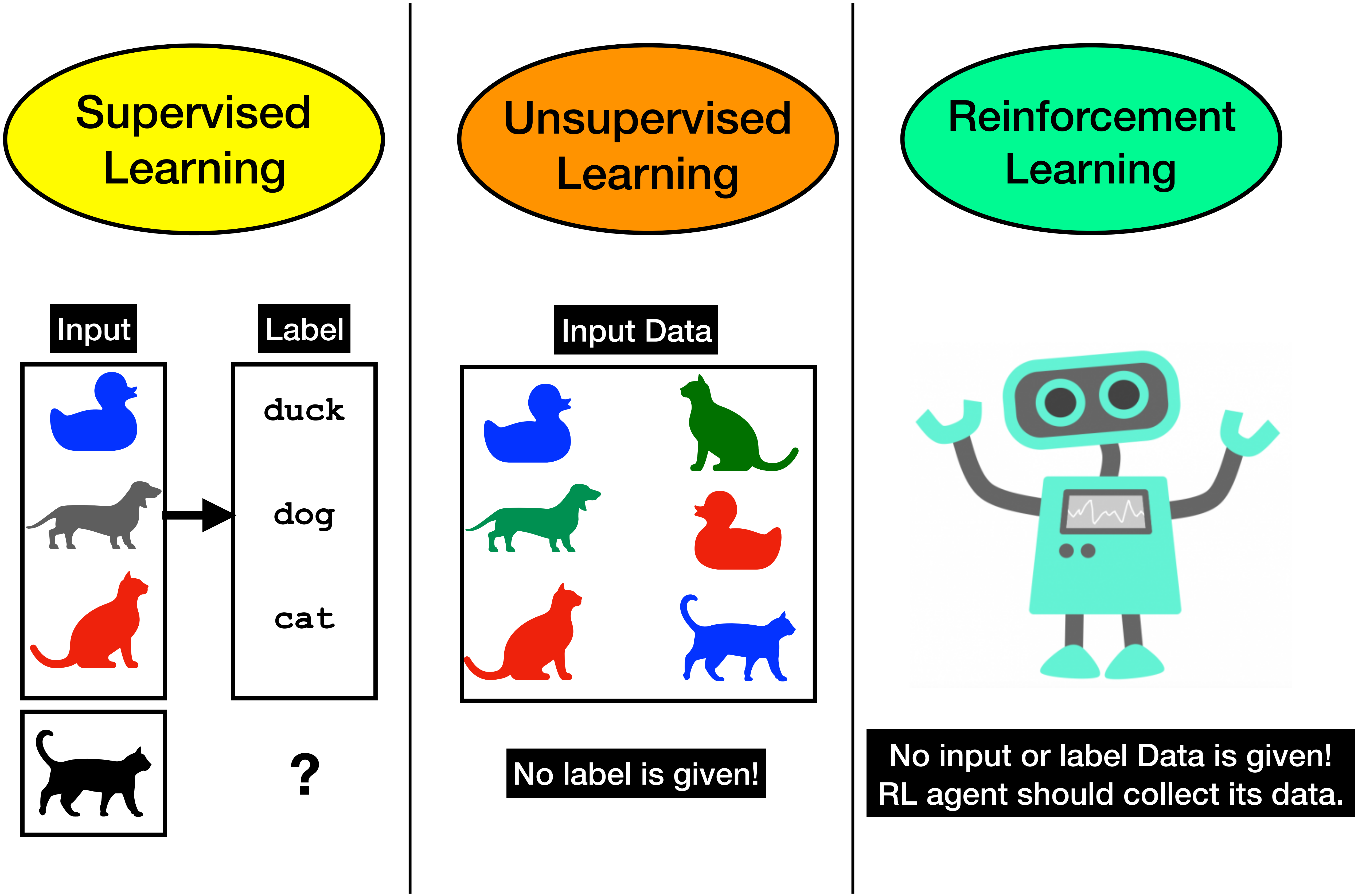
April 26th, 2019

Agenda

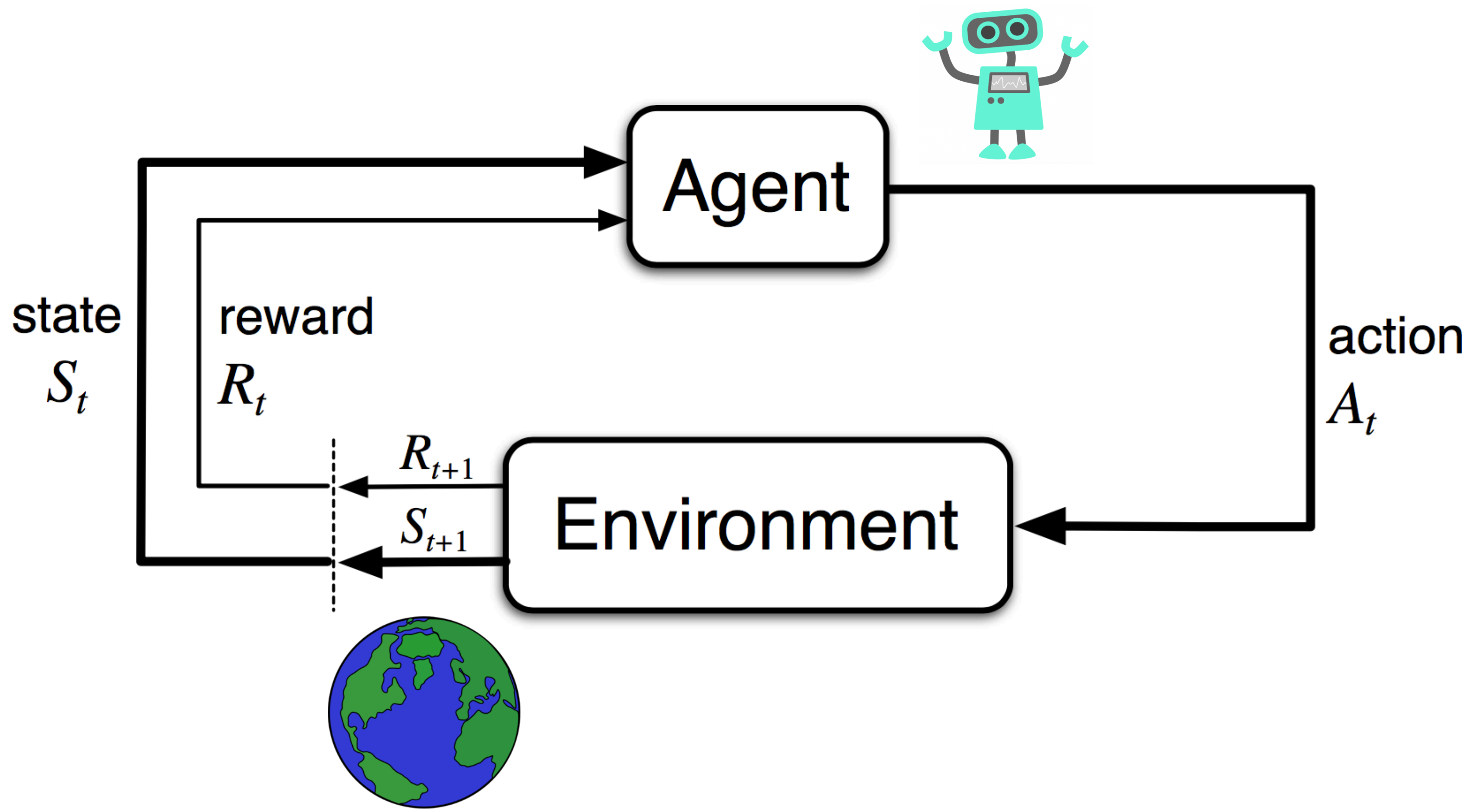
- **Introduction, Motivation, and Objectives**
- **Reinforcement Learning Algorithms**
- **Dissertation Problems and Methods:**
    1. Learning Sparse Representation in Reinforcement Learning
    2. Learning Representations in Hierarchical Reinforcement Learning
    3. Trust-Region Optimization Methods in Deep Learning
    4. Quasi-Newton Optimization in Deep Reinforcement Learning
- **Concluding Remarks and Future Work**

Introduction

Machine Learning



Reinforcement Learning

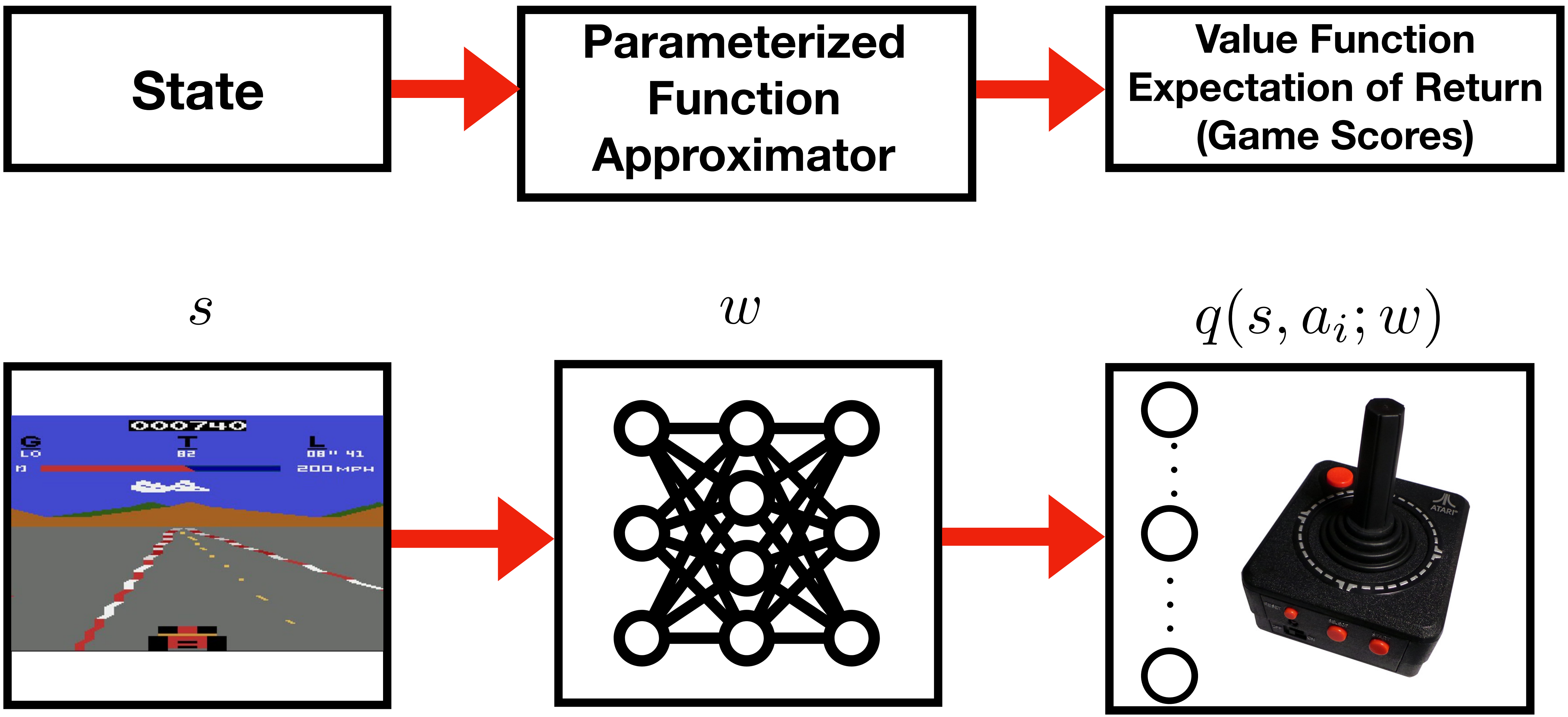


Reinforcement learning (RL) is learning how to map situations (**states**) to **agent**'s decisions (**actions**) to **maximize future rewards (return)** by **interaction** with an **unknown environment.**

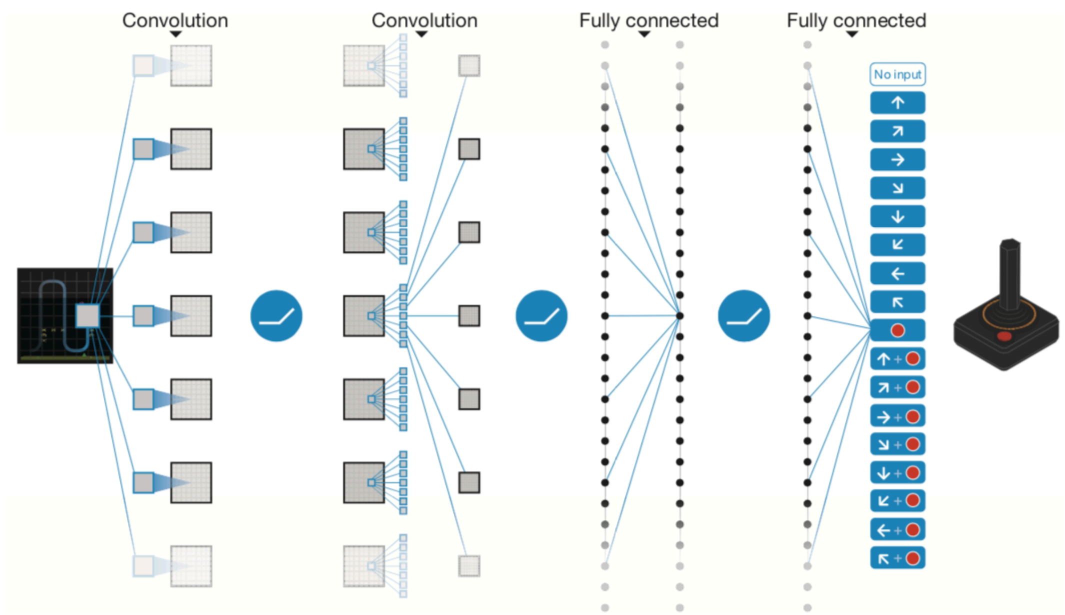
Experience (*s, a, r, s'*) as Data.

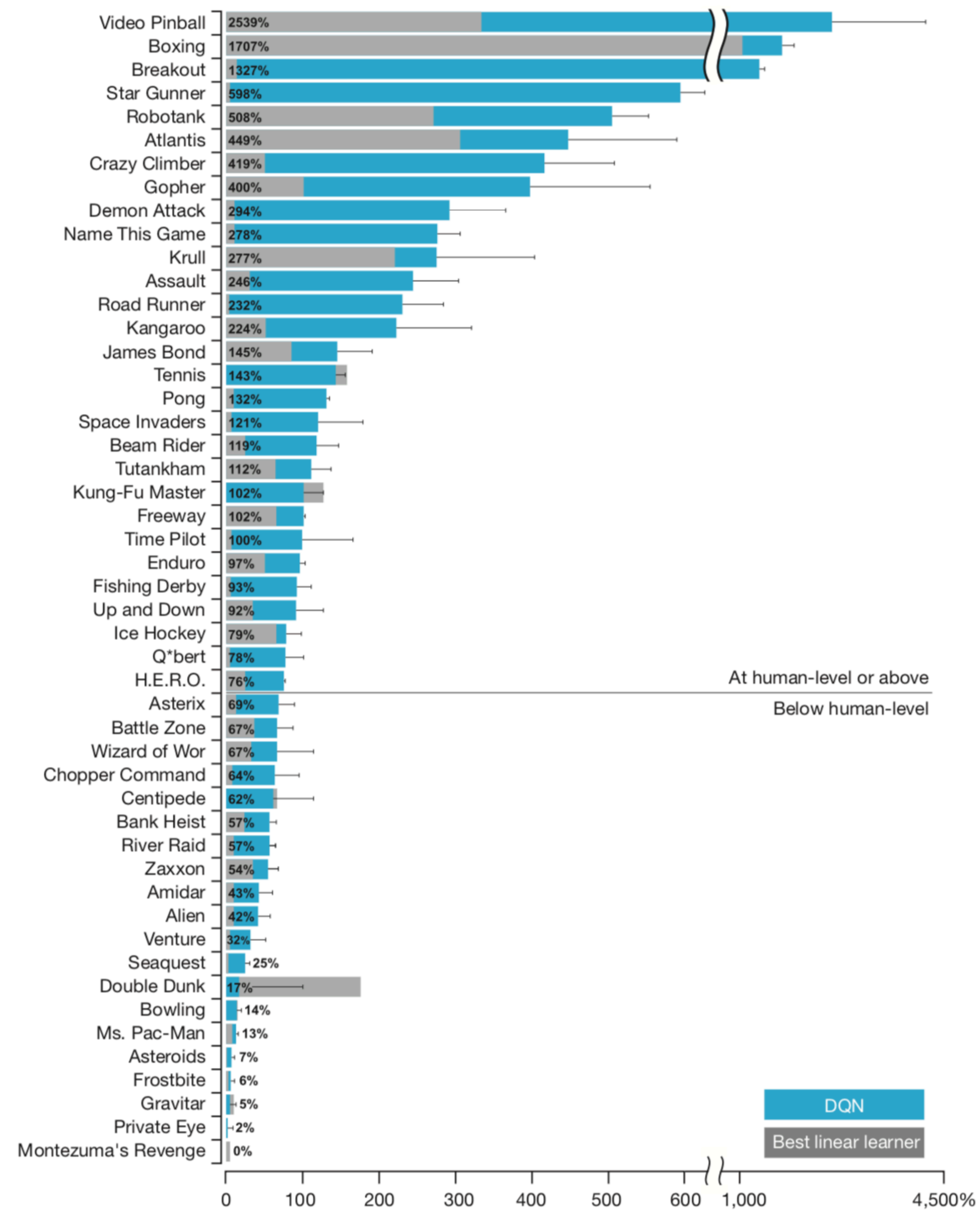
Sutton and Barto (2017). Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, USA, 2nd edition.

Generalization



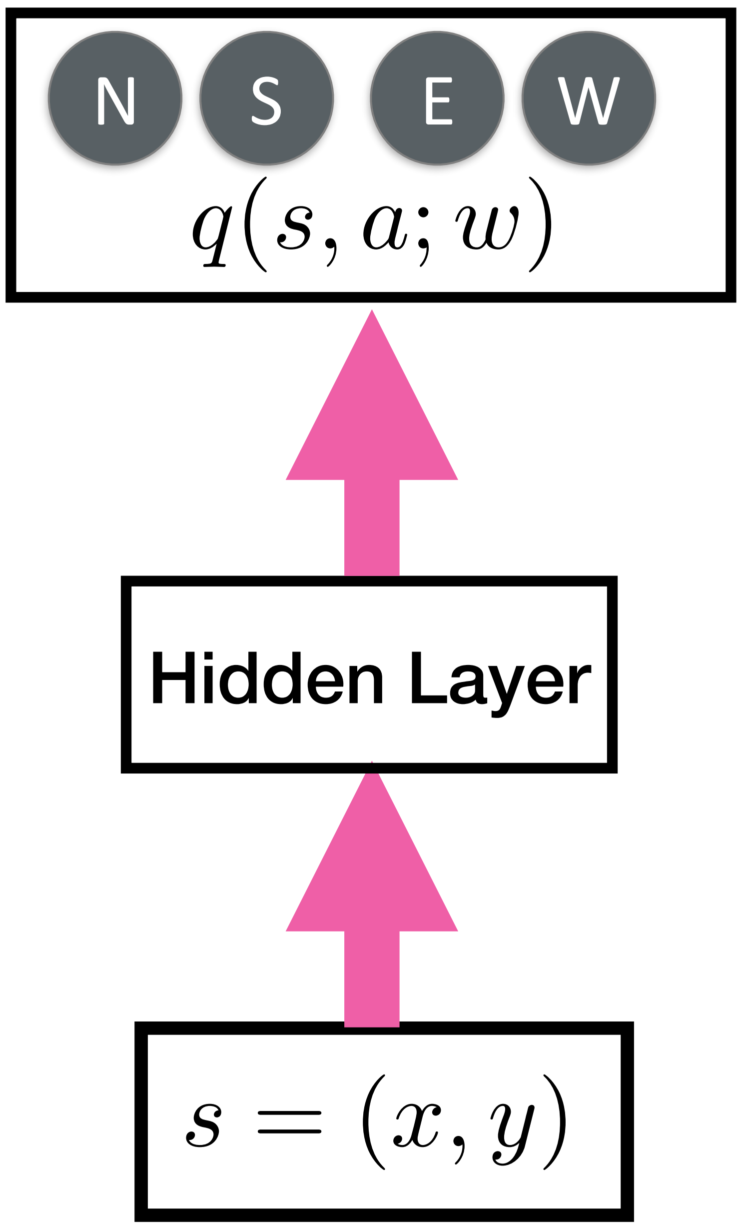
Super-Human Success

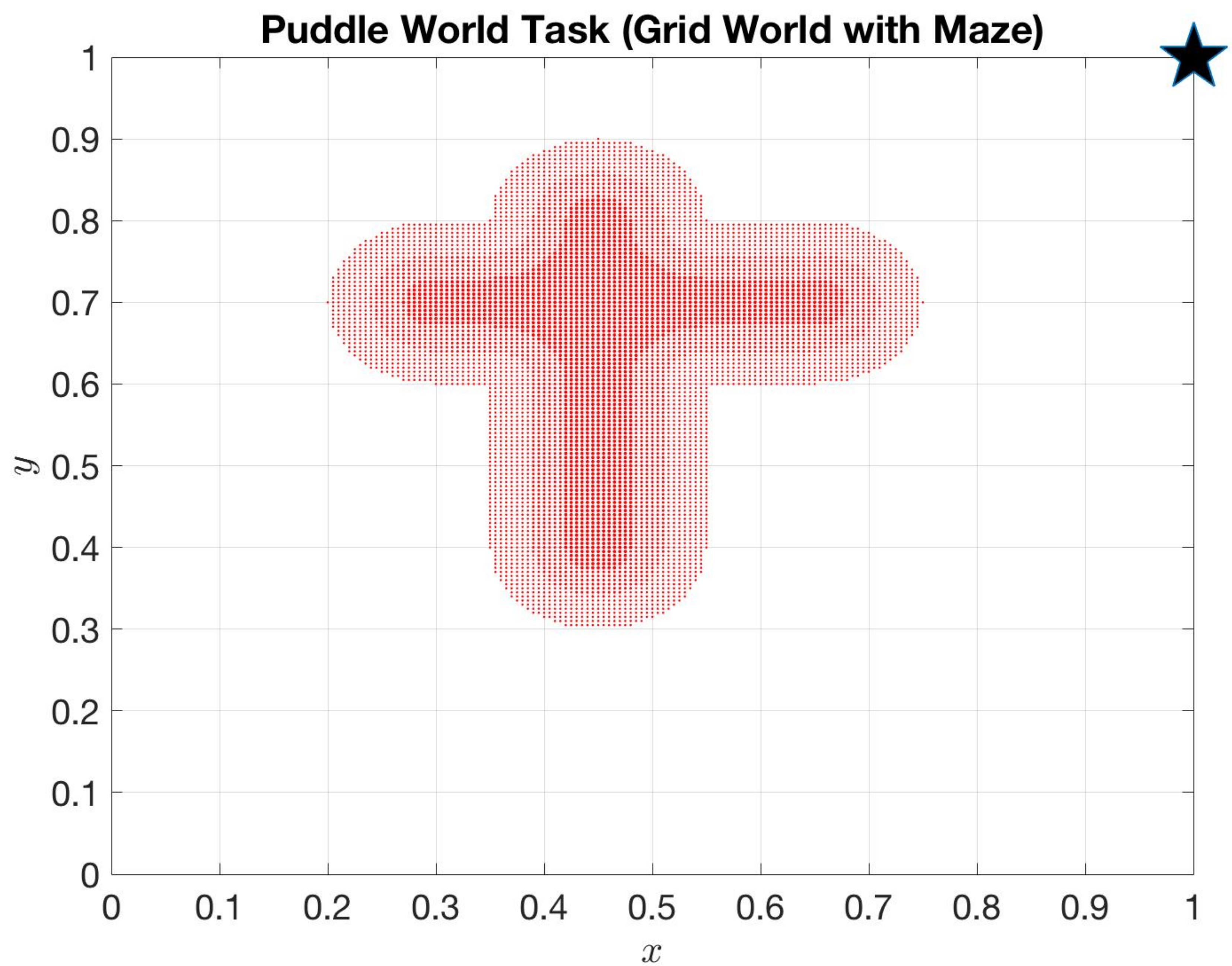




Mnih, et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540):529–533.

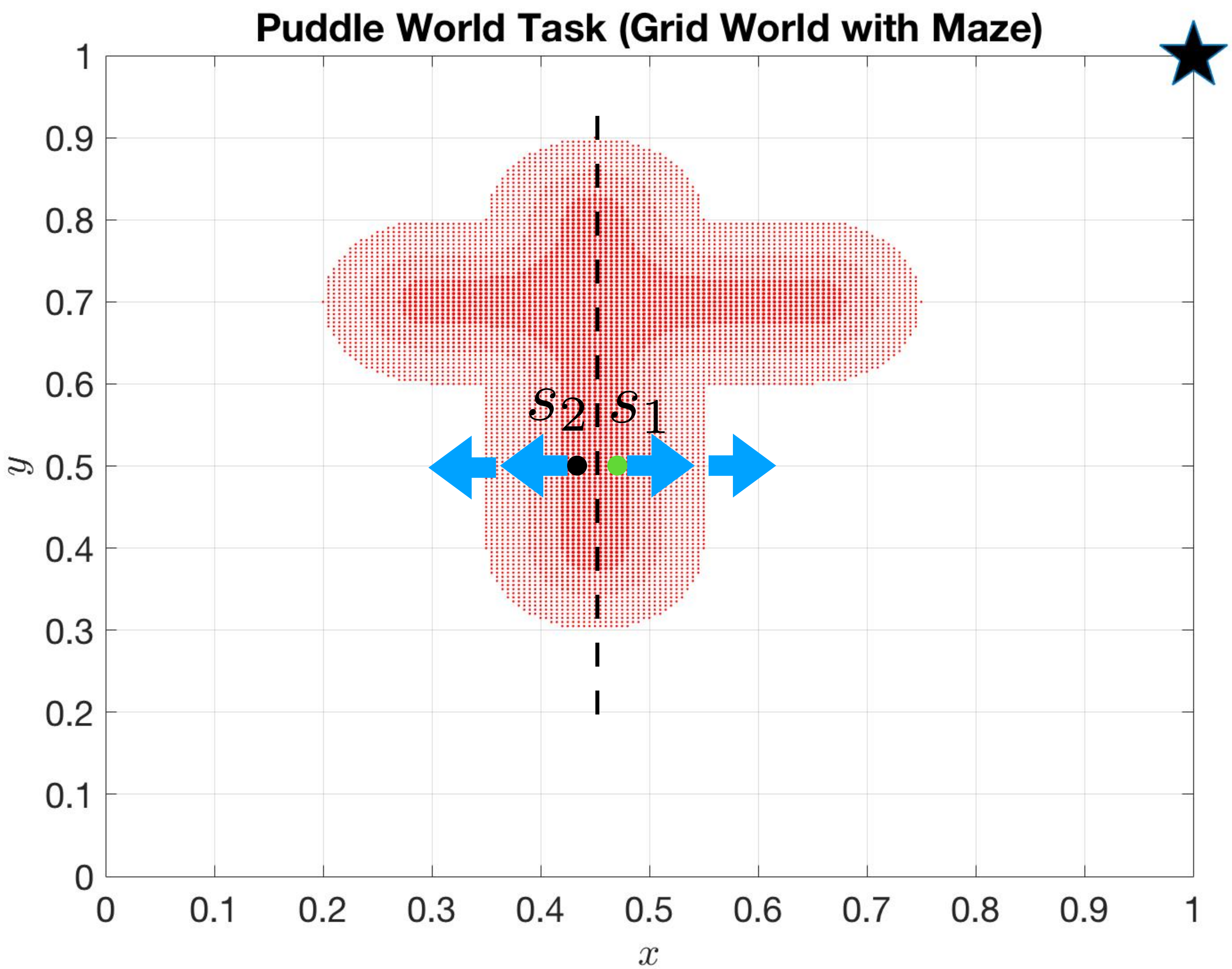
Failure in a simple task





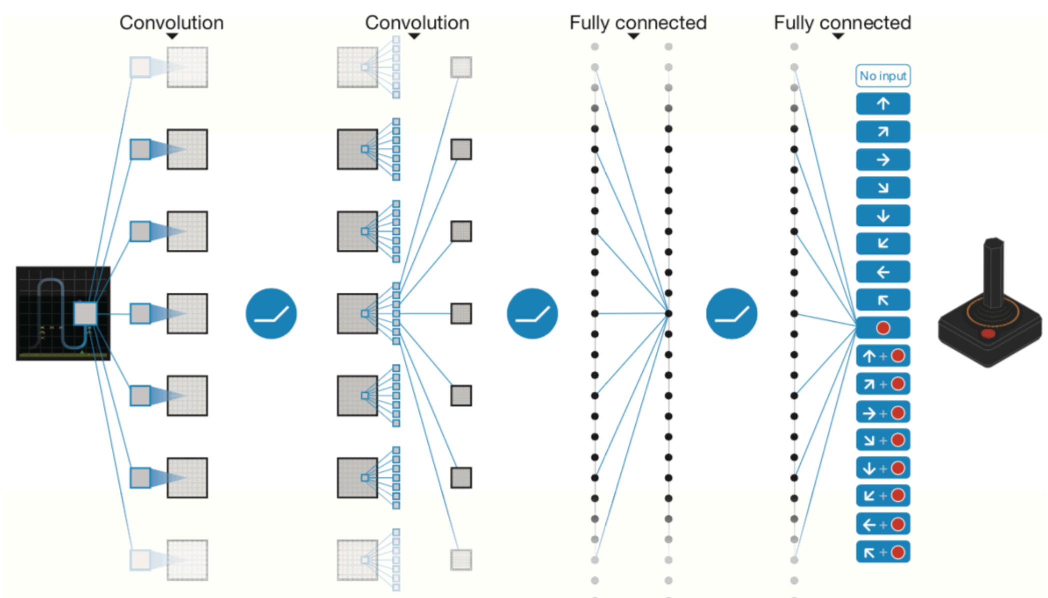
Boyan and Moore (1995). Generalization in reinforcement learning: Safely approximating the value function. NeurIPS.

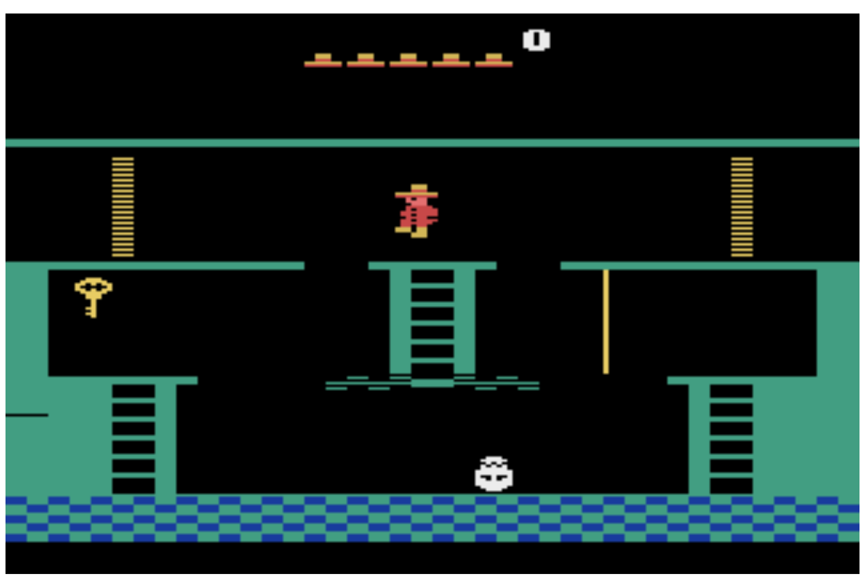
Failure in a simple task

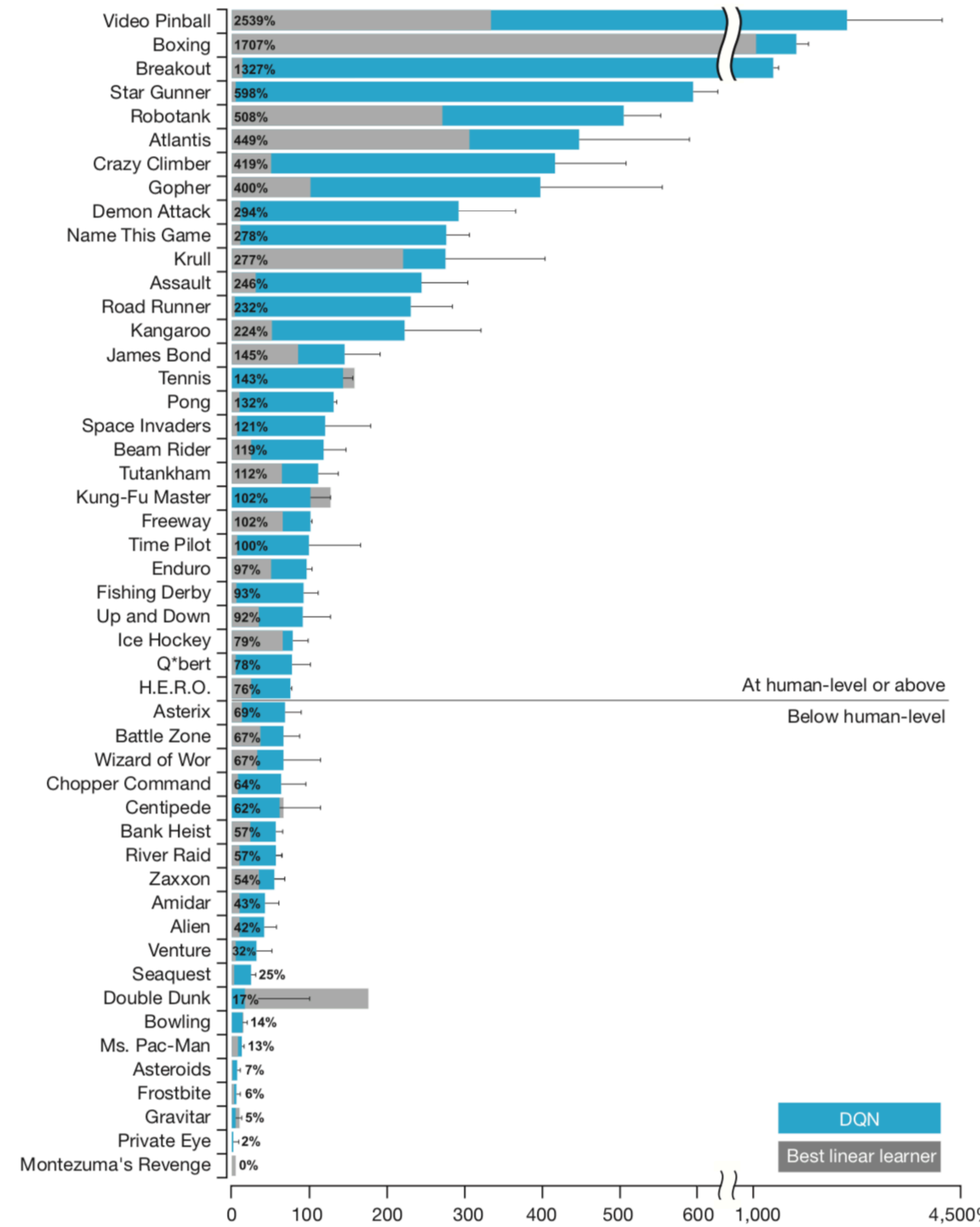


Optimal policy for two similar states can be very different.

Failure in a complex task

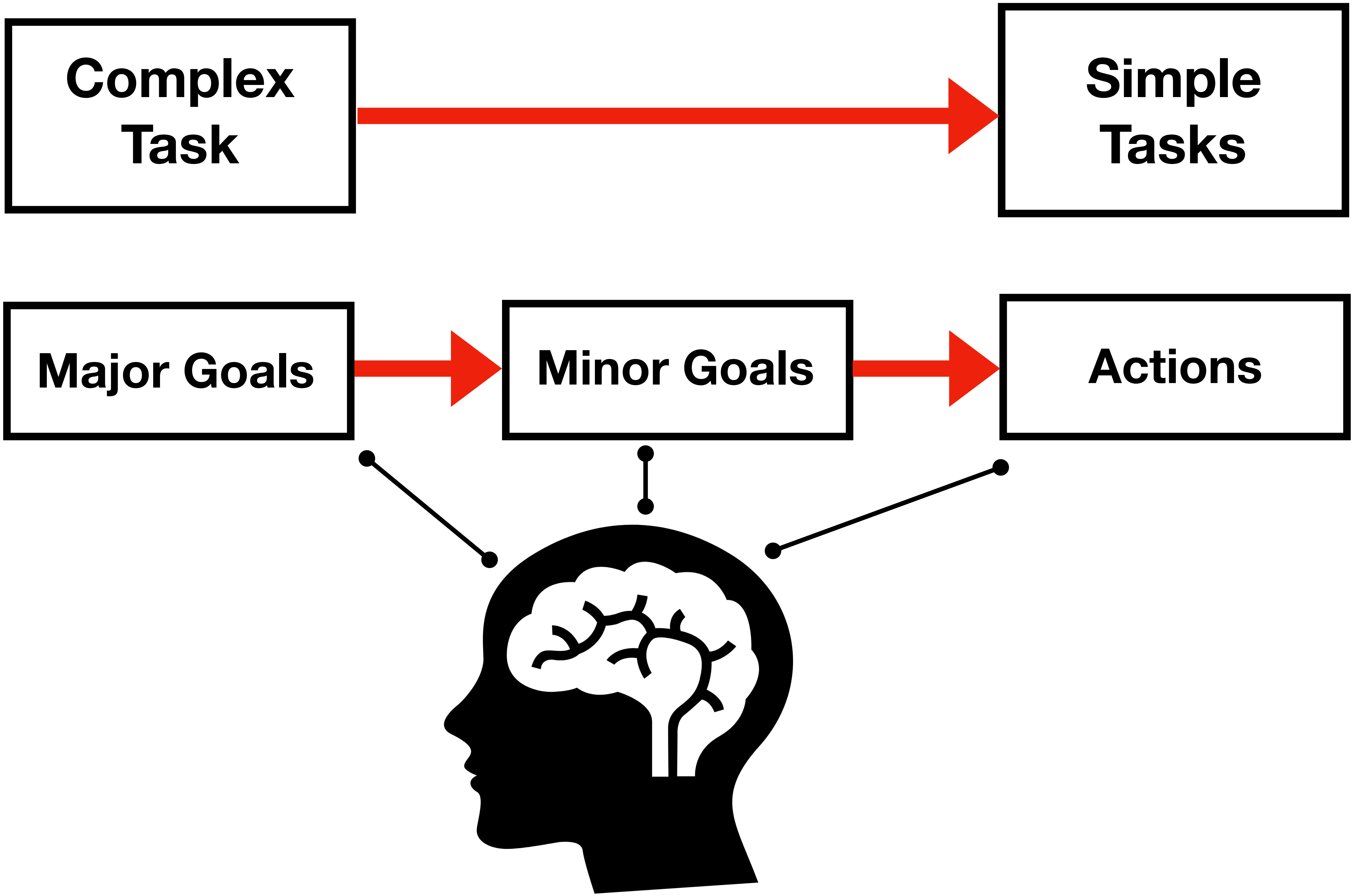




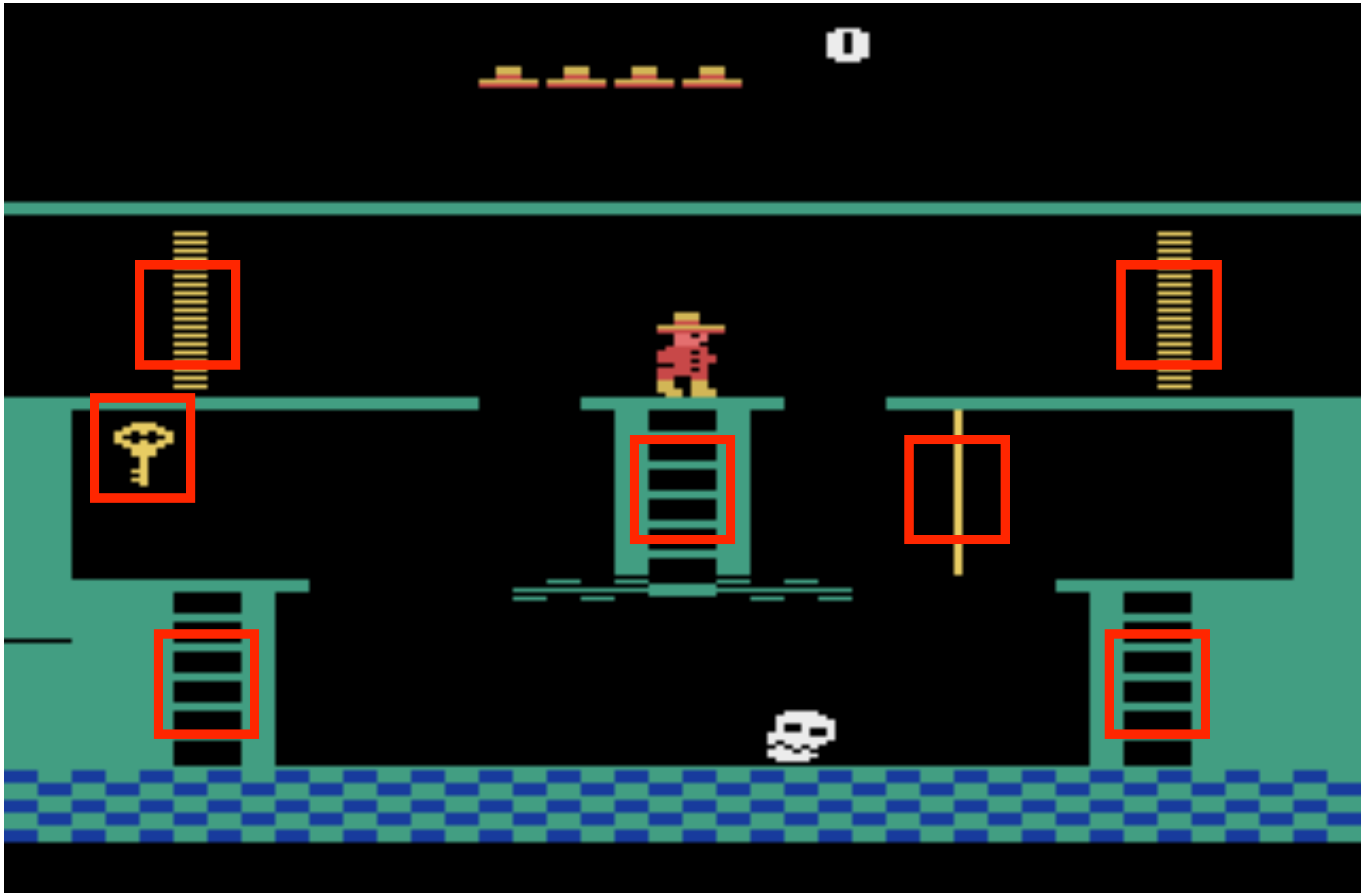


Mnih, et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540):529–533.

Hierarchy in Human Behavior & Brain Structure



Hierarchical Reinforcement Learning

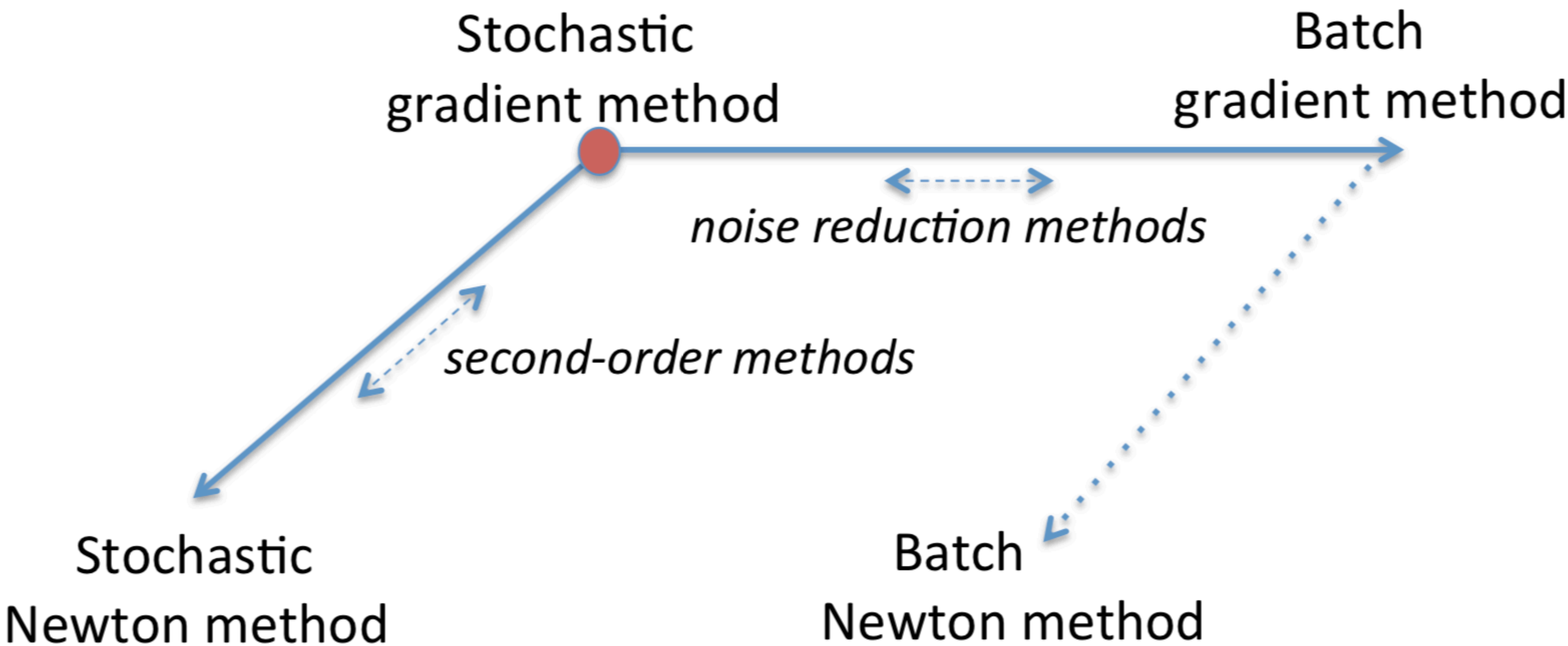


Empirical Risk Minimization in Deep Learning and Deep RL

$$\min_{w \in \mathbb{R}^n} \mathcal{L}(w) \triangleq \frac{1}{N} \sum_{i=1}^{N} \ell_i(w)$$

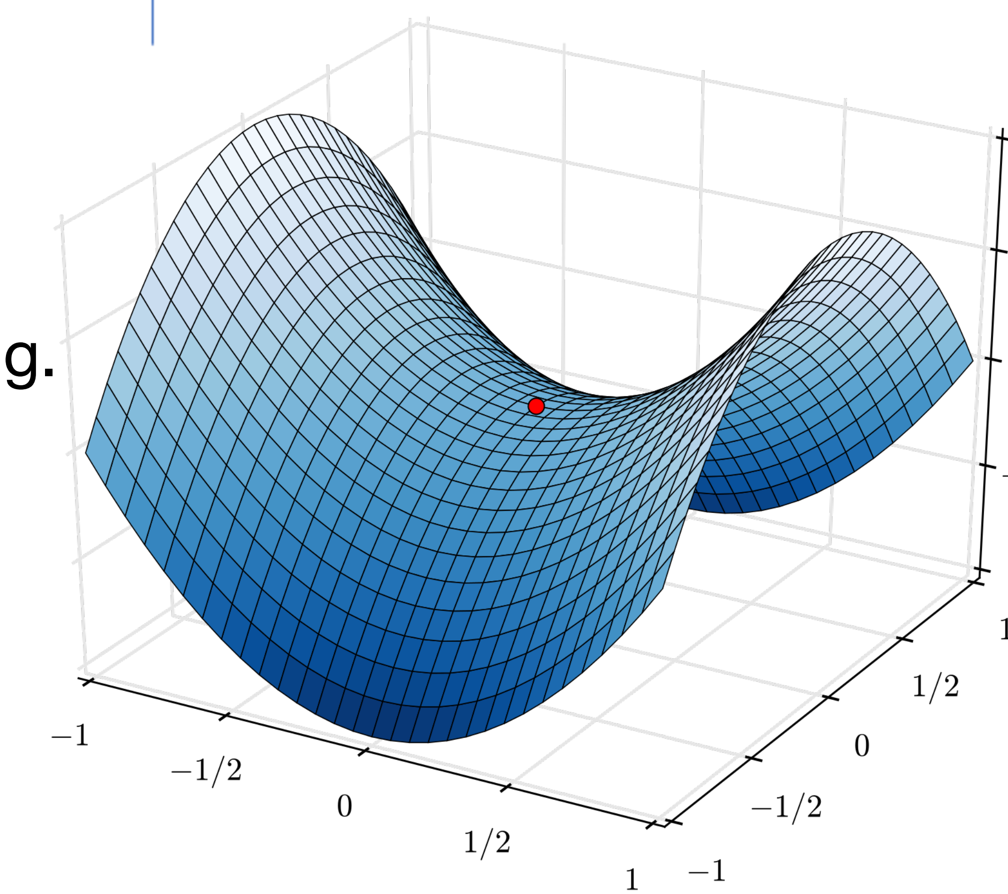
$$\mathcal{L} : \mathbb{R}^n \to \mathbb{R}$$

Optimization Algorithms



Bottou et al., (2016) Optimization methods for large-scale machine learning, print arXiv:1606.04838

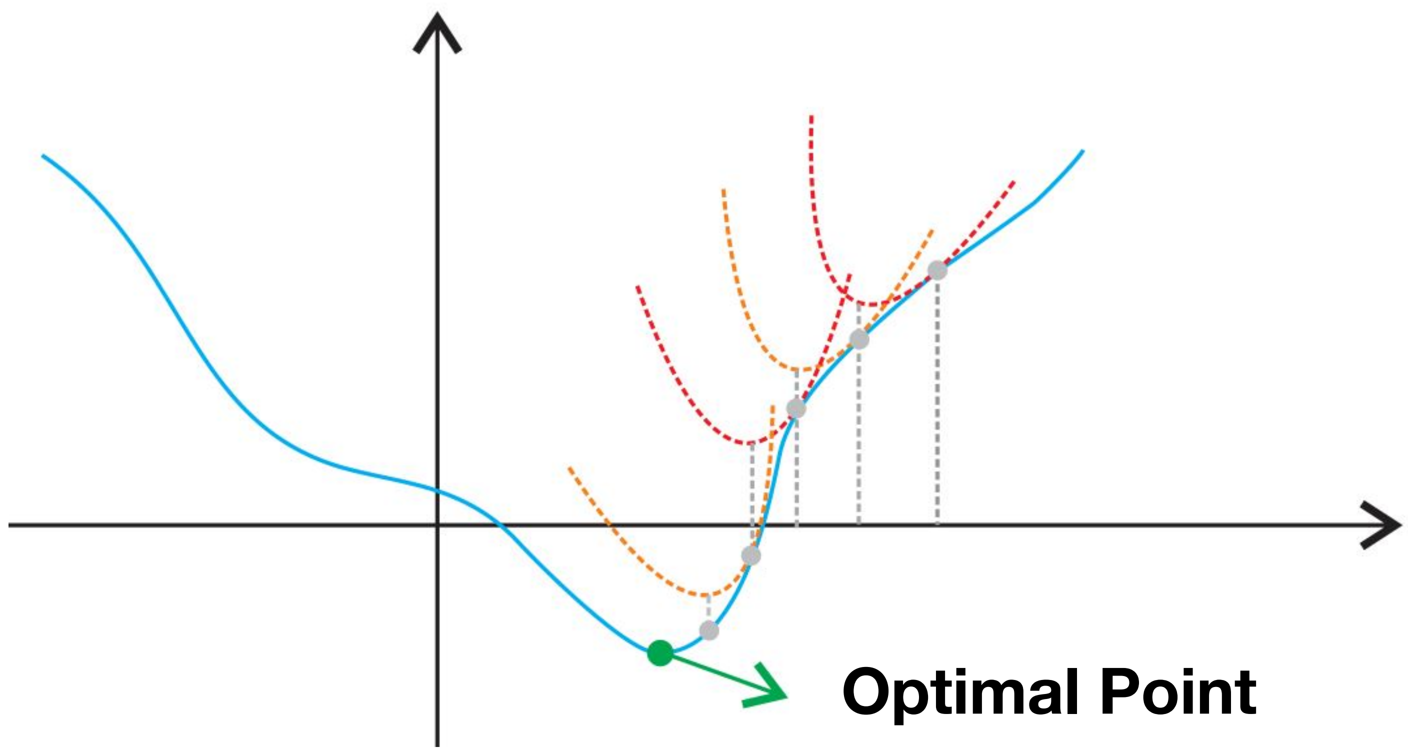
Stochastic Gradient Decent



- Very sensitive to choice of learning rate.
- Very sensitive to the ill-conditioning problem and scaling.
- Requires fine-tuning many hyper-parameters.
- Can stuck in a saddle-point instead of local minimum.
- Sublinear and slow rate of convergence.

H. Robbins, D. Siegmund. (1971). "A convergence theorem for non negative almost supermartingales and some applications". Optimizing methods in statistics.

Second-Order Methods



Advantages:

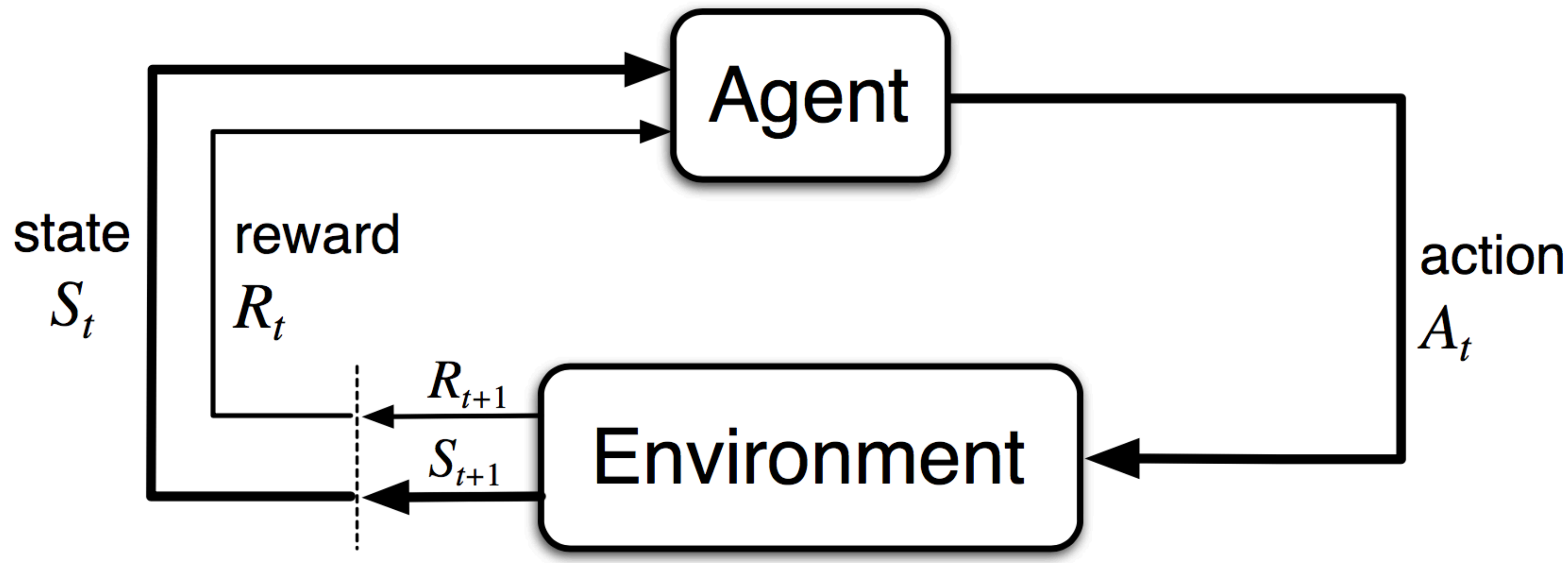
- The rate of convergence is quadratic.
- They are resilient to problem ill-conditioning.
- They involve less parameter tuning.
- They are less sensitive to the choice of hyper-parameters.

Disadvantages:

- Computing the Hessian matrix is very expensive and requires massive storage.
- Computing the inverse of Hessian is not practical.

Reinforcement Learning Algorithms

Reinforcement Learning



We want to maximize expectation of return for each state

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots$$

and find the optimal action-selection policy

$$\pi^* = \arg\max_{\pi} \mathbb{E}\big[G_t | S_t = s\big], \quad \forall s \in \mathcal{S}$$

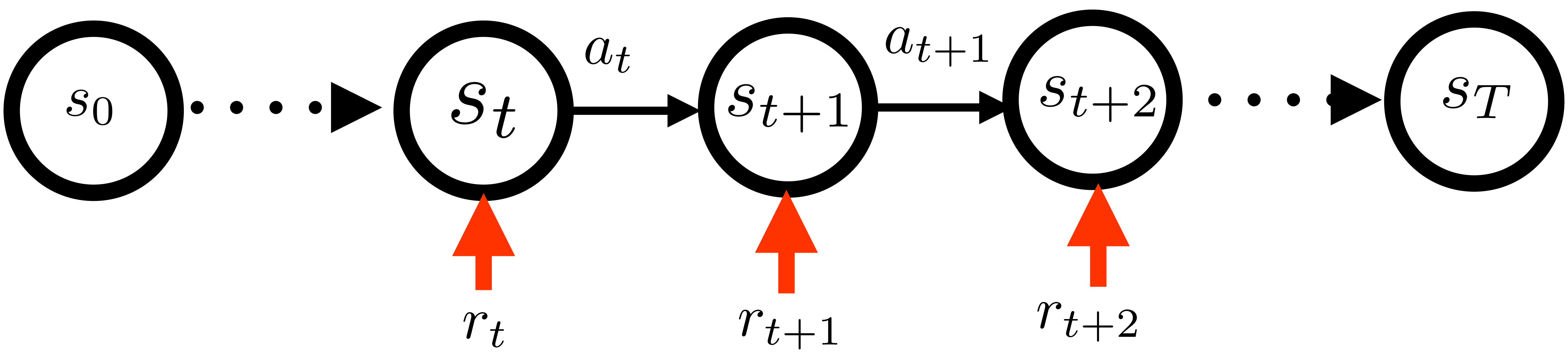
Sutton and Barto (2017). Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, USA, 2nd edition.

Return

Return is the cumulative sum of a future rewards:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots$$

$$\gamma \in (0, 1] \text{ is a discount factor}$$

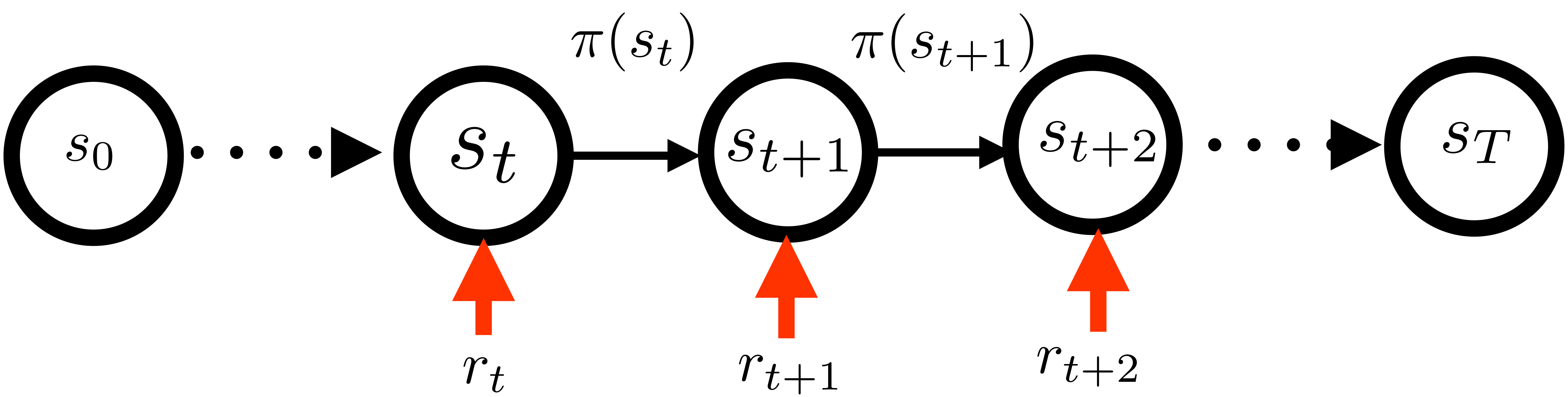


Policy Function

Policy Function: At each time step agent implements a mapping from states to possible actions

$$\pi : \mathcal{S} \rightarrow \mathcal{A}$$

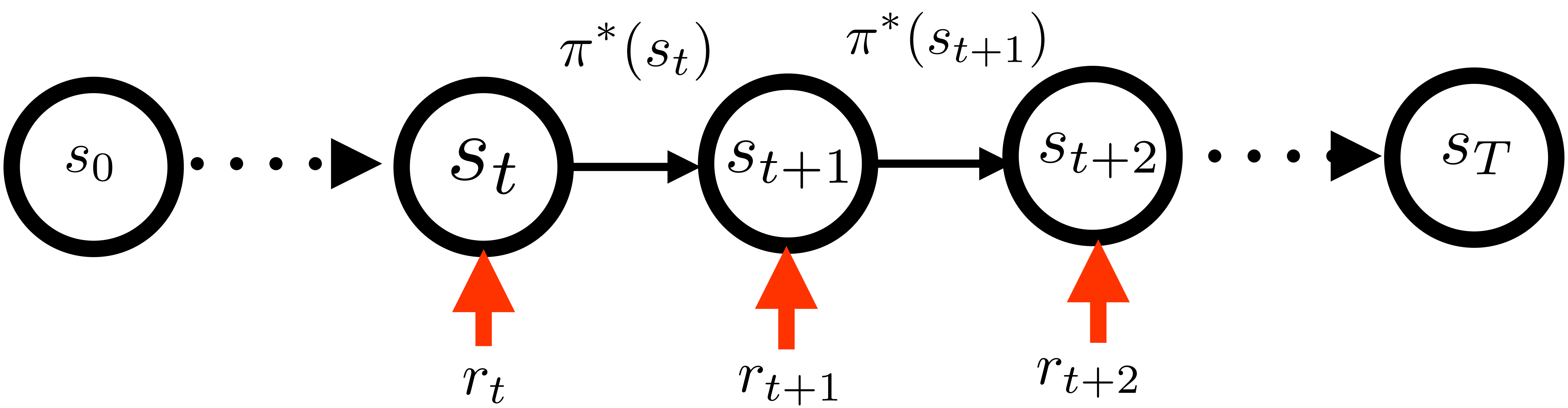
$$a_t = \pi(s_t)$$



Objective of RL

Finding an **optimal policy** that maximizes the expectation of **return**

$$\pi^* = \arg\max_{\pi} \mathbb{E}\big[G_t | S_t = s\big], \quad \forall s \in \mathcal{S}$$



Q-Function

State-Action Value Function (Q-Function) is the expected return when starting from *(s,a)* and following *a policy* thereafter

$$q_\pi(s,a) = \mathbb{E}_\pi\left[G_t \mid S_t = s, A_t = a\right]$$

The optimal *Q*-Function is the maximum expected return.

$$q^*(s,a) = \max_\pi q_\pi(s,a)$$

The optimal policy can be obtained from the optimal *Q*-Function

$$\pi^*(s) = \arg\max_a q^*(s,a)$$

Markov Decision Process

$$(\mathcal{S}, \mathcal{A}, P, R, \gamma)$$

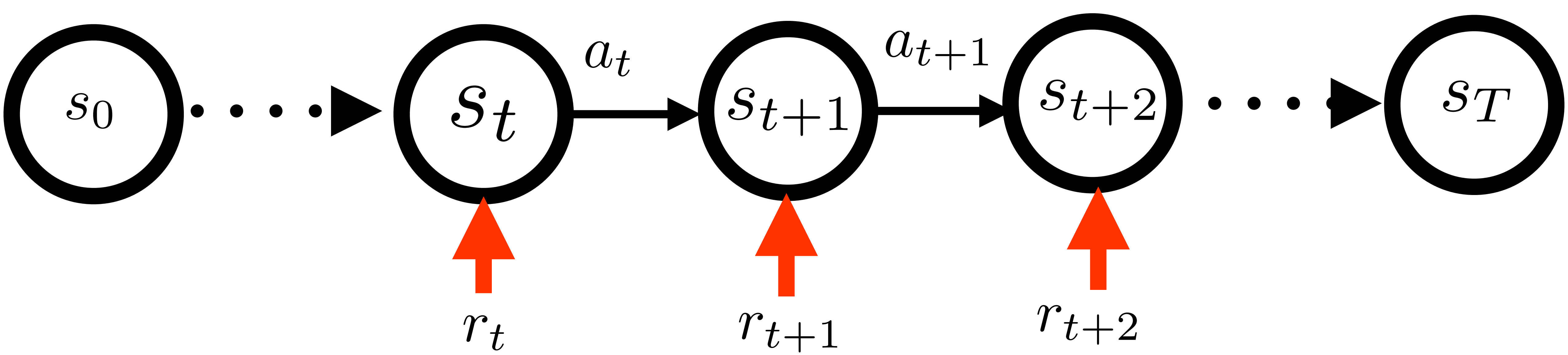
$\mathcal{S}$ is a finite set of states

$\mathcal{A}$ is a finite set of actions

$P(s', s, a) = \Pr(S_{t+1} = s' | S_t = s, A_t = a)$ is state-transition probability

$R(s, a)$ is the expected reward

$\gamma \in [0, 1]$ is the discount factor



Properties of Return and Value Function

The return has a recursive property

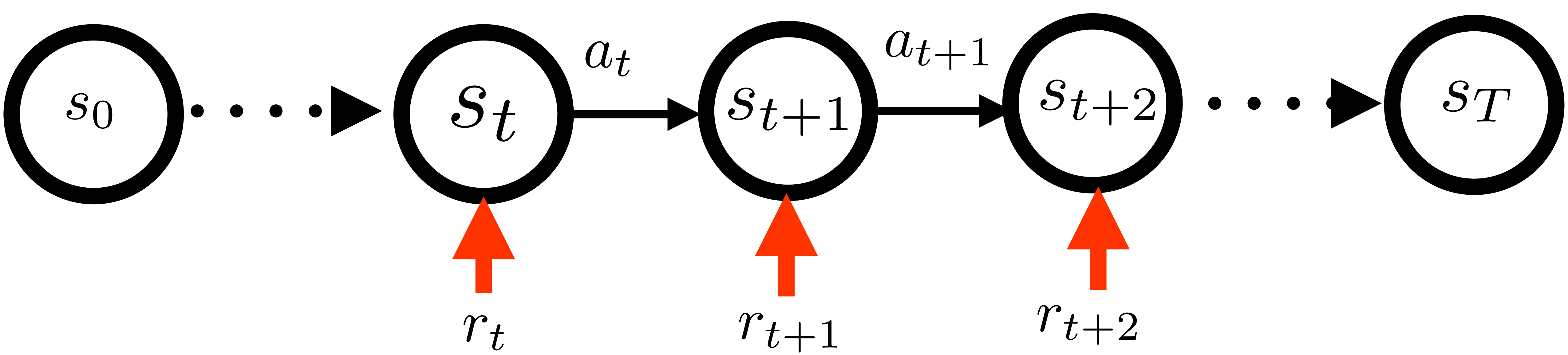
$$G_t = R_{t+1} + \gamma\big(R_{t+2} + \gamma R_{t+3} + \dots\big)$$

$$G_t = R_{t+1} + \gamma G_{t+1}$$

Therefore, there is a recursive property in value function

$$q(s,a) = \mathbb{E}\big[G_t \mid S_t = s, A_t = a\big]$$

$$q(s,a) = \mathbb{E}\big[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a\big]$$



Bellman Optimality Equation

Necessary condition for optimality associated with dynamic programming.

Bellman's Optimality in Expectation form:

$$q^*(s,a) = \mathbb{E}\big[r(s,a) + \gamma \max_{a'} q^*(s',a') \mid S_t = s, A_t = a\big]$$

$$\mathbb{E}\Big[\underbrace{r(s,a) + \gamma \max_{a'} q^*(s',a')}_{\text{target}} - \underbrace{q^*(s,a)}_{\text{prediction}} \Big| s,a\Big] = 0, \ \forall\ s,a$$

Bellman's Optimality in MDP framework (Empirical form):

$$q^*(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \max_{a'} q^*(s',a')$$

Value Iteration

Algorithm Value Iteration

Initialize: $q^0(s,a)$ arbitrarily for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

repeat (for ever)

for all $s \in \mathcal{S}$ **do**

$q^{i+1}(s,a) \leftarrow \left\{R(s,a) + \gamma \sum_{s'} p(s'|s,a) \max_{a'} q^i(s',a')\right\}$

$\pi(s) \leftarrow \arg\max_a q^i(s,a)$

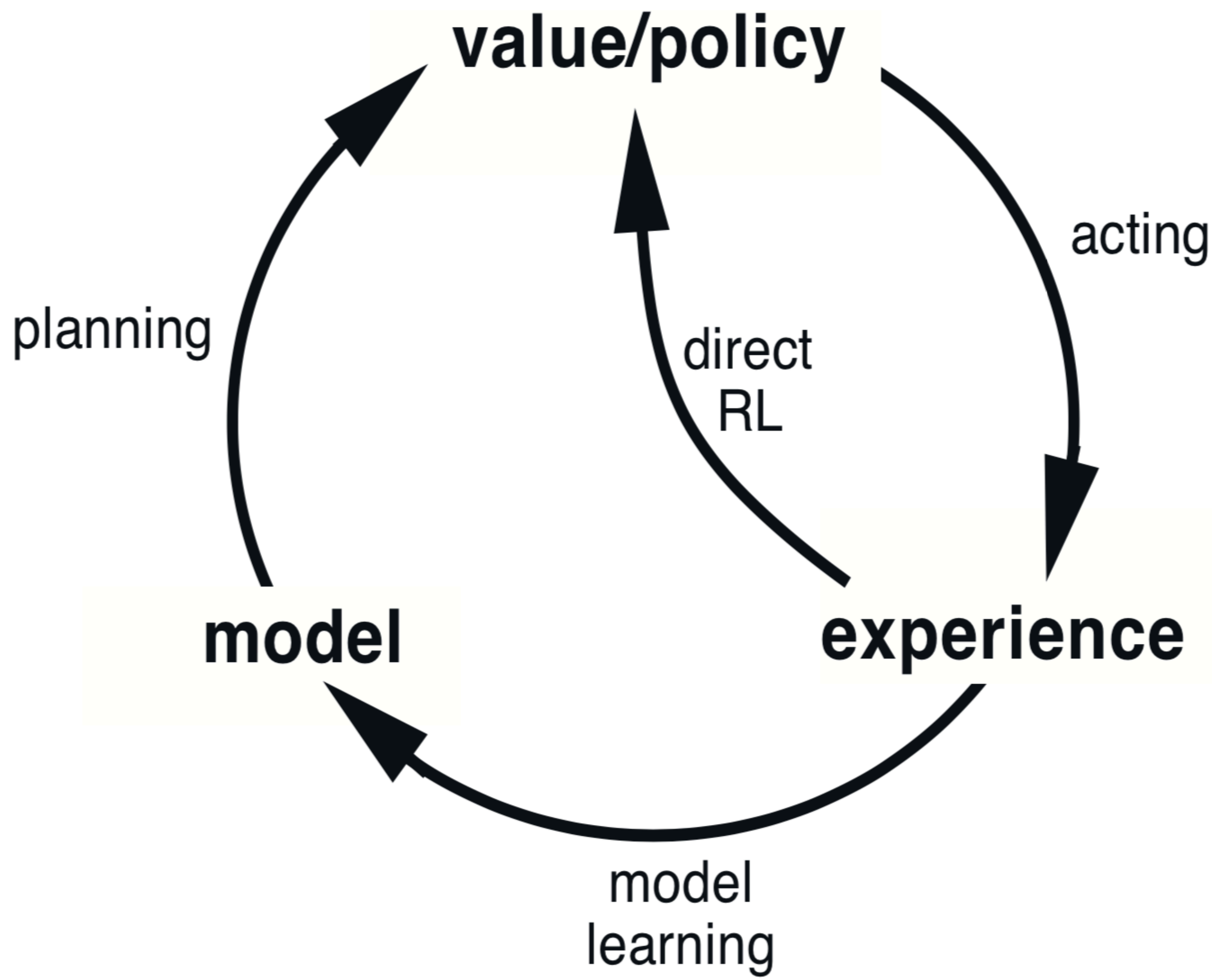
end for

until value function and policy is stable

return $q^*(s,a), \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}$

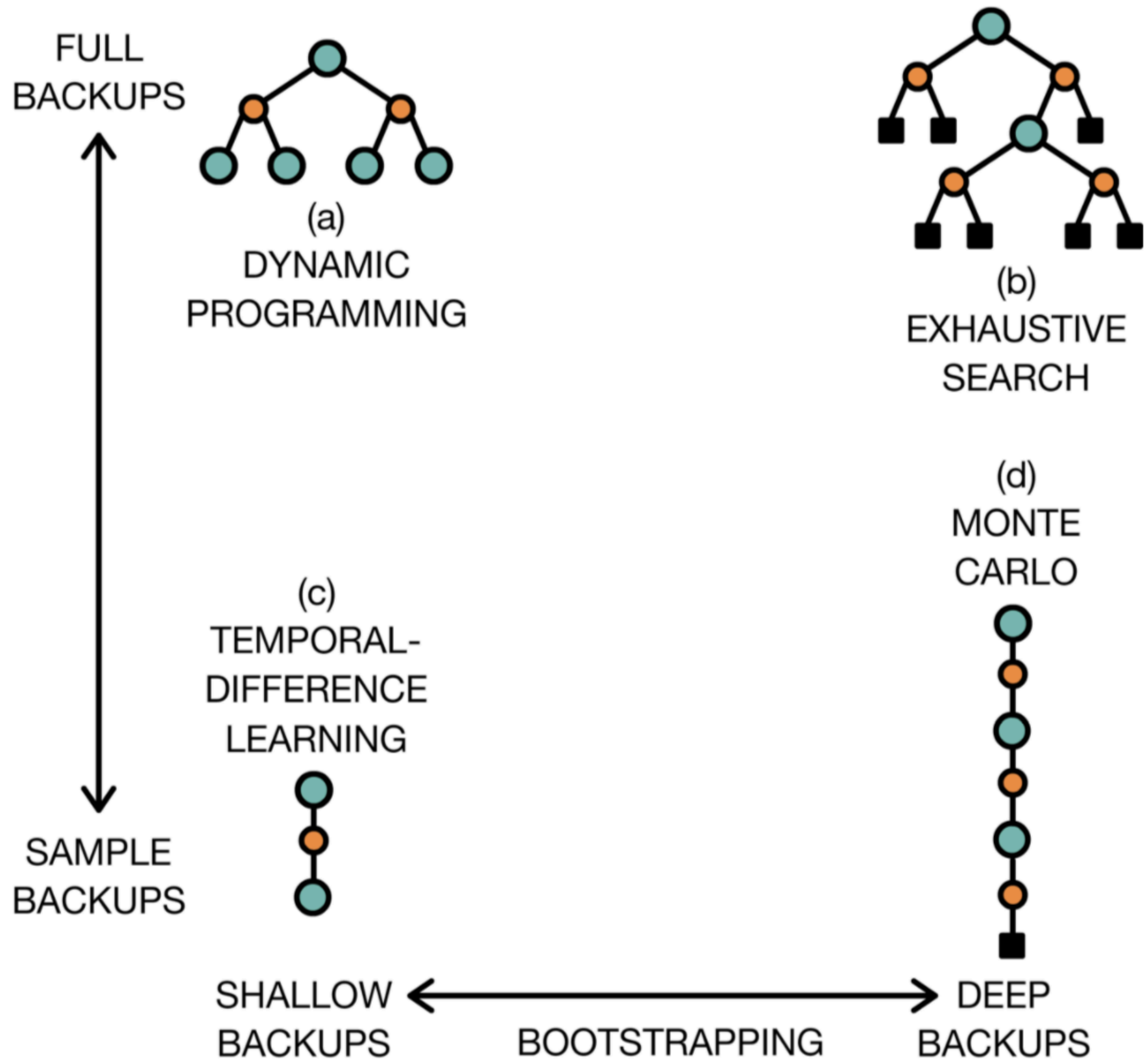
return $\pi^*(s) = \arg\max_a q^*(s,a), \quad \forall s \in \mathcal{S}$

RL Algorithms
Model-Free vs. Model-Based



Sutton and Barto (2017). Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, USA, 2nd edition.

RL Algorithms



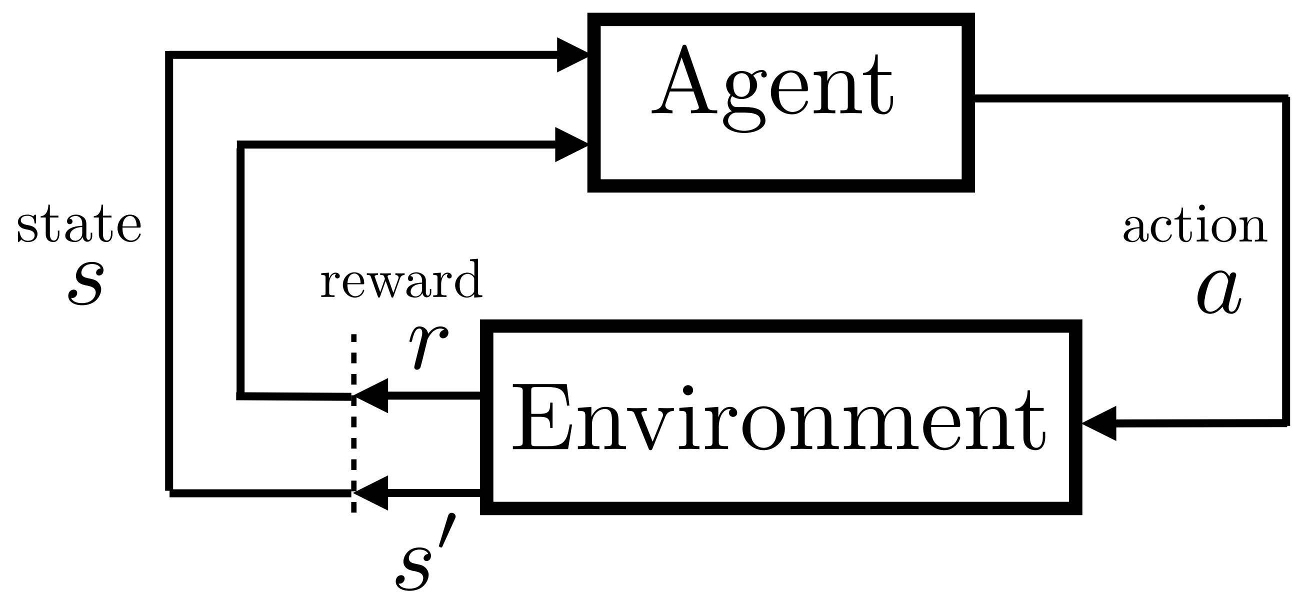
Arulkumaran et al. (2017). Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, 34(6):26–38.

Temporal Difference

- Model free reinforcement learning algorithm, i.e. State-transition probabilities or reward function are not available.
- A combination of Monte Carlo method and Dynamic Programming.
- A powerful computational cognitive neuroscience model of learning in brain.
- Q-Learning and SARSA are two popular TD learning methods.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. Machine Learning, 3:9–44.

Q-Learning



We can update our prediction of the return by computing the TD error

$$q(s,a) \leftarrow q(s,a) + \alpha \underbrace{\left(r + \gamma \max_{a'} q(s',a') - q(s,a) \right)}_{\text{TD error}}$$

$$\underbrace{r + \gamma \max_{a'} q(s',a')}_{\text{target}} \qquad \underbrace{q(s,a)}_{\text{prediction}}$$

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. Machine Learning, 3:9–44.

Q-Learning

Algorithm Q-Learning: Off-Policy TD Learning

Input: policy π to be evaluated

Initialize: $Q(s,a)$ arbitrarily for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$.

repeat (for each episode)

  initialize s

  repeat (for each step of episode)

    choose action a using policy derived by Q (e.g. ϵ-greedy)

    take action a, observe reward r and next state s'

    $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$

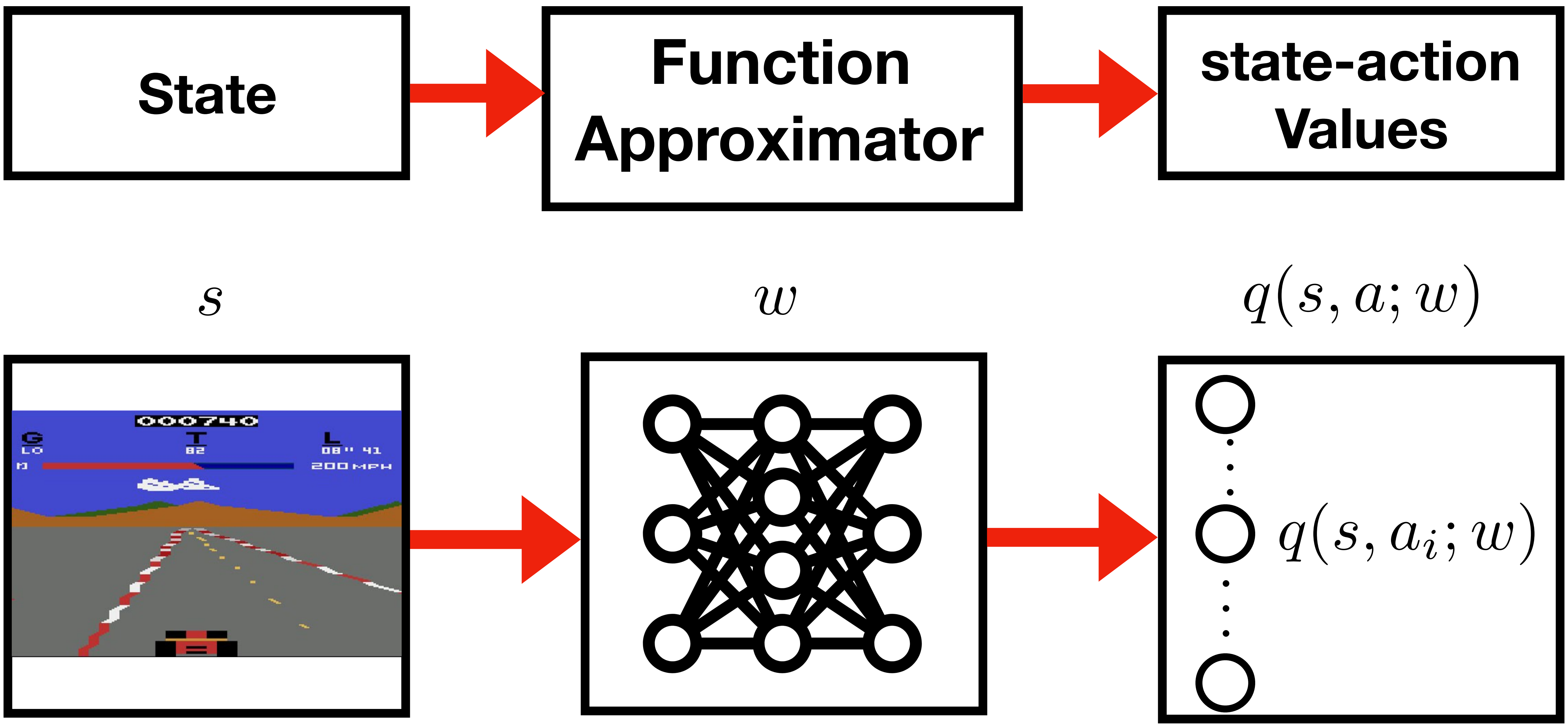
    $s \leftarrow s'$

  until (s is terminal or reaching to max number of steps)

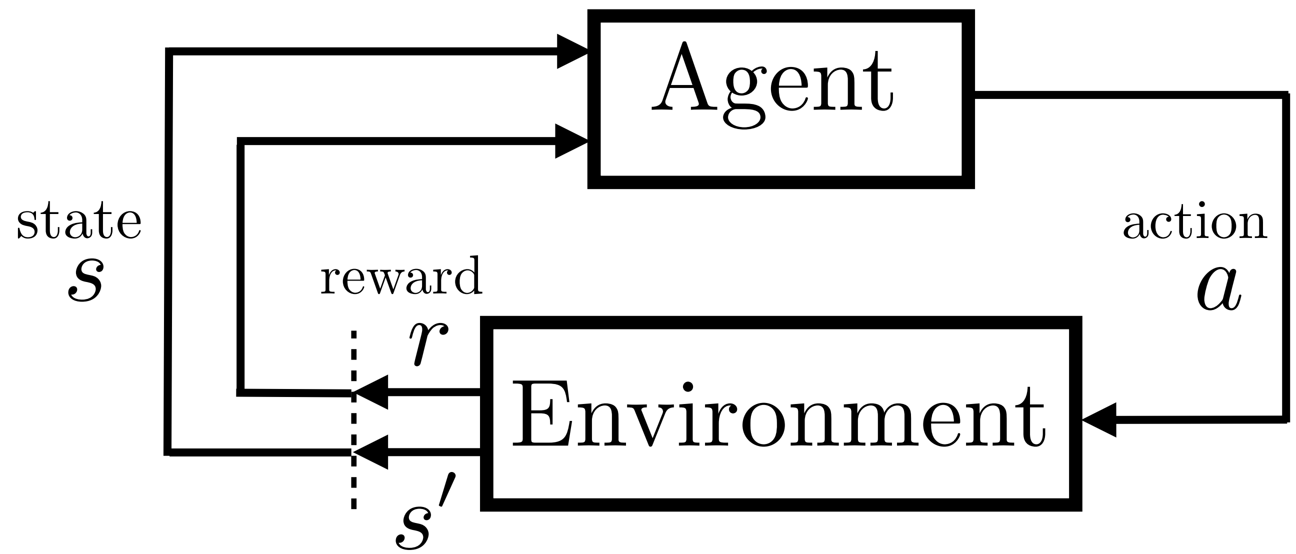
until (convergence or reaching to max number of episodes)

$$\epsilon\text{-greedy}(Q,\epsilon) = \begin{cases} \text{random action } a \in \mathcal{A} & \text{if rand}() < \epsilon \\ \arg\max_a Q(s,a) & \text{otherwise} \end{cases}$$

Generalization



Optimization in Deep RL



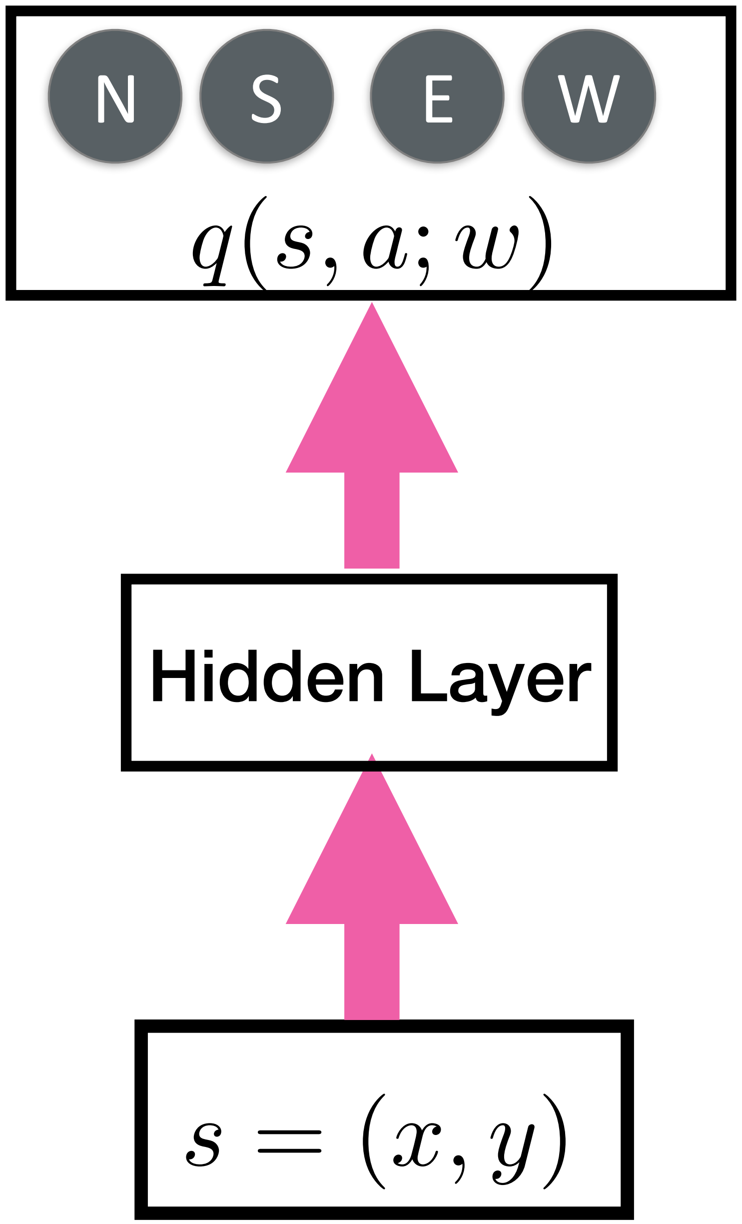
$$\min_{w \in \mathbb{R}} \mathcal{L}(w) \triangleq \frac{1}{N} \sum_{e \in \mathcal{D}} \left(r + \gamma \max_{a'} q(s', a'; w) - q(s, a; w)\right)^2$$

$\mathcal{D} = \{(s, a, s', r)\}$ is Agent's Experiences Memory.

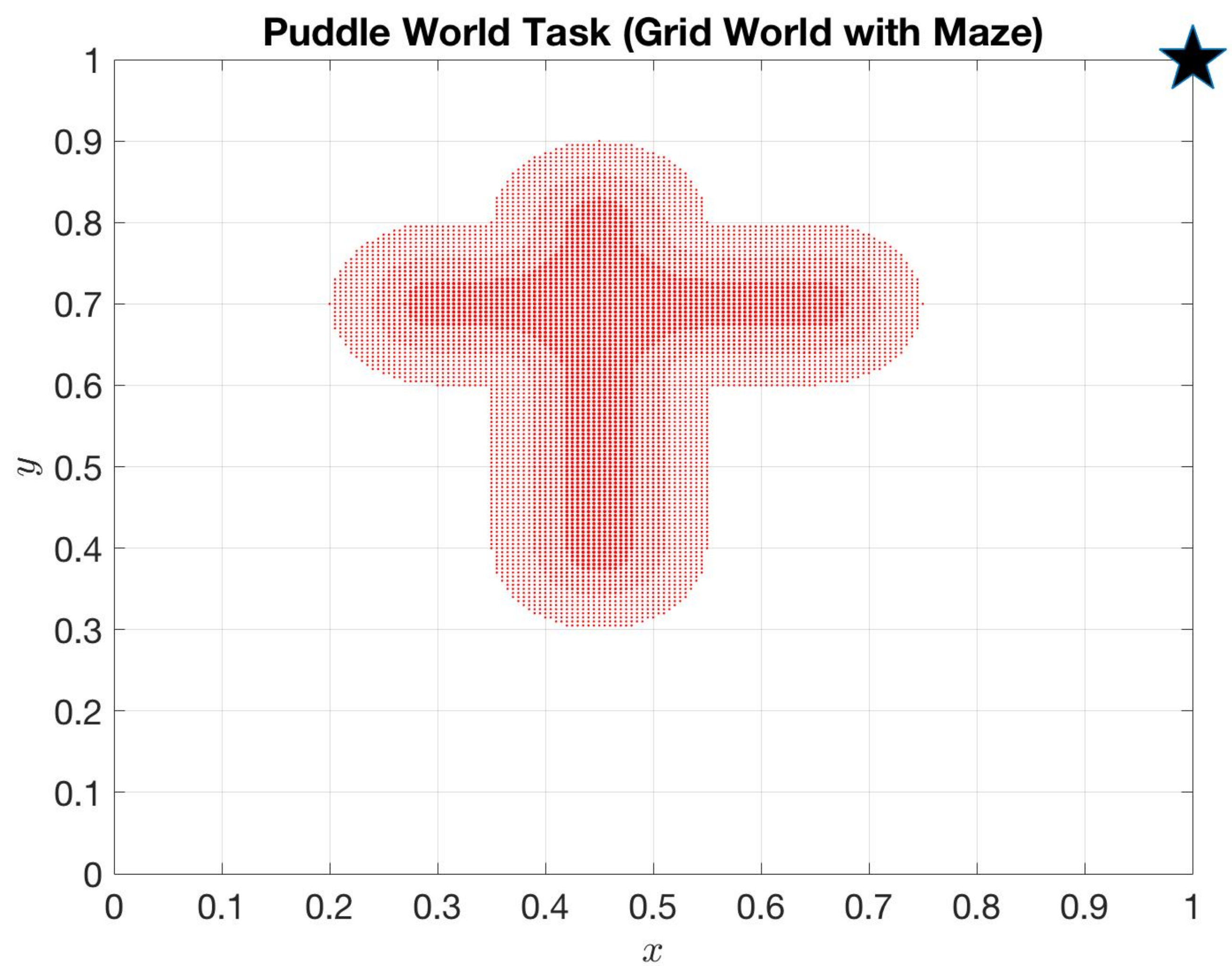
Sutton and Barto (2017). Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, USA, 2nd edition.

Problem 1.
Learning Sparse Representations in Reinforcement Learning

Divergence of Vanilla ANN



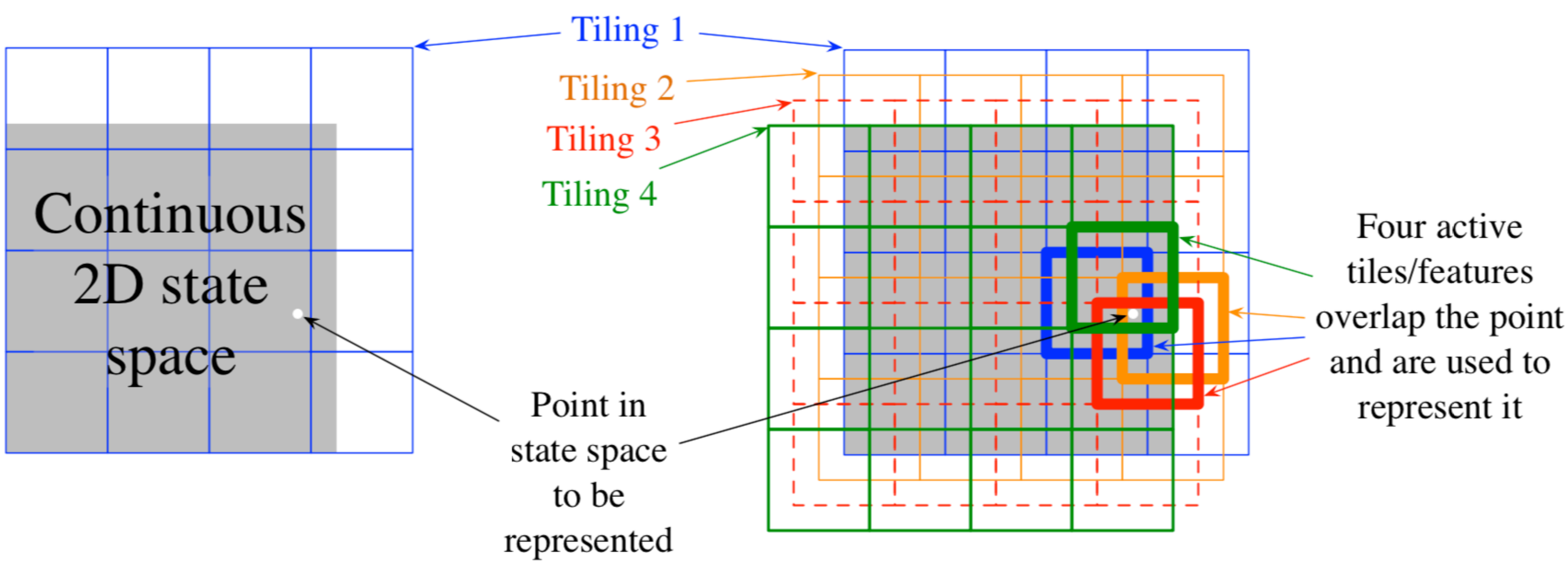
Generalization in Reinforcement Learning: Safely Approximating the Value Function



Boyan and Moore (1995). Generalization in reinforcement learning: Safely approximating the value function. NeurIPS.

Sparse Representation

Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding



Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. NeurIPS.

Problem 1:
Learning Sparse Representation in RL

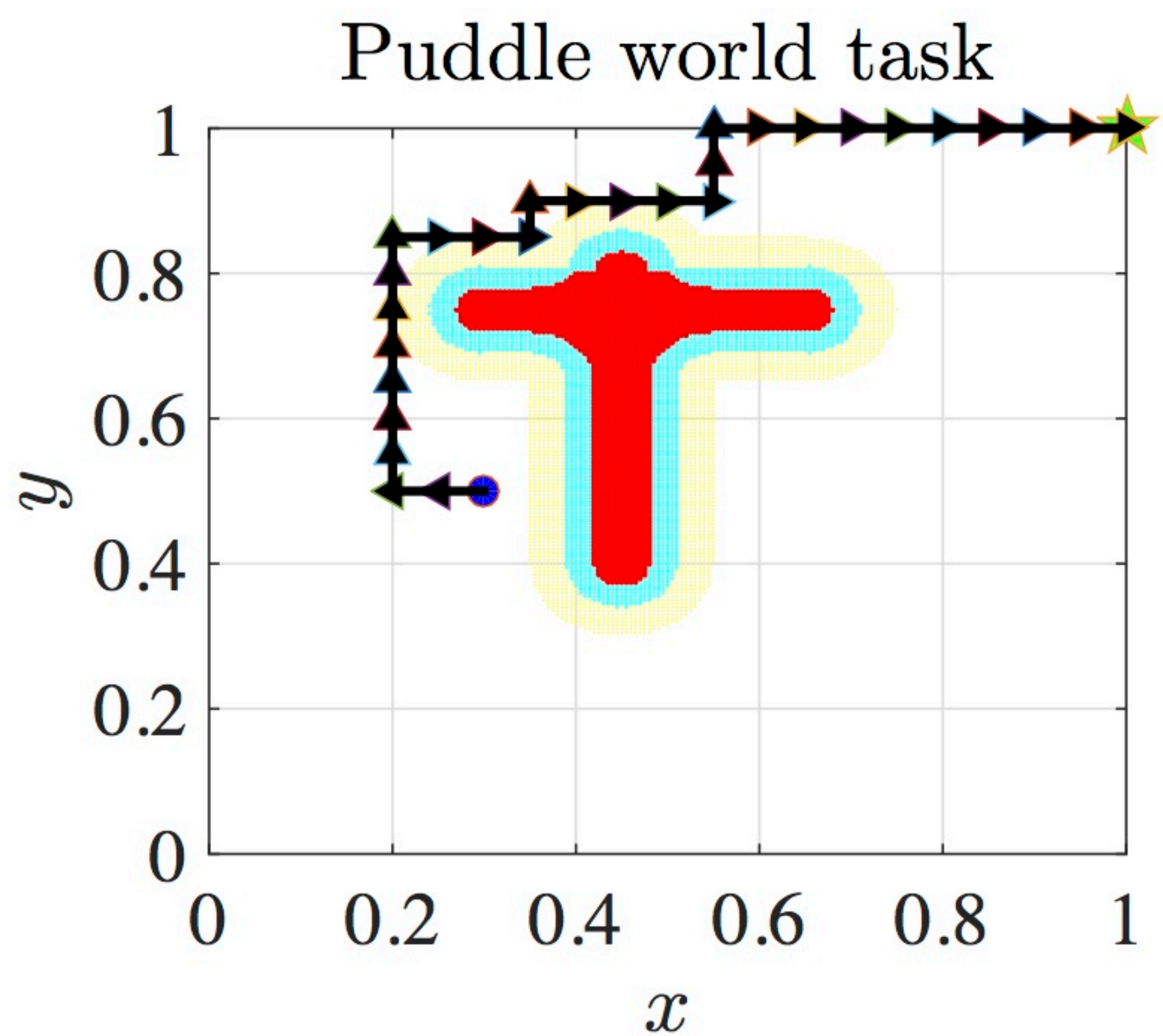
- **Problem Statement**: Generalization over similar states may cause catastrophic interference that unable learning.

- **Objective**: A mechanism for *pattern separation* is required to overcome catastrophic interference when learning highly nonlinear value function.

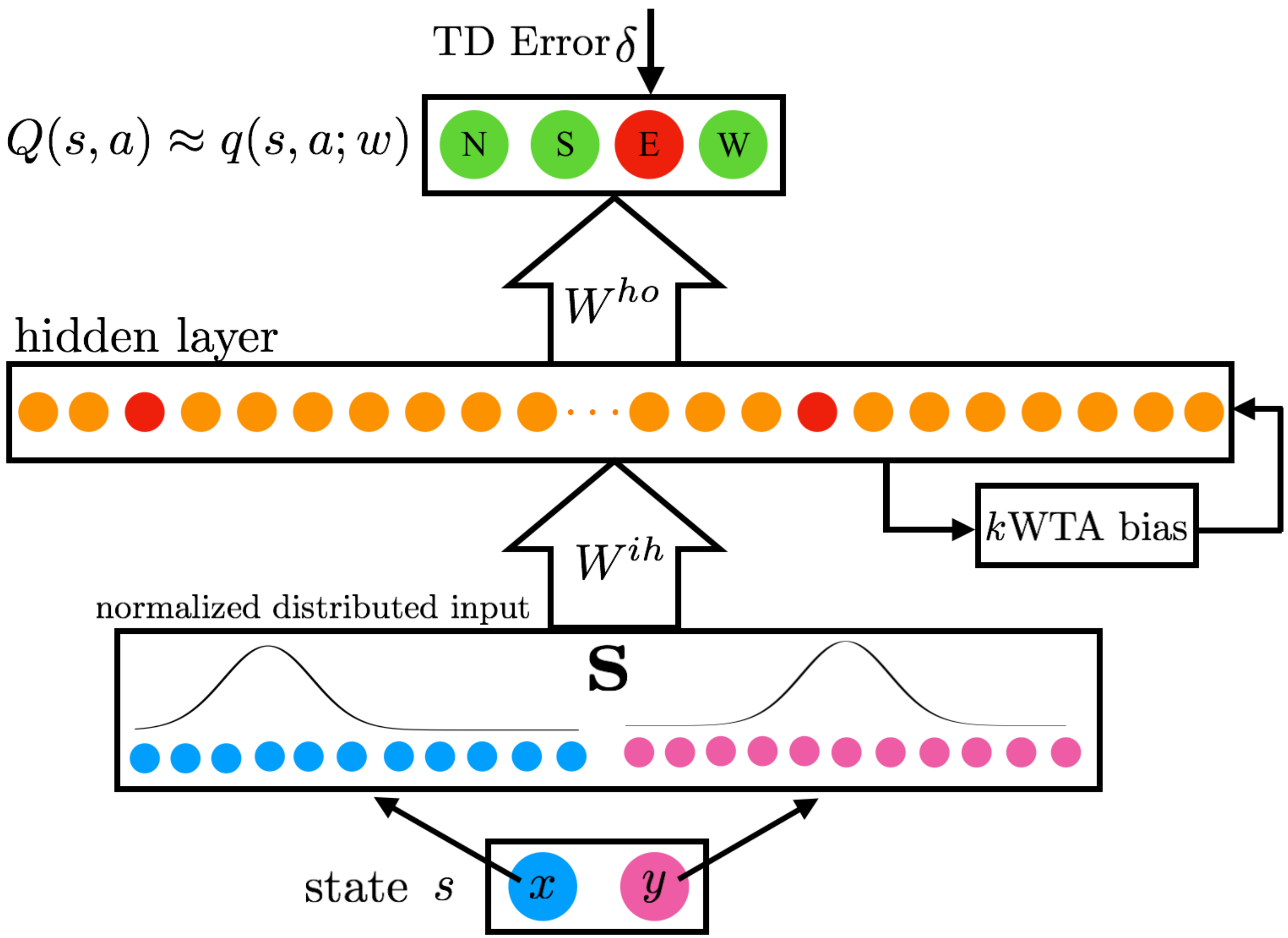
- **Hypothesis**: *Lateral inhibition* mechanism in cortex produce sparse conjunctive representation that helps avoiding catastrophic interference while supporting generalization. This mechanism might help overcoming the catastrophic interference in RL tasks that use neural networks for value function.

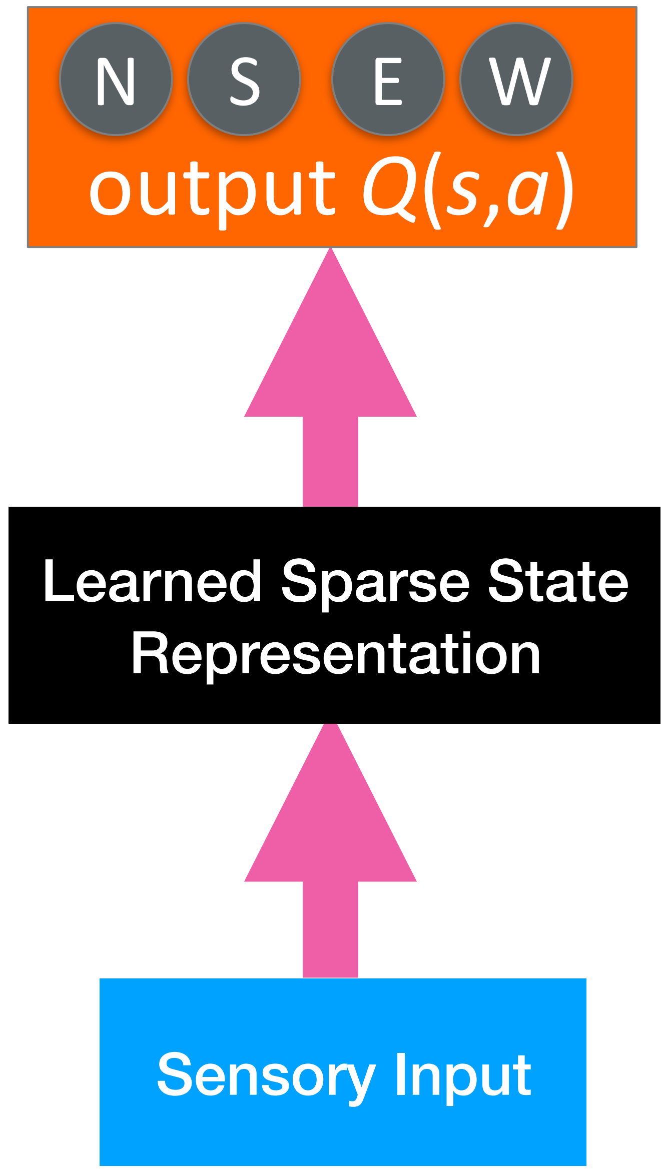
Tasks: Puddle World

$$r(s,a) = \begin{cases} 0, & \text{if } s' \text{ is terminal} \\ -400d, & \text{if } s' \text{ is inside puddle} \\ -2, & \text{if agent bumps the wall} \\ -1, & \textit{otherwise} \end{cases}$$

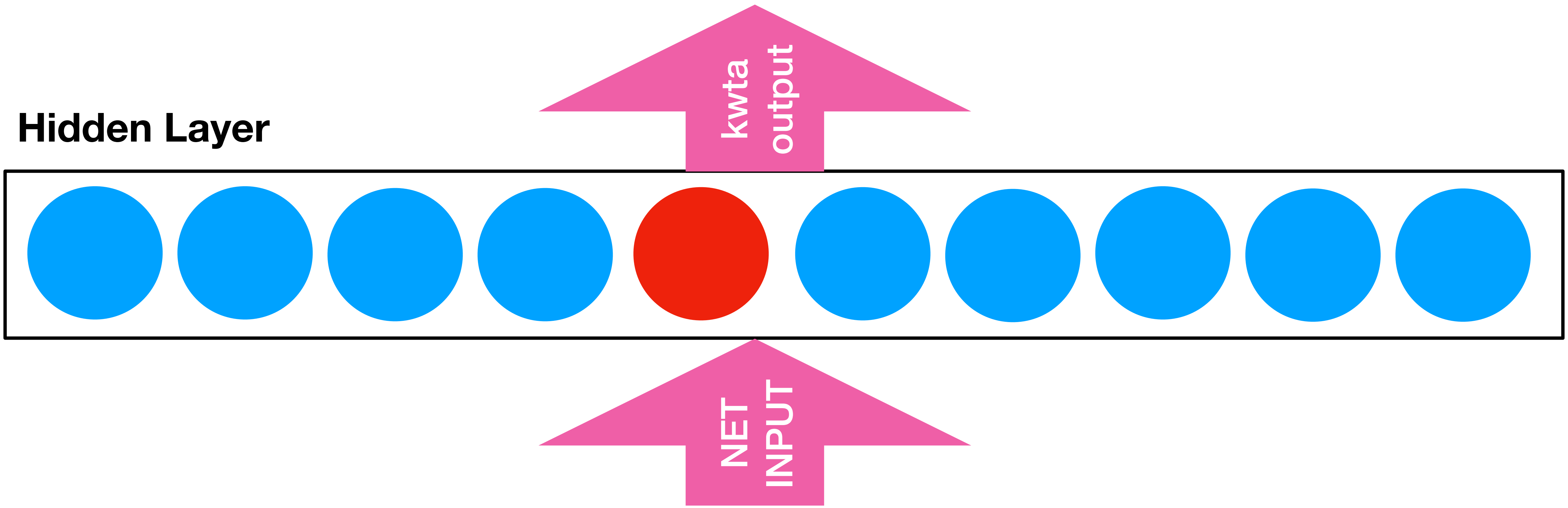


Sparse Representation





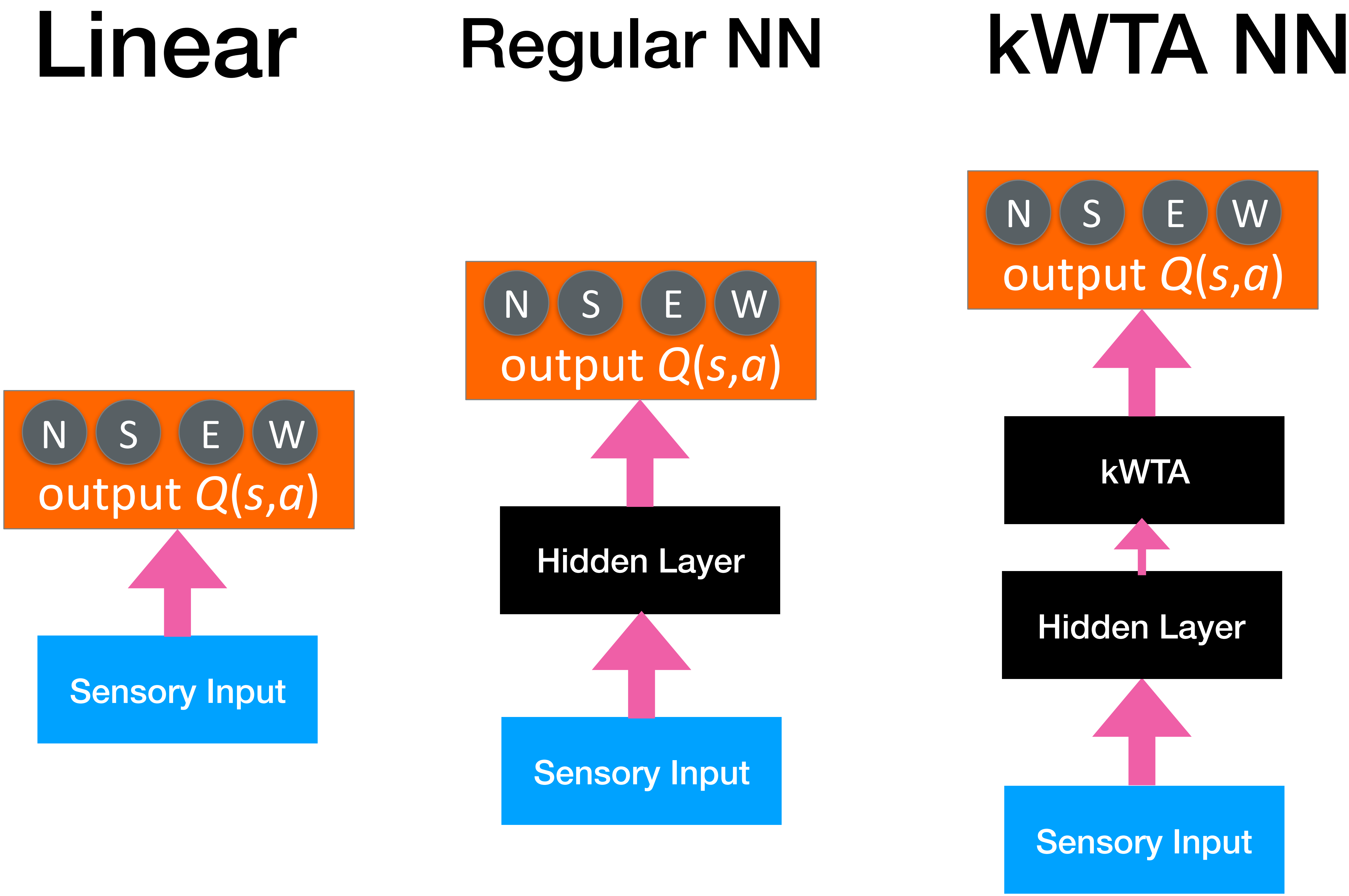
k-Winners-Take-All



- Producing sparse representation in feedforward path, by letting the top *k* active neurons to fire.
- Local smoothness (in apposed to global smoothness of regular NN)
- No need to solve any optimization problem.

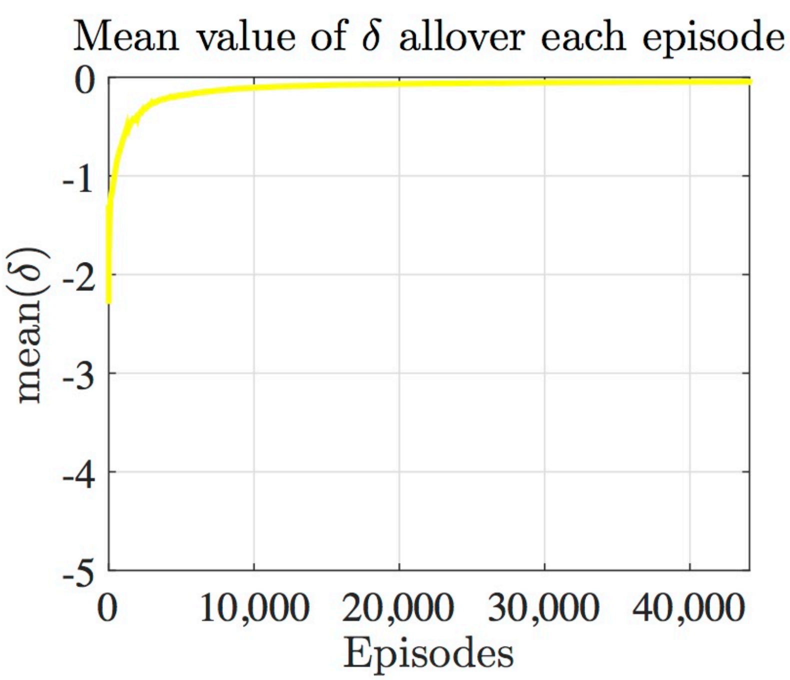
O'Reilly, R. C. and Munakata, Y. (2001). Computational Explorations in Cognitive Neuroscience. MIT Press.

Architectures

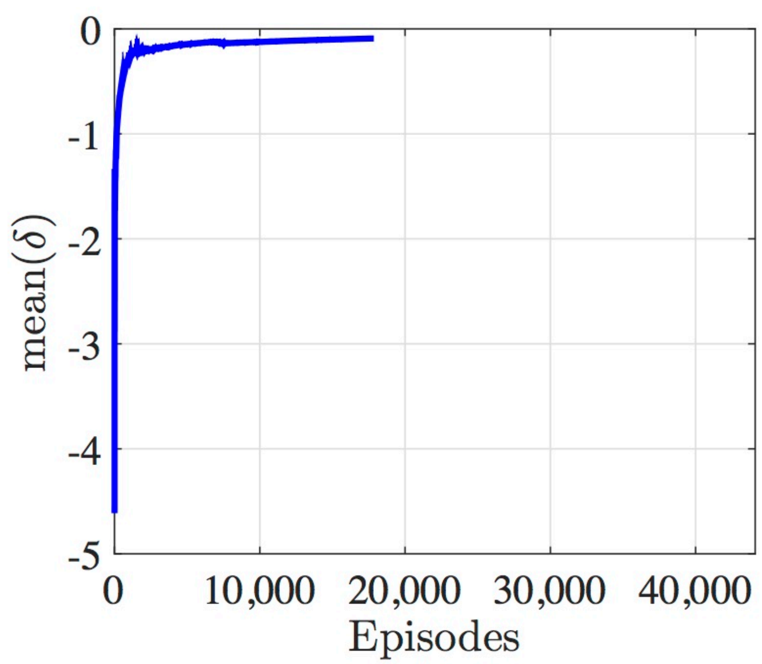


TD error: Puddle World

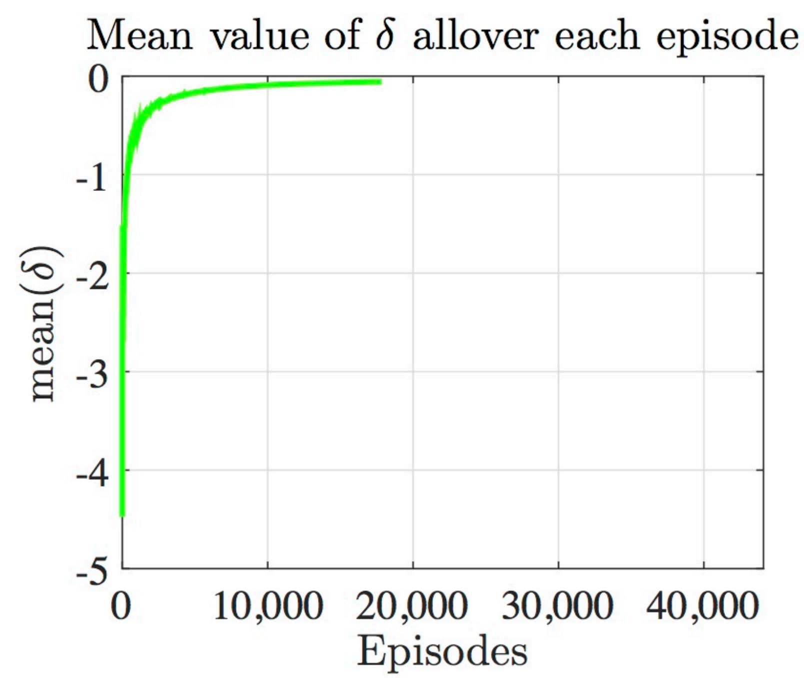
Linear



Regular NN



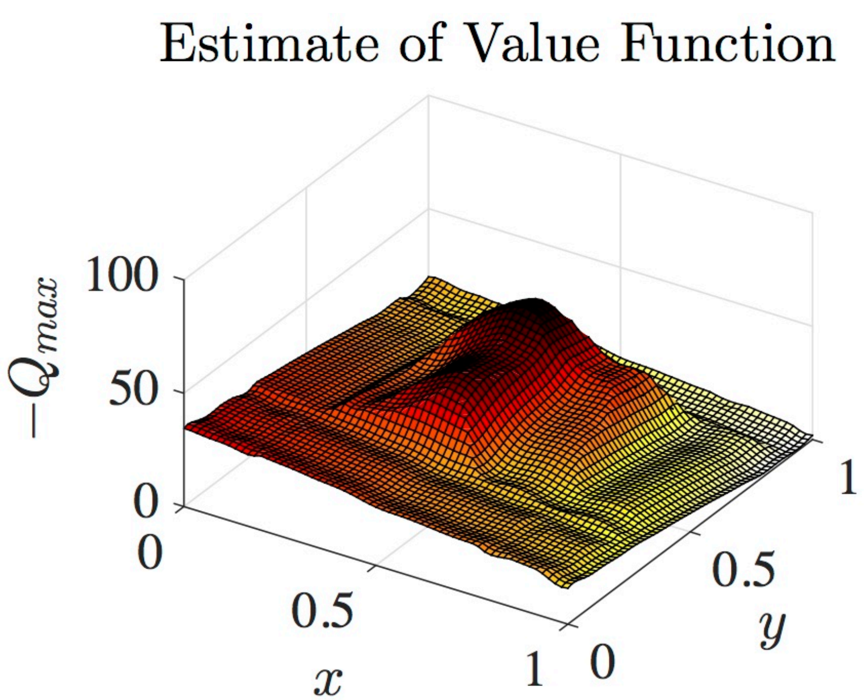
kWTA NN



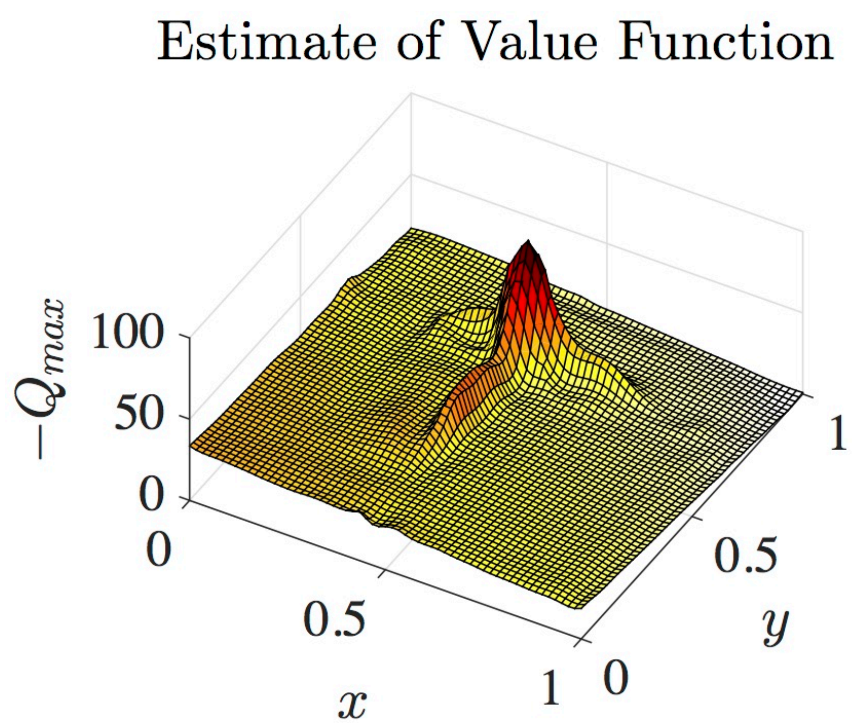
Jacob Rafati and David C. Noelle (2015). Lateral Inhibition Overcomes Limits of Temporal Difference Learning. In 37th Annual Cognitive Science Society Meeting, Pasadena, CA, USA.

Value Function: Puddle World

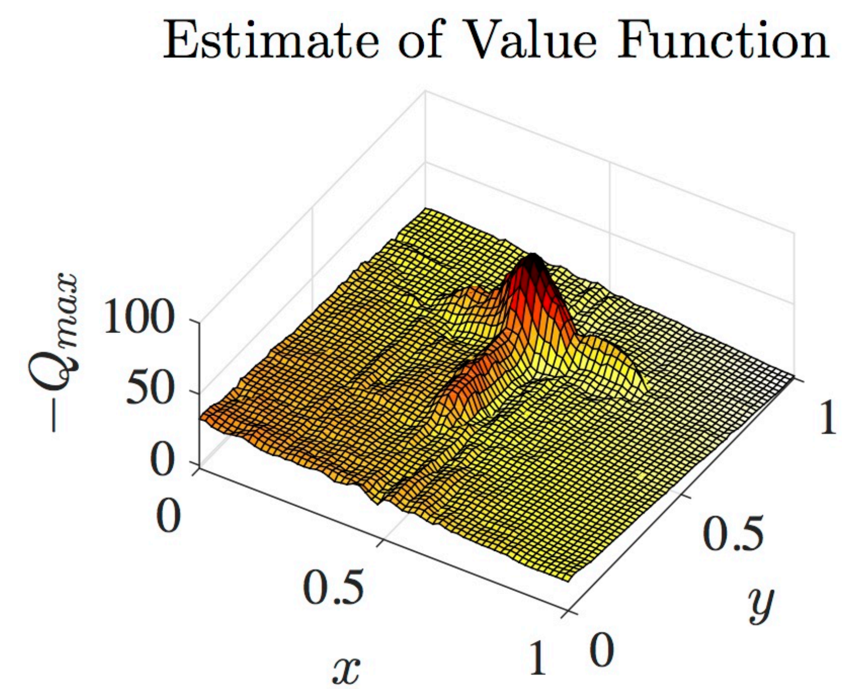
Linear



Regular NN



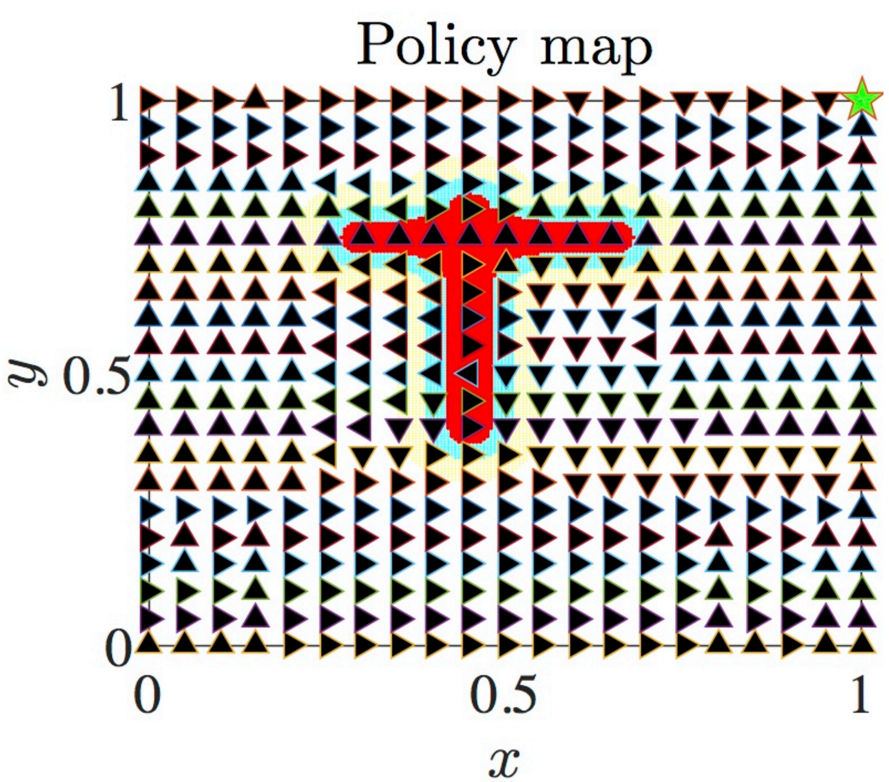
kWTA NN



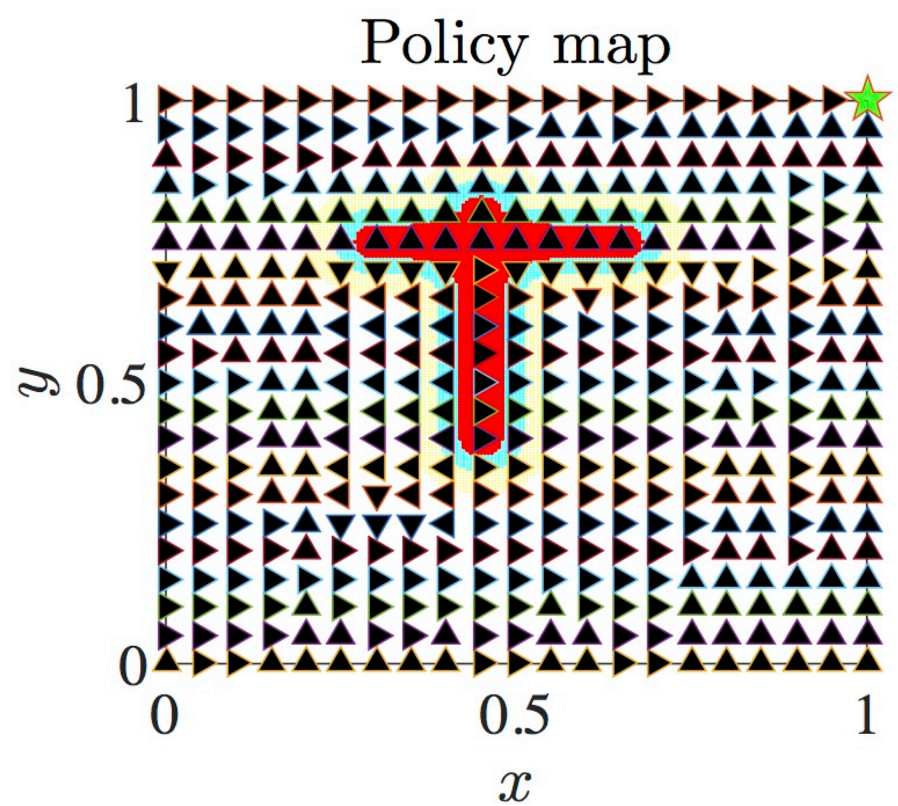
Jacob Rafati and David C. Noelle (2015). Lateral Inhibition Overcomes Limits of Temporal Difference Learning. In 37th Annual Cognitive Science Society Meeting, Pasadena, CA, USA.

Policy Function: Puddle World

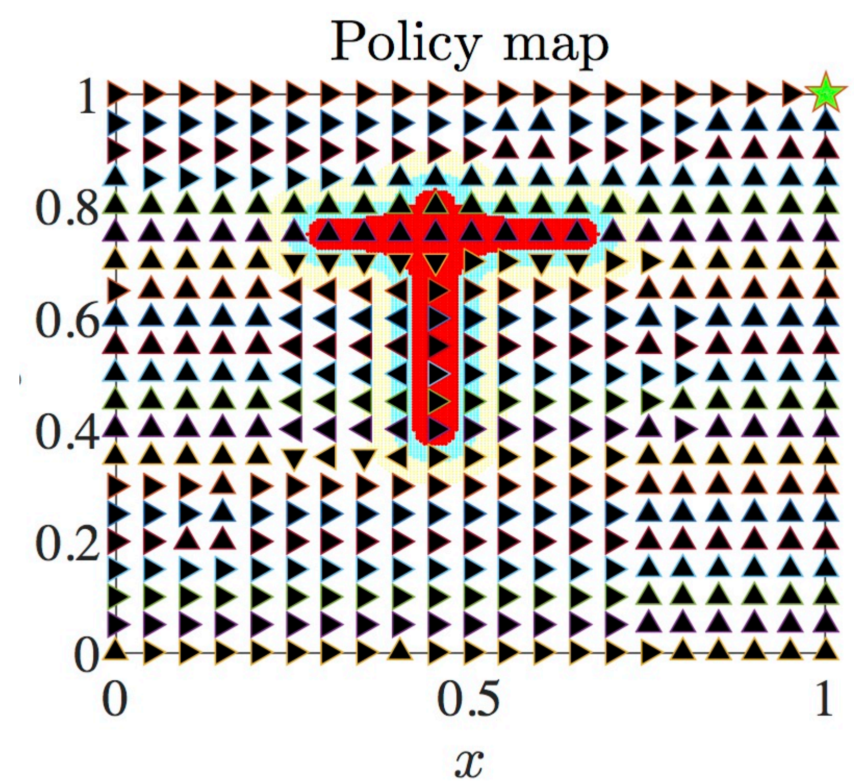
Linear



Regular NN



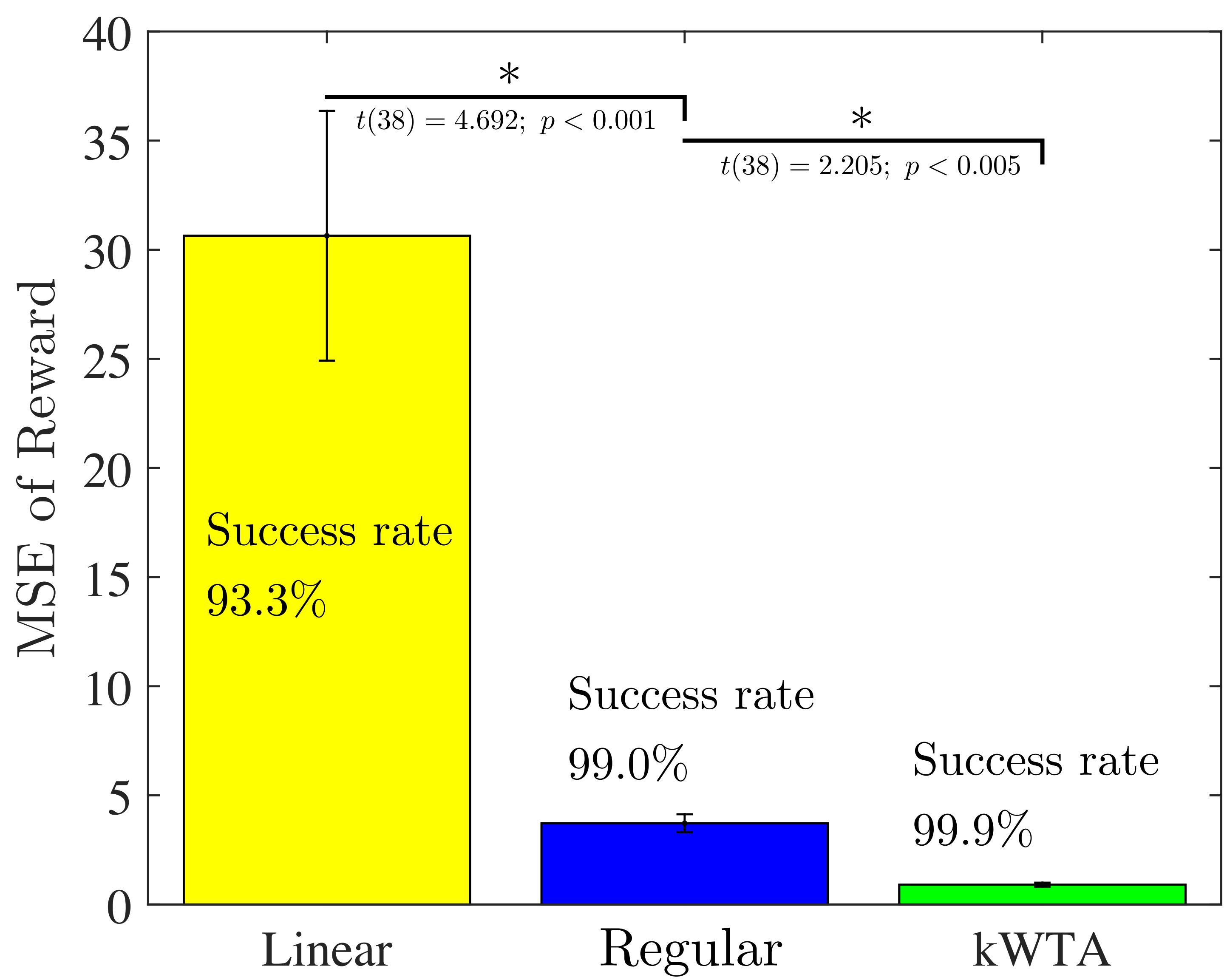
kWTA NN



Jacob Rafati and David C. Noelle (2015). Lateral Inhibition Overcomes Limits of Temporal Difference Learning. In 37th Annual Cognitive Science Society Meeting, Pasadena, CA, USA.

MSE of Reward: Puddle World

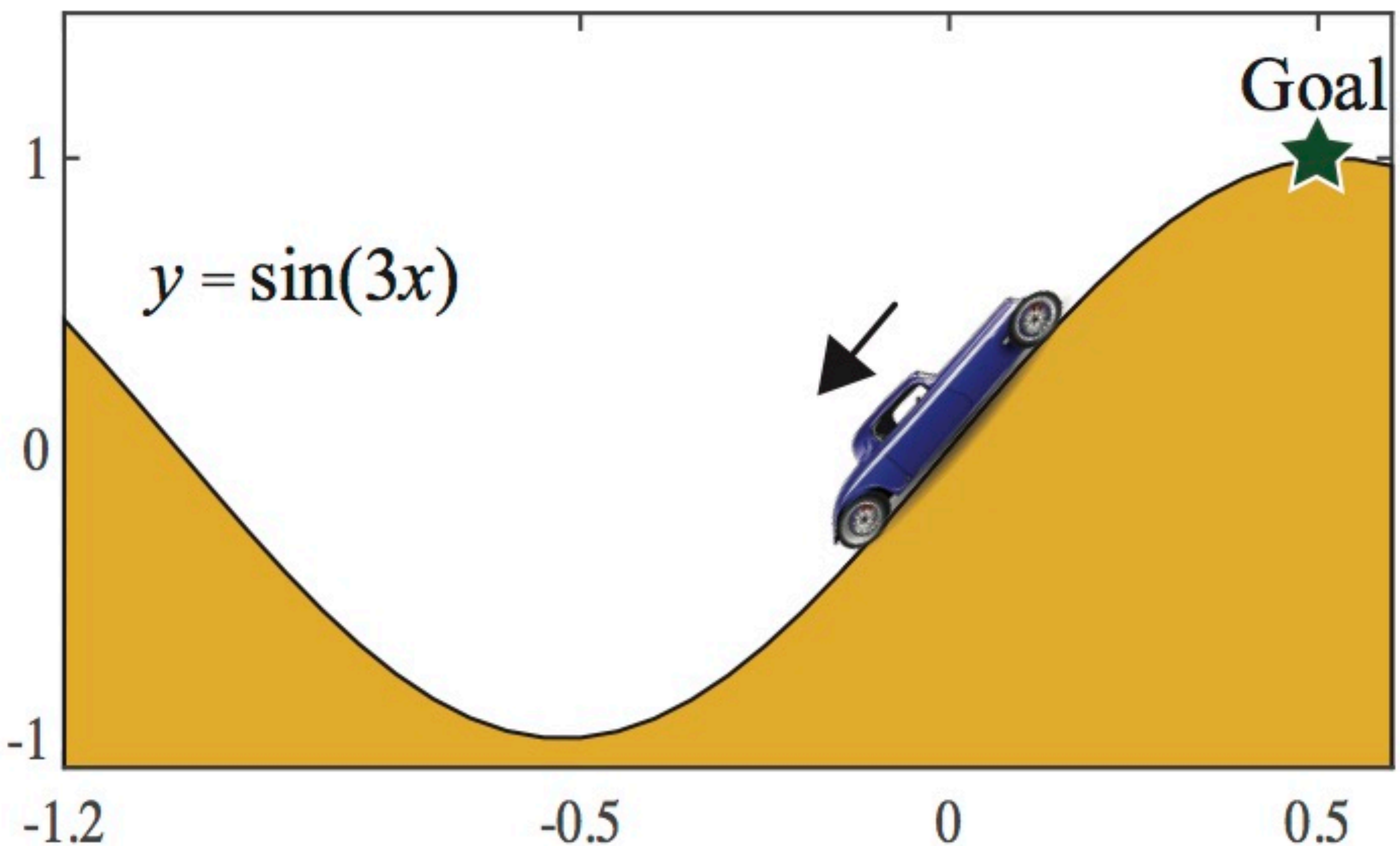
Averaged over 20 simulations of each network type, these columns display the mean squared deviation of accumulated reward from that of optimal performance (Q Table Values). Error bars show one standard error of the mean.



Jacob Rafati and David C. Noelle (2015). Lateral Inhibition Overcomes Limits of Temporal Difference Learning. In 37th Annual Cognitive Science Society Meeting, Pasadena, CA, USA.

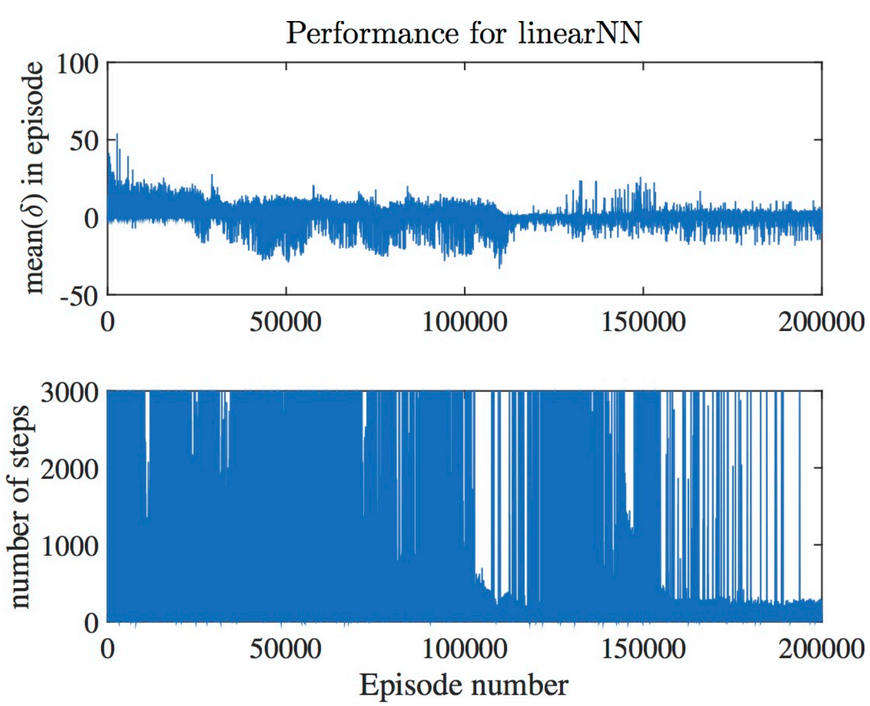
Task: Mountain Car

$$r(s,a) = \begin{cases} 0, & \text{if } s' \text{ is terminal} \\ -1, & otherwise \end{cases}$$

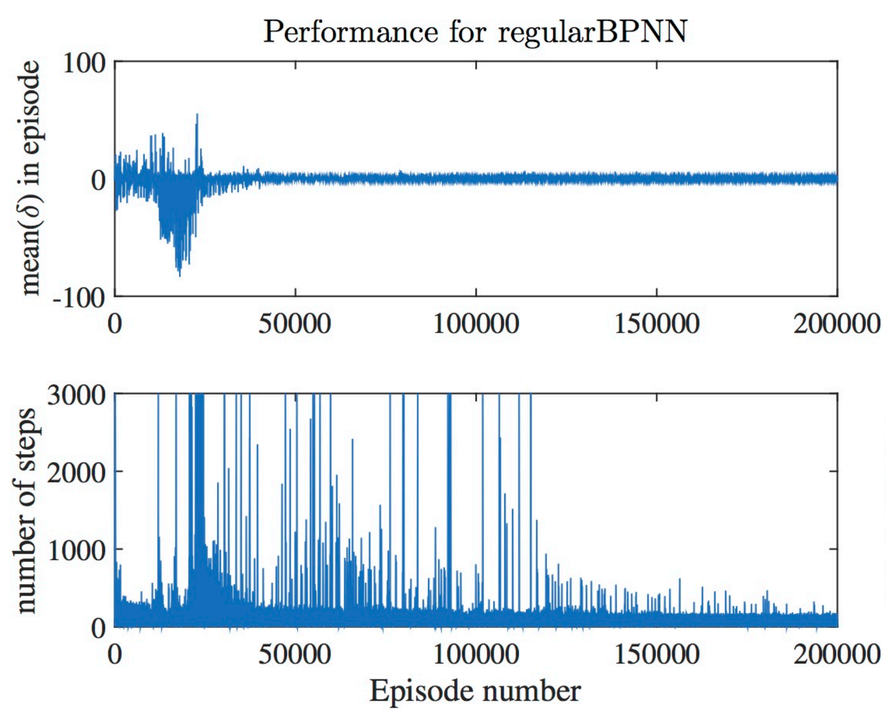


Training performance : Mountain Car

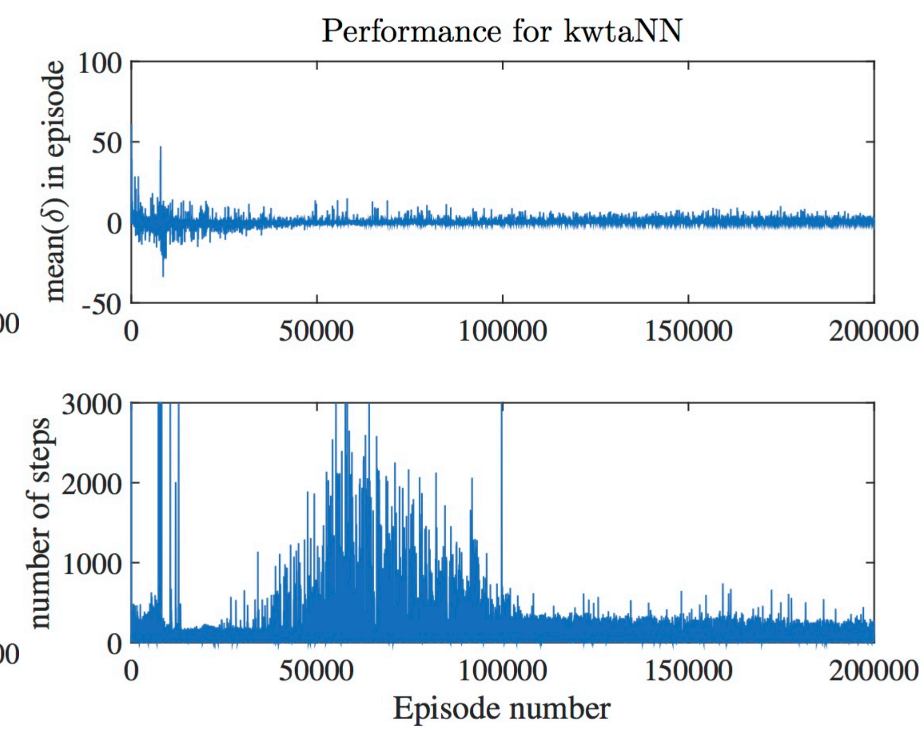
Linear



Regular NN



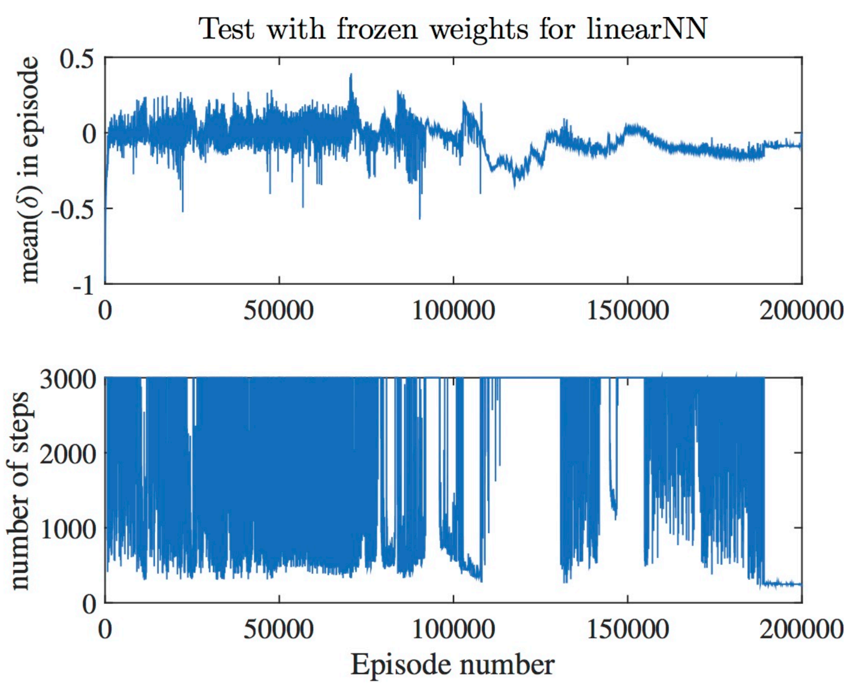
kWTA NN



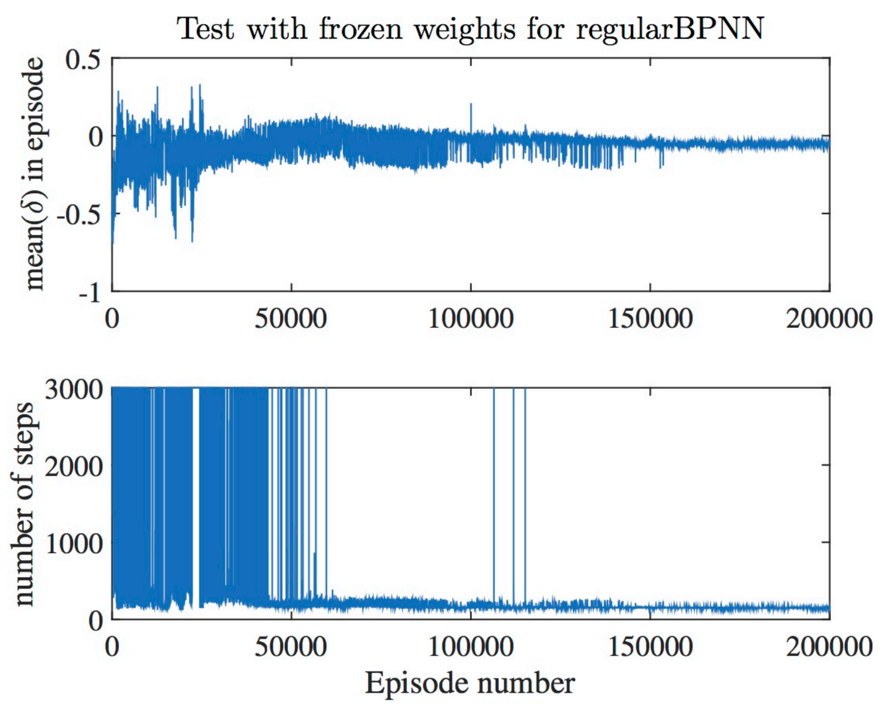
Jacob Rafati and David C. Noelle (2017). Sparse Coding of Learned State Representations in Reinforcement Learning. In Cognitive Computational Neuroscience Conference, New York City, NY.

Test time performance : Mountain Car

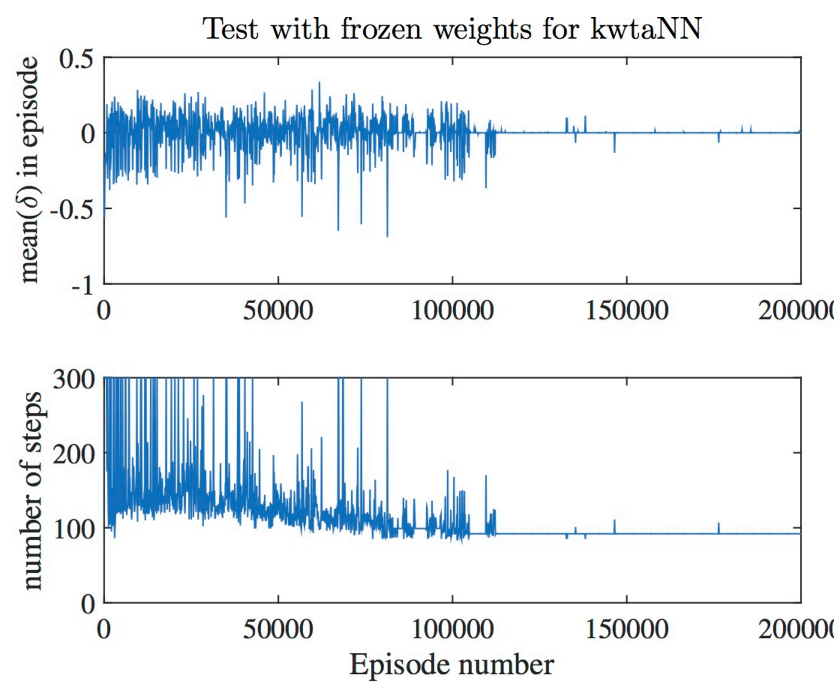
Linear



Regular NN



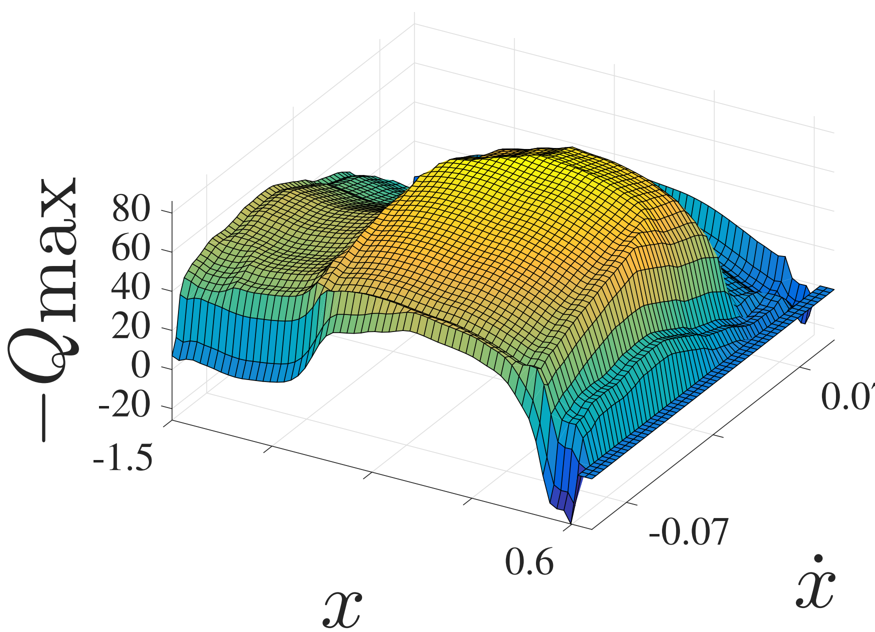
kWTA NN



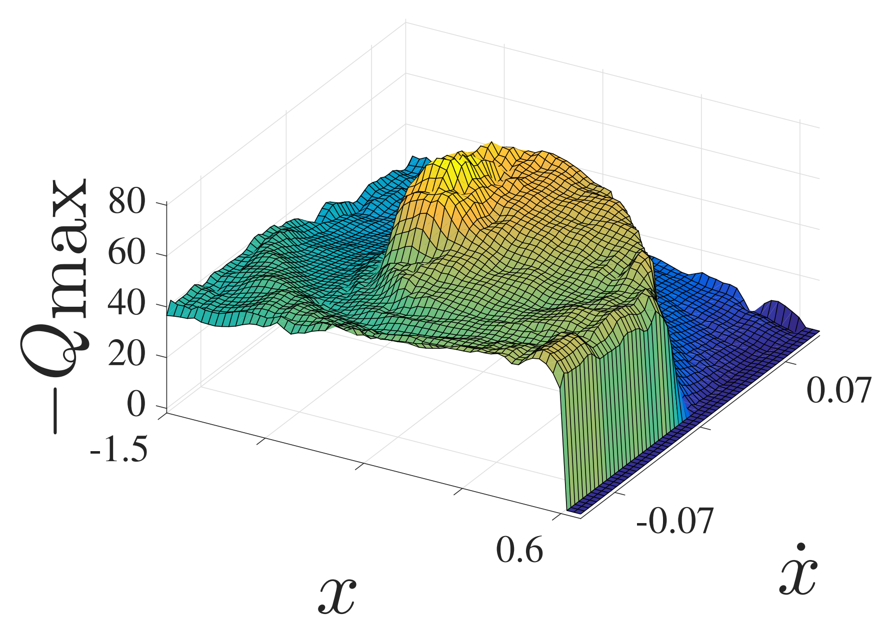
Jacob Rafati and David C. Noelle (2017). Sparse Coding of Learned State Representations in Reinforcement Learning. In Cognitive Computational Neuroscience Conference, New York City, NY.

Value Function : Mountain Car

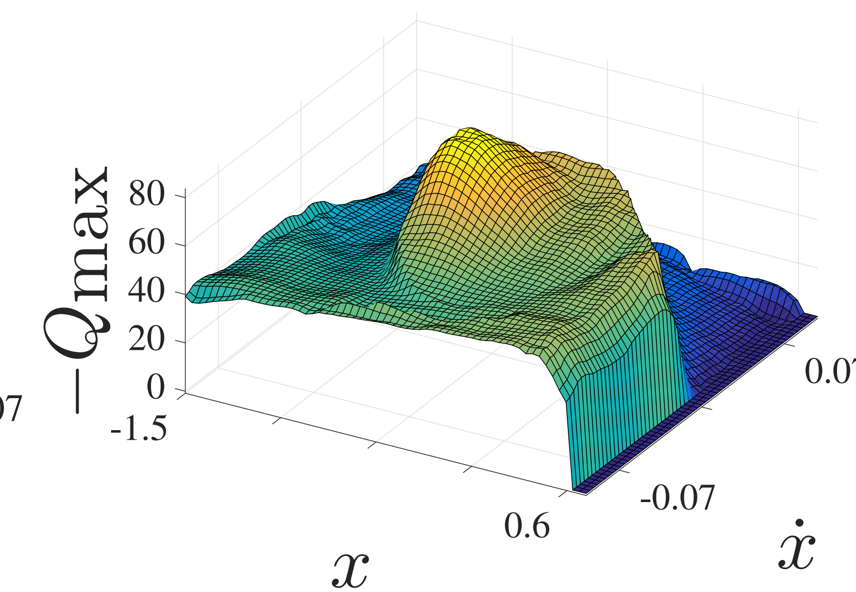
Linear



Regular NN



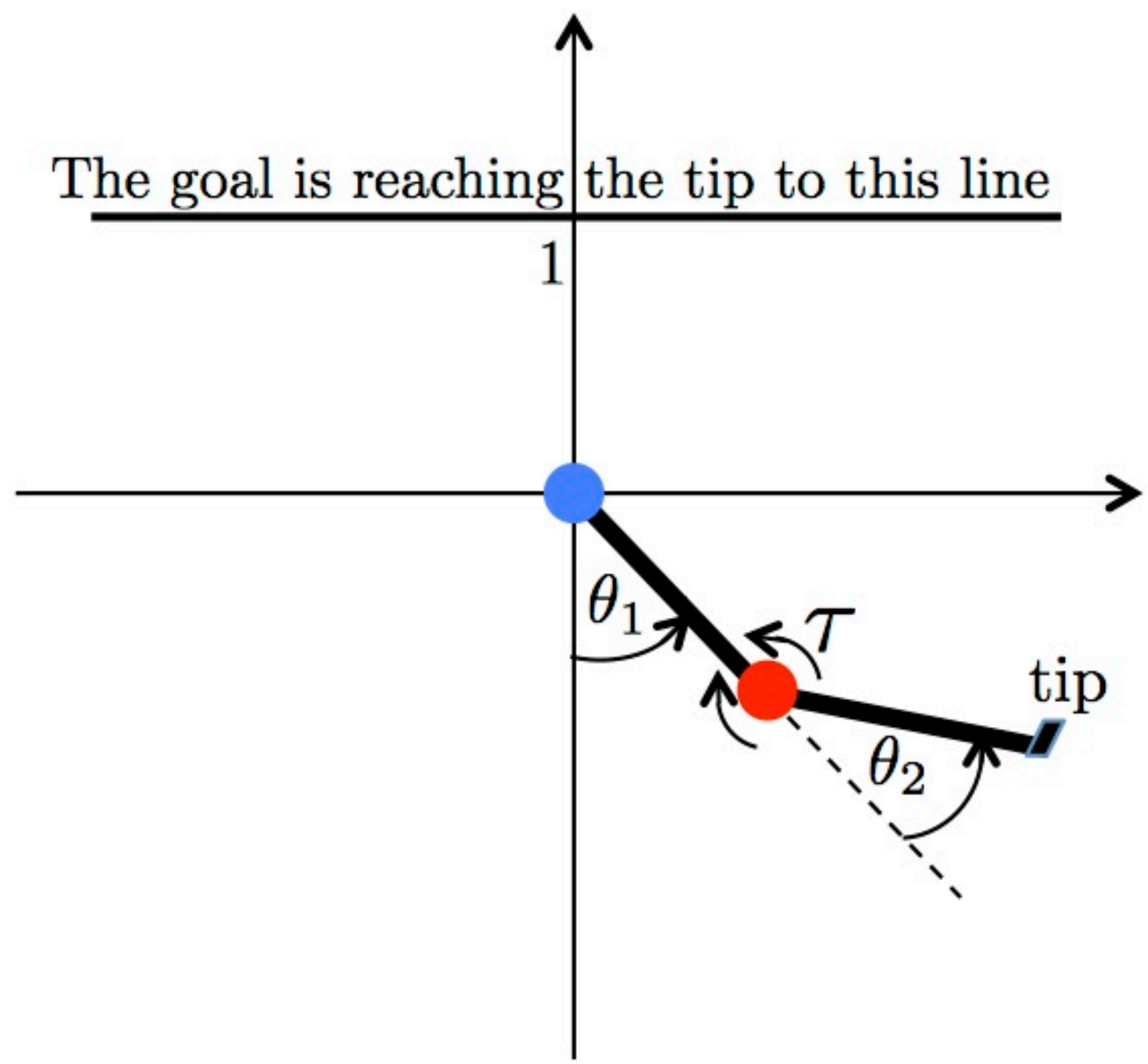
kWTA NN



Jacob Rafati and David C. Noelle (2017). Sparse Coding of Learned State Representations in Reinforcement Learning. In Cognitive Computational Neuroscience Conference, New York City, NY.

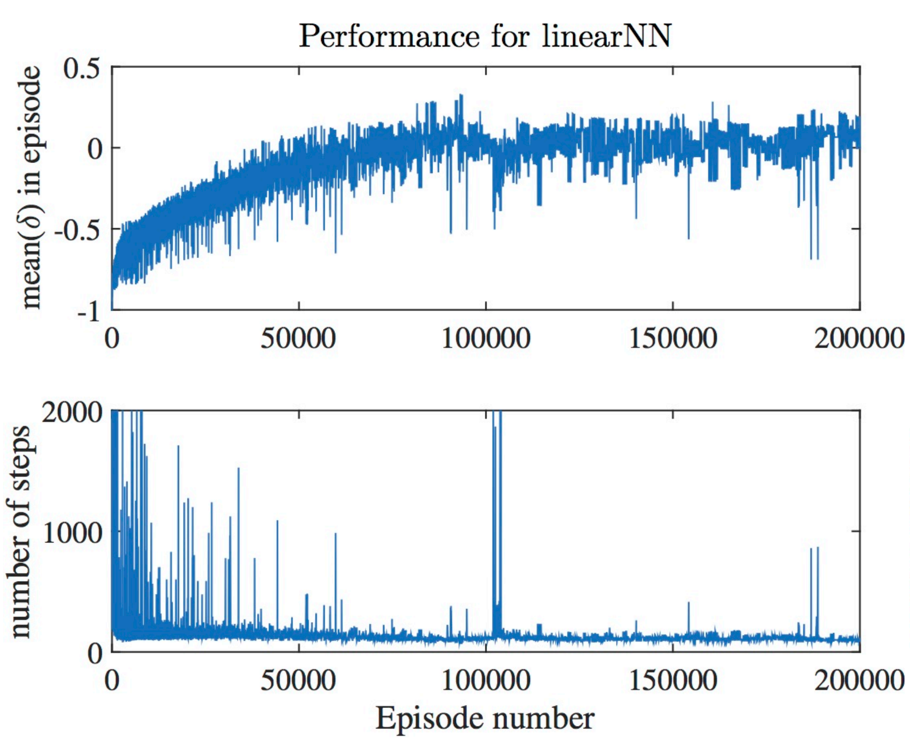
Task: Acrobat Task

$$r(s,a) = \begin{cases} 0, & \text{if } s' \text{ is terminal} \\ -1, & otherwise \end{cases}$$

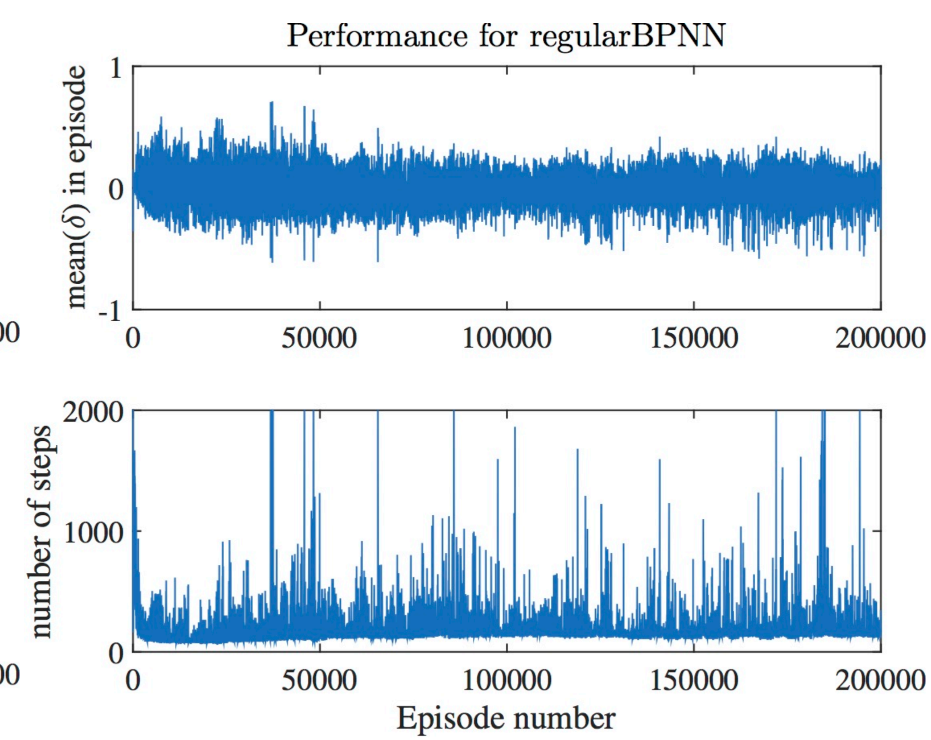


Training performance : Acrobat

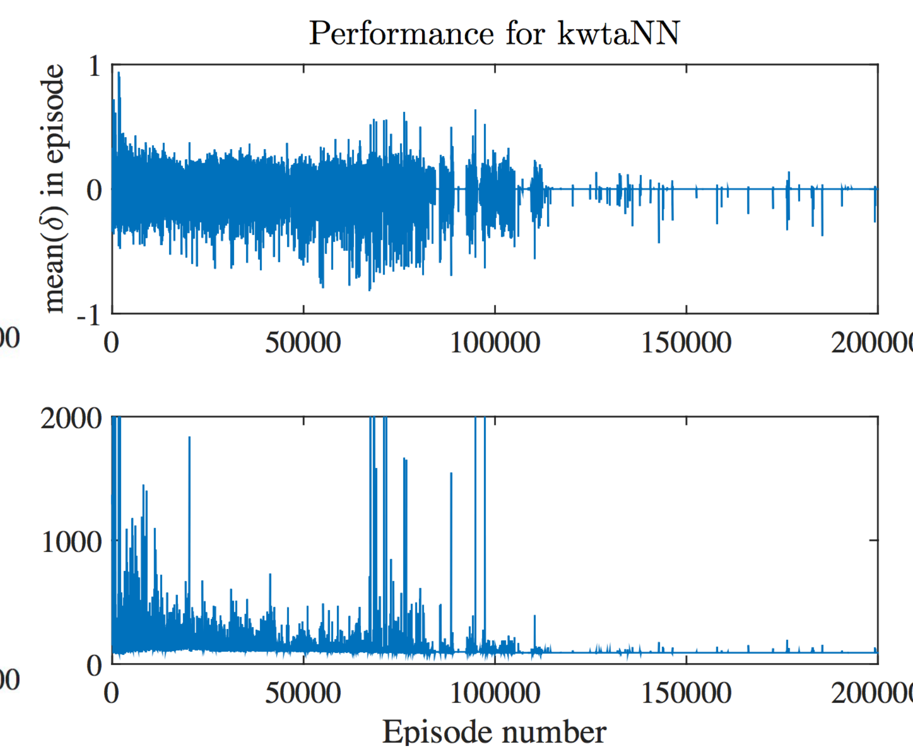
Linear



Regular NN



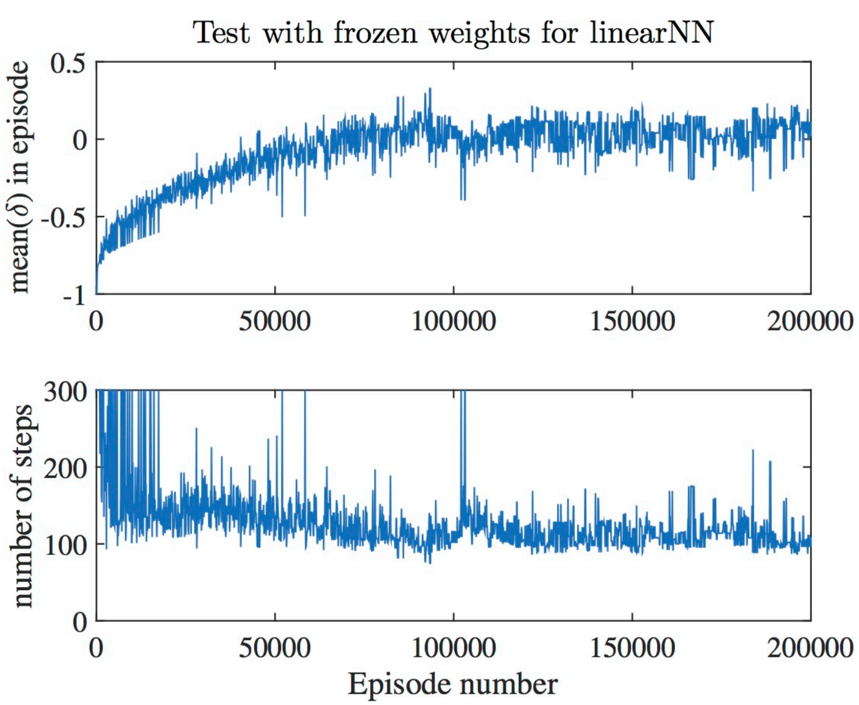
kWTA NN



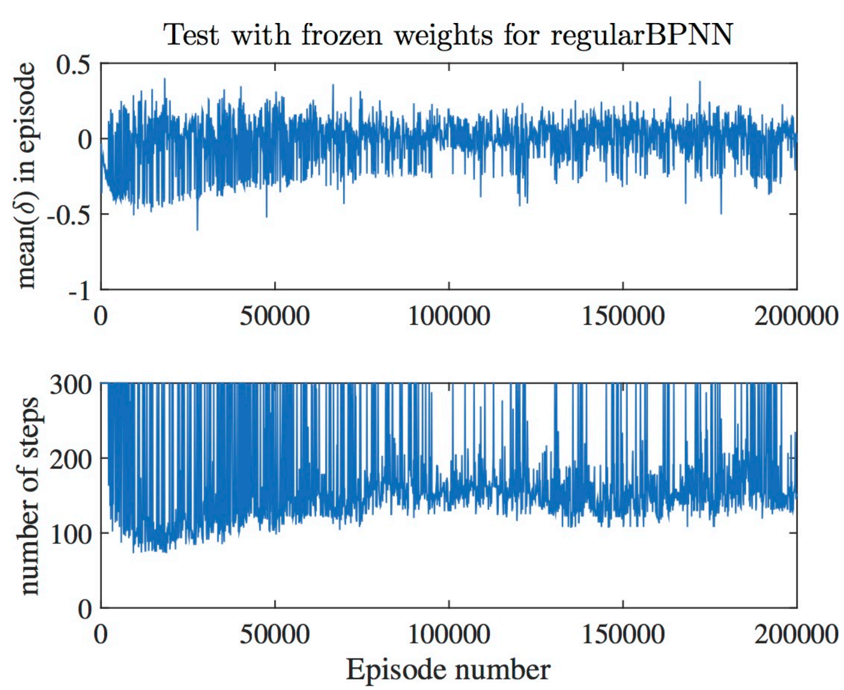
Jacob Rafati and David C. Noelle (2017). Sparse Coding of Learned State Representations in Reinforcement Learning. In Cognitive Computational Neuroscience Conference, New York City, NY.

Test time performance : Acrobat

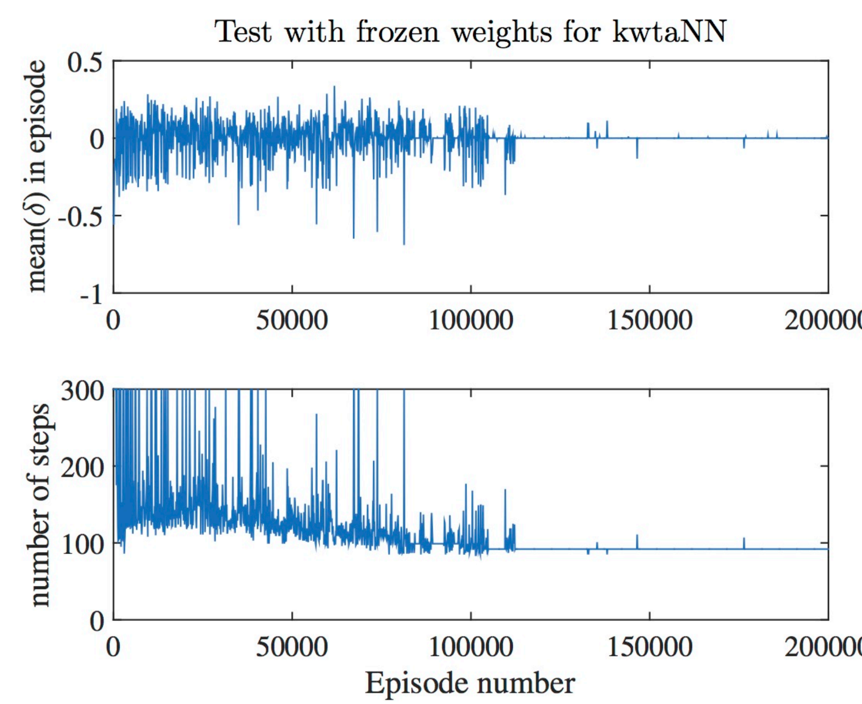
Linear



Regular NN



kWTA NN



Jacob Rafati and David C. Noelle (2017). Sparse Coding of Learned State Representations in Reinforcement Learning. In Cognitive Computational Neuroscience Conference, New York City, NY.

Conclusions

- Inspired by the lateral inhibition that appears in cortical areas, we implemented a state-action value function approximator that utilizes a *k*-Winners-Take-All mechanism.

- The simulation results demonstrate that a mechanism for learning sparse conjunctive codes for the agent's sensory state can help overcome learning problems observed when using TD Learning with a value function approximation.

Future Work

- Using sparse conjunctive representation of the agent's state not only can help in the solving of the simple reinforcement learning tasks, but it might also help improve the learning of some large-scale tasks, too. In the future, we will extend this work to the deep reinforcement learning framework.

- A deep CNN equipped with a *k*-Winners-Take-All mechanism in the fully connected layers can also be used for supervised learning. This is particularly interesting in applications such as image recognition, when two images look similar (they share similar features), but they belong to different classes.

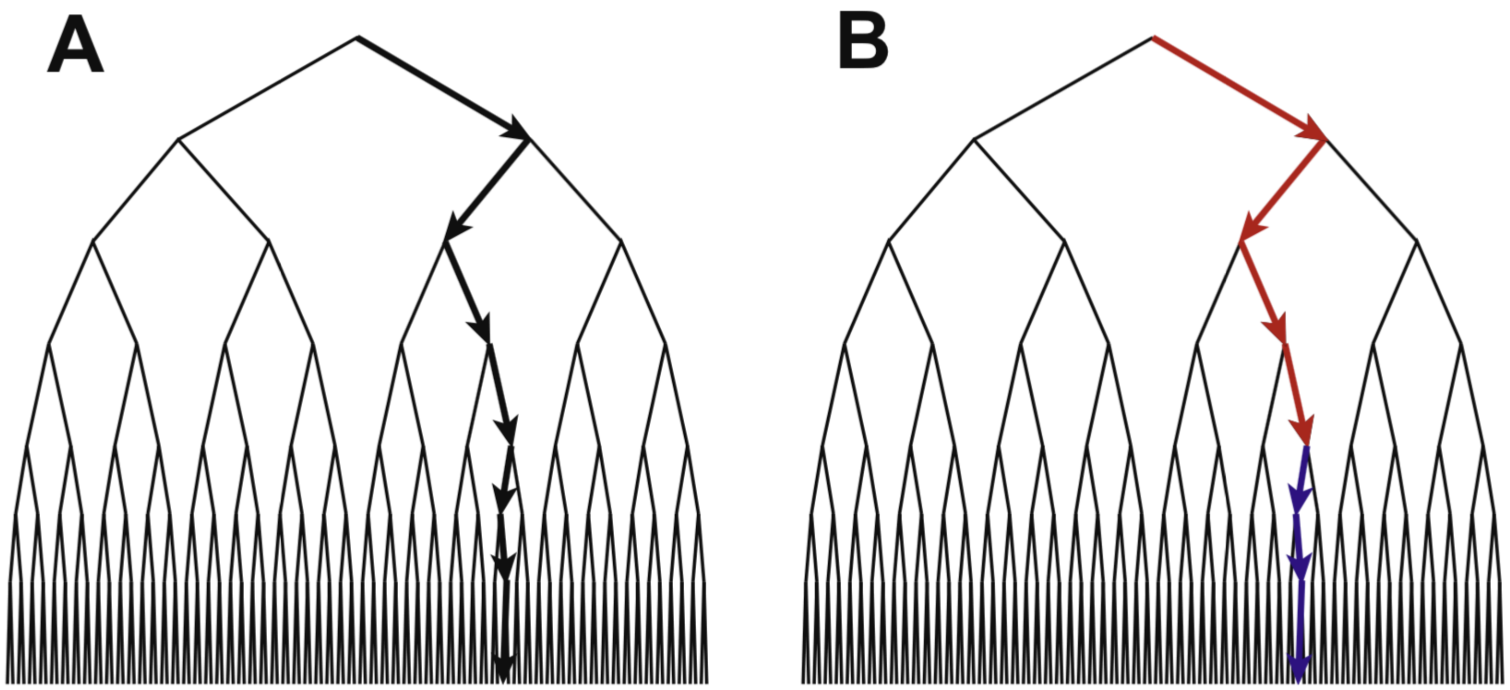
Publications

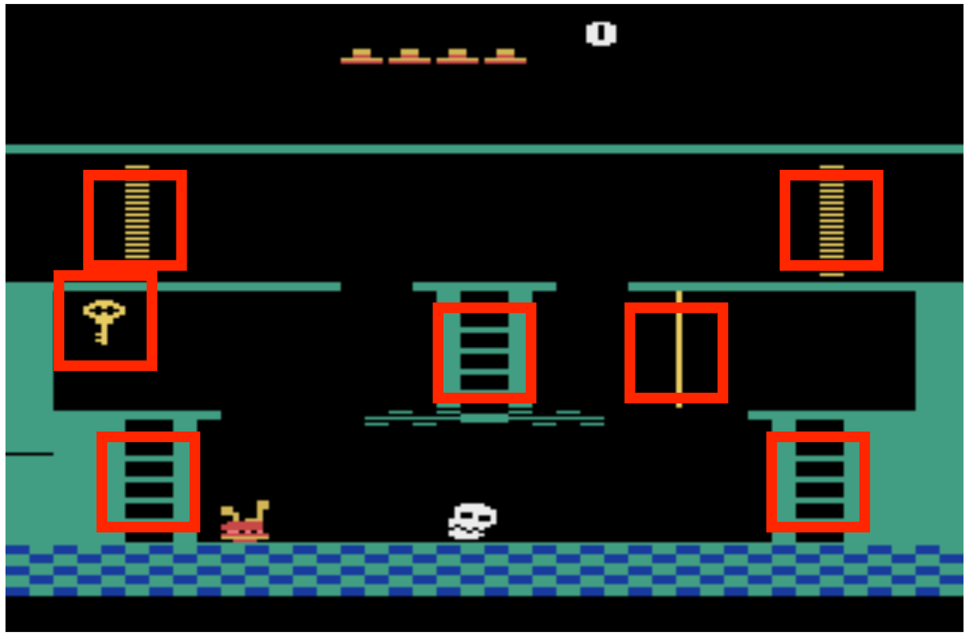
- Jacob Rafati and David C. Noelle (2015). Lateral Inhibition Overcomes Limits of Temporal Difference Learning. In 37th Annual Cognitive Science Society Meeting, Pasadena, CA, USA.

- Jacob Rafati and David C. Noelle (2017). Sparse Coding of Learned State Representations in Reinforcement Learning. In Cognitive Computational Neuroscience Conference, New York City, NY.

Problem 2.
Learning Representations in Model-Free Hierarchical Reinforcement Learning

Sparse Feedback & Scalability

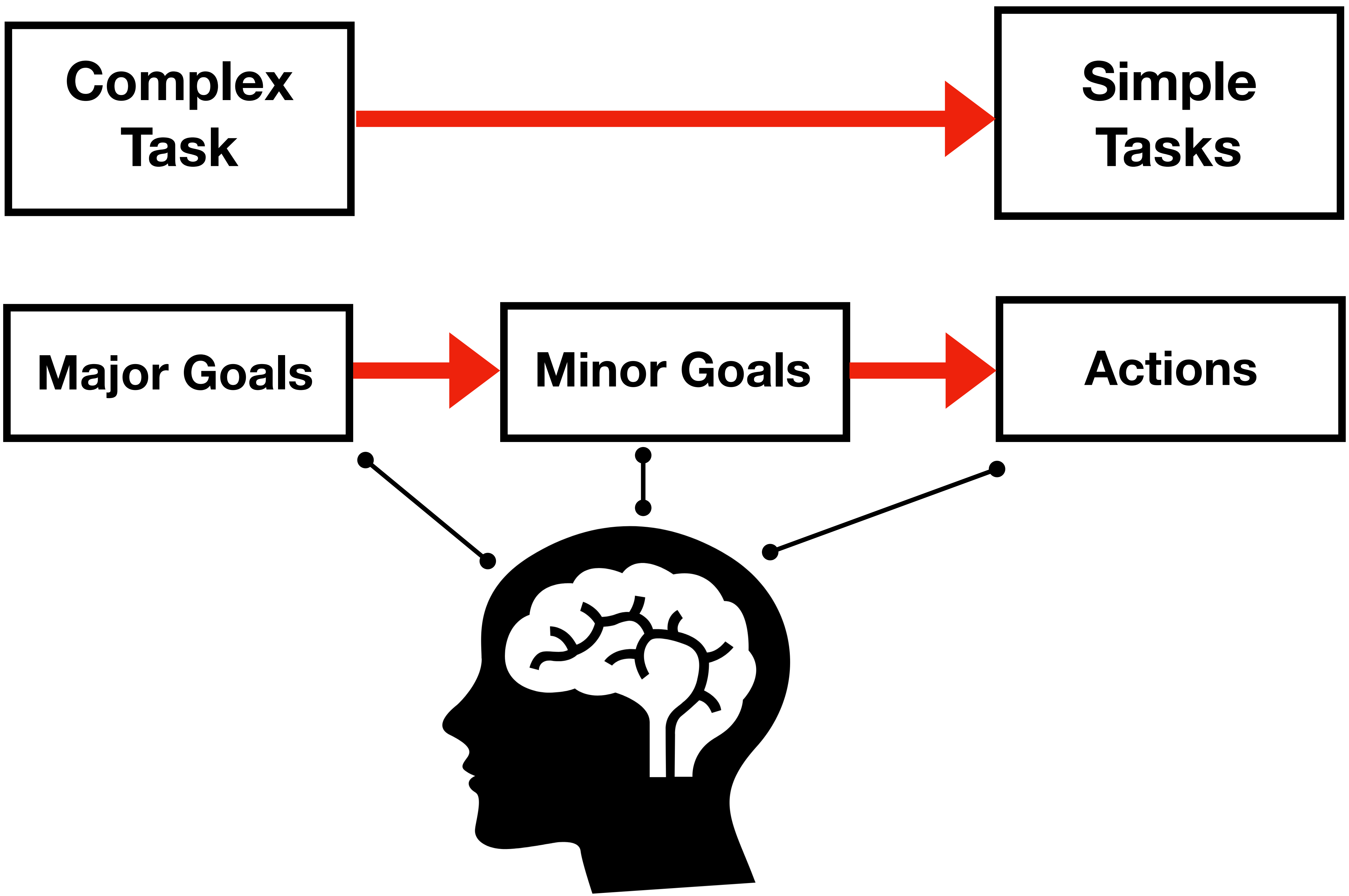




Subgoals

Botvinick et al. (2009). Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective. Cognition, 113(3).

Hierarchy in Human Behavior & Brain Structure



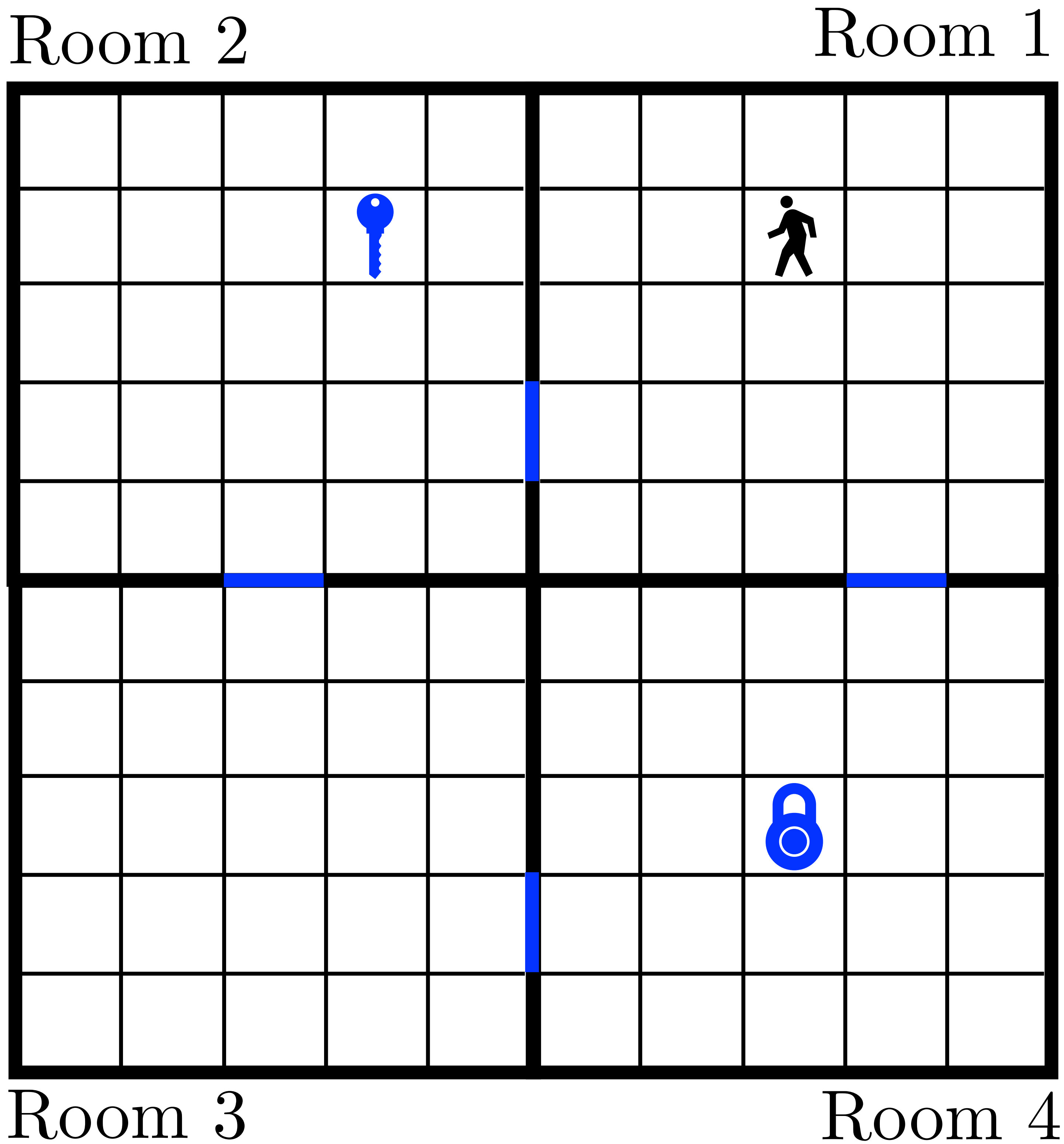
Problem 2:

Learning Representations in model-free HRL

Objective:

1. Learning to operate over different levels of *temporal abstraction*.
2. Efficiently exploring the state-space while learning reusable skills through *intrinsic motivation*.
3. Automatic Subgoal Discovery in large-scale tasks with sparse delayed feedback within model-free HRL framework.
4. Learning Representations in a unified framework.

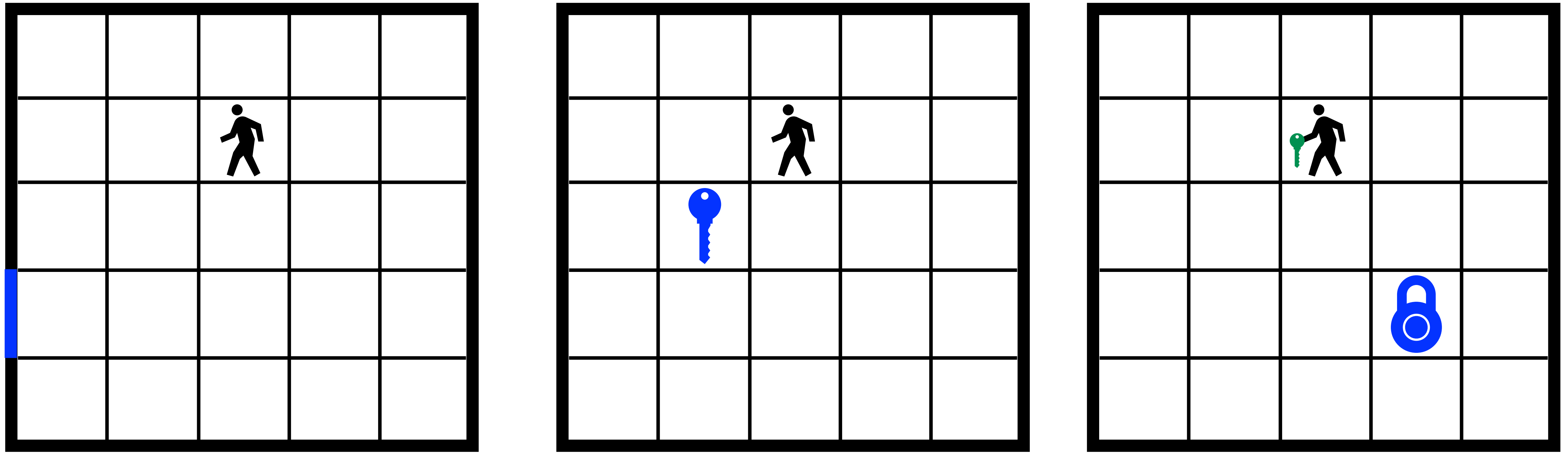
4 Rooms Task



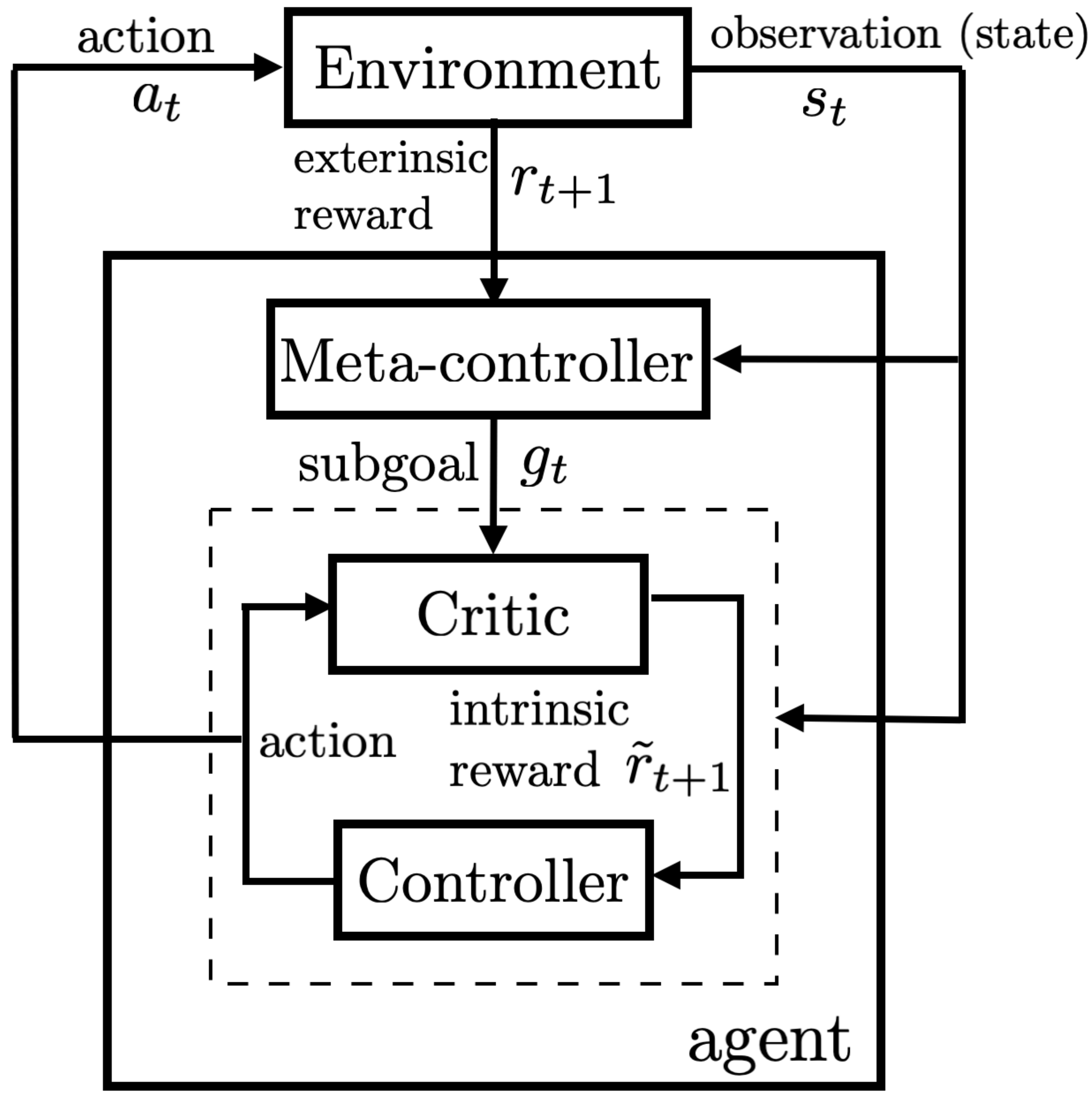
Hierarchical Reinforcement Learning Subproblems

- **Subproblem 1:** Learning a meta-policy to choose a proper subgoal.
- **Subproblem 2:** Developing skills through intrinsic motivation learning.
- **Subproblem 3:** Automatic subgoal discovery.

Developing skills through Intrinsic Motivation

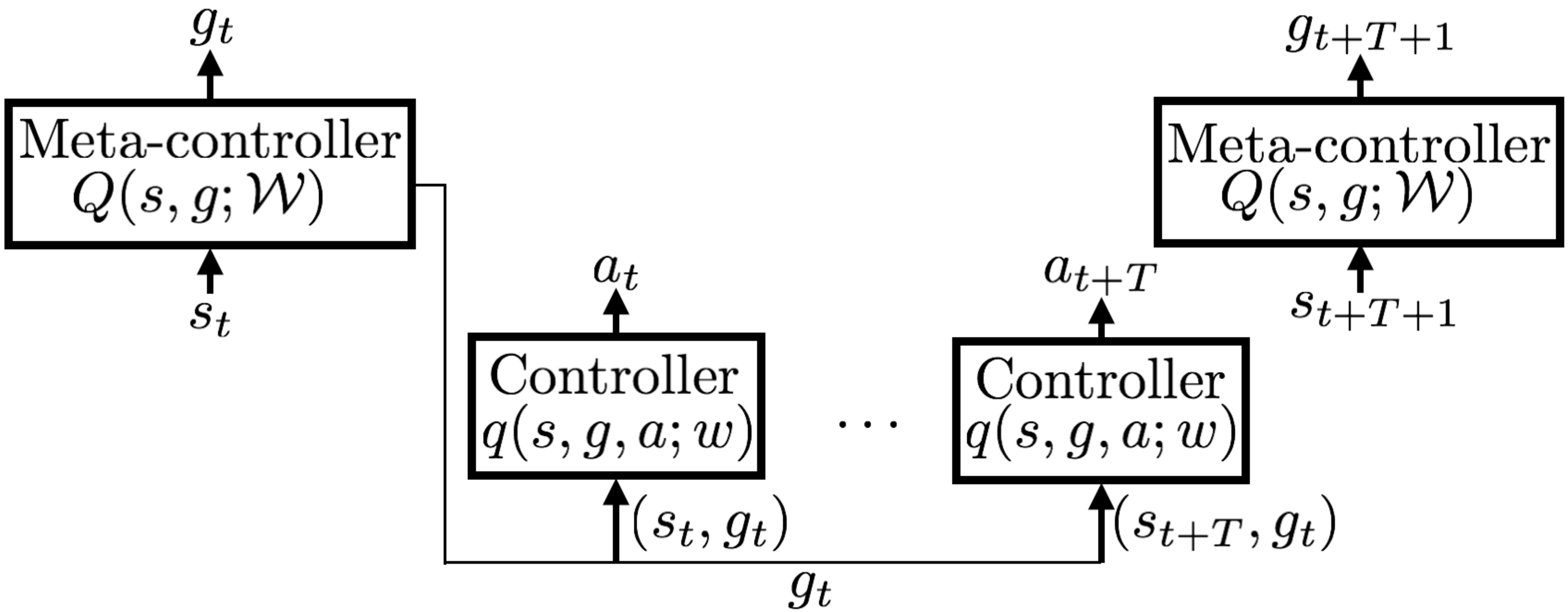


Meta-controller/Controller Framework



Kulkarni et al. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. NeurIPS.

Subproblems 1 and 2: Integration of Temporal Abstraction and Intrinsic Motivation Learning



Kulkarni et al. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. NeurIPS.

Meta-controller/Controller Framework

Meta-controller's loss function

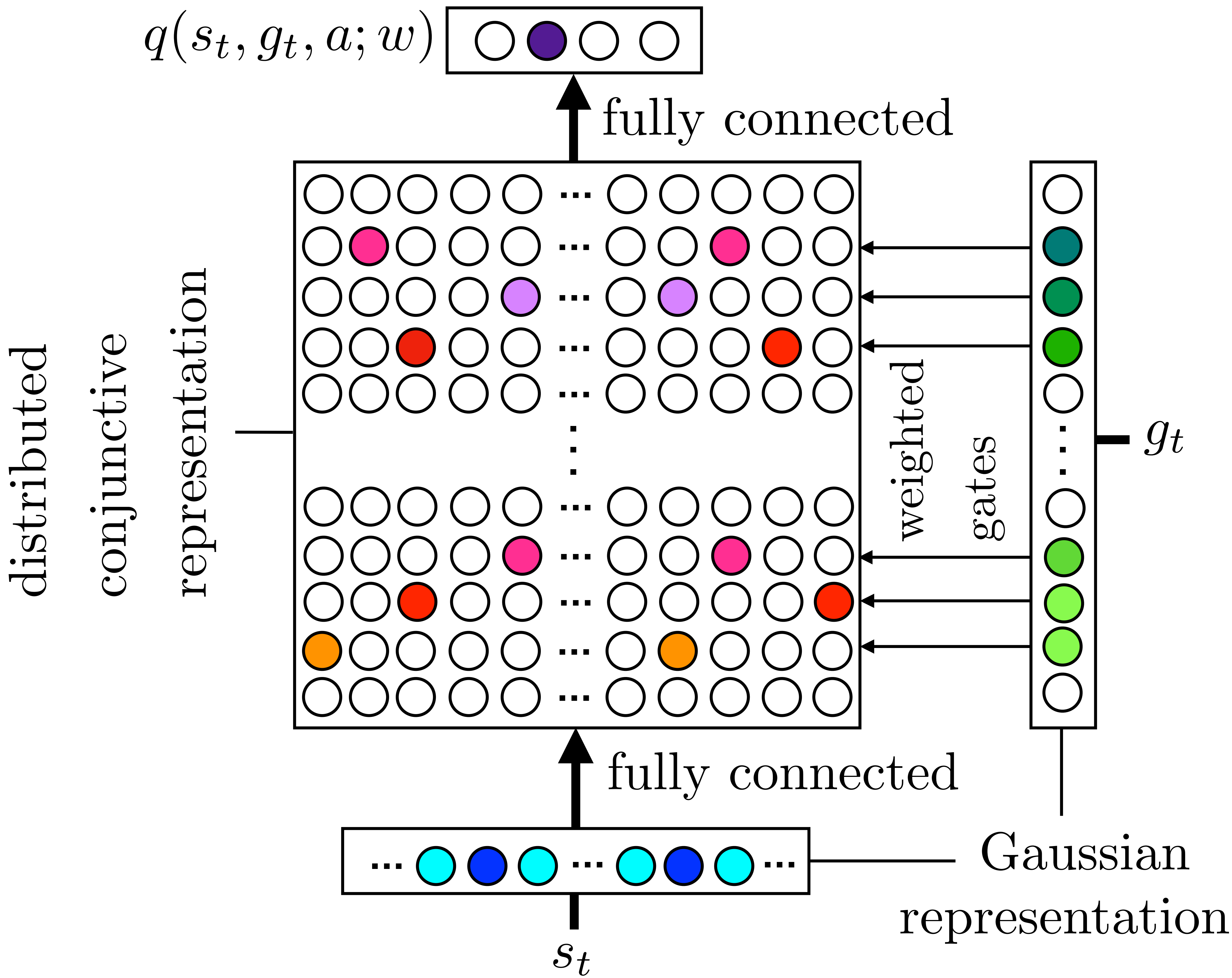
$$\mathcal{L}(\mathcal{W}) \triangleq \mathbb{E}_{(s,g,G,s_{t'})\sim\mathcal{D}_2}\left[\left(G + \gamma \max_{g'} Q(s_{t'}, g'; \mathcal{W}) - Q(s, g; \mathcal{W})\right)^2\right]$$

Controller's loss function

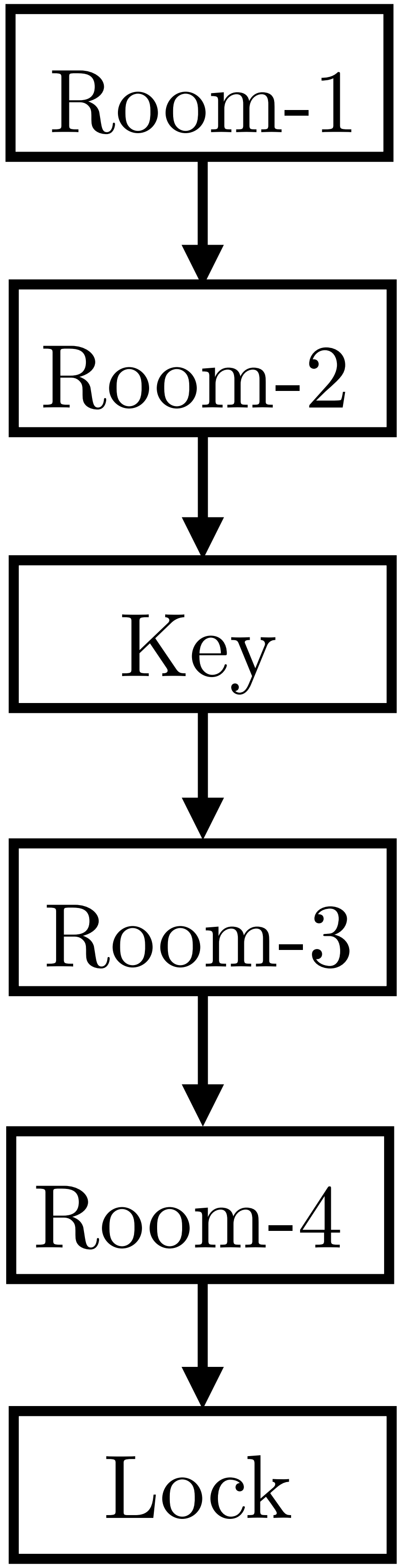
$$L(w) \triangleq \mathbb{E}_{(s,g,a,\tilde{r},s')\sim\mathcal{D}_1}\left[\left(\tilde{r} + \gamma \max_{a'} q(s', g, a'; w) - q(s, g, a; w)\right)^2\right]$$

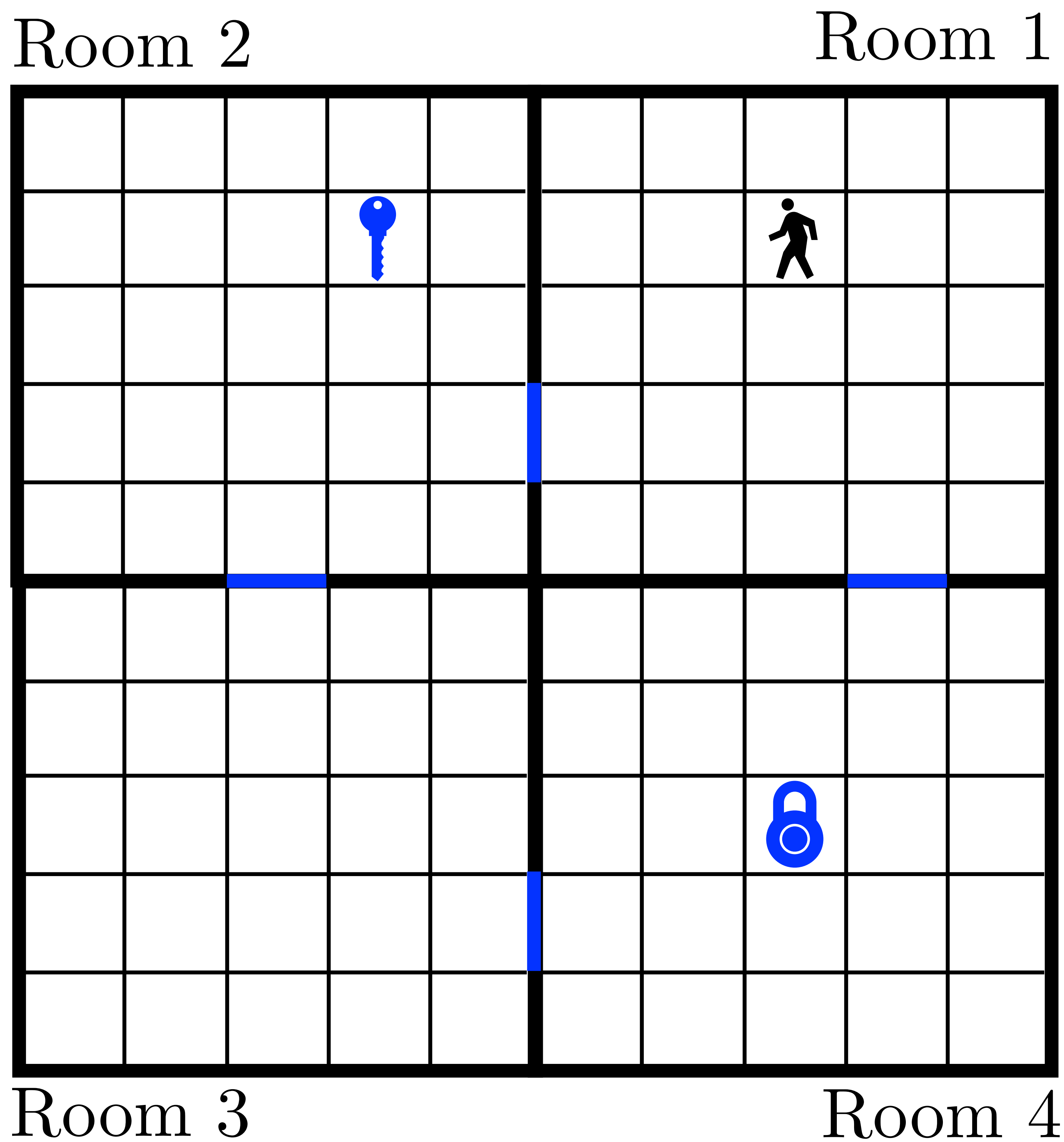
Kulkarni et al. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. NeurIPS.

State-Goal Q Function

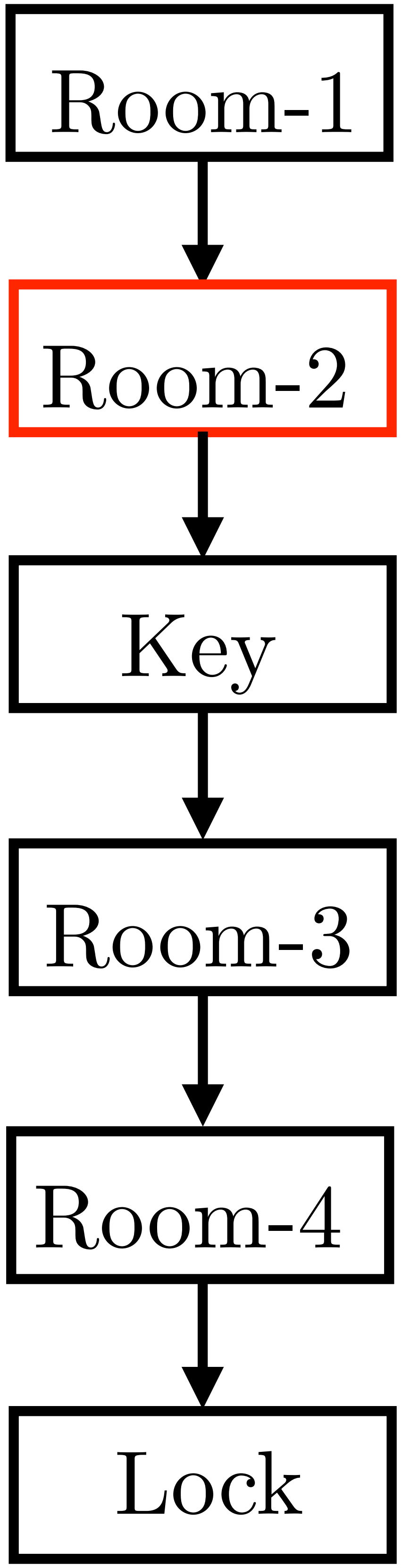


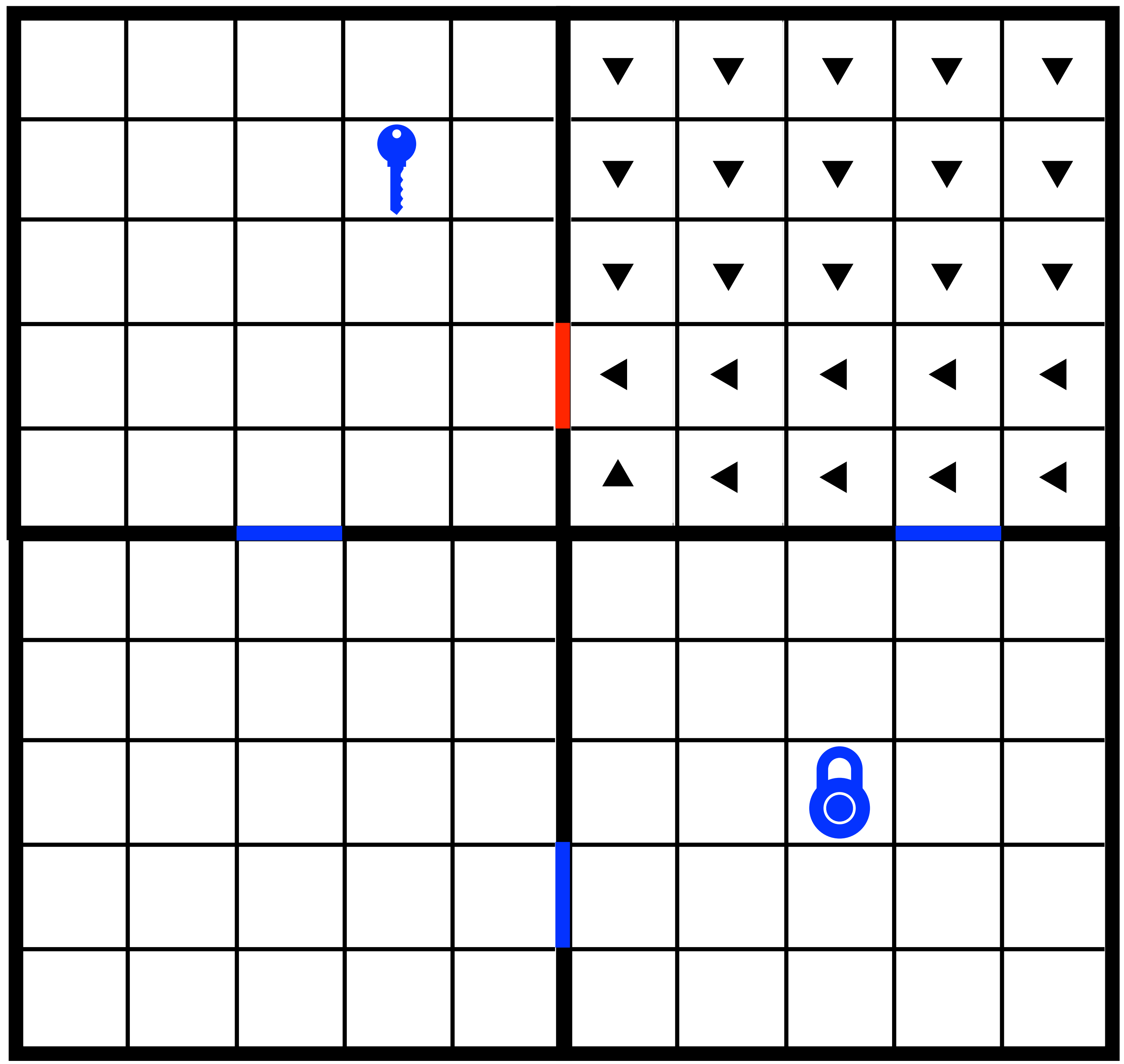
Reusing the skills



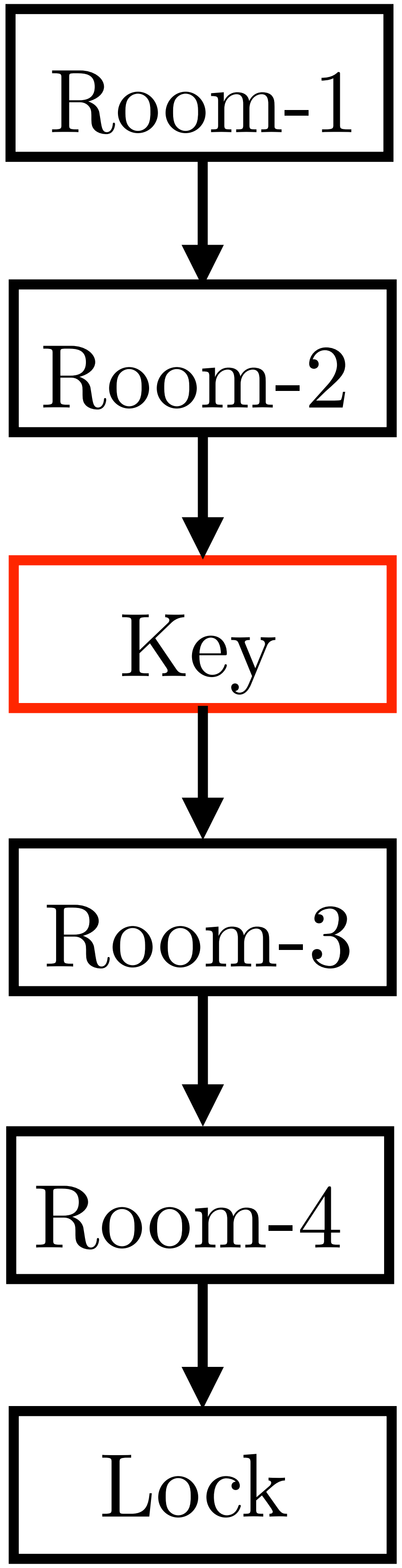


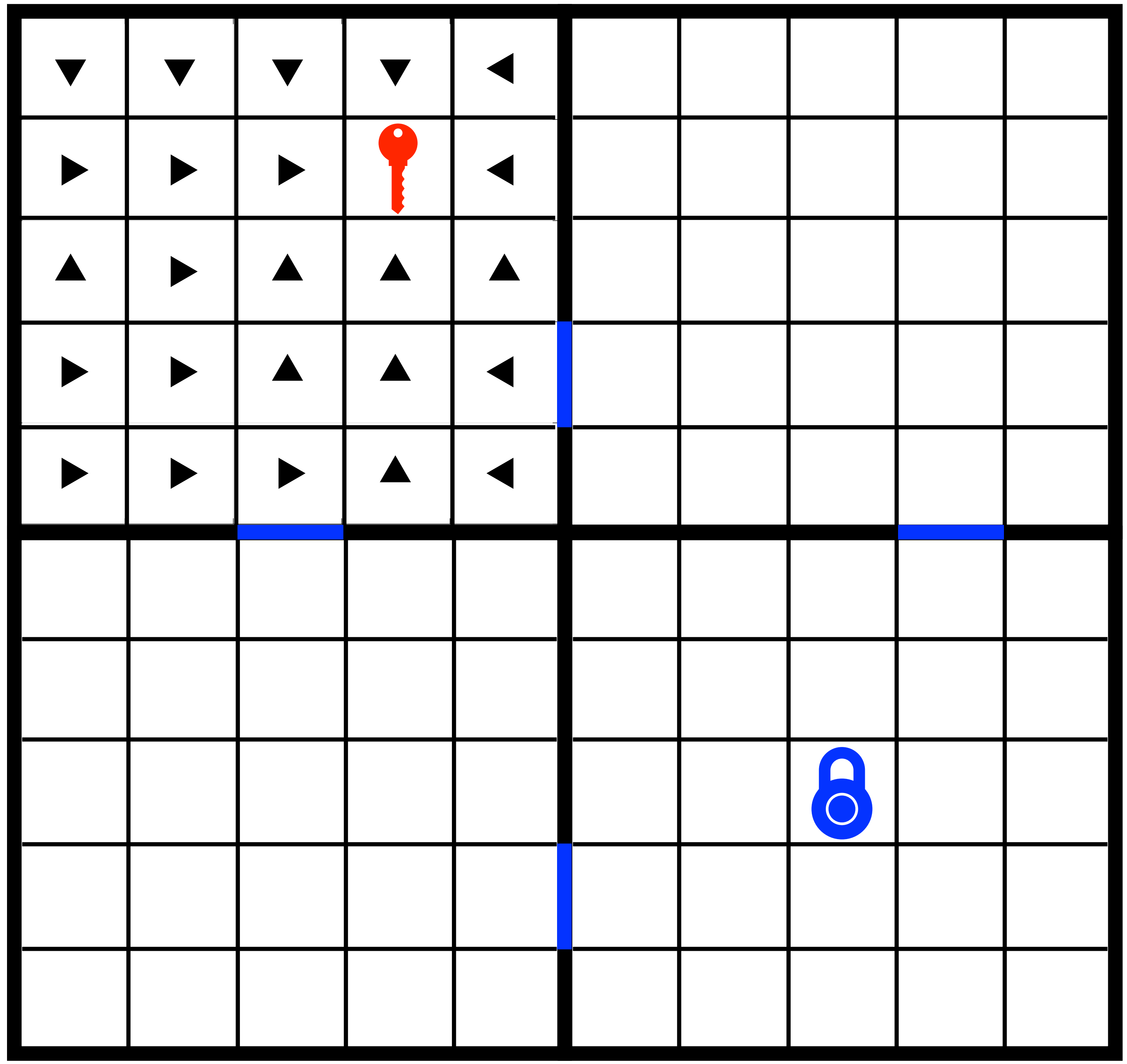
Reusing the skills



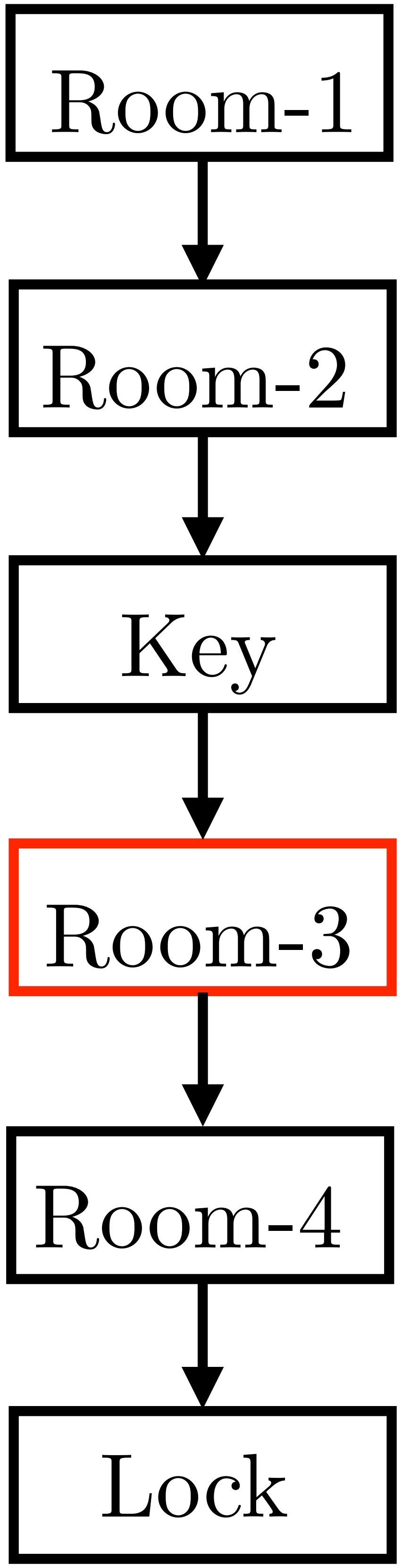


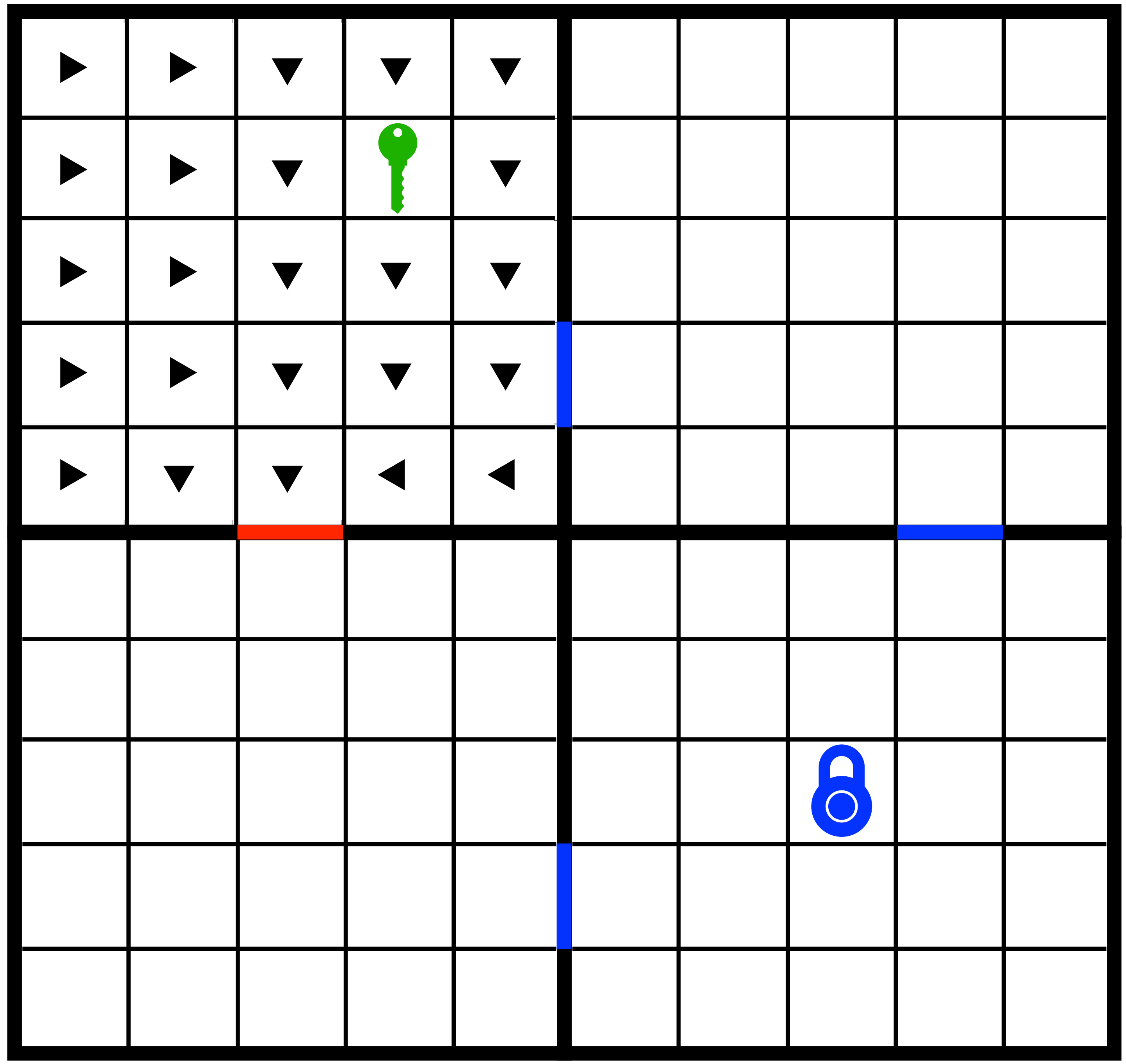
Reusing the skills



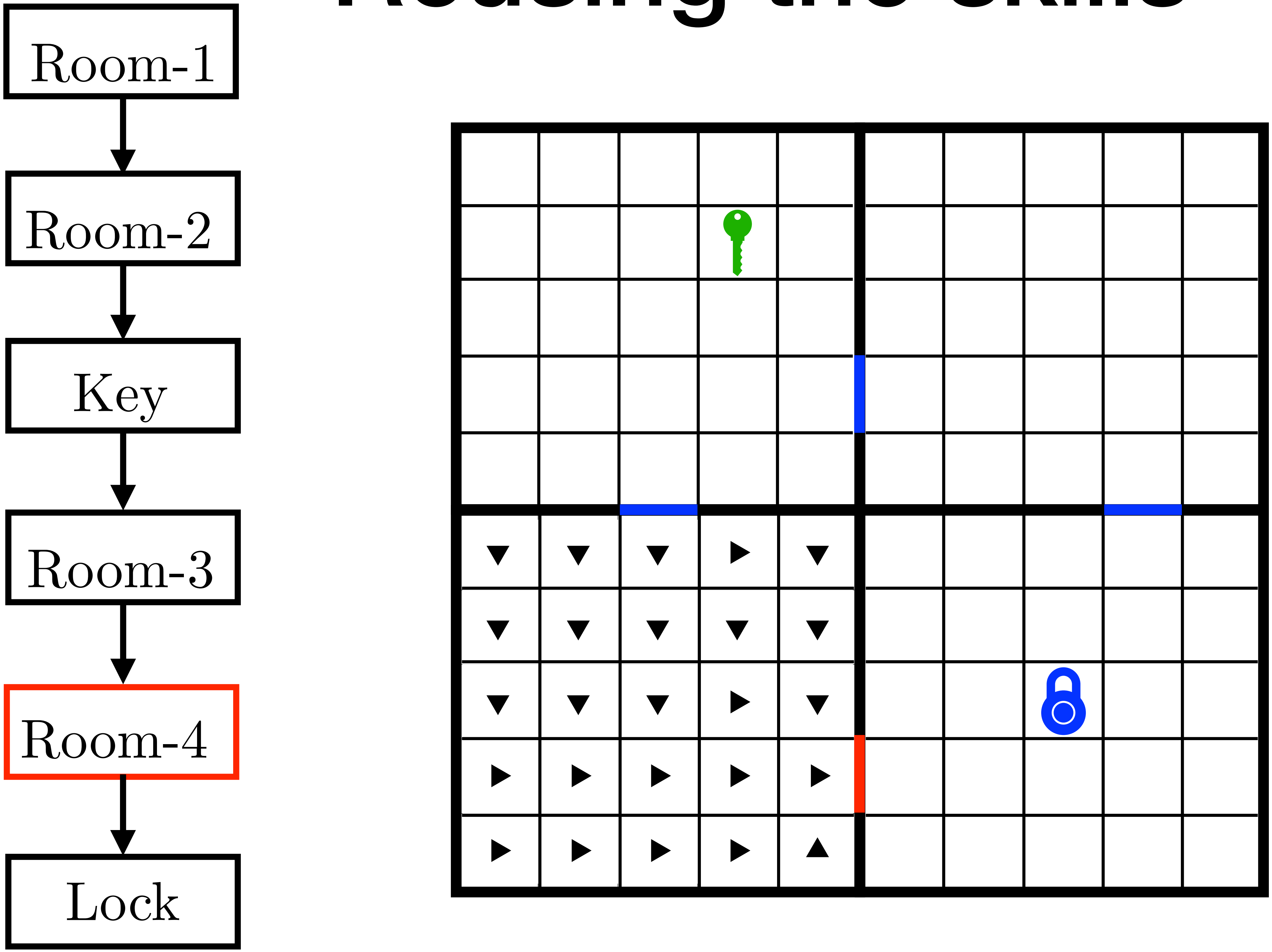


Reusing the skills

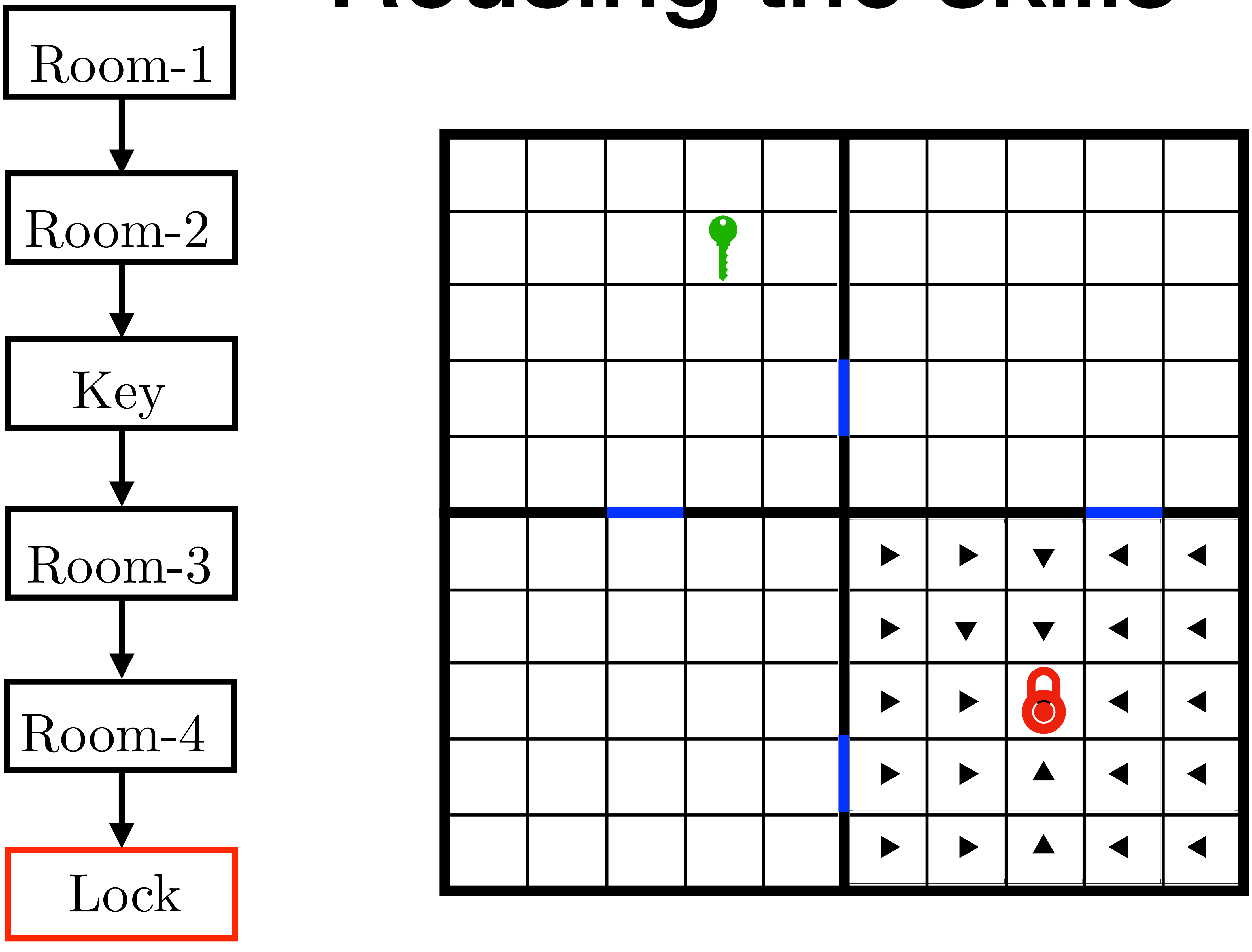




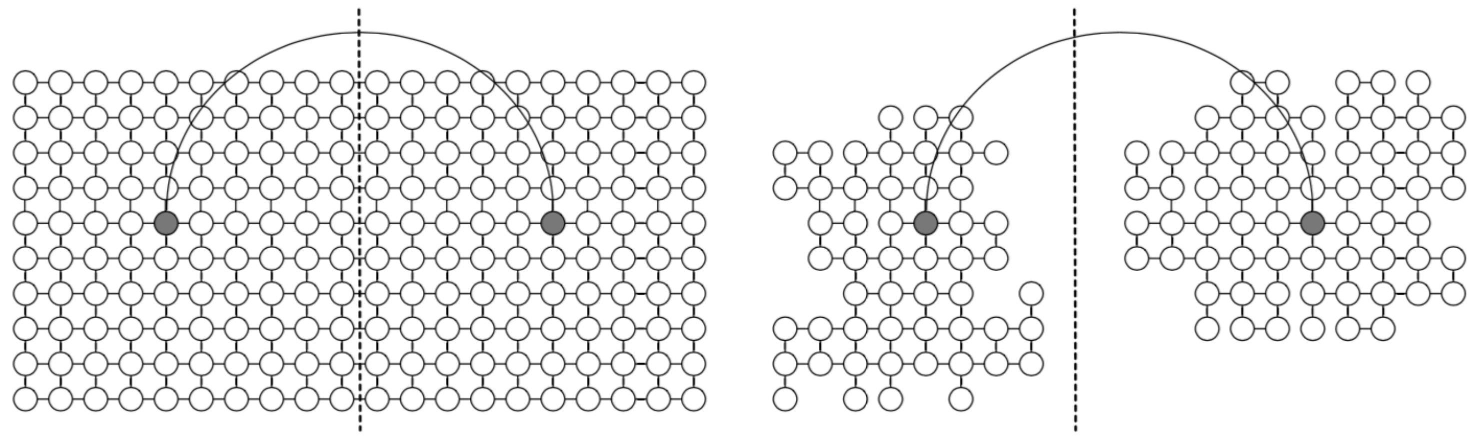
Reusing the skills



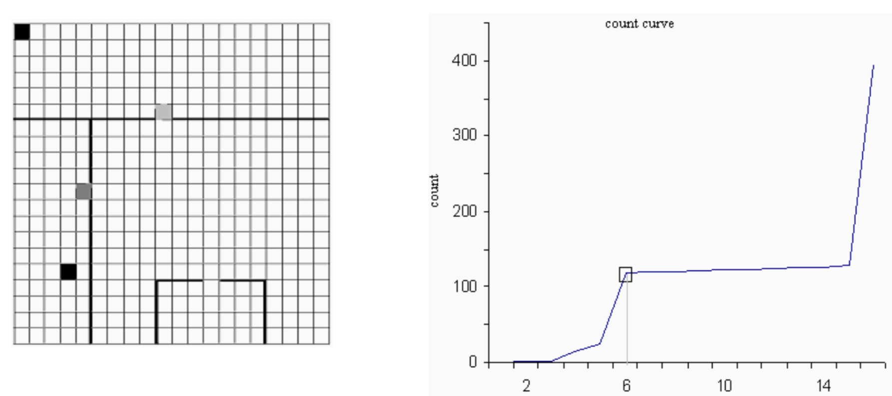
Reusing the skills



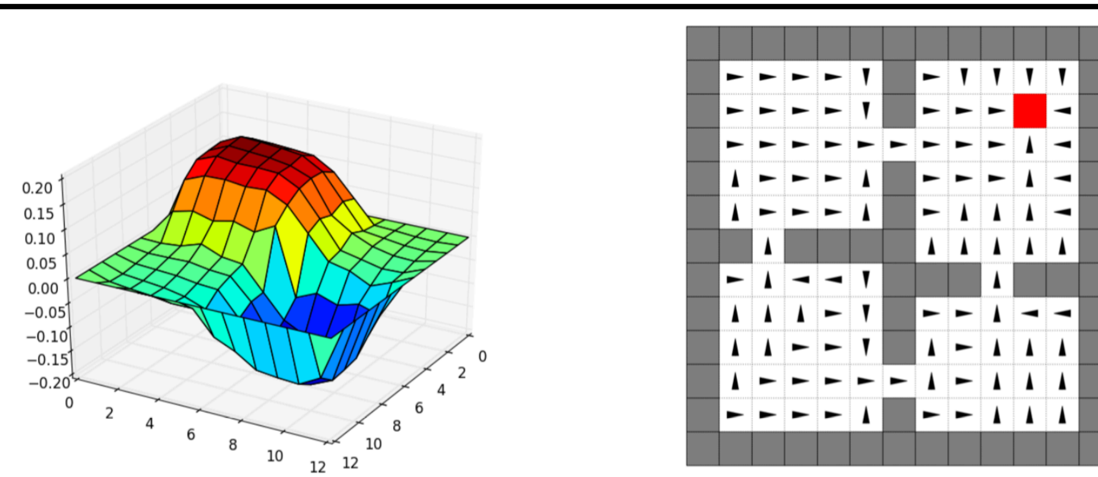
Subproblem 3. Subgoal Discovery



Simsek et al. (2005). Identifying useful subgoals in reinforcement learning by local graph partitioning. ICML.



Goel, S. and Huber, M. (2003). Subgoal discovery for hierarchical reinforcement learning using learned policies. FLAIRS.



Machado et al. (2017). A Laplacian Frame- work for Option Discovery in Reinforcement Learning. ICML.

Subproblem 3. Subgoal Discovery

- Discovering promising states to pursue, i.e. finding subgoals set.
- Implementing subgoal discovery algorithm for large-scale model free RL problem, i.e. without access to environment models (e.g. state-transition probabilities, reward function).
- Unsupervised learning on the limited past experience memory collected during intrinsic motivation learning.

Subproblem 3. Candidate Subgoals

- It is close to a rewarding state.
- It represents a set of states, at least some of which tend to be along a state transition path to a rewarding state.

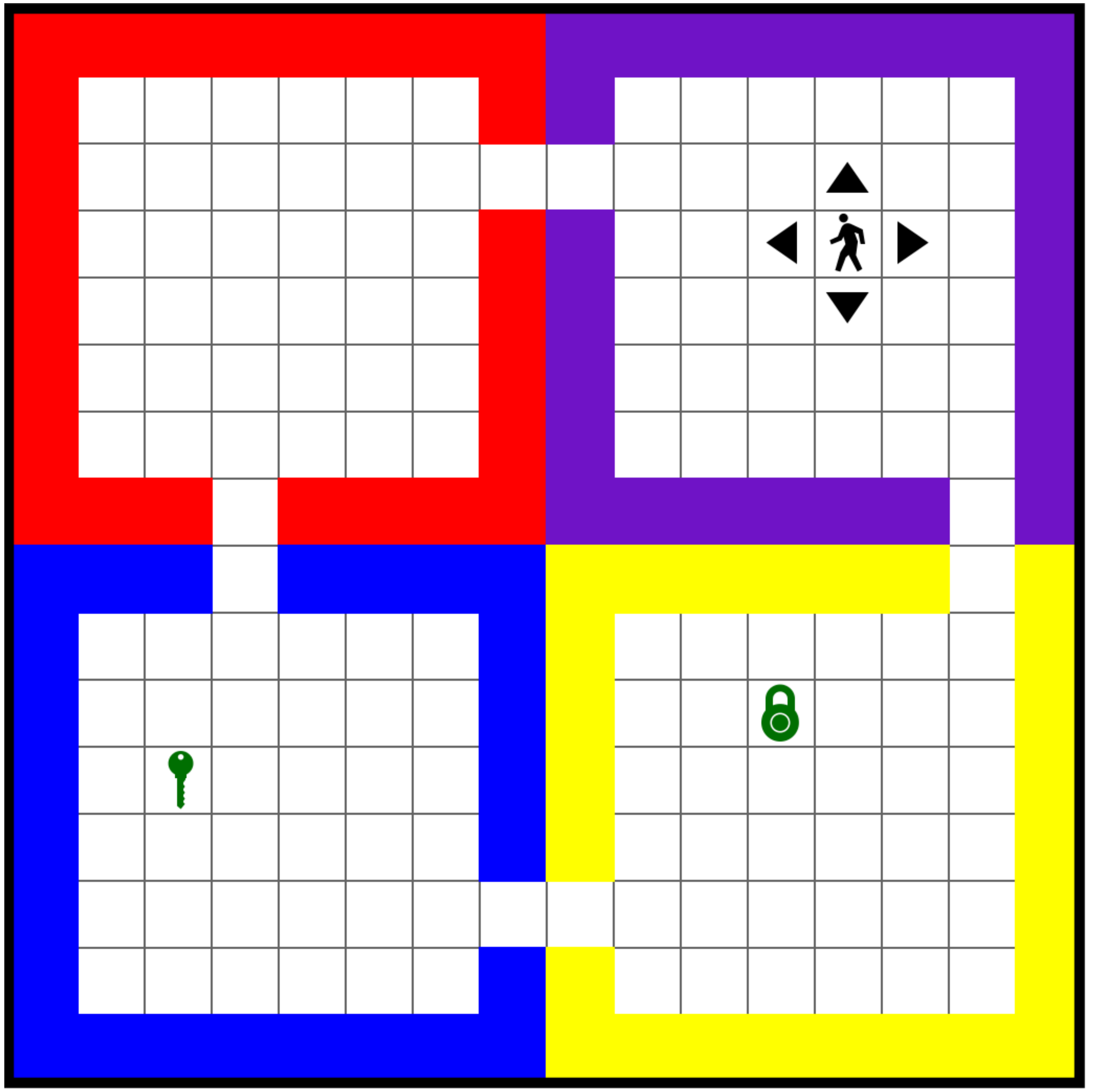
  - Centroids of K-means clusters (e.g. rooms)
  - Outliers as potential subgoals (e.g. key, box)
  - Boundary of two clusters (e.g. doorway)

Unsupervised Subgoal Discovery

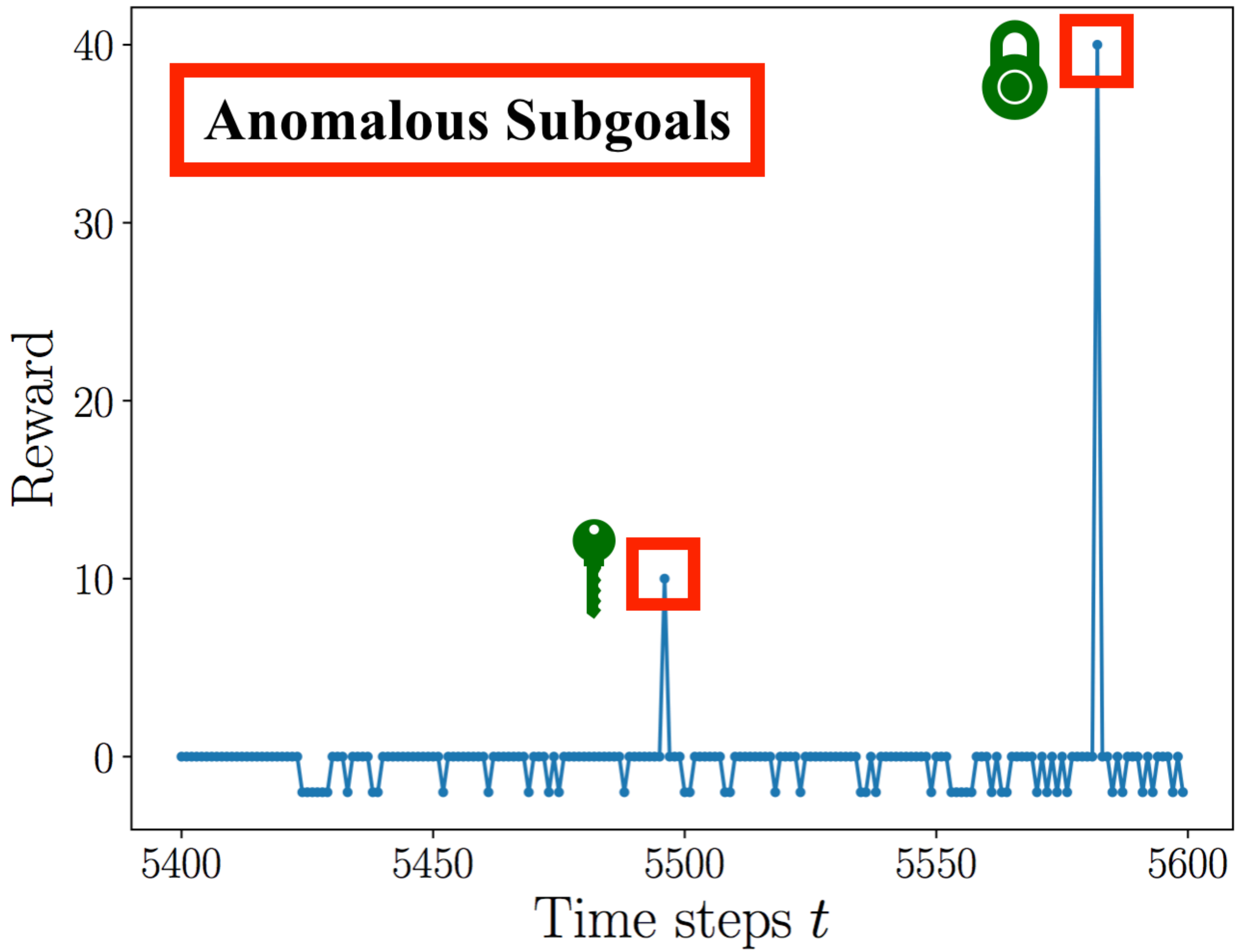
Algorithm 10 Unsupervised Subgoal Discovery Algorithm

for each $e = (s, a, r, s')$ stored in $\mathcal{D}$ **do**
  if experience e is an outlier (anomaly) **then**
    Store s' to the subgoals set $\mathcal{G}$
    Remove e from $\mathcal{D}$
  end if
end for
Fit a K-means Clustering Algorithm on $\mathcal{D}$ using previous centroids as initial points
Store the updated centroids to the subgoals set $\mathcal{G}$

Rooms Task

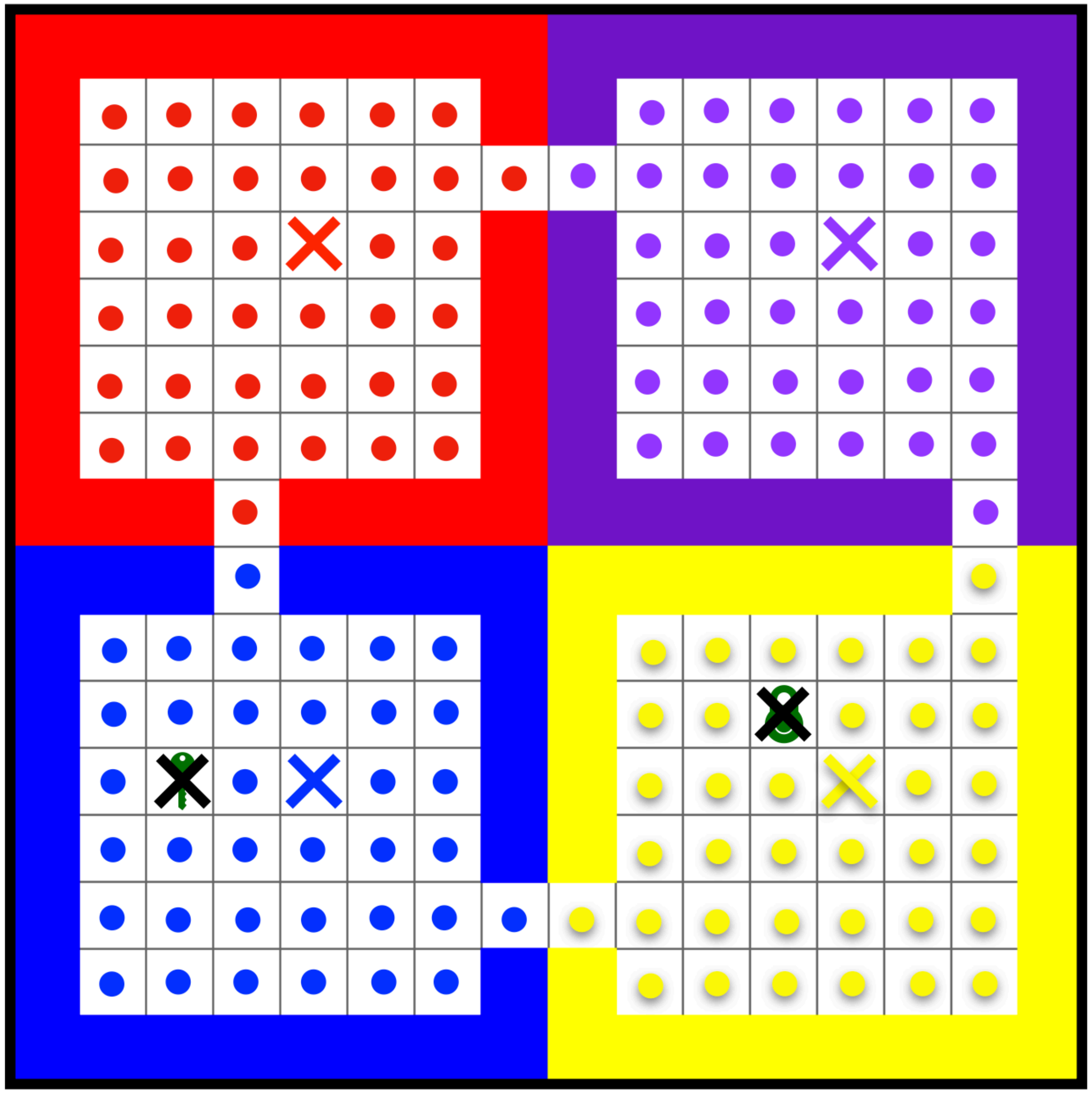


Anomaly Detection



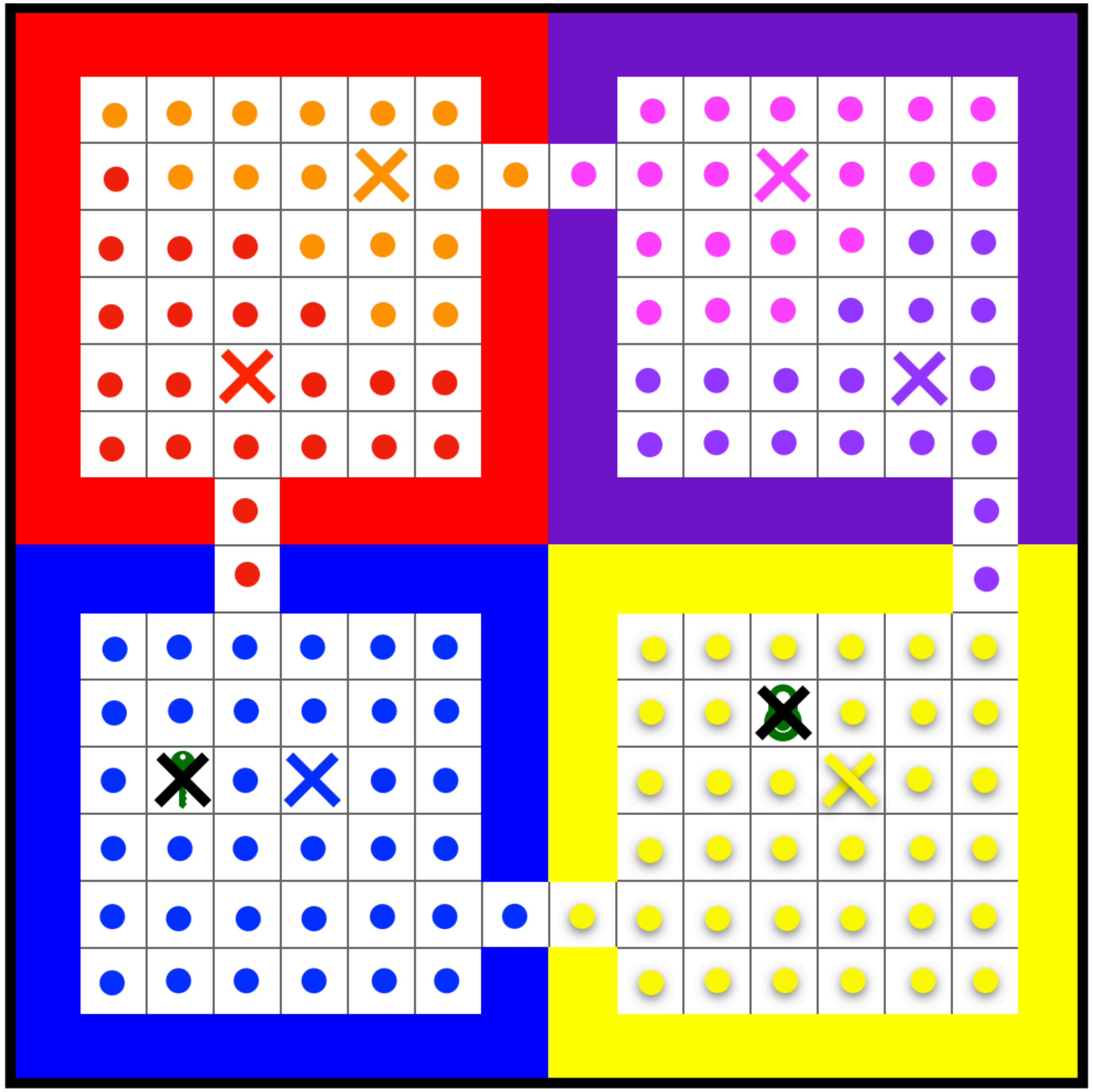
K-Means Clustering

$$K = 4$$



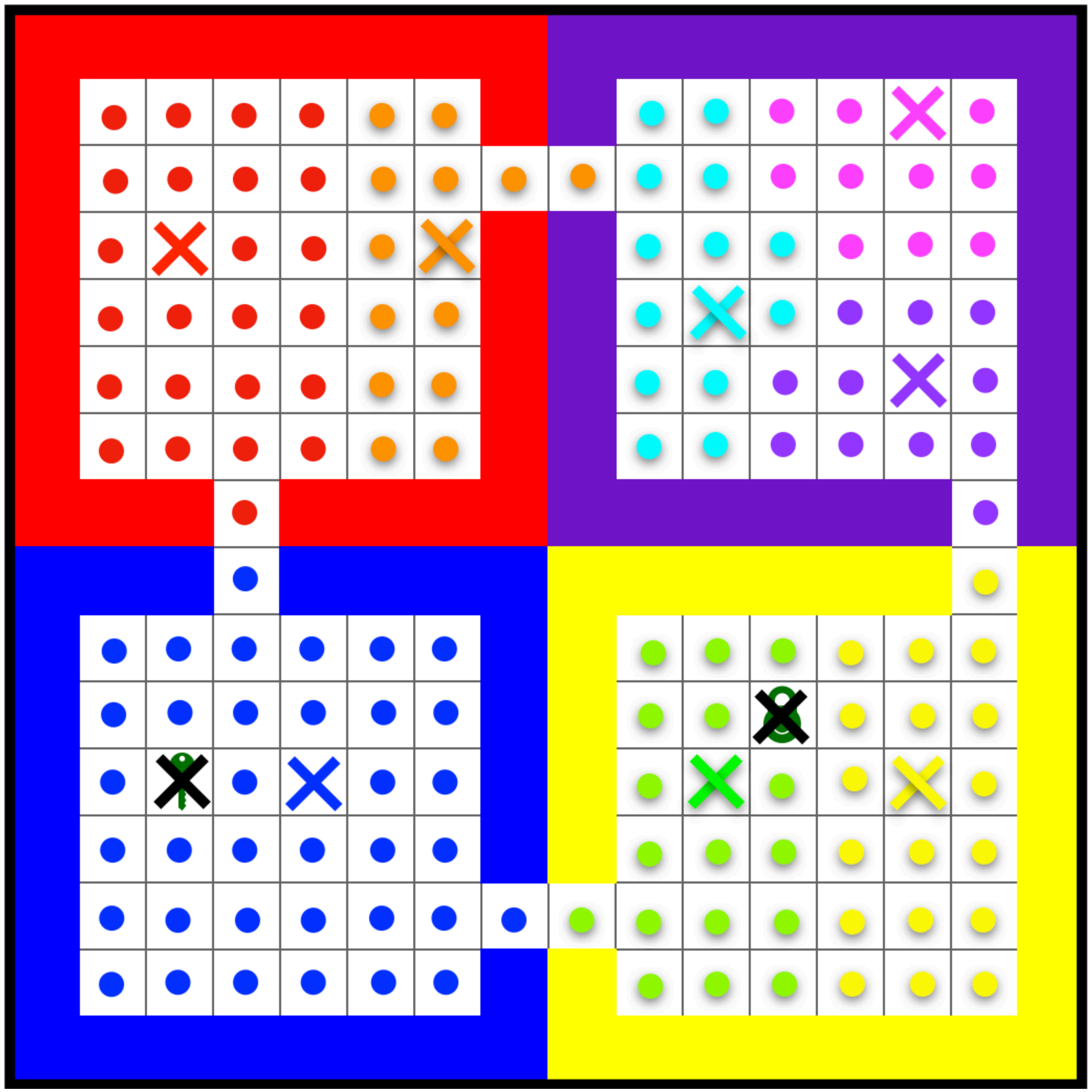
K-Means Clustering

$$K = 6$$



K-Means Clustering

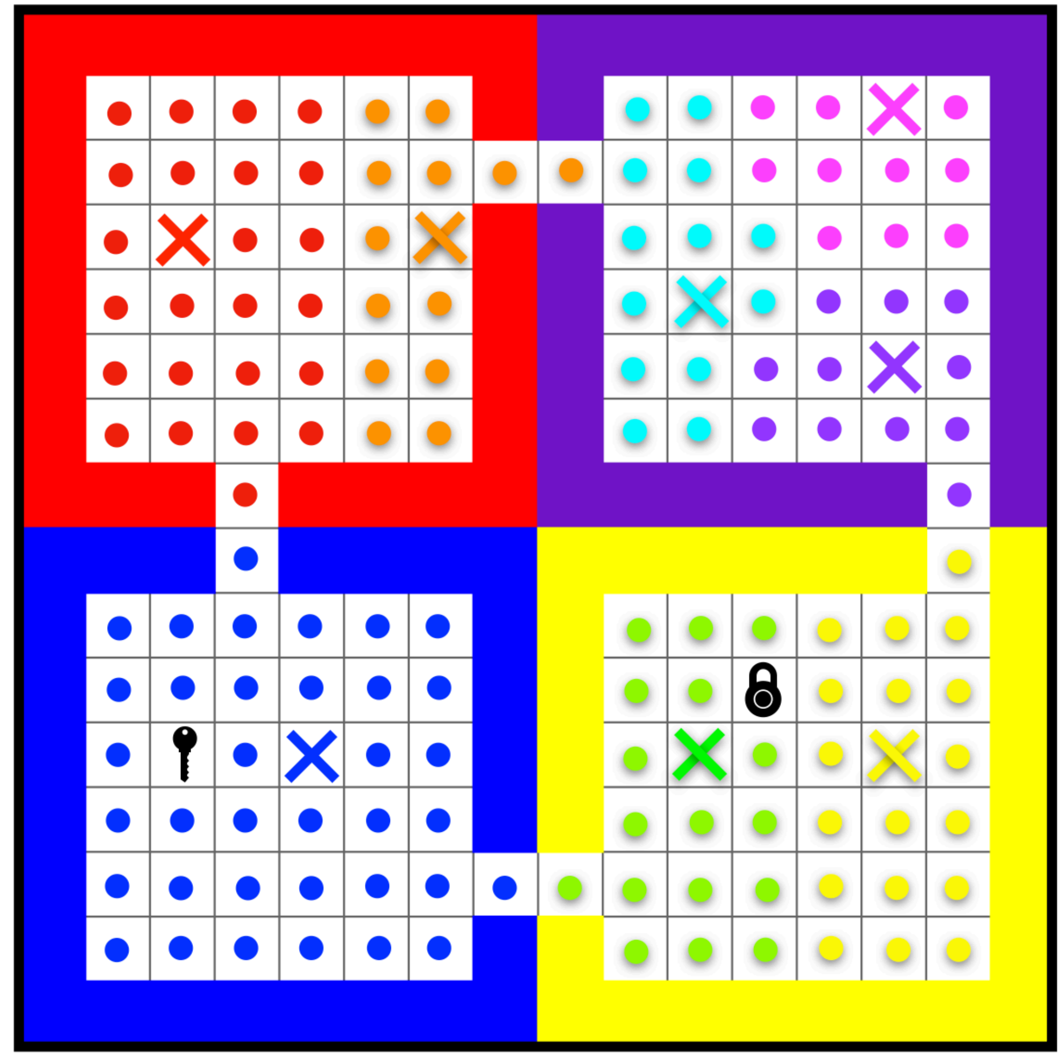
$$K = 8$$



Mathematical Interpretations

Value of a state in RL:

$$V_\pi(s) \triangleq \mathbb{E}\Big[\sum_{t=0}^{\mathcal{T}} \gamma^t r_t | s, \pi\Big]$$



Value of a state in regards to the meta-controller's value function

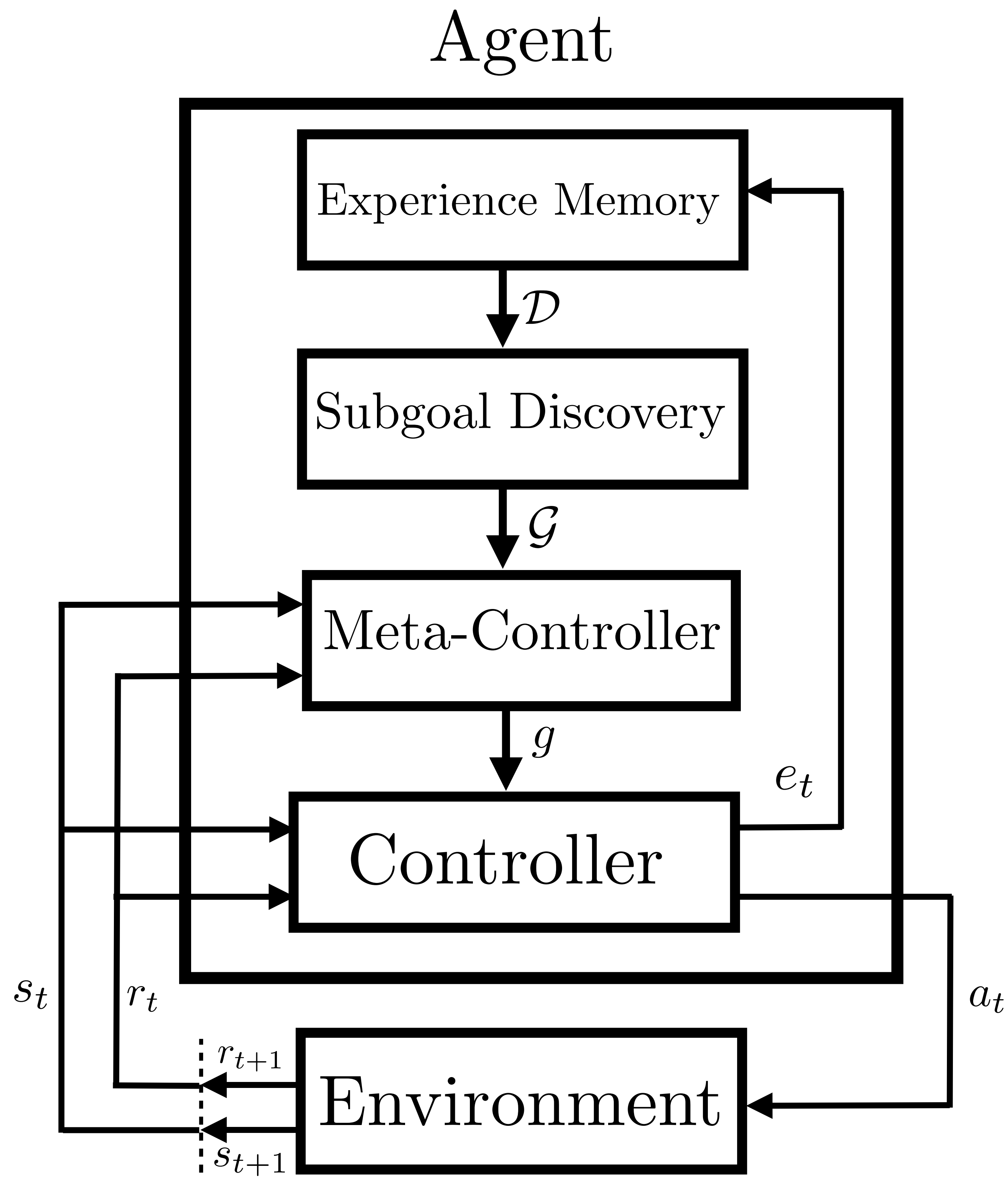
$$V(s) \approx Q(s, g_1) + \gamma^{T_1} Q(g_1, g_2) + \gamma^{T_1+T_2} Q(g_2, g_3) + \ldots$$

In tasks with sparse rewards, states within a cluster have similar values.

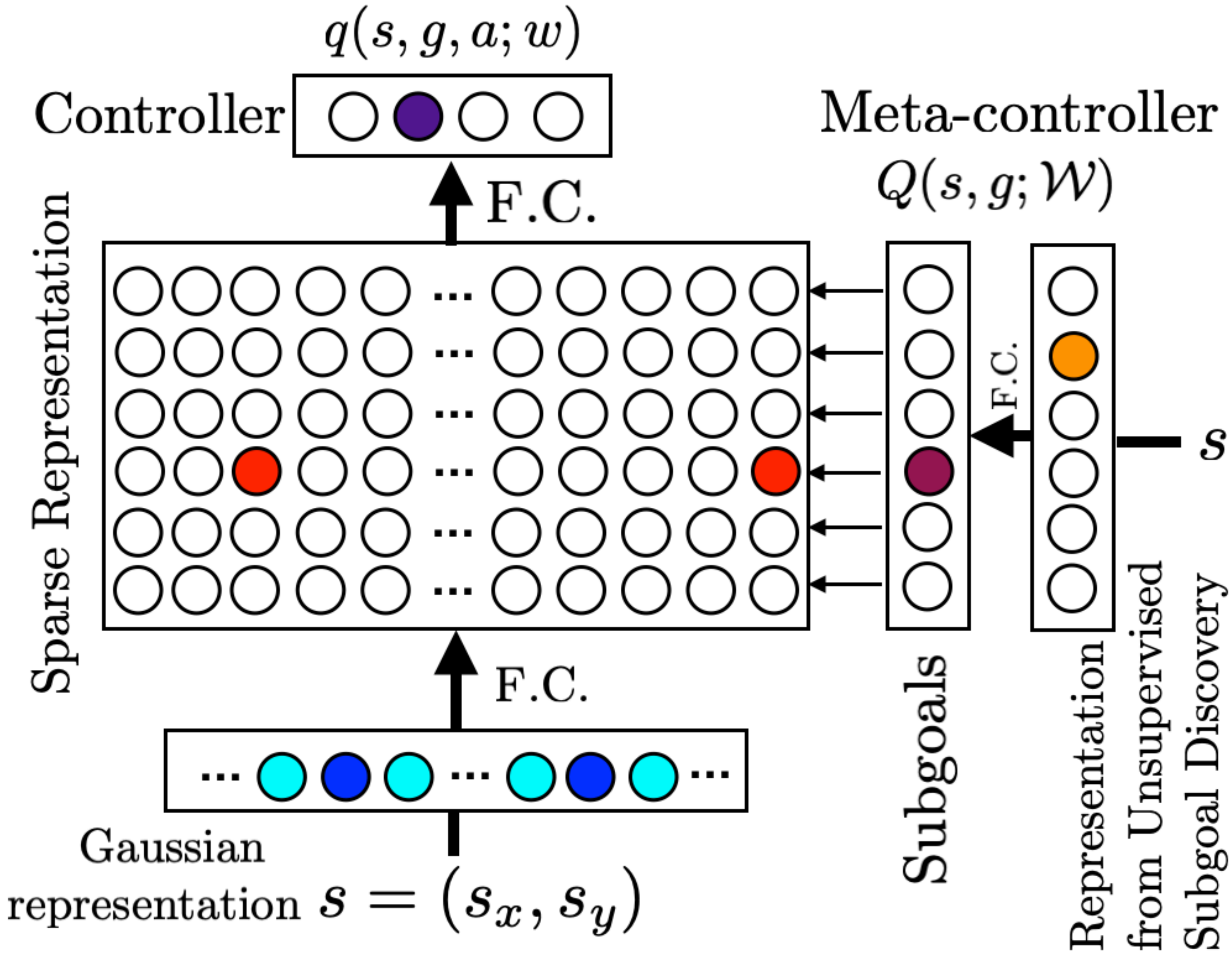
Unification of HRL Subproblems

- Implementing a model-free HRL framework that makes it possible to integrate automatic subgoal discovery, intrinsic motivation, and temporal abstraction.
- Learning subgoal-selection policy and action-policy simultaneously.
- The unification element should only use agent's experience memory (trajectories).

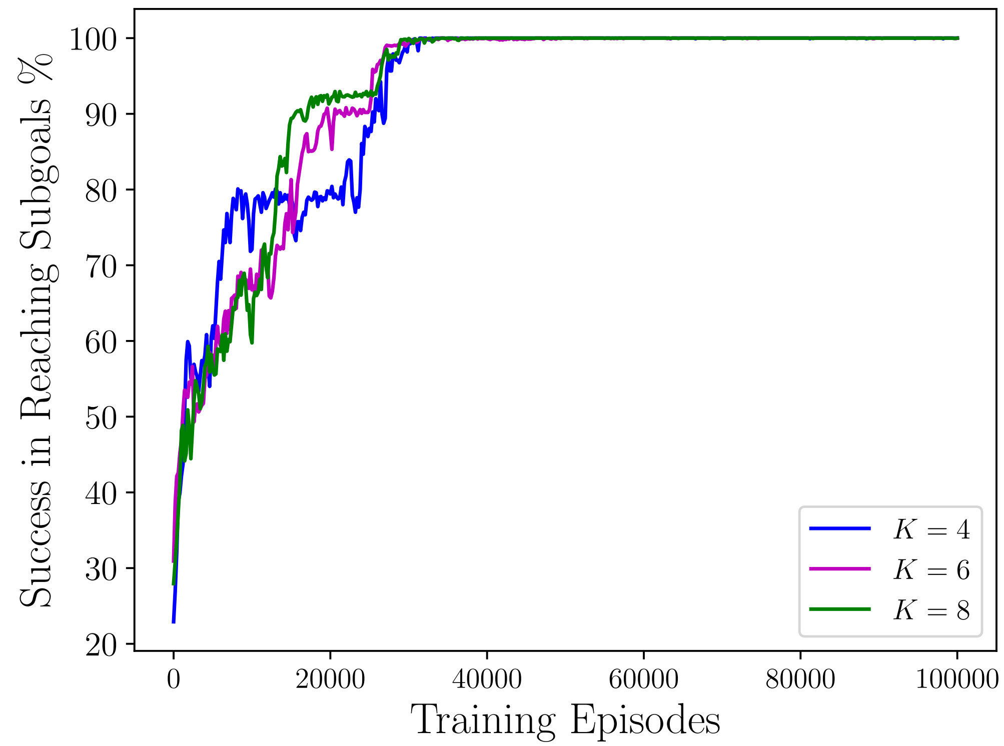
Unified Model-Free HRL

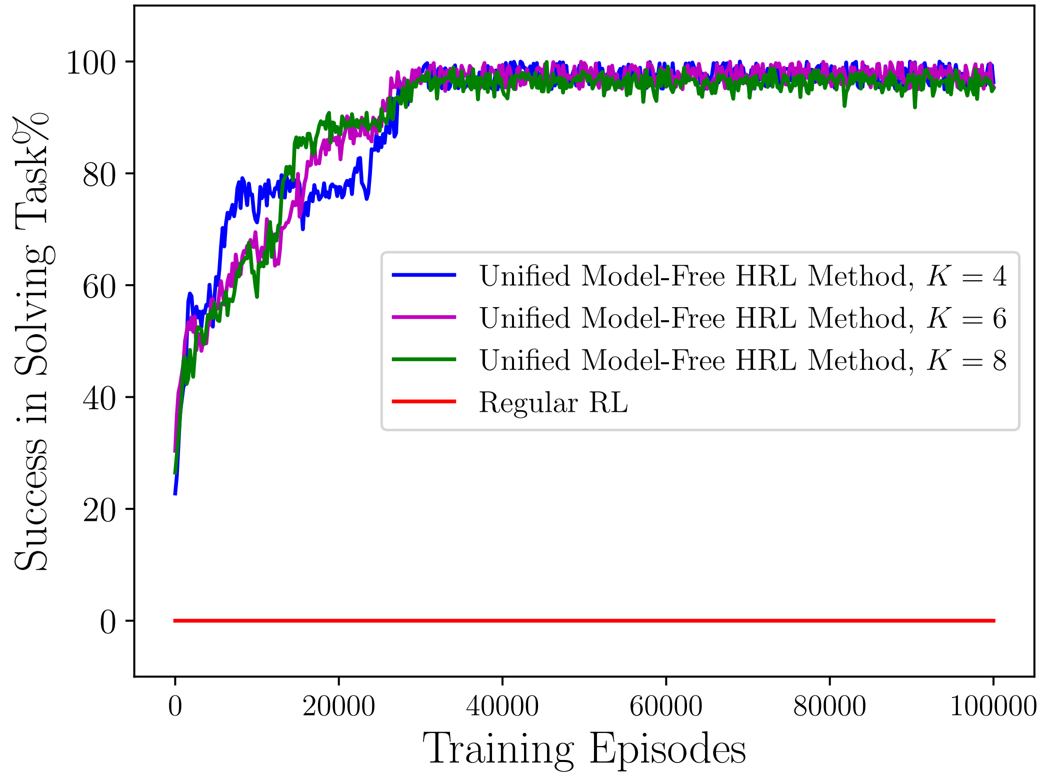


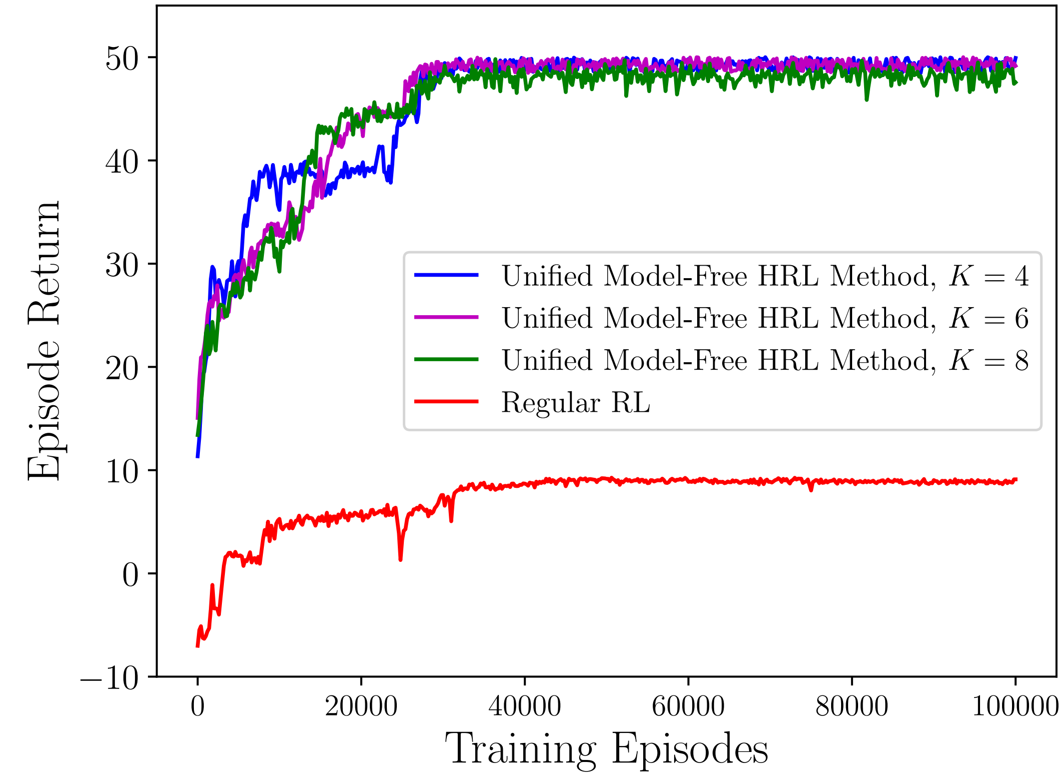
Neural Networks for Meta-controller and controller Rooms Task



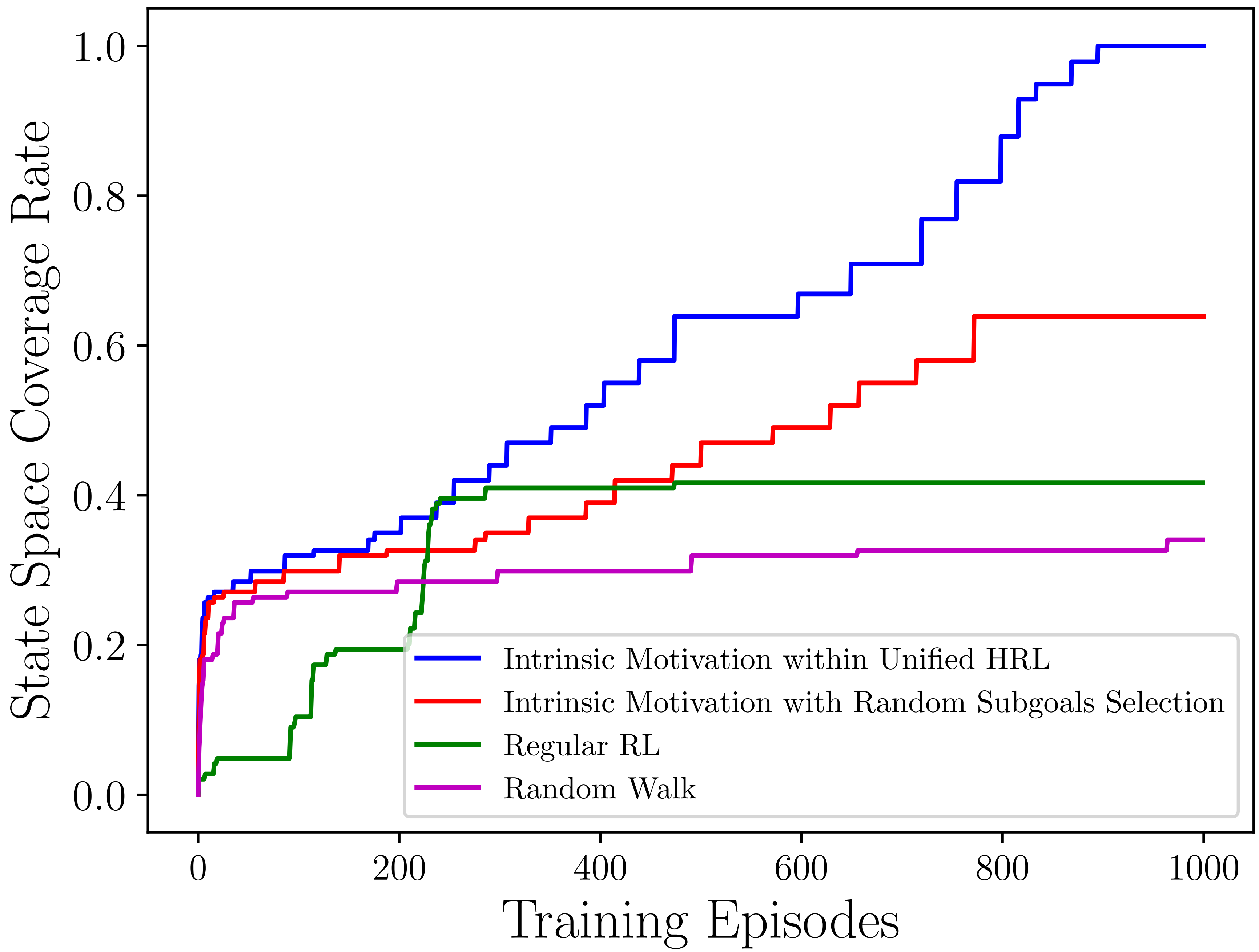
Results — 4-Rooms task



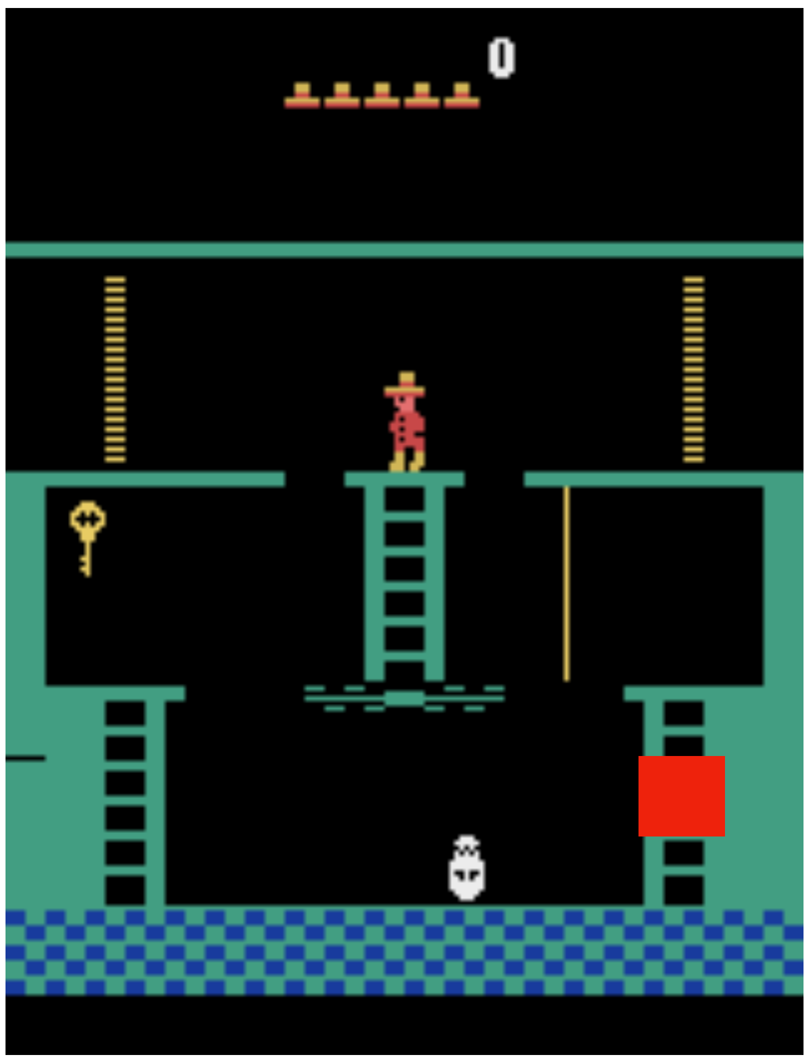




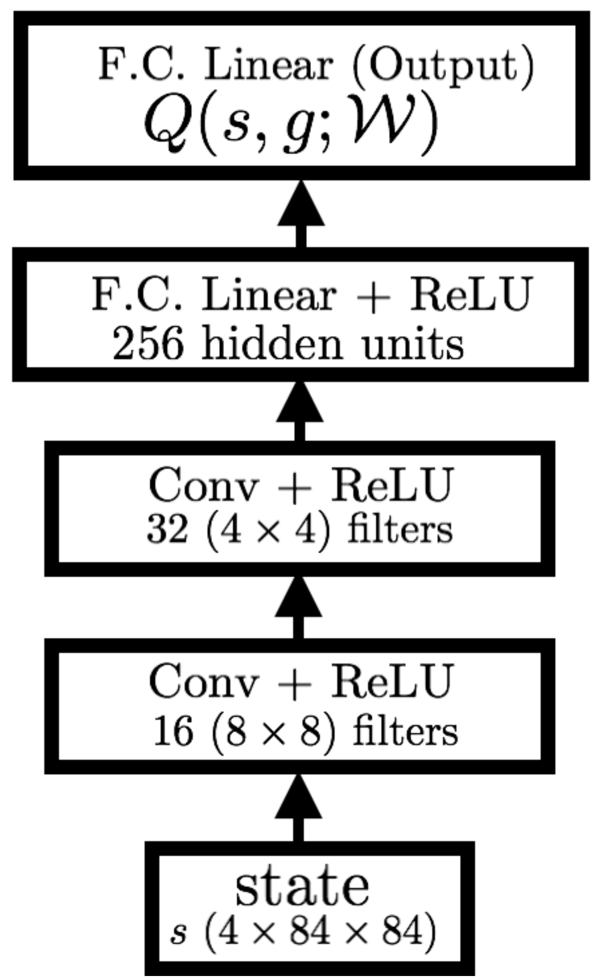
The Role of Intrinsic Motivation in Efficient Exploration of Rooms



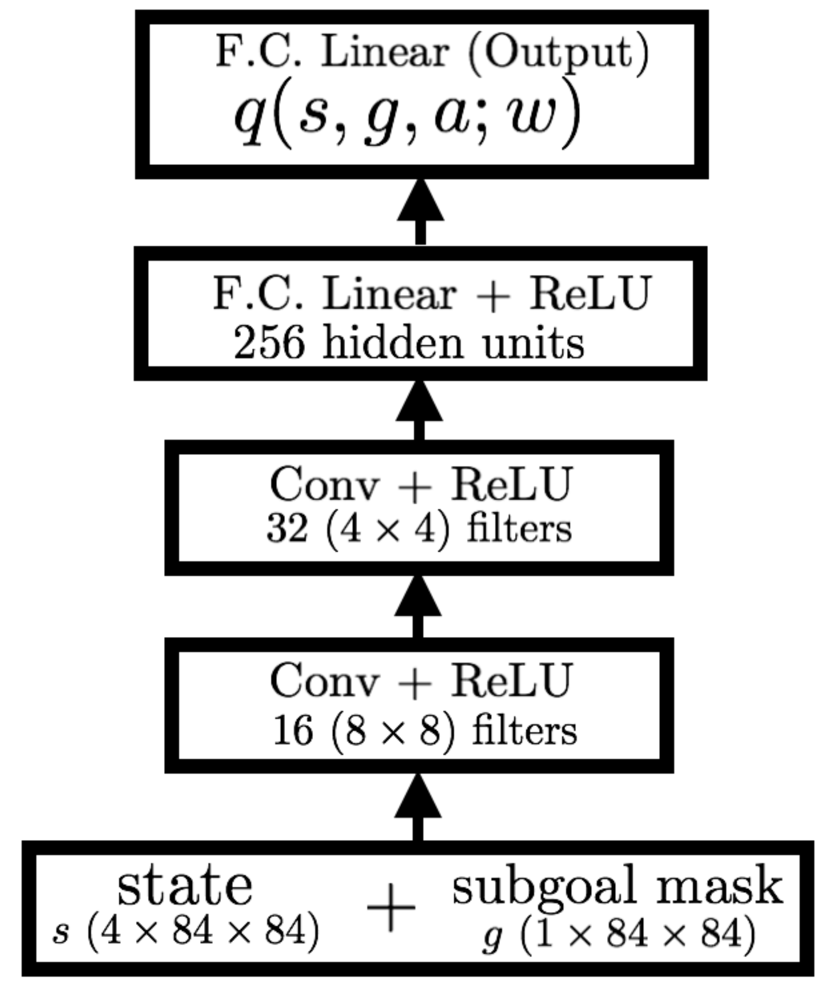
Neural Networks for Meta-controller and controller
Montezuma's Revenge



Meta-Controller

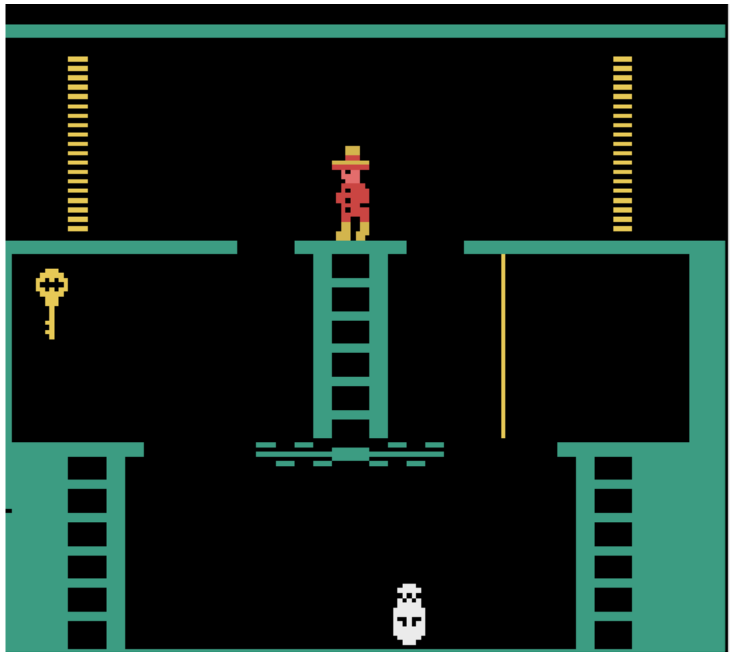


Controller

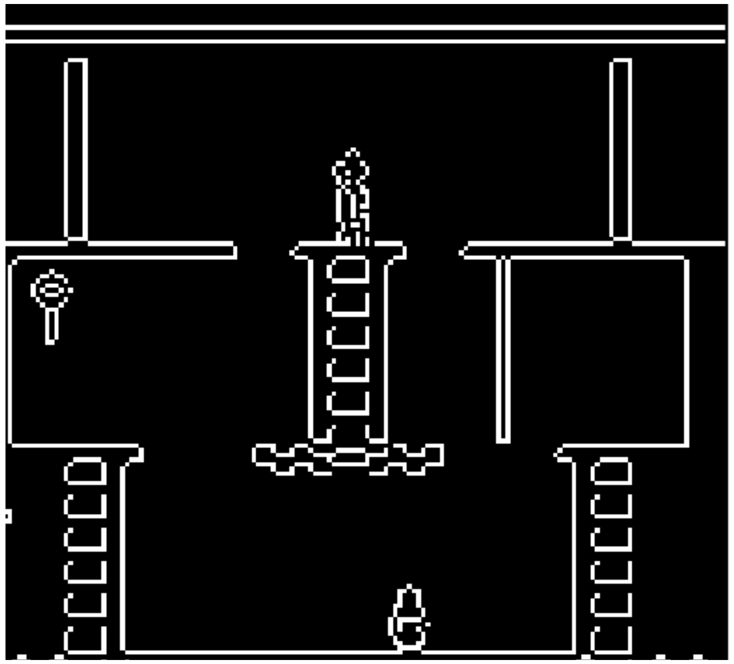


Computer Vision methods for finding “interesting” Initial Subgoals for Intrinsic Motivation Learning

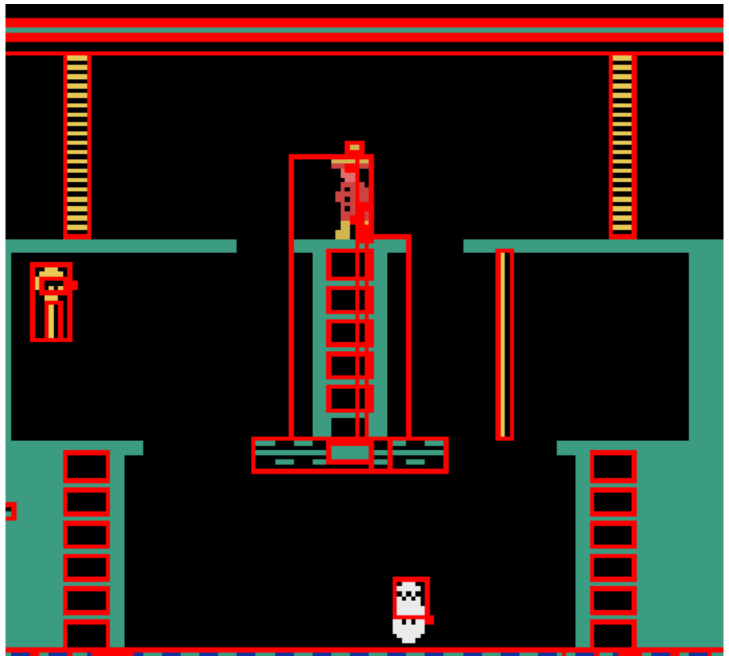
Original



Edge Detection

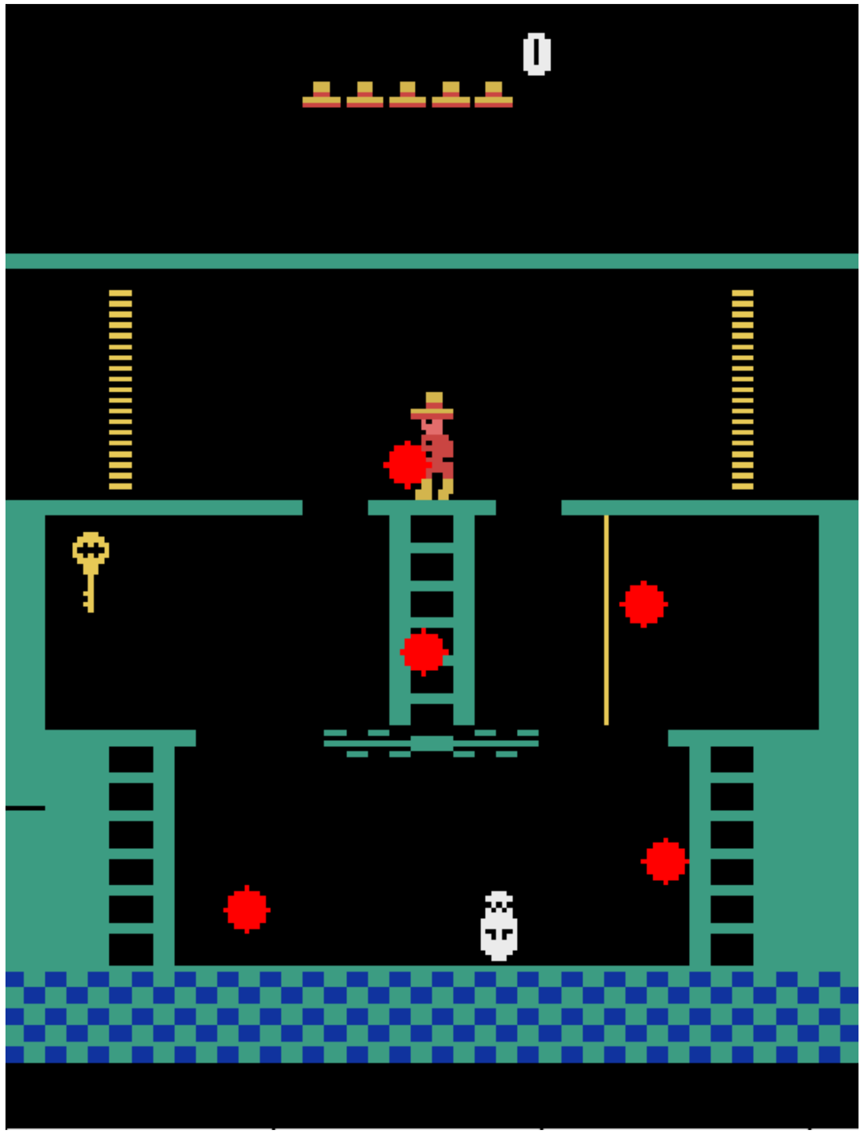


Bounding Box

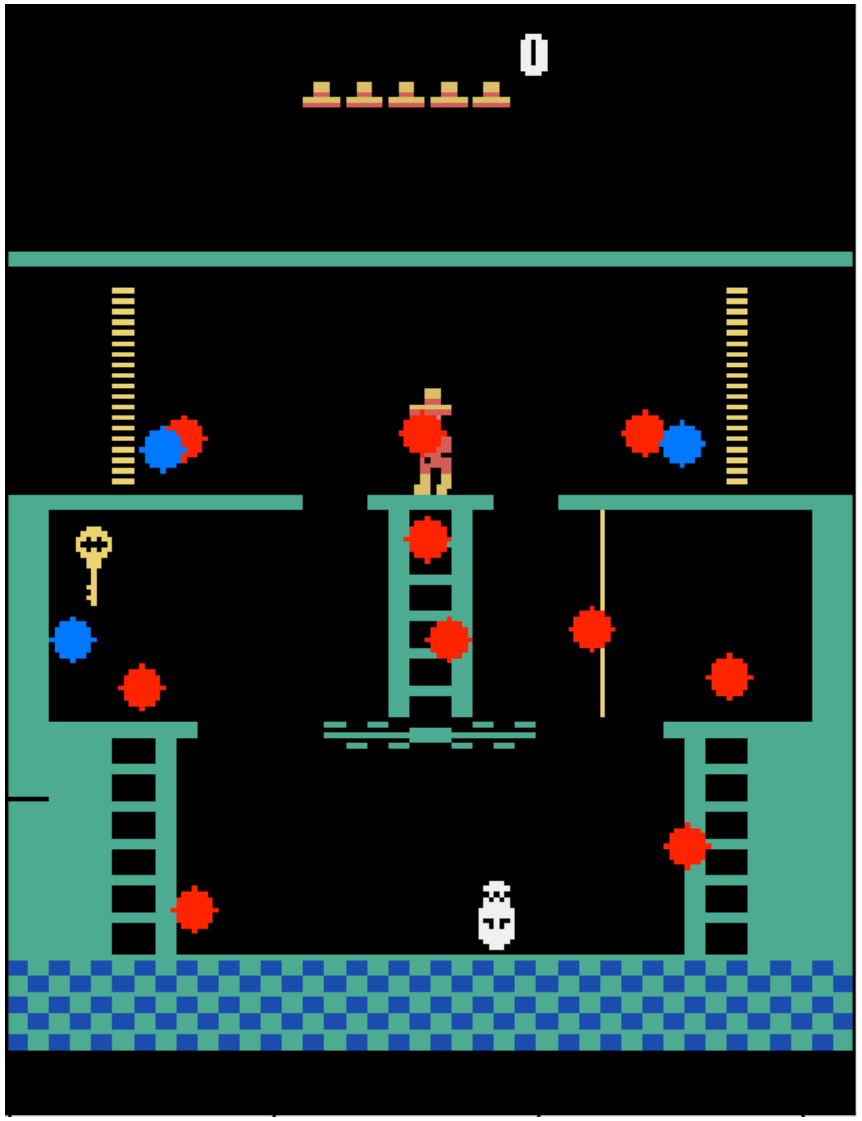


Unsupervised Subgoal Discovery for Montezuma's Revenge

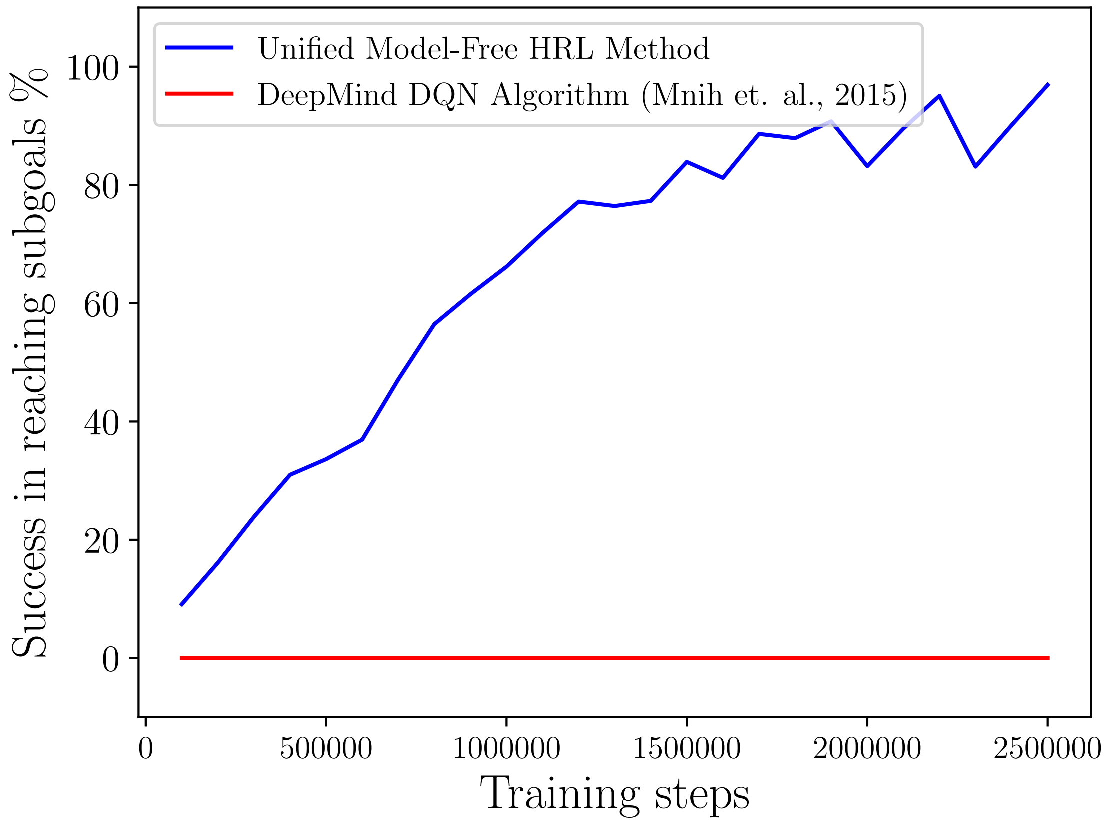
Random Walk

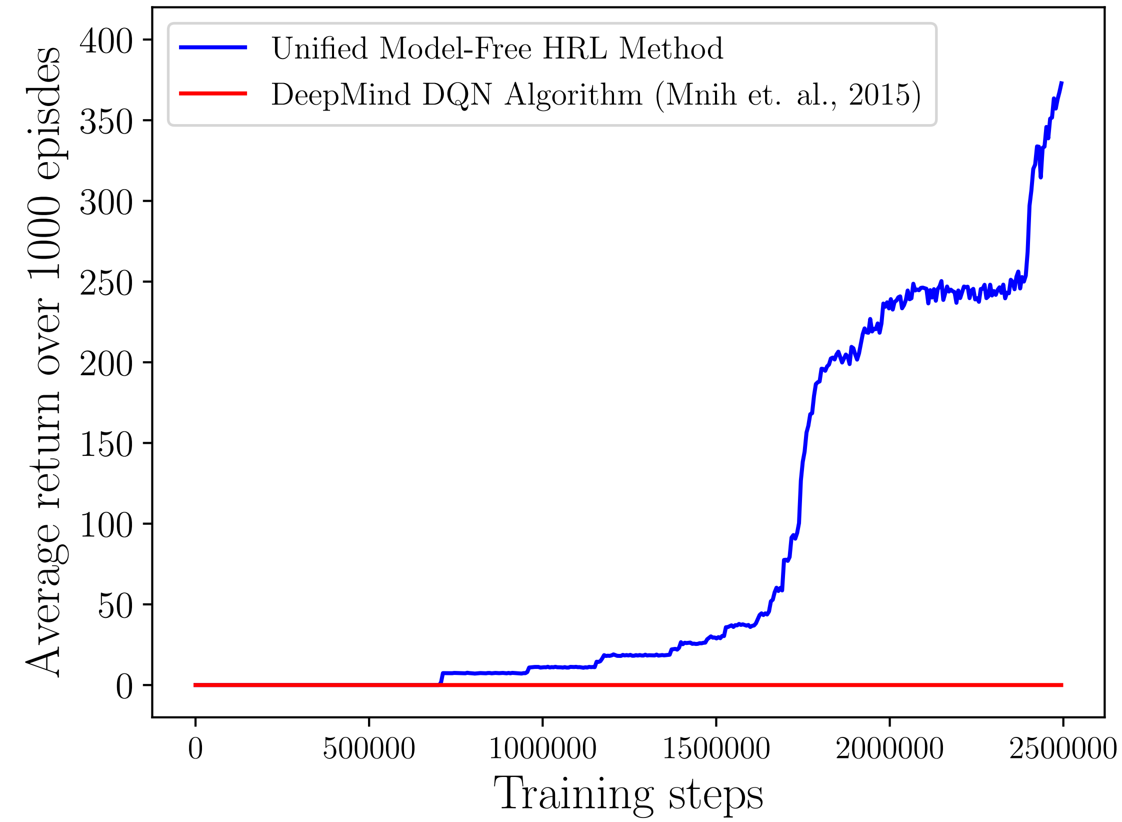


Our Method

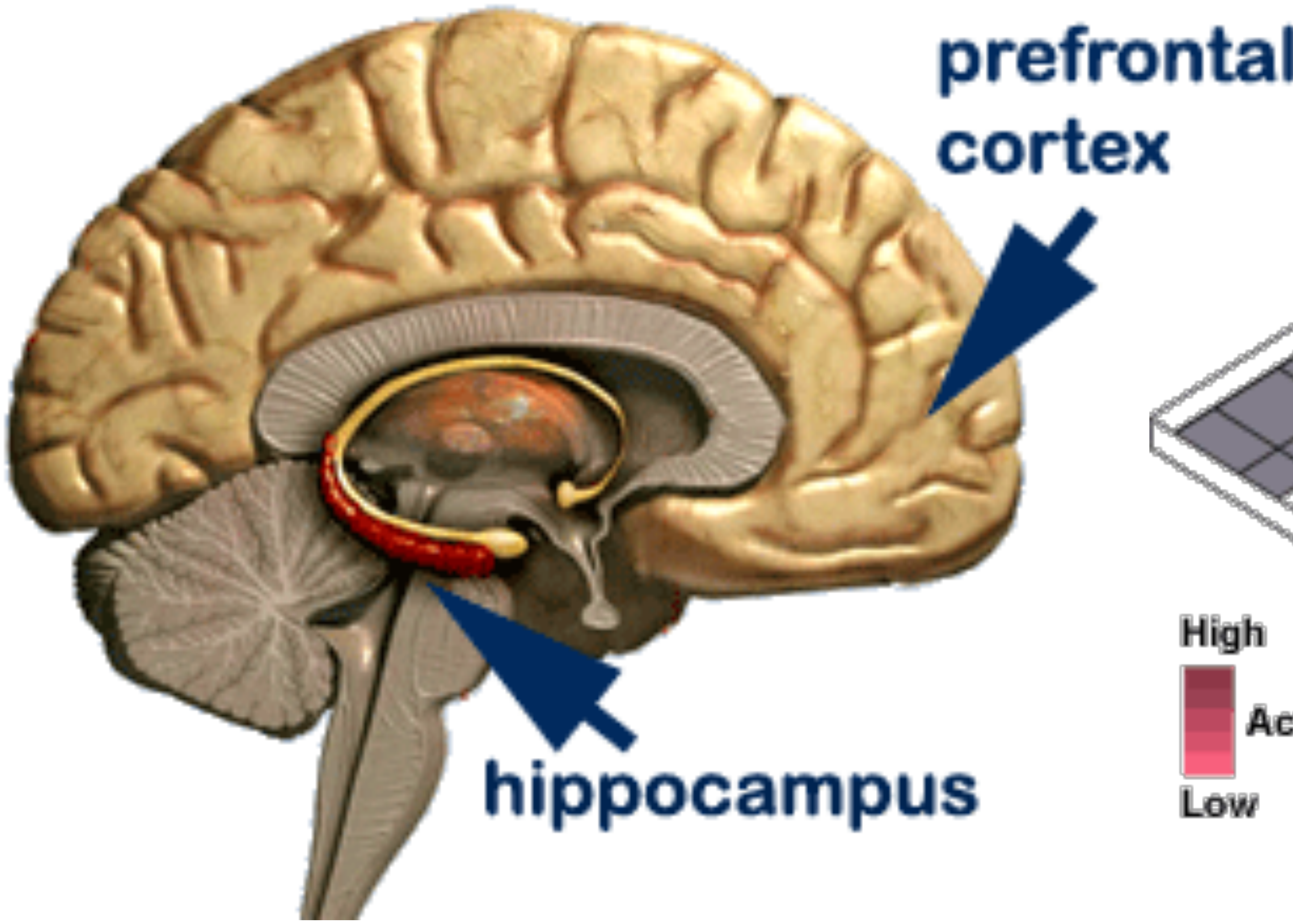


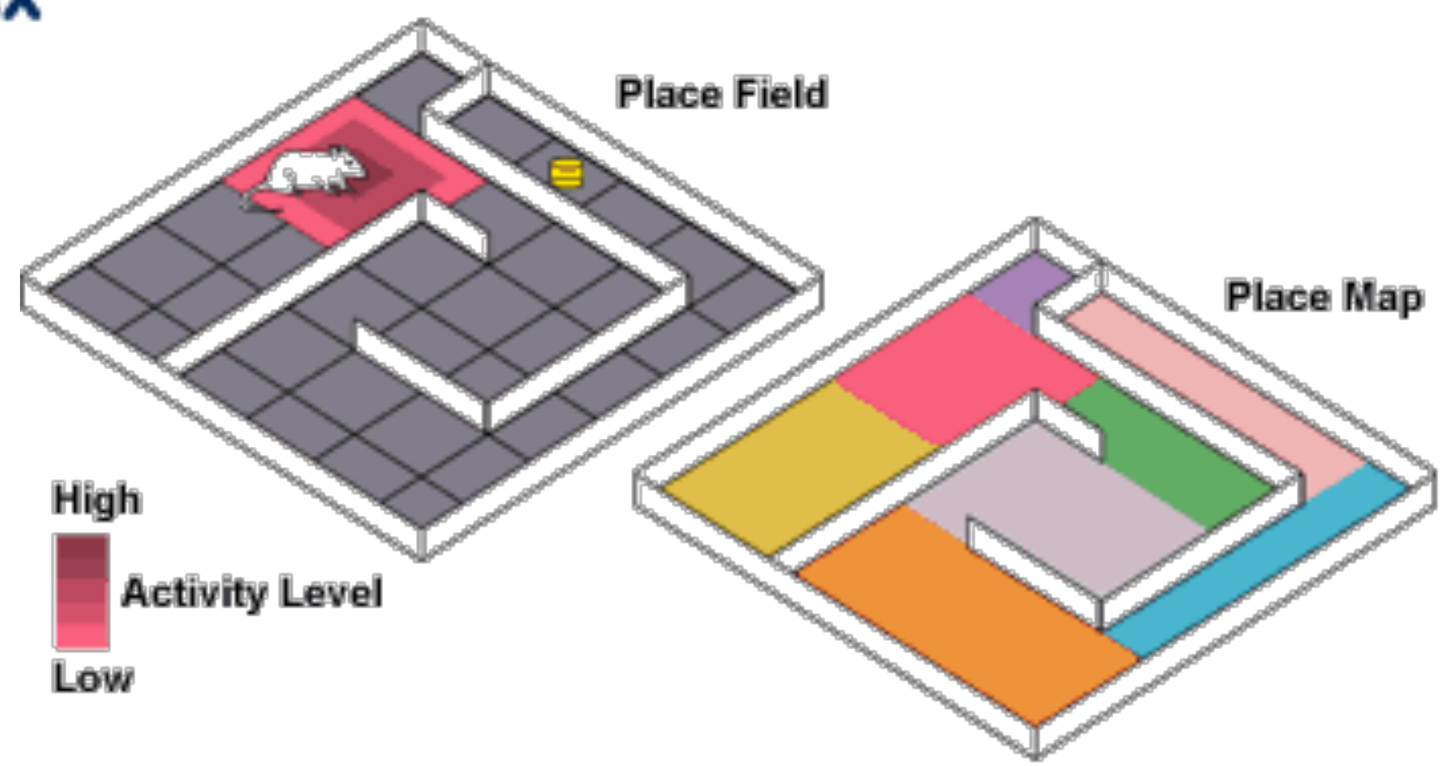
Results for Montezuma's Revenge





Neural Correlates of Unsupervised Subgoal Discovery





Strange et al. (2014). Functional organization of the hippocampal longitudinal axis. Nature Reviews Neuroscience, 15(10):655–669.
Chalmers et al. (2016). Computational properties of the hippocampus increase the efficiency of goal-directed foraging through hierarchical reinforcement learning. Frontiers in Computational Neuroscience, 10.

Conclusions

- We proposed and demonstrated a novel model-free HRL method for subgoal discovery using unsupervised learning over a small memory of experiences (trajectories) of the agent.
- When combined with an intrinsic motivation learning mechanism, this method learns subgoals and skills together, based on experiences in the environment.
- Intrinsic motivation learning provides efficient exploration scheme in tasks with sparse rewards that leads to successful subgoal discovery.
- We offered an HRL approach that does not require a model of the environment, making it suitable for larger-scale applications.

Future Work

- Learning Representations in model-based HRL in order to **plan** in case the direct experience is expensive (e.g. autonomous driving)
- Combining model-free and model-based HRL and solving entire game of Montezuma's Revenge using model-based HRL.
- Implement Computational Cognitive Neuroscience model of the model-free HRL framework.
- Empirical (fMRI, EEG) study on Neural correlates of unsupervised subgoal discovery and unified model-free/model-based HRL.
- Study on phenomenological interpretations of HRL.
- Improving the subgoal initialization with more advanced computer vision algorithms such as attention-based vision.

Yoshimi, J. (2017). Modeling consciousness using cognitive maps. Mind and Matter, 15(1):29–47.

Publications

- Jacob Rafati, David C. Noelle. (2019). Unsupervised Subgoal Discovery Method for Learning Hierarchical Representations. In 7th International Conference on Learning Representations, ICLR 2019 Workshop on "Structure & Priors in Reinforcement Learning", New Orleans, LA, USA.

- Jacob Rafati, David C. Noelle. (2019). Unsupervised Methods For Subgoal Discovery During Intrinsic Motivation in Model-Free Hierarchical Reinforcement Learning. In 33rd AAAI Conference on Artificial Intelligence (AAAI-19). Workshop on Knowledge Extraction From Games. Honolulu, Hawaii. USA.

- Jacob Rafati, and David C. Noelle (2019). Learning Representations in Model-Free Hierarchical Reinforcement Learning. In 33rd AAAI Conference on Artificial Intelligence (AAAI-19), Honolulu, Hawaii.

- Jacob Rafati, and David C. Noelle (2019). Learning Representations in Model-Free Hierarchical Reinforcement Learning. arXiv e-print (arXiv:1810.10096).

Problem 3.
Trust-Region Optimization Methods in Deep Learning

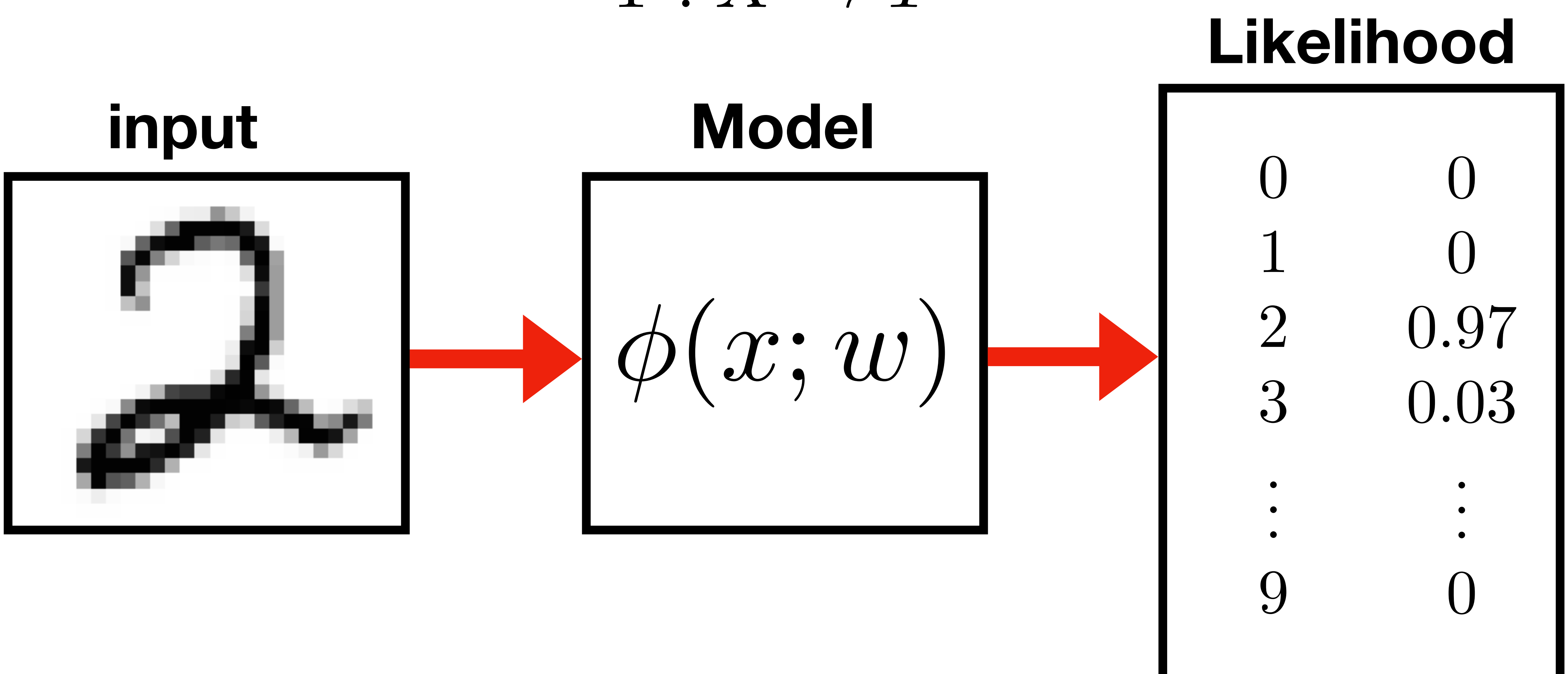
Supervised Learning

Features
$X = \{x_1, x_2, \ldots, x_i, \ldots, x_N\}$

Labels
$T = \{t_1, t_2, \ldots, t_i, \ldots, t_N\}$

Objective: Learning a mapping from data to labels,

$$\Phi : X \rightarrow T$$



Loss Function

$$y_i = \phi(x_i; w)$$
$$y_{ij} = p(y_{ij} = C_j | x_i; w)$$

Target

$$t = \begin{bmatrix}0, & 0, & 1, & 0, & \dots, & 0\end{bmatrix}$$

Prediction

$$y = \begin{bmatrix}0, & 0, & 0.97, & 0.03, & \dots, & 0\end{bmatrix}$$

Cross-Entropy Loss:

$$\ell(t, y) = -t.\log(y) - (1-t).\log(1-y)$$

Empirical Risk

$$\mathcal{L}(w) = \frac{1}{N}\sum_{i=1}^{N} \ell(t_i, \phi(x_i, w))$$

Empirical Risk Minimization

$$\min_{w \in \mathbb{R}^n} \mathcal{L}(w) \triangleq \frac{1}{N} \sum_{i=1}^{N} \ell_i(w)$$

$$\mathcal{L} : \mathbb{R}^n \to \mathbb{R}$$

- $n \text{ and } N$ are both large in modern applications.
- $\mathcal{L}(w)$ is a non-convex and nonlinear function.
- $\nabla^2 \mathcal{L}(w)$ is ill-conditioned.
- Computing full gradient, $\nabla \mathcal{L}$ is expensive.
- Computing Hessian, $\nabla^2 \mathcal{L}$ is not practical.

Quasi-Newton Methods

Construct a **low-rank** update of Hessian approximation with first-order gradient informations:

$$B_k \approx \nabla^2 \mathcal{L}(w_k)$$

Find the search direction by Minimizing the **Quadratic Model** of the objective function

$$p_k = \underset{p \in \mathbb{R}^n}{\operatorname{argmin}} \ \mathcal{Q}_k(p) \triangleq g_k^T p + \frac{1}{2} p^T B_k p^T$$

Secant Condition

- Symmetric
- Easy and Fast Computation
- Satisfy Secant Condition

Displacement Vector:

$$s_k \triangleq w_{k+1} - w_k$$

Gradients Difference Vector:

$$y_k \triangleq \nabla\mathcal{L}(w_{k+1}) - \nabla\mathcal{L}(w_k)$$

A Taylor expansion of the gradient difference will lead to

$$\nabla\mathcal{L}(w_{k+1}) - \nabla\mathcal{L}(w_{k+1}) \approx \nabla^2\mathcal{L}(w_{k+1})(w_{k+1} - w_k)$$

Quasi Newton matrices should satisfy **Secant Condition**

$$B_{k+1}s_k = y_k$$

Quasi-Newton Methods

Advantages:

- The rate of convergence is super-linear.
- They are resilient to problem ill-conditioning.
- The second derivative, Hessian matrix, is not required.
- They only use the gradient information to construct quasi-Newton matrices.

Disadvantages:

- The cost of storing the gradient informations can be expensive.
- The quasi-Newton matrix can be dense.
- The quasi-Newton matrix grow in size and rank in large-scale problems.

J. Nocedal and S. J. Wright. (2006). Numerical Optimization. 2nd ed. New York. Springer.

Broyden **F**letcher **G**oldfarb **S**hanno.

Broyden (1970). The convergence of a class of double-rank minimization algorithms: general considerations. SIAM Journal of Applied Mathematics, 6(1):76– 90.
Fletcher (1970). A new approach to variable metric algorithms. The Computer Journal, 13(3):317–322.
Goldfarb (1970). A family of variable-metric methods derived by variational means. Mathematics of computation, 24(109):23–26.
Shanno (1970). Conditioning of quasi-Newton methods for function minimization. Mathematics of computation, 24(111):647–656.

Broyden **F**letcher **G**oldfarb **S**hanno.

- Positive definite matrices.
- 2-rank updates.
- Quasi Newton matrices should satisfy **Secant Condition** $B_{k+1}s_k = y_k$

BFGS update formula:

$$B_{k+1} = B_k - \frac{1}{s_k^T B_k s_k} B_k s_k s_k^T B_k + \frac{1}{y_k^T s_k} y_k y_k^T,$$

With an initial matrix:

$$B_0 = \gamma_k I$$

Displacement Vector:

$$s_k \triangleq w_{k+1} - w_k$$

Gradients Difference Vector:

$$y_k \triangleq \nabla\mathcal{L}(w_{k+1}) - \nabla\mathcal{L}(w_k)$$

J. Nocedal and S. J. Wright. (2006). Numerical Optimization. 2nd ed. New York. Springer.

Compact Representation

Limited Memory Storage, *m* most recent vectors

$$S_k = \begin{bmatrix} s_{k-m} & \ldots & s_{k-1} \end{bmatrix} \qquad Y_k = \begin{bmatrix} y_{k-m} & \ldots & y_{k-1} \end{bmatrix}$$

L-BFGS has a Compact Representation

$$B_k = B_0 + \Psi_k M_k \Psi_k^T$$

where

$$\Psi_k = \begin{bmatrix} B_0 S_k \; Y_k \end{bmatrix}, \quad M_k = \begin{bmatrix} -S_k^T B_0 S_k & -L_k \\ -L_k^T & D_k \end{bmatrix}^{-1}$$

LDU decomposition

$$S_k^T Y_k = L_k + D_k + U_k$$

Problem 3:
Optimization Methods in Deep Learning

Objective:

- Implementing fast and robust stochastic quasi-Newton optimization in deep learning.
- Multi-batch Stochastic L-BFGS in Line Search strategy.
- Multi-Batch Stochastic L-BFGS in Trust-Region strategy.

Quasi-Newton Matrices

- Symmetric
- Easy and Fast Computation
- Satisfy Secant Condition

Displacement Vector:

$$s_k \triangleq w_{k+1} - w_k$$

Gradients Difference Vector:

$$y_k \triangleq \nabla\mathcal{L}(w_{k+1}) - \nabla\mathcal{L}(w_k)$$

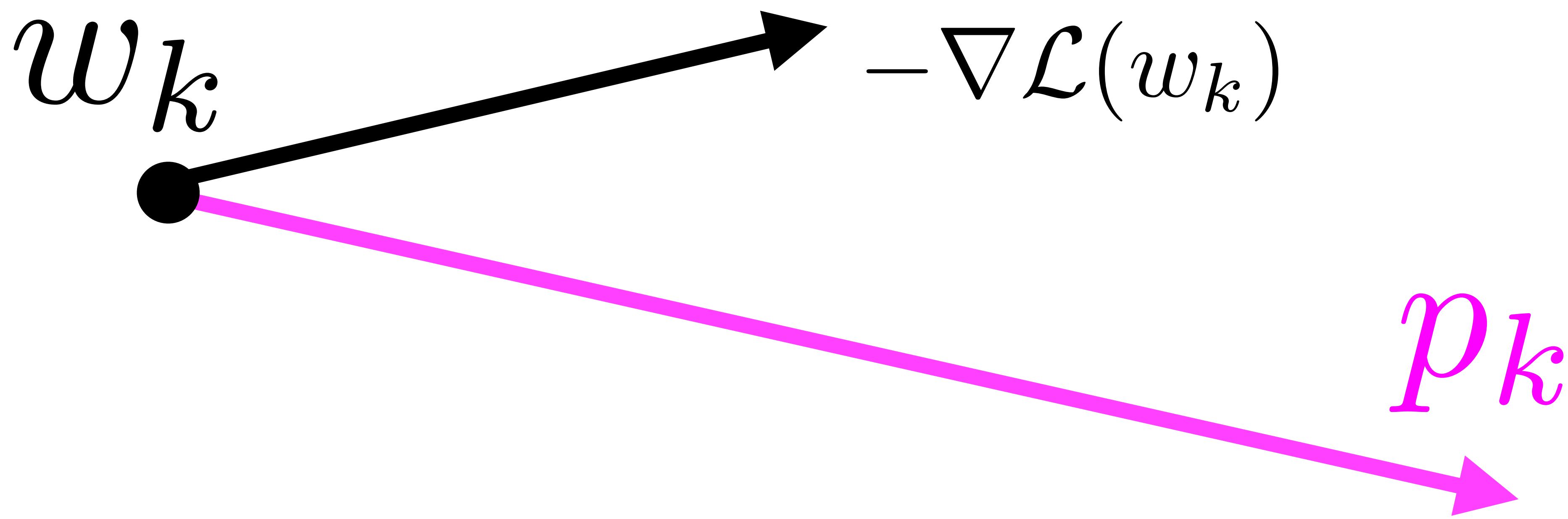
A Taylor expansion of the gradient difference will lead to

$$\nabla\mathcal{L}(w_{k+1}) - \nabla\mathcal{L}(w_{k+1}) \approx \nabla^2\mathcal{L}(w_{k+1})(w_{k+1} - w_k)$$

Quasi Newton matrices should satisfy **Secant Condition**

$$B_{k+1}s_k = y_k$$

Line Search Strategy



Quadratic model of objective function

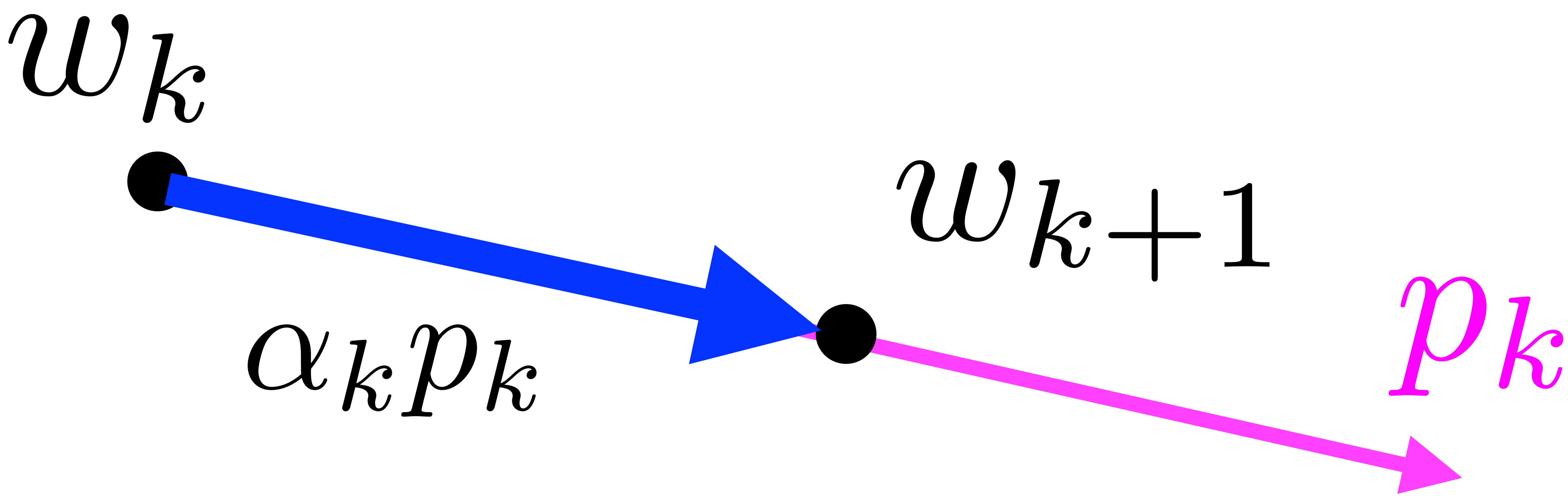
$$p_k = \operatorname*{argmin}_{p\in\mathbb{R}^n}\ \mathcal{Q}_k(p) \triangleq g_k^T p + \frac{1}{2} p^T B_k p^T$$

if B_k is positive definite:

$$p_k = B_k^{-1} g_k$$

J. Nocedal and S. J. Wright. (2006). Numerical Optimization. 2nd ed. New York. Springer.

Line Search Strategy



Next we should find a proper step size by solving

$$\alpha_k = \min_{\alpha} \mathcal{L}(w_k + \alpha p_k)$$

Instead we satisfy the sufficient decrease and curvature conditions known as

Wolfe conditions

$$\mathcal{L}(w_k + \alpha_k p_k) \leq \mathcal{L}(w_k) + c_1 \alpha_k \nabla \mathcal{L}(w_k)^T p_k$$

$$\nabla \mathcal{L}(w_k + \alpha_k p_k)^T p_k \geq c_2 \nabla f(w_k)^T p_k$$

J. Nocedal and S. J. Wright. (2006). Numerical Optimization. 2nd ed. New York. Springer.

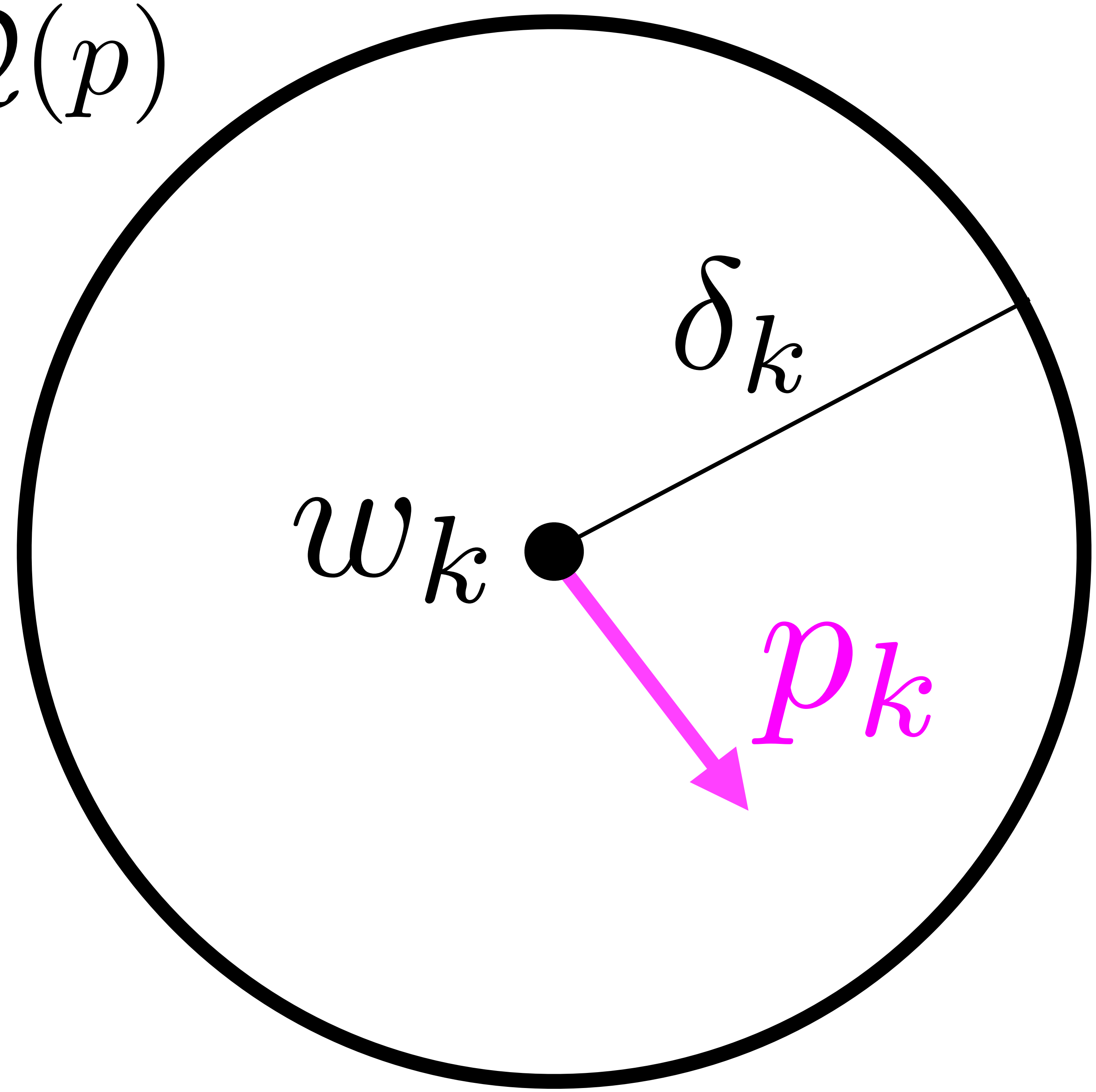
Line-search strategy

- Compute *stochastic* gradient
- Compute the quasi-Newton matrix, B_k
- Compute the search direction, $p_k = B_k^{-1} g_k$
- Compute step length using Wolfe Conditions
- Update parameters, $w_{k+1} \leftarrow w_k + \alpha_k p_k$

Trust-Region Srategy

$$p_k = \arg\min_{p \in \mathbb{R}^n} Q(p)$$

$$\text{s.t. } \|p\|_2 \leq \delta_k$$



Toint et al. (2000), Trust Region Methods, SIAM.

Optimality Conditions

Theorem.

A vector, p^*, is a global solution for trust-region subproblem,

$$\arg\min_p \mathcal{Q}(p) \text{ s.t. } \|p\|_2 \leq \delta,$$

if and only if

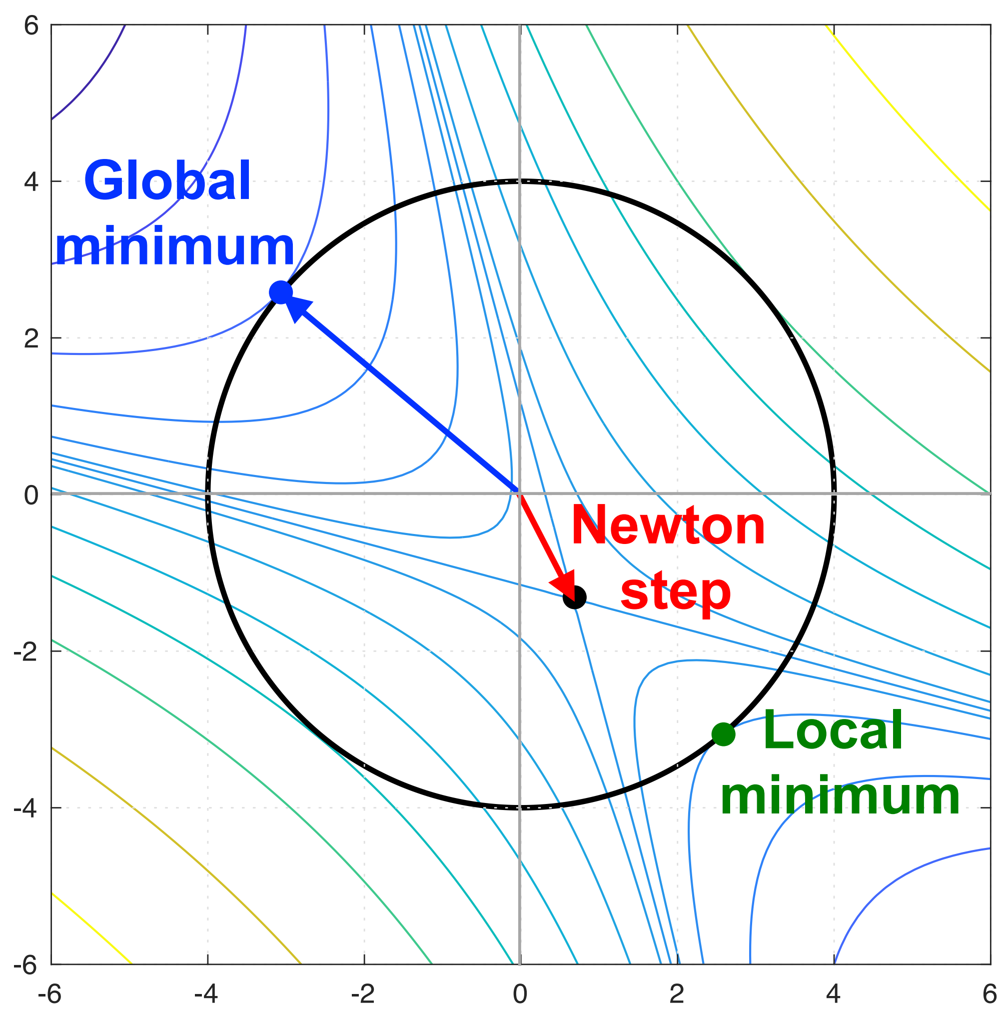
(i) $\|p^*\|_2 \leq \delta$

(ii) There exists a unique $\sigma^* \geq 0$ such that

(iii) $(B + \sigma^* I)p^* = -g,$ and

(iv) $\sigma^*(\delta - \|p^*\|_2) = 0$

Moreover, if $B + \sigma^* I$ is positive definite, then the global minimizer is unique.



J. J. Mor´e and D. C. Sorensen, (1984) Newton's method, in Studies in Mathematics, Volume 24. Studies in Numerical Analysis, Math. Assoc., pp. 29–82.

Trust-Region strategy

Compute *stochastic* gradient

Compute the quasi-Newton matrix, B_k

Compute the search direction by solving TR subproblem

$$p_k = \arg\min_p \mathcal{Q}_k(p) \text{ s.t. } \|p\|_2 \leq \delta_k$$

Compute the ratio of actual reduction to prediction reduction

$$\rho_k = \frac{\mathcal{L}(w_k) - \mathcal{L}(w_k + p_k)}{-\mathcal{Q}_k(p_k)}$$

Update TR radius, δ_k

Update parameters, $w_{k+1} \leftarrow w_k + \alpha_k p_k$

Solving Trust-region subproblem

Eigen-decomposition of $B_k = B_0 + \Psi_k M_k \Psi_k^T$

$$B_k = P \begin{bmatrix} \Lambda + \gamma_k I & 0 \\ 0 & \gamma_k I \end{bmatrix} P^T$$

Sherman-Morrison-Woodbury Formula gives a closed form solution to optimal search direction

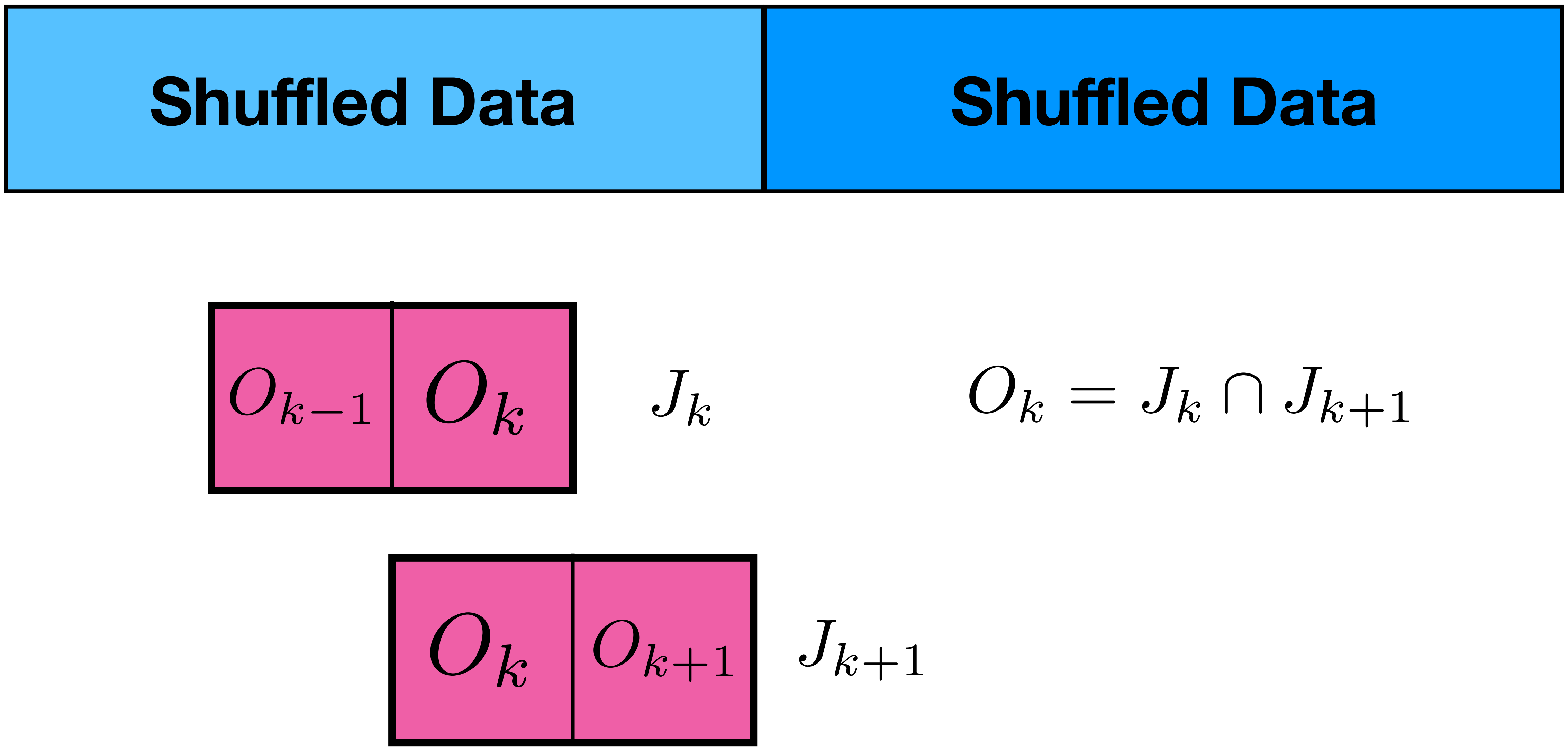
$$p_k^* = -\frac{1}{\tau^*} \left[I - \Psi_k (\tau^* M_k^{-1} + \Psi_k^T \Psi_k)^{-1} \Psi_k^T \right] g_k,$$

σ^* is the Lagrange multiplier, there are fast and accurate methods to find it. (See Brust et al (2017).)

$$\tau^* = \gamma_k + \sigma^*$$

L. Adhikari et al. (2017) “Limited-memory trust-region methods for sparse relaxation,” in Proc.SPIE, vol. 10394.
Brust et al, (2017). “On solving L-SR1 trust-region subproblems,” Computational Optimization and Applications, vol. 66, pp. 245–266.

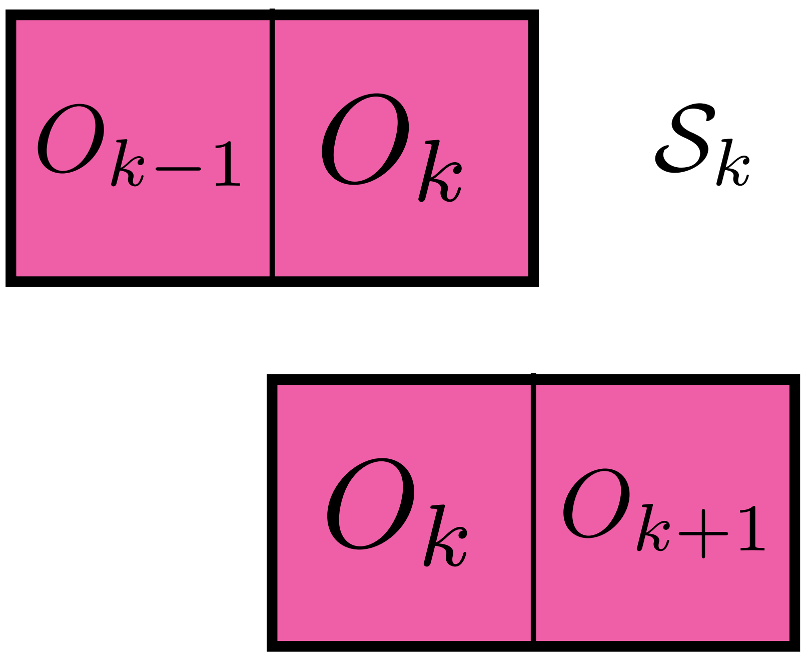
Multi-Batch L-BFGS



$$O_k = J_k \cap J_{k+1}$$

Berahas et al. (2016). "A multi-batch L-BFGS method for machine learning," in Advances in Neural Information Processing Systems 29, pp. 1055–1063.

Computing gradients



$\mathcal{S}_k$

$\mathcal{S}_{k+1}$

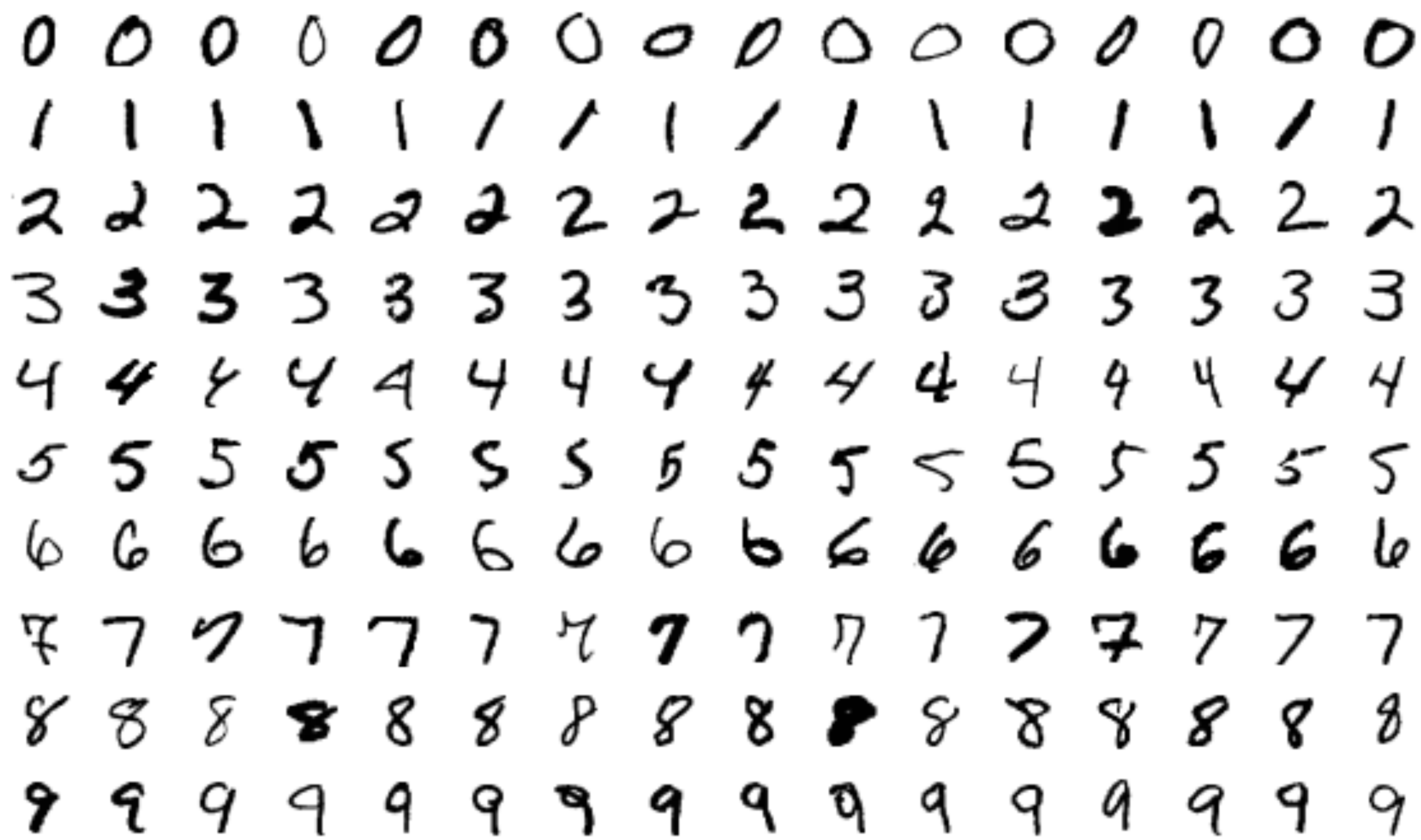
$$O_k = \mathcal{S}_k \cap \mathcal{S}_{k+1}$$

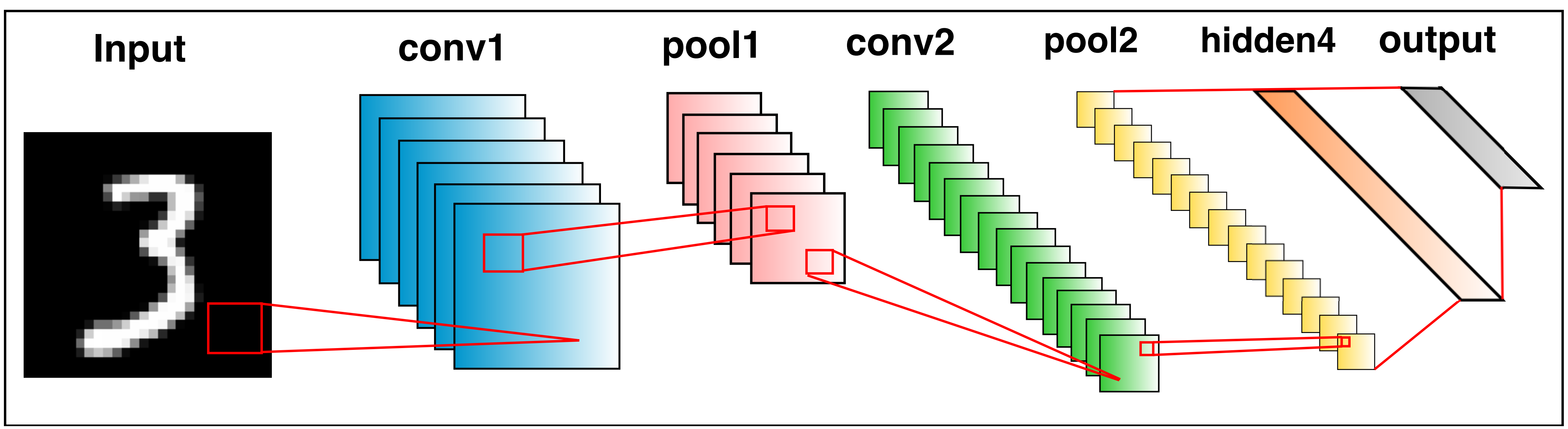
$$g_k = \nabla \mathcal{L}(w_k)^{(\mathcal{S}_k)} = \frac{1}{|\mathcal{S}_k|} \sum_{i \in J_k} \nabla \mathcal{L}_i(w_k)$$

$$y_k = \nabla \mathcal{L}(w_{k+1})^{(O_k)} - \nabla \mathcal{L}(w_k)^{(O_k)}$$

Berahas et al. (2016). “A multi-batch L-BFGS method for machine learning,” in Advances in Neural Information Processing Systems 29, pp. 1055–1063.

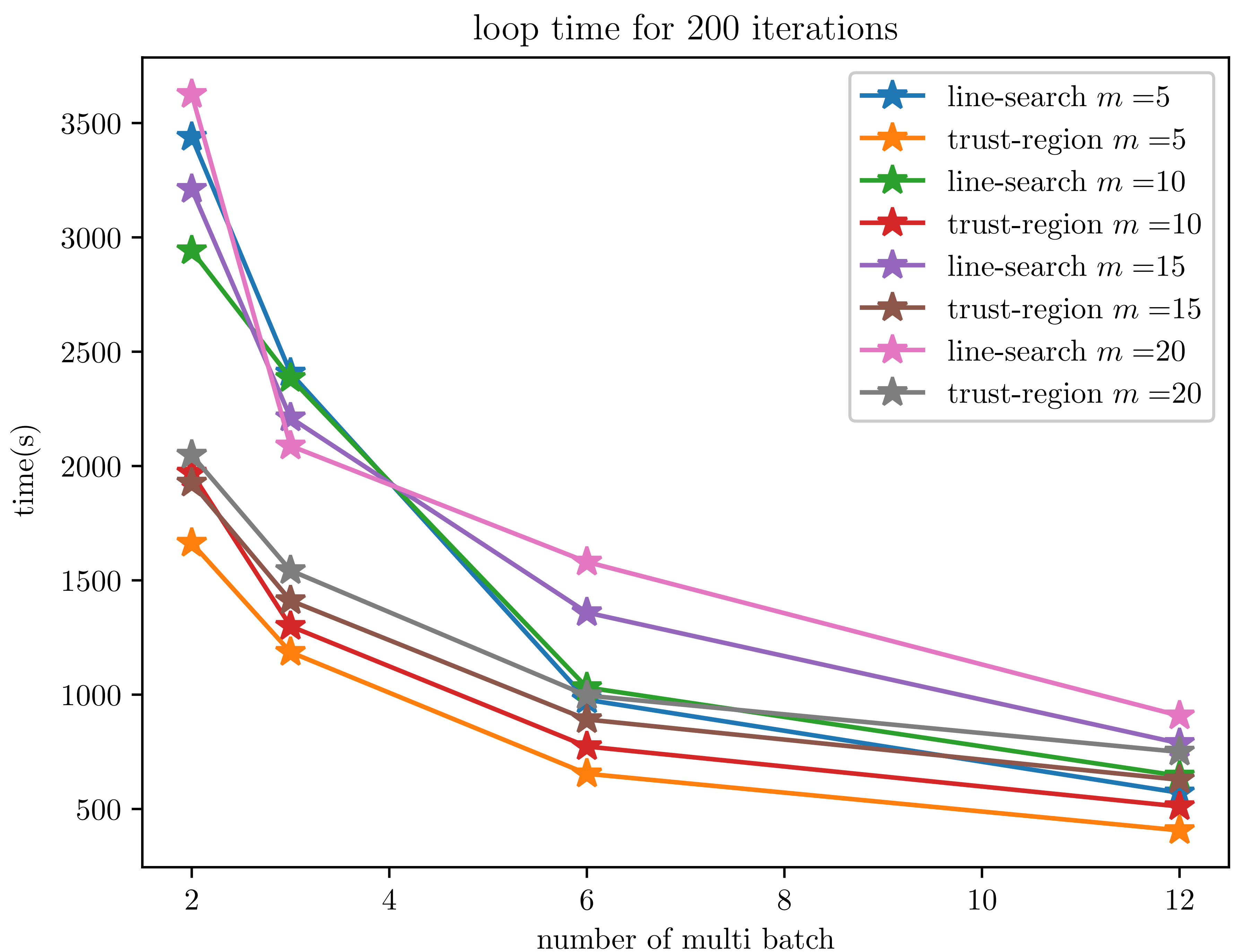
Experiment on MNIST





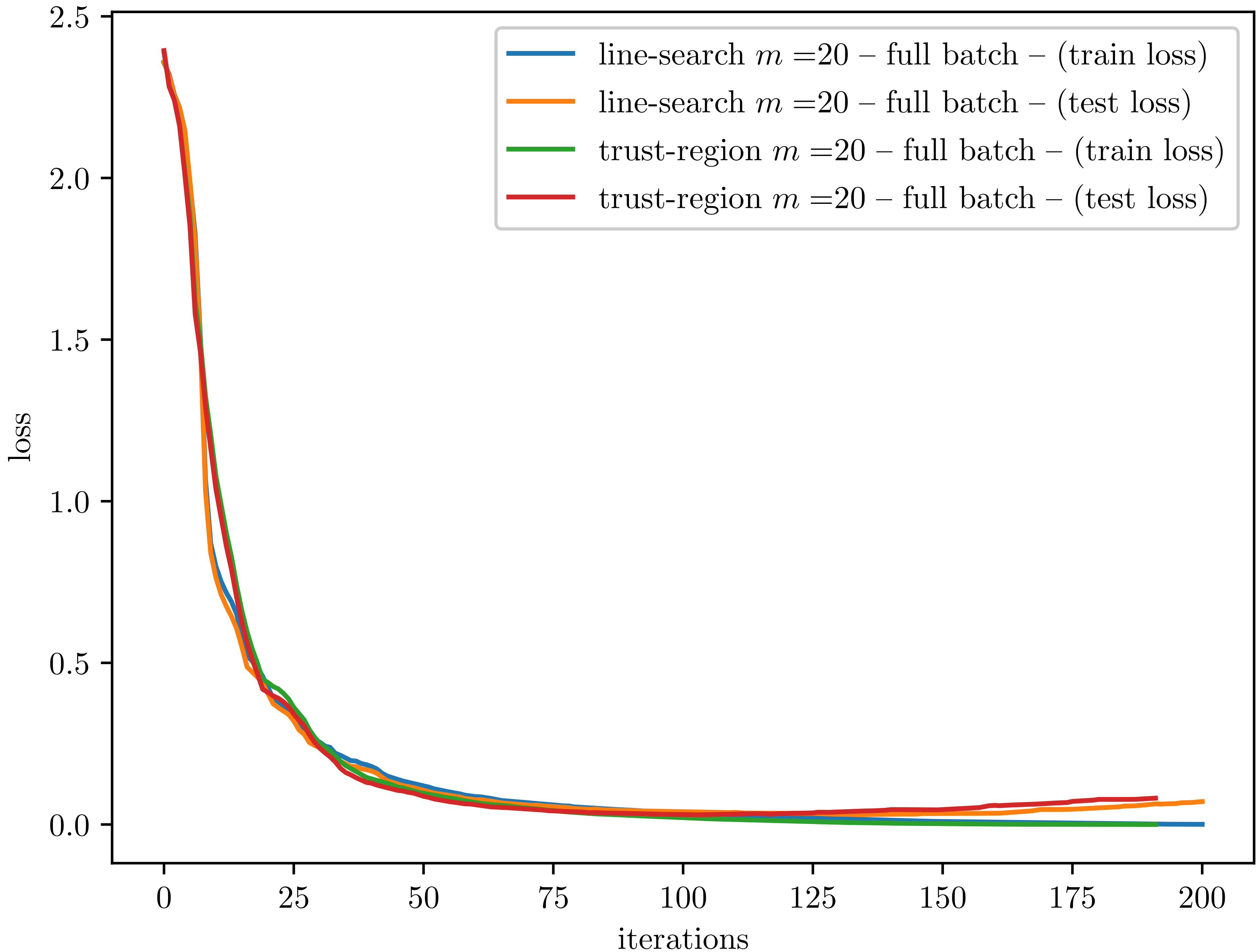
Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

Trust-region vs Line-Search Training Time - MNIST



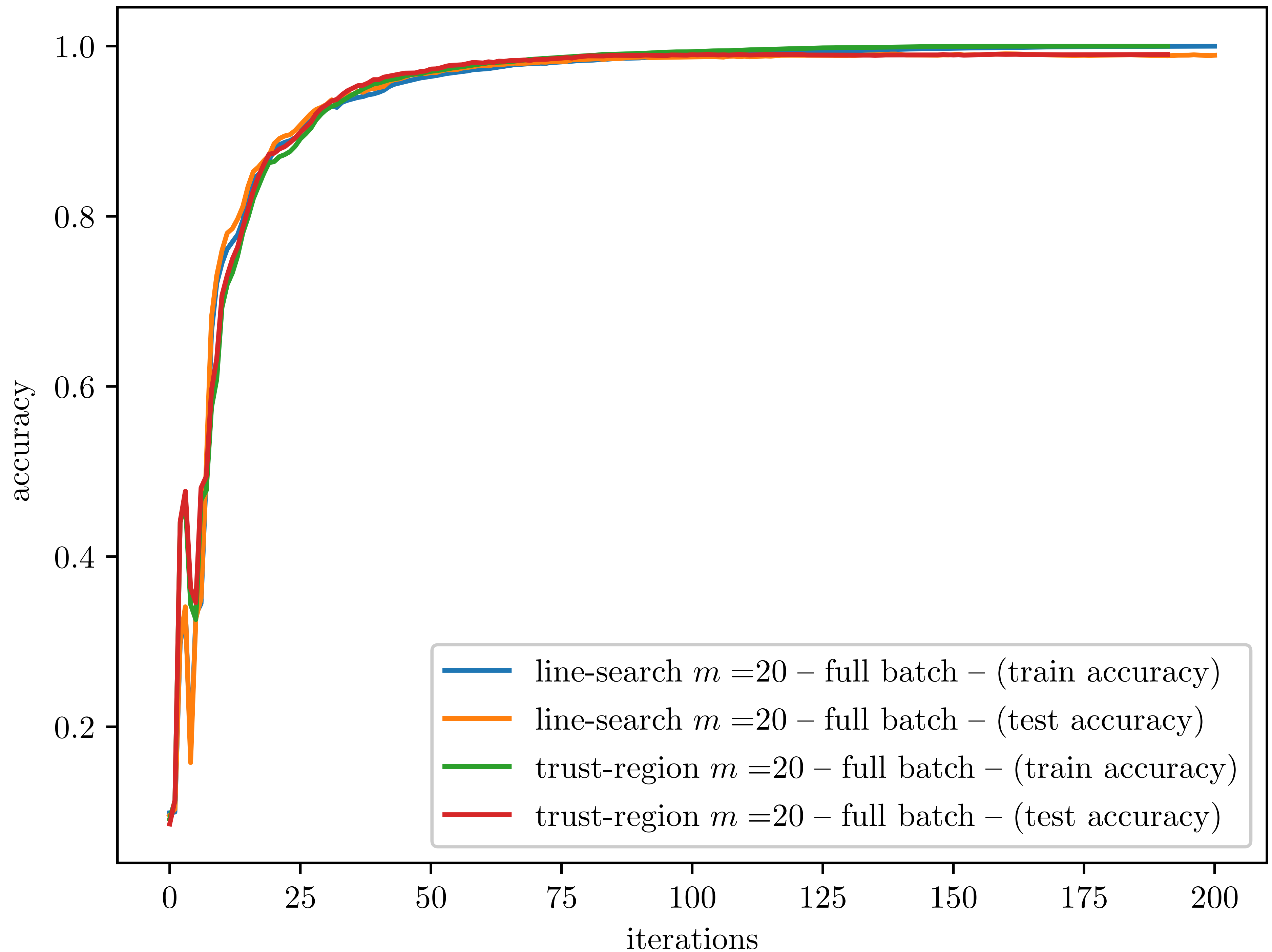
Rafati et al. (2018). Trust-Region Minimization Algorithm for Training Responses. In 26th European Signal Processing Conference, Rome, Italy.

Trust-region vs Line-Search Training/Test Loss - MNIST



Rafati et al. (2018). Trust-Region Minimization Algorithm for Training Responses. In 26th European Signal Processing Conference, Rome, Italy.

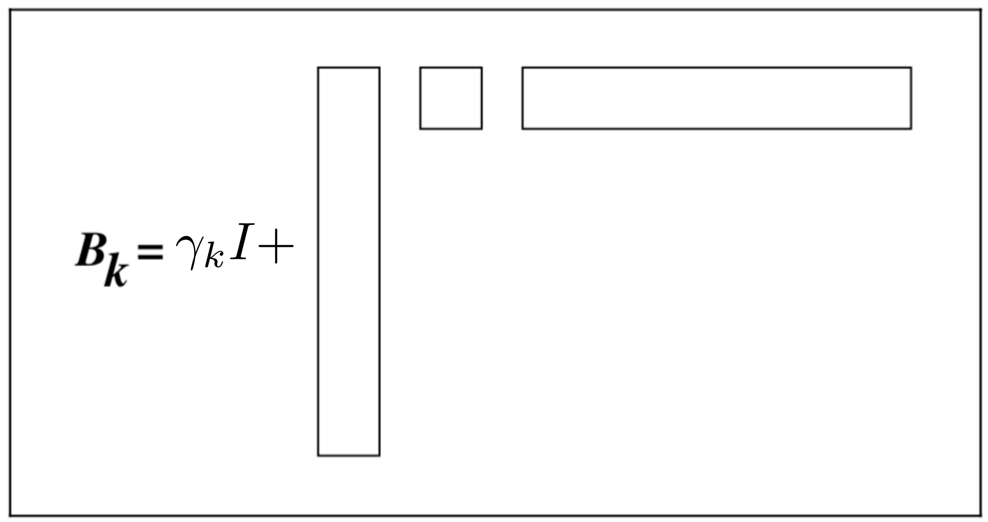
Trust-region vs Line-Search Training/Test Accuracy - MNIST



Rafati et al. (2018). Trust-Region Minimization Algorithm for Training Responses. In 26th European Signal Processing Conference, Rome, Italy.

Initialization Methods for L-BFGS

$$B_k = \gamma_k I + \Psi_k M_k \Psi_k^T$$



How can we improve the performance by choosing a proper initialization

$$B_0 = \gamma_k I, \quad \gamma_k > 0$$

Initialization Method I

$$B_0 = \gamma_k I, \quad \gamma_k > 0$$

Method I is used in literature. This method finds the best initial positive matrix that best represent the Hessian, i.e. spectral estimate of Hessian.

$$\gamma_k = \frac{y_{k-1}^T y_{k-1}}{s_{k-1}^T y_{k-1}}$$

J. Nocedal and S. J. Wright. (2006). Numerical Optimization. 2nd ed. New York. Springer.

Initialization Methods II,III

For a quadratic function,

$$\mathcal{L}(w) = \frac{1}{2} w^T H w + g^T w$$

We have

$$\nabla^2 \mathcal{L}(w) = H$$

Therefore *H* satisfies the Secant Equations

$$S_k^T H S_k = S_k^T Y_k$$

Erway et al. (2018). “Trust-Region Algorithms for Training Responses: Machine Learning Methods Using Indefinite Hessian Approximations,” ArXiv e-prints.

Initialization Methods II,III

Consider the compact representation of L-BFGS

$$B_k - \gamma_k I = \Psi_k M_k \Psi_k^T$$

We should bound γ_k in order to avoid a false curvature condition in $\Psi_k M_k \Psi_k^T$.

By solving a general Eigenvalue problem

$$Az = \lambda Bz$$

$$\gamma_k < \lambda_{\min}$$

Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

Initialization Method II

$$B_0 = \gamma_k I, \quad \gamma_k > 0$$

The general Eigenvalue problem

$$(L_k + D_k + L_k^T)z = \lambda S_k^T S_k z$$

LDU decomposition

$$S_k^T Y_k = L_k + D_k + U_k$$

Choose a positive value less than the smallest eigenvalue

$$\gamma_k < \lambda_{\min}$$

Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

Initialization Method III

$$B_k = \gamma_k I + \Psi_k M_k \Psi_k^T$$

$$\Psi_k = \begin{bmatrix} \gamma_k S_k & Y_k \end{bmatrix}, \quad M_k = \begin{bmatrix} -\gamma_k S_k^T S_k & -L_k \\ -L_k^T & D_k \end{bmatrix}^{-1}$$

The general Eigenvalue problem considering nonlinearity

$$A^* z = \lambda B^* z$$

$$A^* = L_k + D_k + L_k^T - S_k^T Y_k \tilde{D} Y_k^T S_k - \gamma_{k-1}^2 (S_k^T S_k \tilde{A} S_k^T S_k),$$
$$B^* = S_k^T S_k + S_k^T S_k \tilde{B} Y_k^T S_k^T + S_k^T Y_k \tilde{B}^T S_k^T S_k.$$

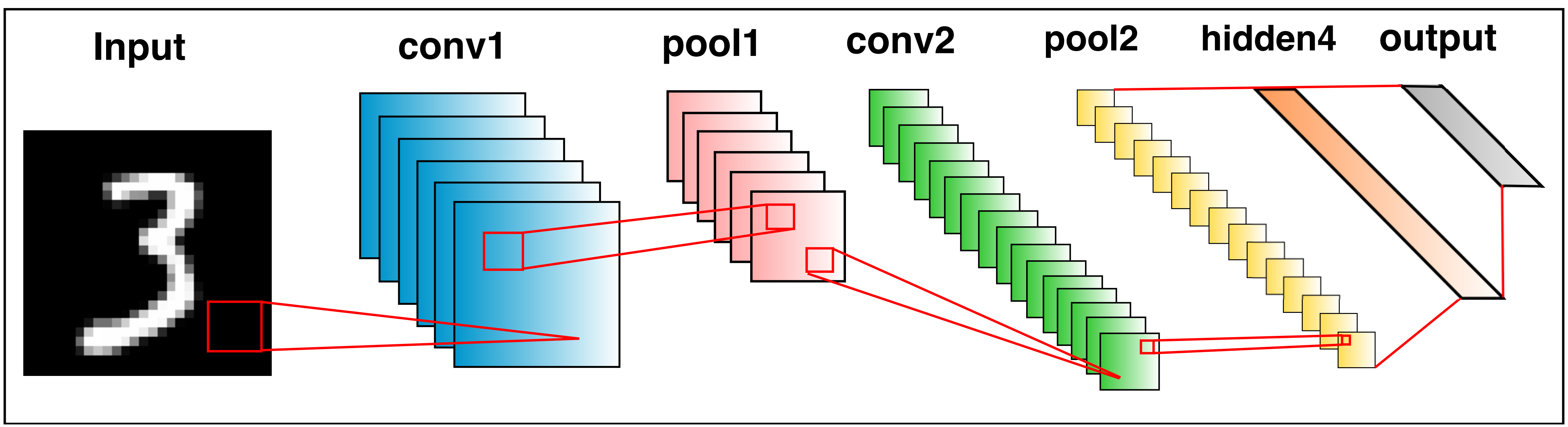
Choose a positive value less than the smallest eigenvalue

$$\gamma_k < \lambda_{\min}$$

Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

Experiment on MNIST

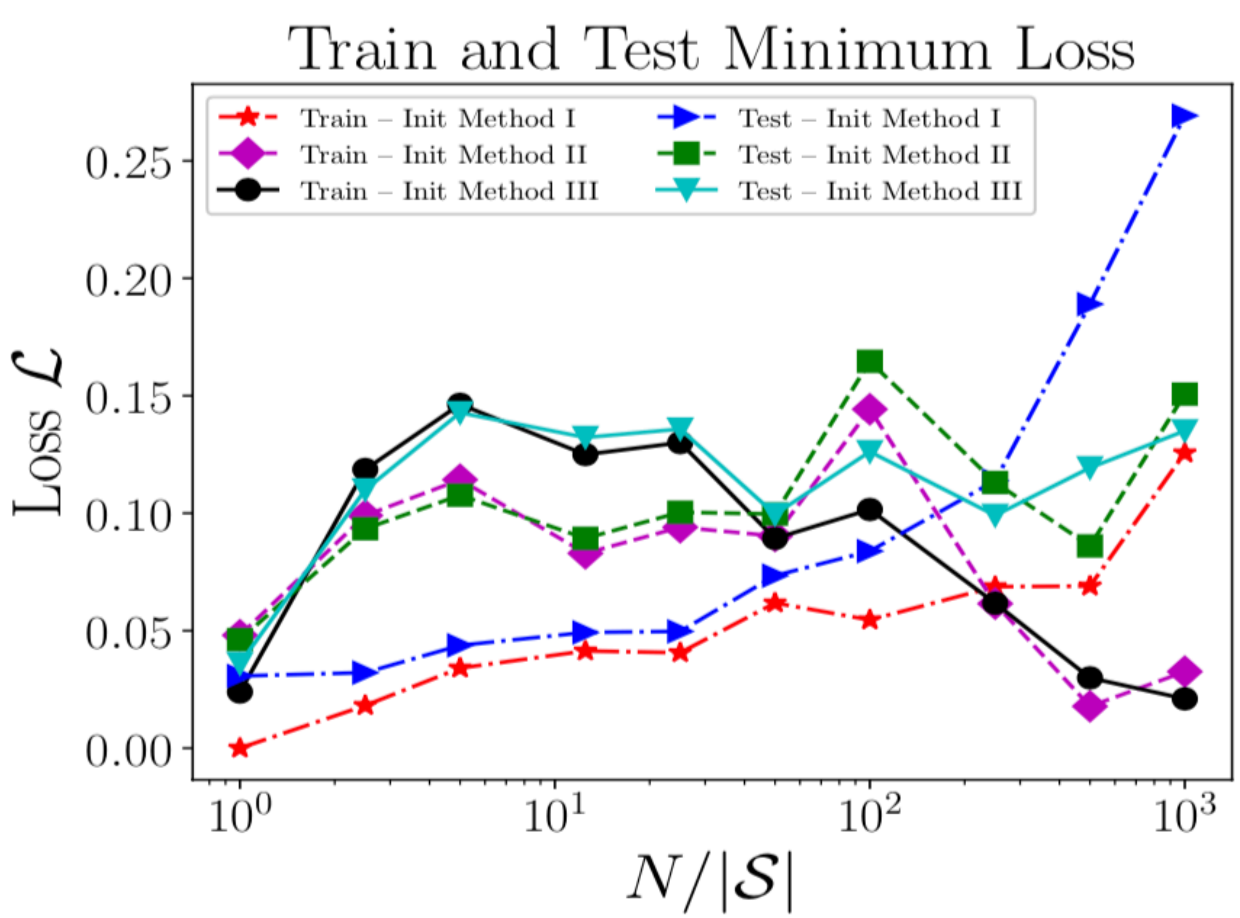
Initialization	Source	Formula
Method I	Solve the optimization problem: $\gamma_k = \arg\min_\gamma \|B_0^{-1} y_{k-1} - s_{k-1}\|_2^2$	$\gamma_k = \max\left\{1, \frac{y_{k-1}^T y_{k-1}}{s_{k-1}^T y_{k-1}}\right\}$
Method II	Solve the generalized eigenvalue problem: $(L_k + D_k + L_k^T)z = \lambda S_k^T S_k z$	$\gamma_k = \begin{cases} \max\{1, 0.9\lambda_{\min}\} & \text{if } \lambda_{\min} > 0, \\ \text{Use Method I} & \text{if } \lambda_{\min} \le 0. \end{cases}$
Method III	Solve the generalized eigenvalue problem: $A^* z = B^* \lambda z$	$\gamma_k = \begin{cases} \max\{1, 0.9\lambda_{\min}\} & \text{if } \lambda_{\min} > 0, \\ \text{Use Method I} & \text{if } \lambda_{\min} \le 0. \end{cases}$



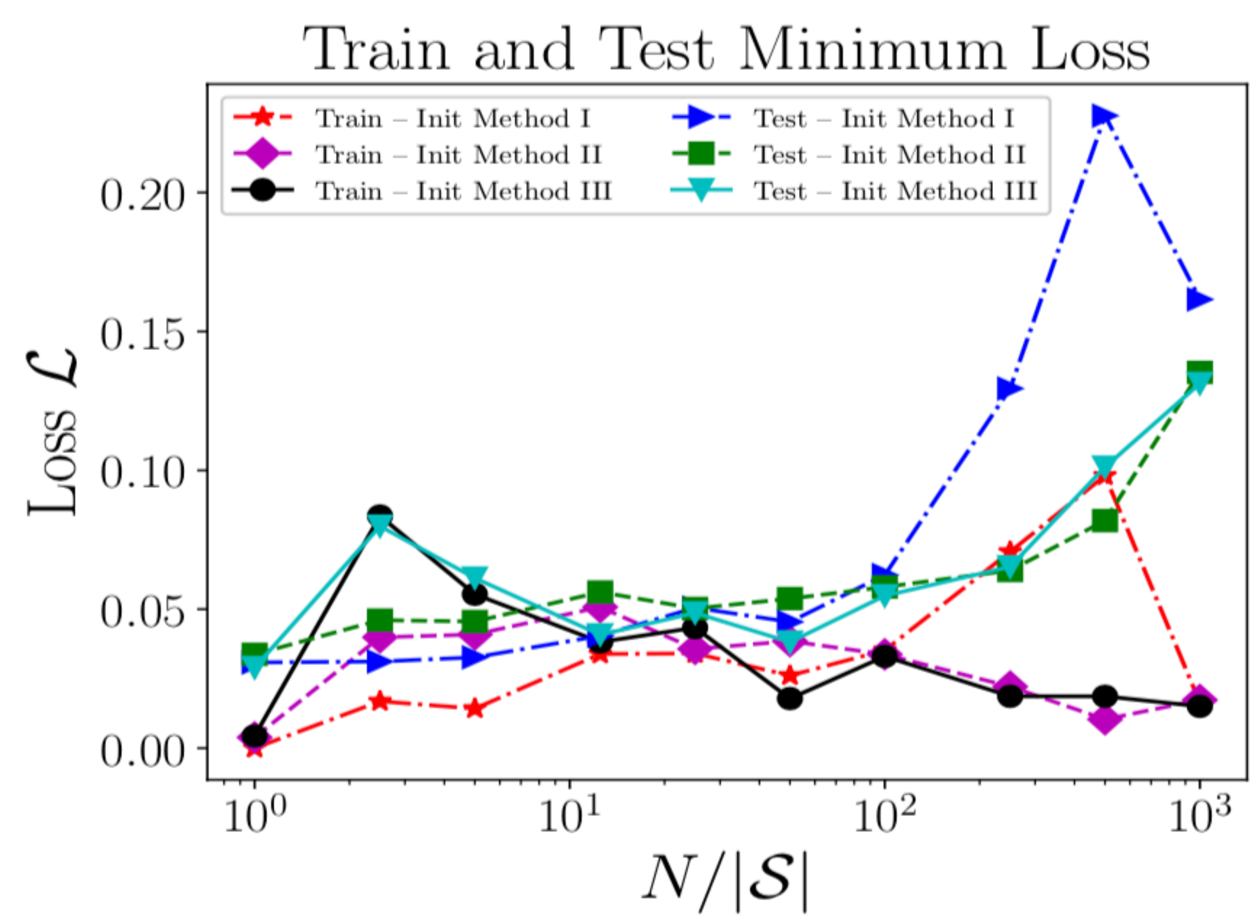
Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

Training/Test Loss

$m = 10$



$m = 20$

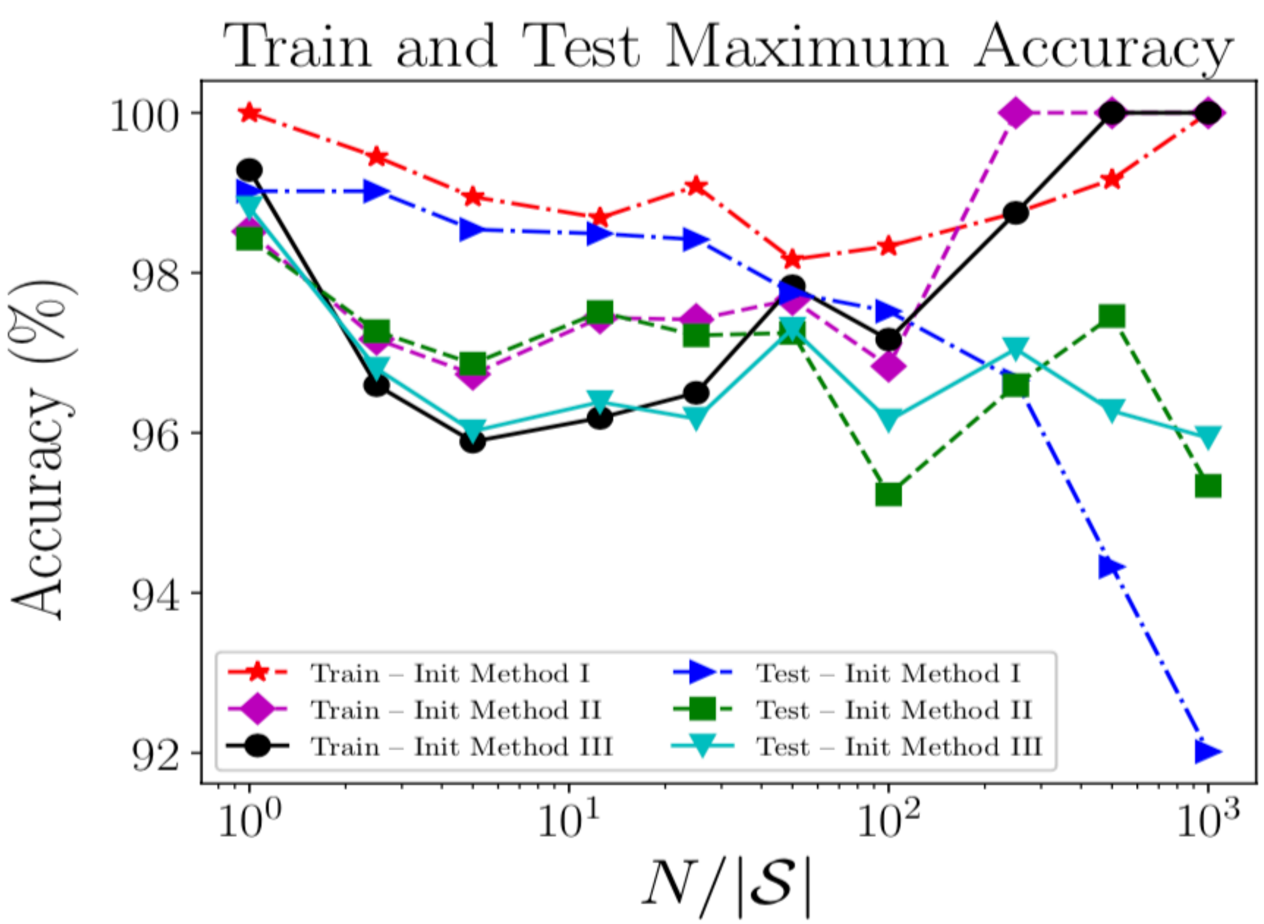


Full ⟷ **Stochastic** **Full** ⟷ **Stochastic**

Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

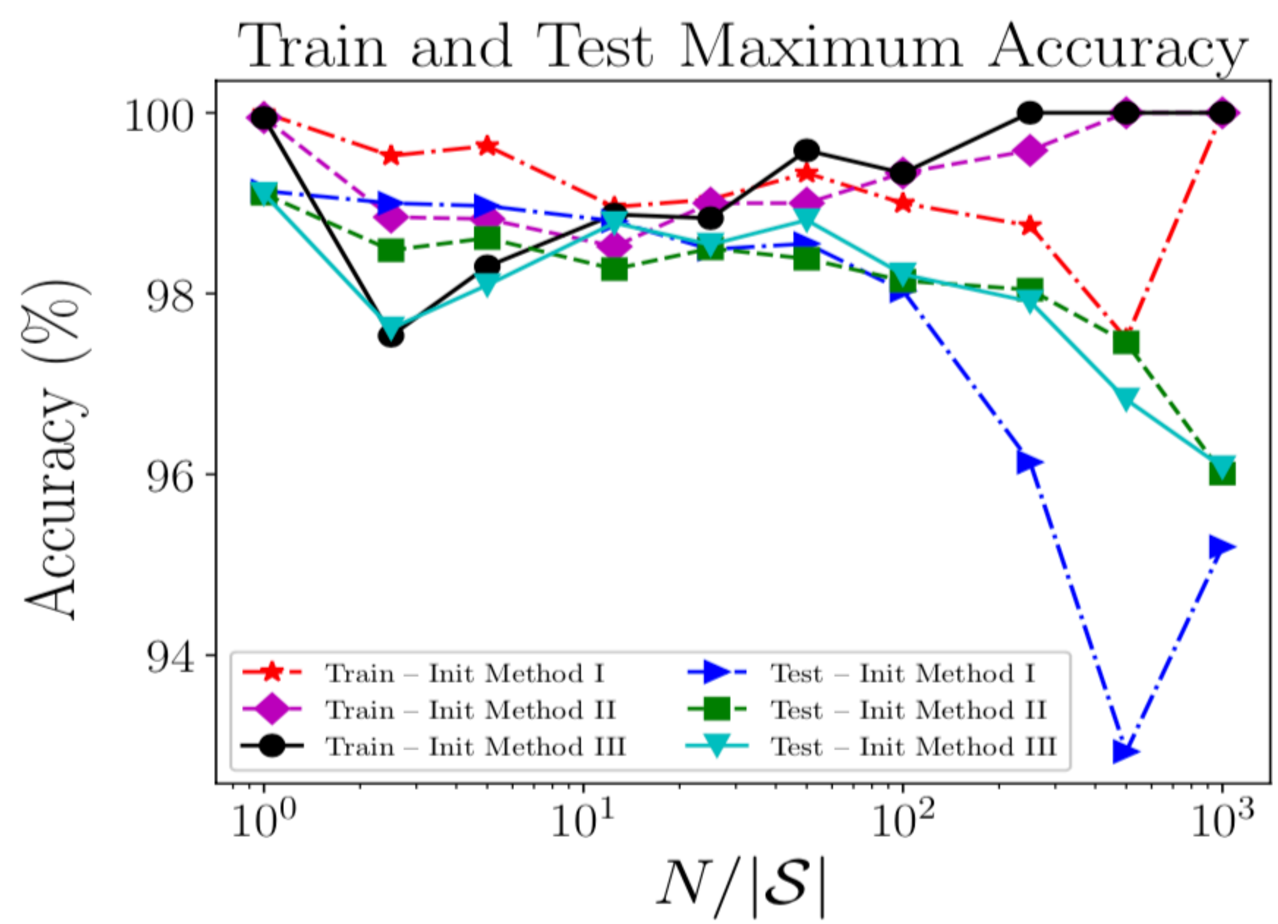
Training/Test Accuracy

$m = 10$



Full ⟷ **Stochastic**

$m = 20$

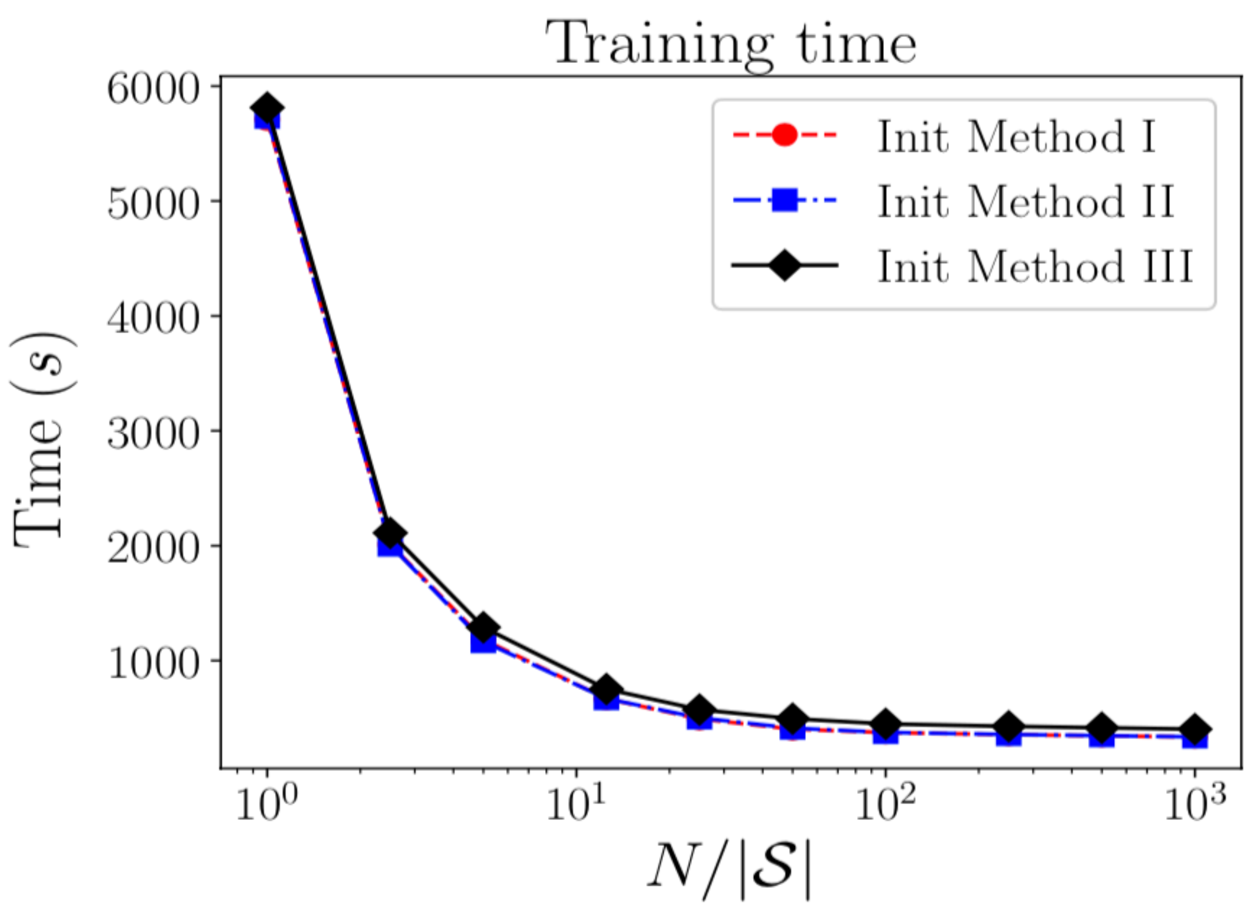


Full ⟷ **Stochastic**

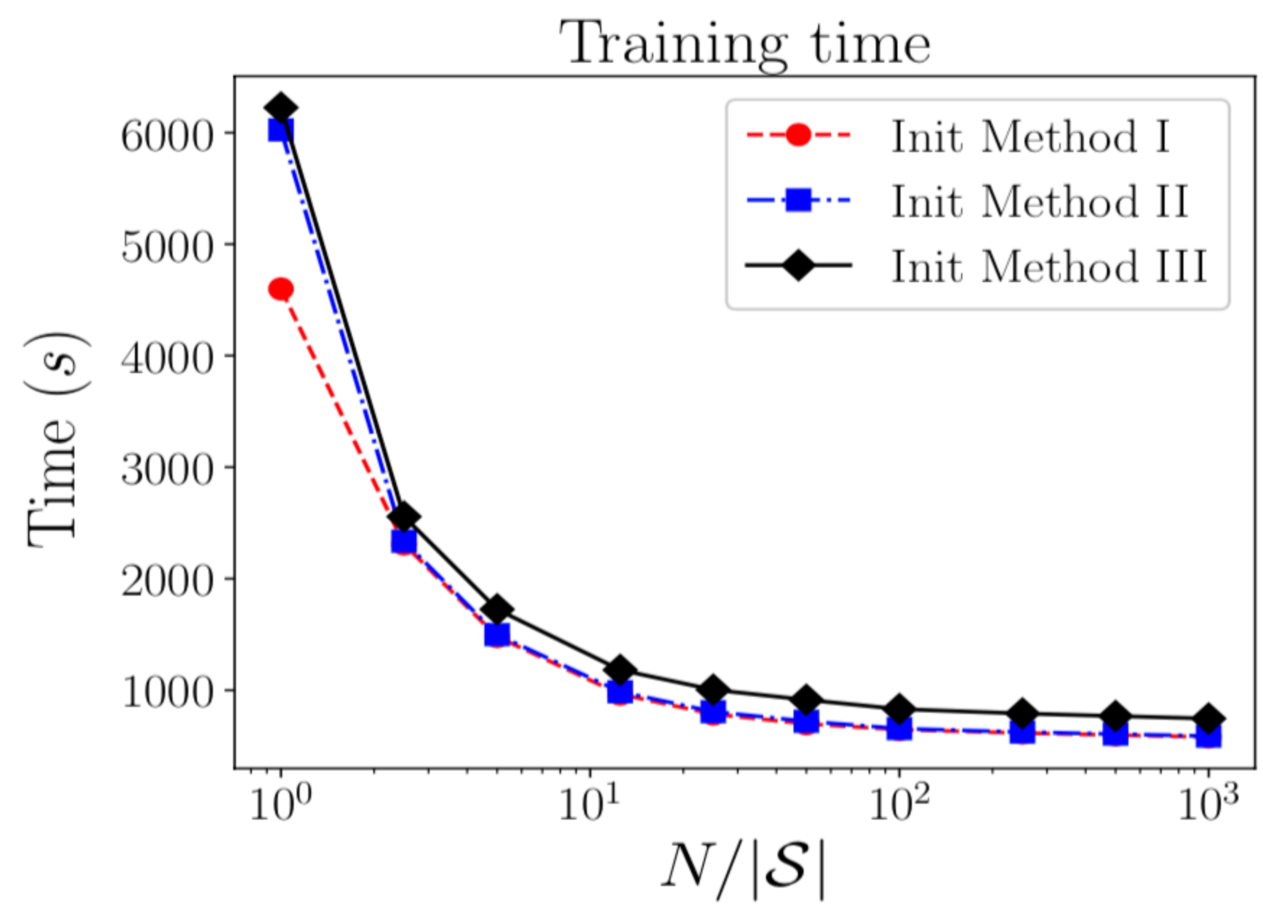
Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

Training Time

$m = 10$



$m = 20$



Full ⟷ **Stochastic**

Full ⟷ **Stochastic**

Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. (ICMLA 2018), Orlando, Florida.

Conclusions

- We proposed and demonstrated an optimization method based on the limited memory quasi-Newton method known as L-BFGS as an alternative to the gradient descent methods typically used to train deep neural networks.
- We considered both line-search and trust-region frameworks. Trust-region is much faster than line-search since it doesn't require satisfying the sufficient decrease and curvature conditions.
- Trust-region radius can shrink or expand and is more flexible in choosing alternative search direction since when the secant condition doesn't satisfy.
- We investigated three methods for initializing L-BFGS matrices in a trust-region optimization framework in order to avoid false curvature conditions. The proposed methods require solving a cheap general eigenvalue problems and offer upper bounds for initial values.

Future Work

- The true Hessian is indefinite, and using indefinite quasi-Newton matrices, like Symmetric Rank 1 (SR1), or Full Broyden Class (FBC) within trust-region methods might lead to better convergence properties. We will study these methods in a future work.
- We will examine these optimization methods on larger deep learning problems, such as image recognition, healthcare, etc.

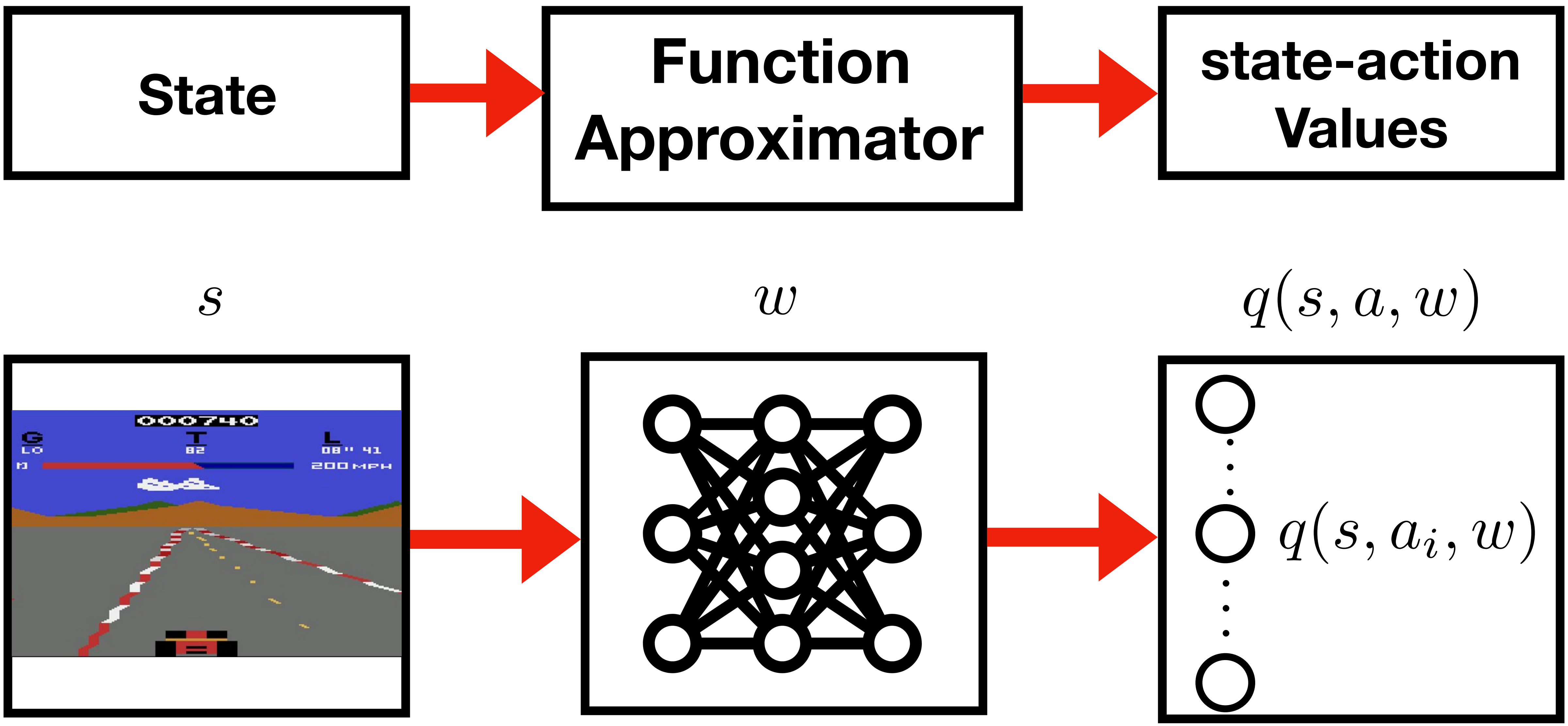
Publications

- Jacob Rafati, Omar DeGuchy and Roummel F. Marcia (2018). Trust-Region Minimization Algorithm for Training Responses (TRMinATR): The Rise of Machine Learning Techniques. In 26th European Signal Processing Conference, Rome, Italy.

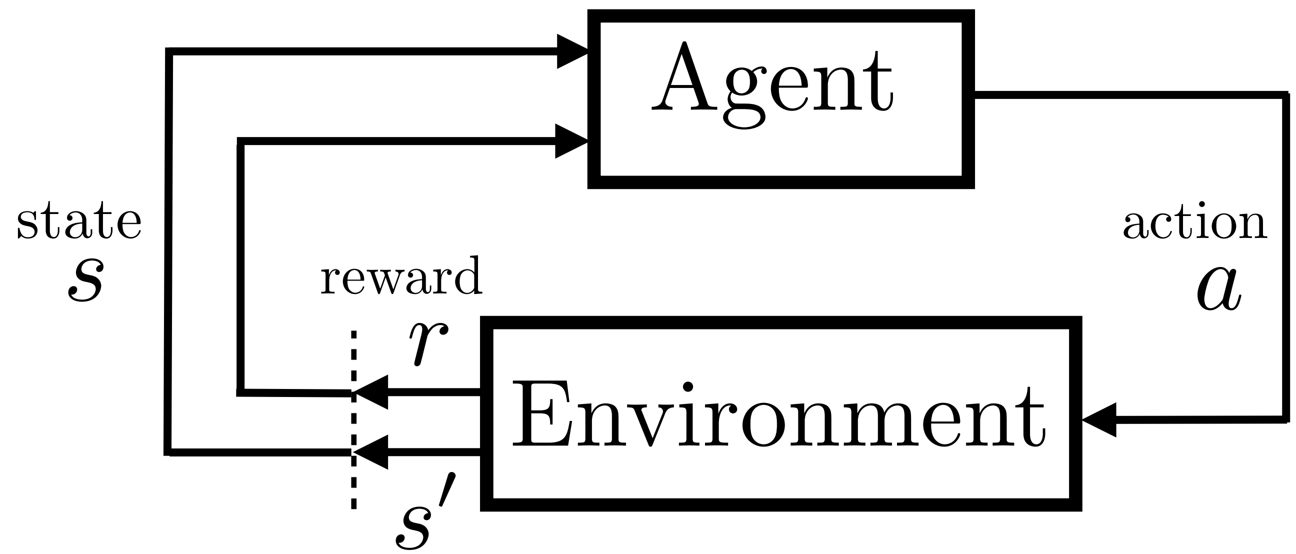
- Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. In 17th IEEE International Conference on Machine Learning and Applications (ICMLA 2018), Orlando, Florida.

Problem 4.
Quasi-Newton Optimization in Deep RL

Generalization



Empirical Risk Minimization in Deep RL



$$\min_{w \in \mathbb{R}} \mathcal{L}(w) \triangleq \frac{1}{N} \sum_{e \in \mathcal{D}} \left(r + \gamma \max_{a'} q(s', a'; w) - q(s, a; w)\right)^2$$

$\mathcal{D} = \{(s, a, s', r)\}$ is Agent's Experiences Memory.

Sutton and Barto (2017). Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, USA, 2nd edition.

Computing gradients

O_{k-1}	O_k

$$O_k = \mathcal{D}$$

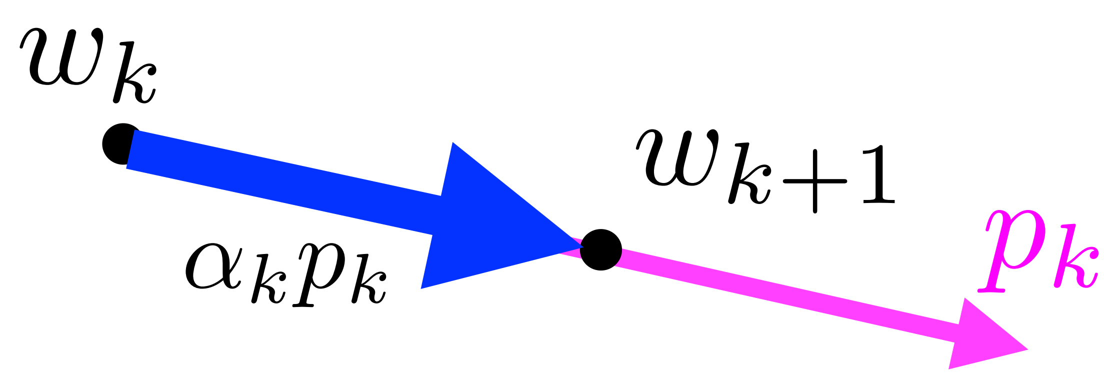
O_k	O_{k+1}

To efficiently compute gradient and gradient differences, we can reuse the previous iteration's gradient

$$g_k = \frac{1}{2}\left(\nabla\mathcal{L}(w_k)^{(O_{k-1})} + \nabla\mathcal{L}(w_k)^{(O_k)}\right)$$

$$y_k = \nabla f(w_{k+1})^{(O_k)} + \nabla f(w_k)^{(O_k)}$$

Line Search Method



Search step, the minimizer of quadratic model

$$p_k = B_k^{-1} \nabla \mathcal{L}(w_k)$$

There is a formula to update inverse of BFGS matrices, $H_k = B_k^{-1}$

$$H_{k+1} = \left(I - \frac{y_k s_k^T}{y_k^T s_k}\right) H_k \left(I - \frac{s_k y_k^T}{y_k s_k^T}\right) + \frac{y_k y_k^T}{y_k s_k^T}$$

Instead we satisfy the sufficient decrease and curvature conditions known as

$$\text{Wolfe conditions}$$

$$\mathcal{L}(w_k + \alpha_k p_k) \leq \mathcal{L}(w_k) + c_1 \alpha_k \nabla \mathcal{L}(w_k)^T p_k$$

$$\nabla \mathcal{L}(w_k + \alpha_k p_k)^T p_k \geq c_2 \nabla f(w_k)^T p_k$$

J. Nocedal and S. J. Wright. (2006). Numerical Optimization. 2nd ed. New York. Springer.

L-BFGS Two-Loop Recursion

Algorithm L-BFGS two-loop recursion.

$\mathbf{q} \leftarrow g_k = \nabla \mathcal{L}(w_k)$
for $i = k-1, \ldots, k-m$ **do**
  $\alpha_i = \frac{\mathbf{s}_i^T q}{\mathbf{y}_i^T \mathbf{s}_i}$
  $\mathbf{q} \leftarrow \mathbf{q} - \alpha_i \mathbf{y}_i$
end for
$\mathbf{r} \leftarrow H_0 q$
for $i = k-1, \ldots, k-m$ **do**
  $\beta = \frac{\mathbf{y}_i^T r}{\mathbf{y}_i^T \mathbf{s}_i}$
  $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{s}_i(\alpha_i - \beta)$
end for
return $-\mathbf{r} = -H_k g_k$

Computational time is $4mn$.

Convergence Analysis

Theorem. Under these assumptions:

$\mathcal{L}(w)$ is strongly convex, and twice differentiable.

$\forall w, \ \exists \lambda, \Lambda > 0$ such that $\lambda I \preceq \nabla^2 \mathcal{L}(w) \mathcal{L} \preceq \Lambda I$, i.e. Hessian is bounded.

$\forall w, \ \exists \eta > 0$ such that $\|\nabla \mathcal{L}(w)\|^2 \leq \eta^2$, i.e. Gradient does not explode.

$\exists \lambda', \Lambda' > 0 \ such \ that \ \lambda' I \preceq H_k \preceq \Lambda' I.$

If we bound the step size, $\alpha < \frac{1}{2\lambda\lambda'}$, we can compute an upper-bound for loss

$$\mathcal{L}(w_k) - \mathcal{L}(w^*) \leq (1 - 2\alpha\lambda\lambda')^k [\mathcal{L}(w_0) - \mathcal{L}(w^*)] + [1 - (1 - 2\alpha\lambda\lambda')^k] \frac{\alpha^2 \Lambda'^2 \Lambda \eta^2}{4\lambda'\lambda}$$

i.e. the loss function will converge to a neighborhood of the optimal loss.

Jacob Rafati, and Roummel F. Marcia (2019). Quasi-Newton Optimization in Deep Q-Learning for Playing ATARI Games. arXiv e-print (arXiv:1811.02693).

Value Optimality

Theorem.

If α_k satisfies

$$\left|1 - \alpha_k \nabla Q_k^T H_k \nabla Q_k + \frac{\alpha_k^2}{2} \nabla Q_k^T H_k \nabla^2 Q_k H_k \nabla \mathcal{L}_k \right| < 1,$$

then

$$\|Q_{k+1} - Q^*\|_\infty < \|Q_k - Q^*\|_\infty.$$

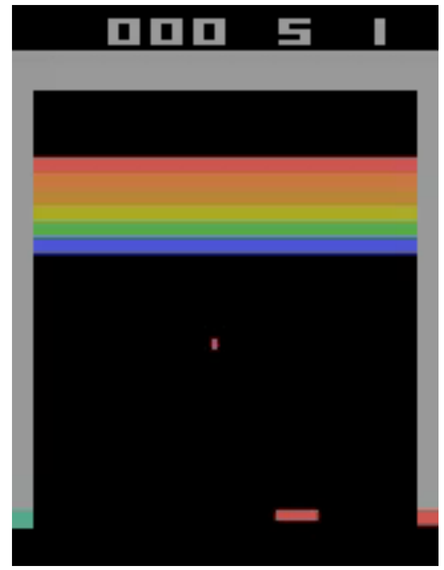
By applying another condition on step size,
the value function theoretically converges to optimal values.

Jacob Rafati, and Roummel F. Marcia (2019). Quasi-Newton Optimization in Deep Q-Learning for Playing ATARI Games. arXiv e-print (arXiv:1811.02693).

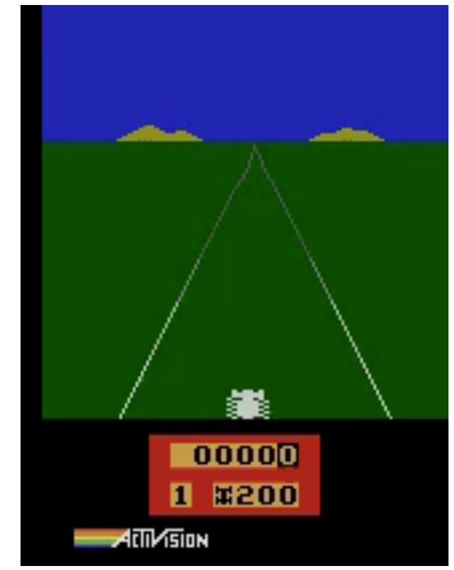
Experiments - ATARI 2600

Beam Rider

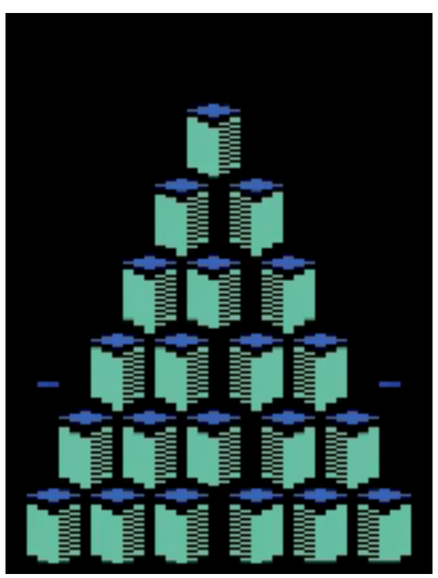
Breakout



Enduro

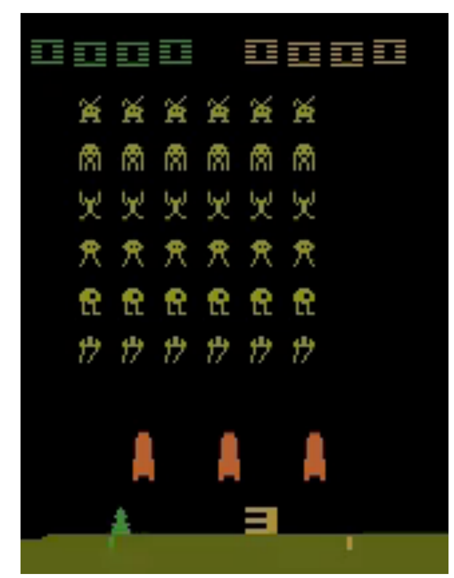


Qbert



Seaquest

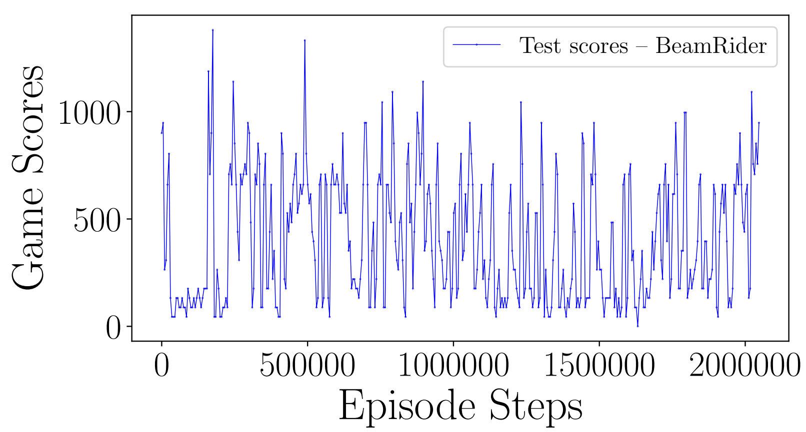
Space Invadors



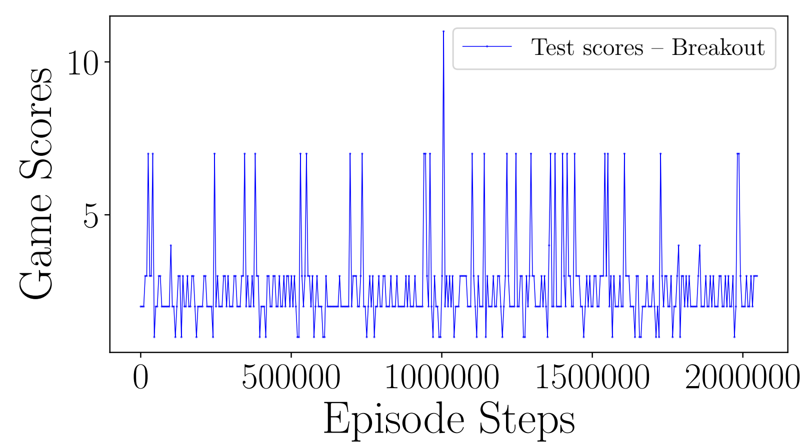
We used one NVIDIA Tesla K40 GPU with 12GB GDDR5 RAM on MERCED clusters.

Test Scores

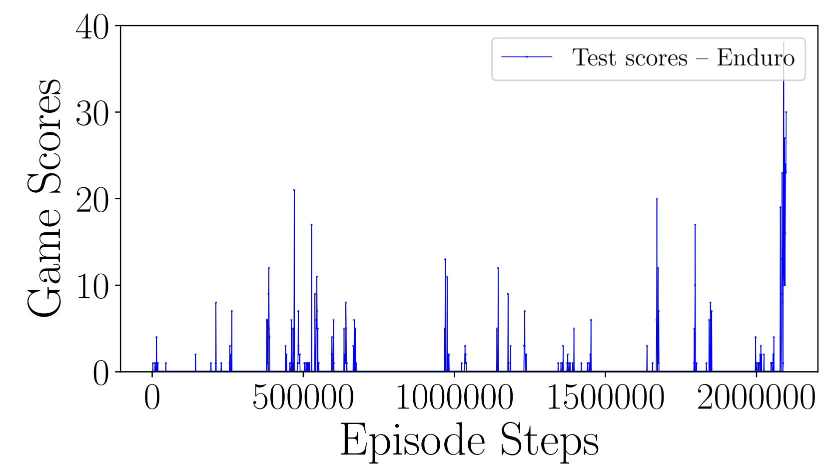
Beam Rider



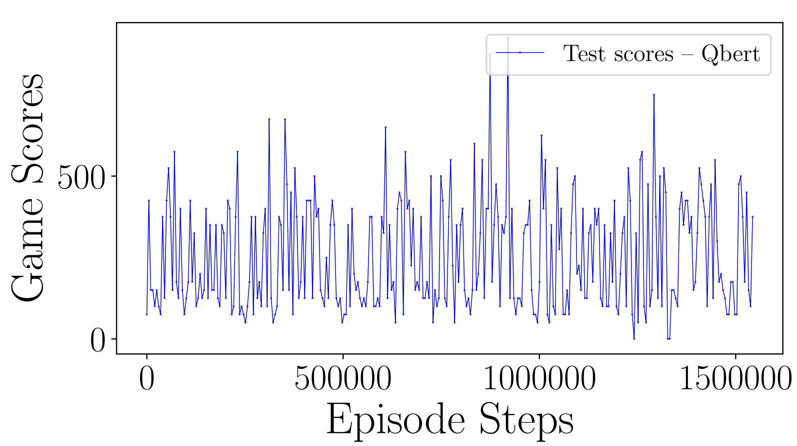
Breakout



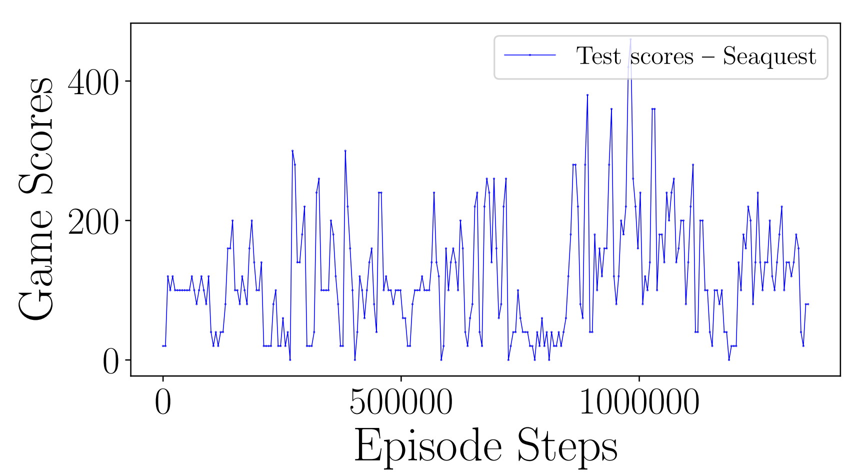
Enduro



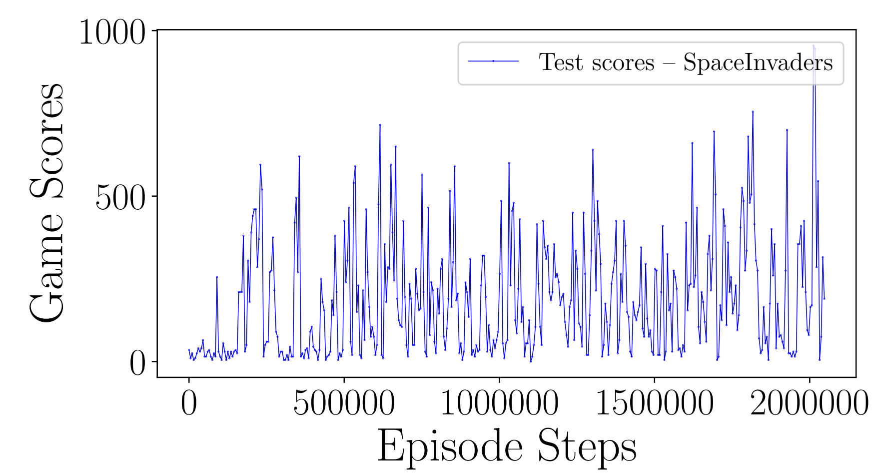
Qbert



Seaquest



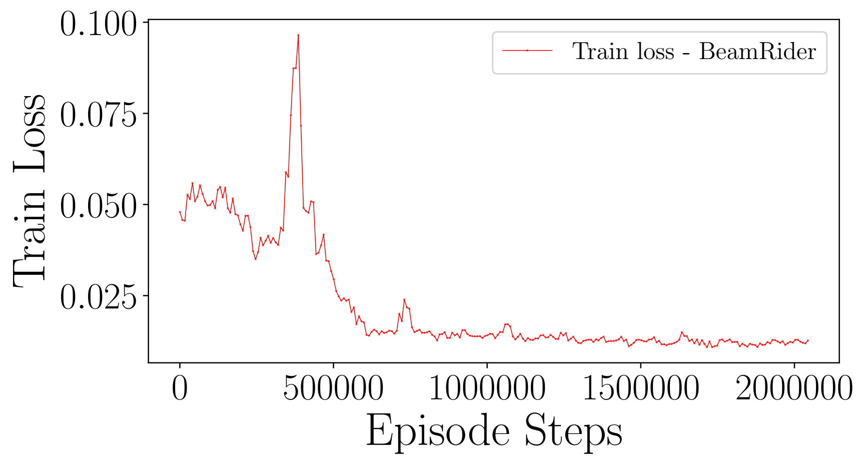
Space Invadors



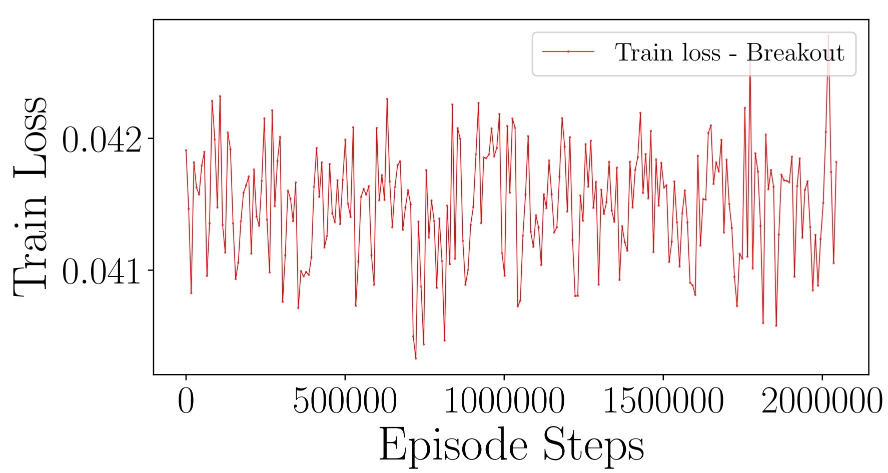
Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693).

Train Loss

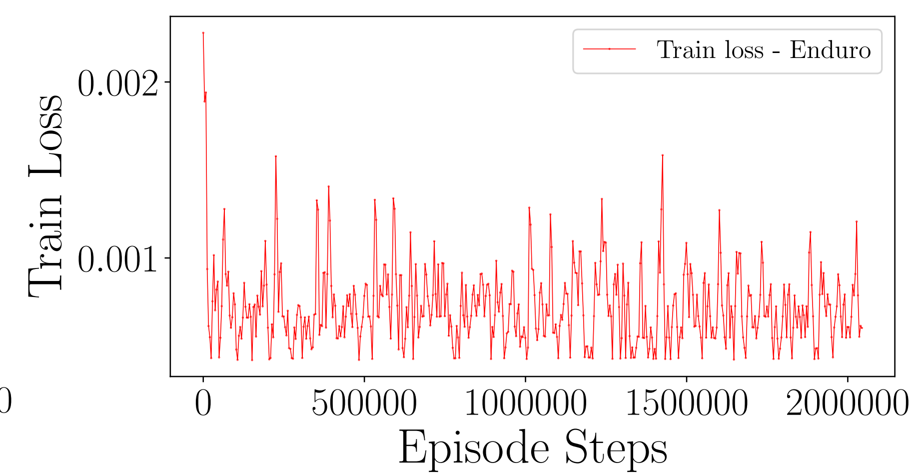
Beam Rider



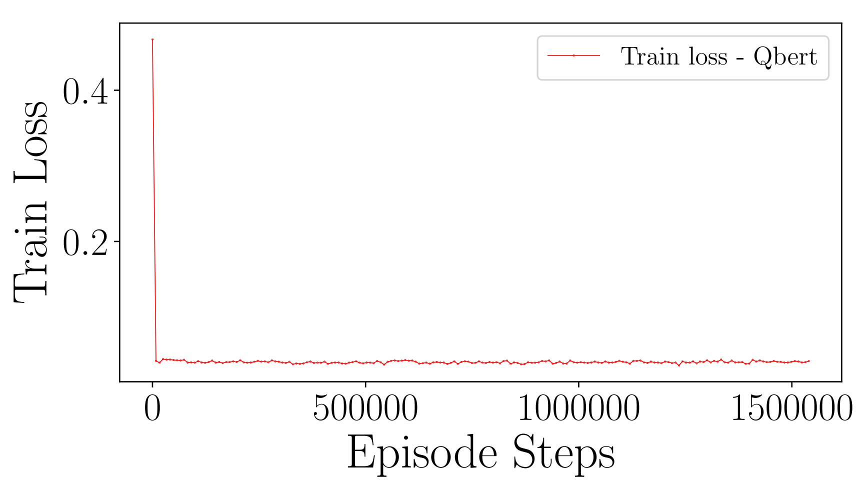
Breakout



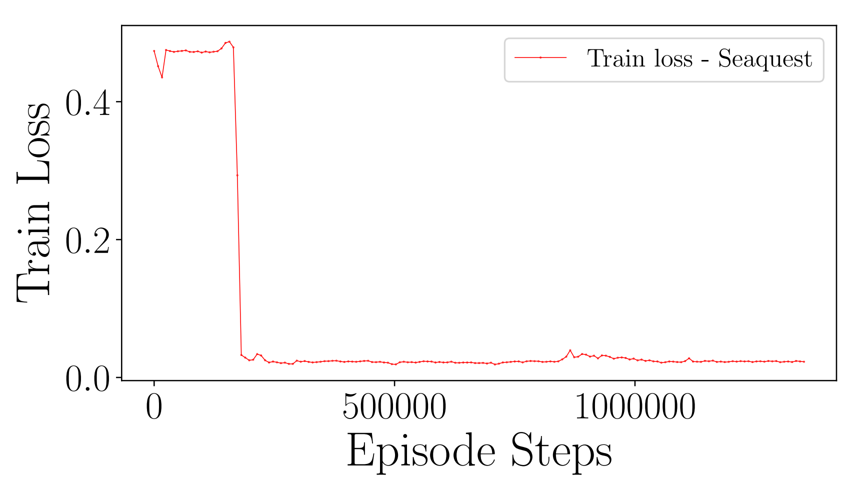
Enduro



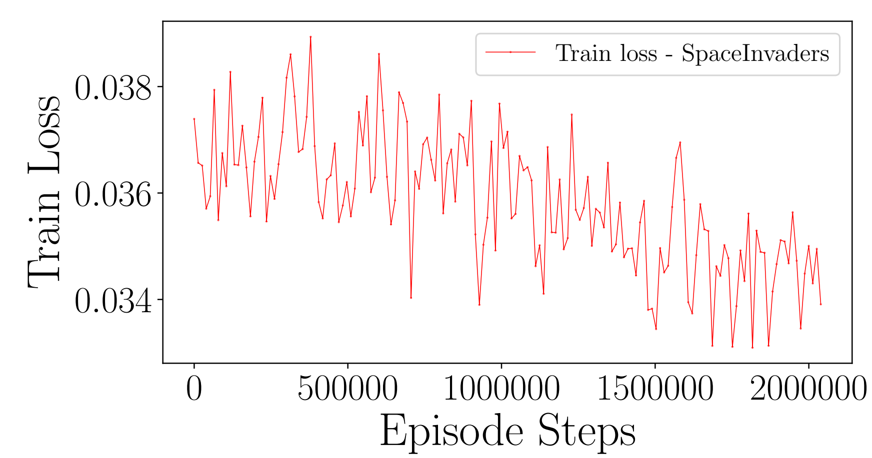
Qbert



Seaquest

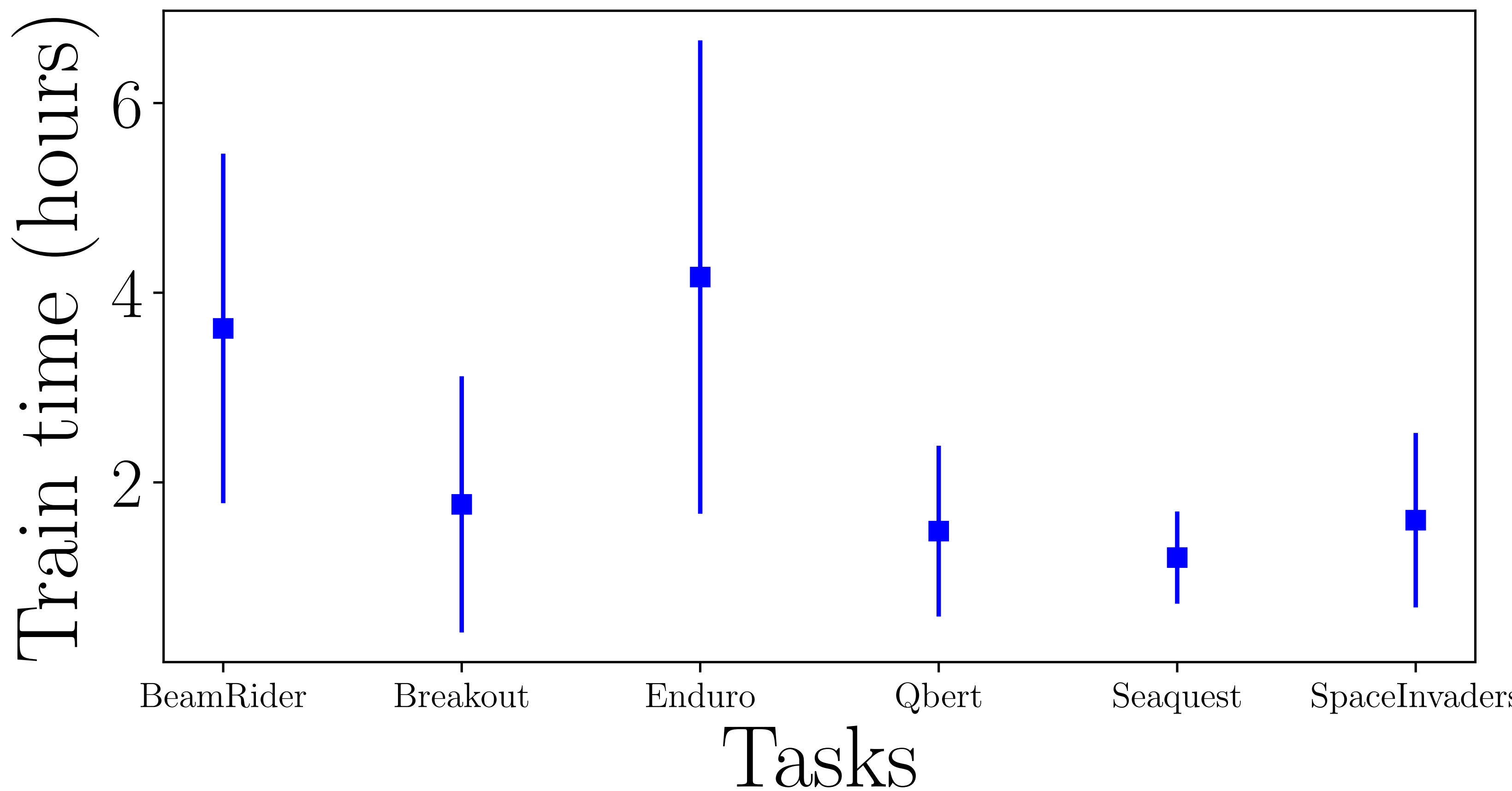


Space Invadors



Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693).

Training Time for different batch and different L-BFGS memory



$$\text{Coefficient of Variation} = \frac{\text{std}}{\text{mean}} \qquad \text{CV} \approx 50\%$$

Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693).

Training Time — L-BFGS vs SGD

Method	BR	BO	EO	Q*B	SQ	SI
SGD	4	2	8	8	8	1
Our method	4	2	4	2	1	1

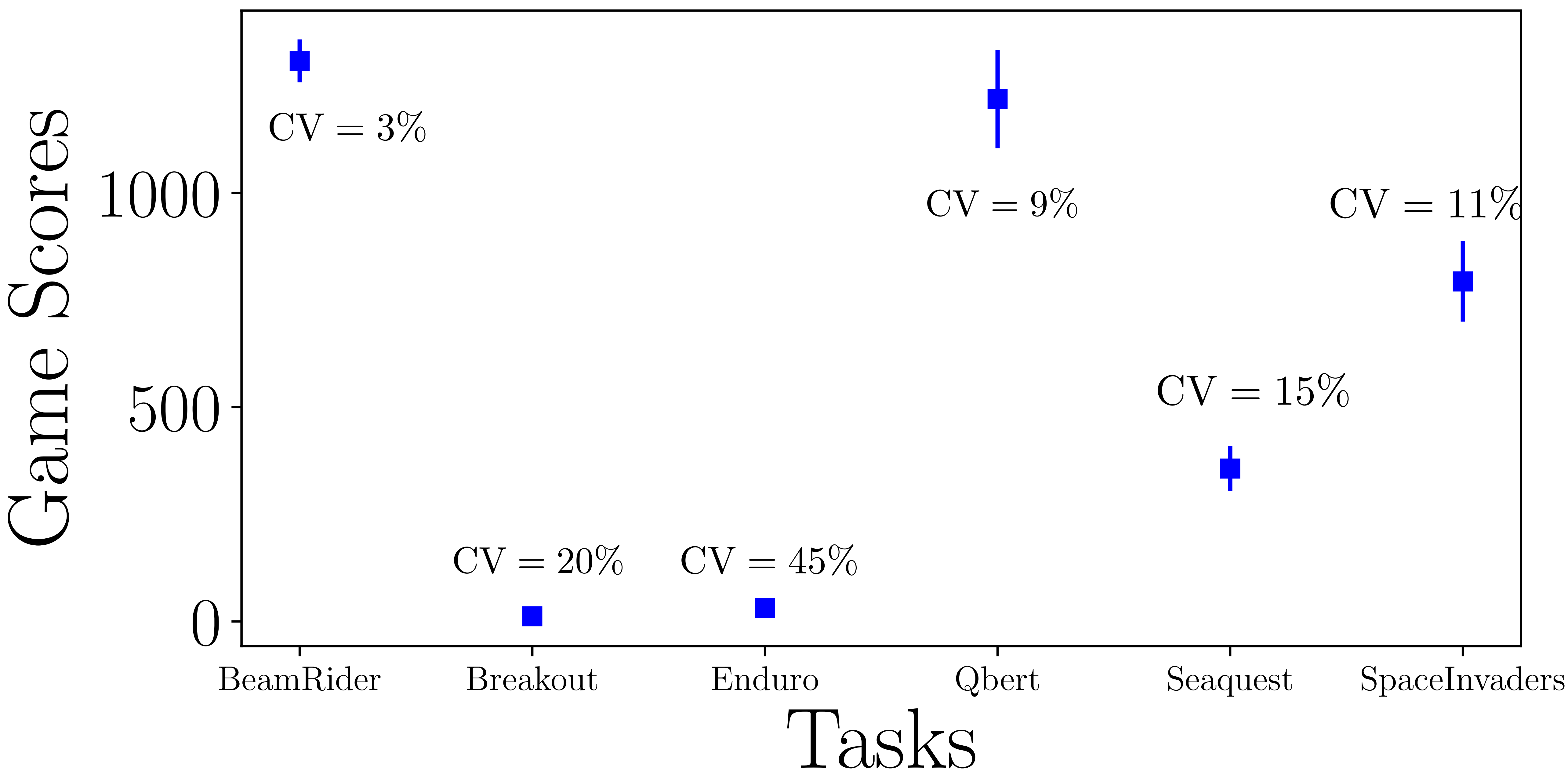
$$\frac{\text{Cost of Algorithm}}{\text{Cost of DQN}} = \frac{(L/b)(zbn + 4mn)}{(L_s/f)(b_s n)} = \frac{L}{L_s}\left(\frac{fz}{b_s} + \frac{4fm}{bb_s}\right).$$

For same number of Q-learning steps, $\frac{L}{L_s} = 1$

$$\frac{\text{Computation time of our algorithm}}{\text{Computation time of SGD algorithm}} \approx 0.63$$

Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693).

Max Scores for different batch and different L-BFGS memory



Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693).

Best Scores

Method	BR	BO	EO	Q*B	SQ	SI
Random	354	1.2	0	157	110	179
Human	7456	31	368	18900	28010	3690
Sarsa (Bellemare et al., 2013)	996	5.2	129	614	665	271
Contingency (Bellemare et al., 2012)	1743	6	159	960	723	268
HNeat Pixel (Hausknecht et al., 2014)	1332	4	91	1325	800	1145
DQN (Mnih et al., 2013)	4092	168	470	1952	1705	581
TRPO, Single path (Schulman et al., 2015)	1425	10	534	1973	1908	568
TRPO, Vine (Schulman et al., 2015)	859	34	431	7732	7788	450
SGD	804	13	2	1325	420	735
Our method	1380	18	49	1525	600	955

Bellemare et al. (2012). Investigating contingency awareness using ATARI 2600 games. AAAI.
Bellemare et al. (2013). The arcade learning environment: An evaluation platform for general agents. Journal of Artificial Intelligence Research, 47:253–279.
Hausknecht et al. (2014). A neuroevolution approach to general ATARI game playing. IEEE Transactions on Computational Intelligence and AI in Games, 6(4):355–366.
Schulman et al. (2015). Trust region policy optimization. ICML.
Mnih, et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540):529–533.
Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693)

Conclusions

- We proposed and implemented a novel optimization method based on line search limited-memory BFGS for deep reinforcement learning framework.
- Due to the nonconvex and nonlinear loss function in deep reinforcement learning, our numerical experiments show that using the curvature information in computing the search direction leads to a more robust convergence.
- Our proposed deep L-BFGS Q-Learning method is designed to be efficient for parallel computations in GPU.
- Our method is much faster than the existing methods in the literature, and it is memory efficient since it does not need to store a large experience replay memory.

Future Work

- We will consider the optimization methods based on trust-region methods.
- We will study methods based on indefinite quasi-Newton, like SR1 and Full Broyden Class.
- We will study the Hessian-free optimization methods for model-free RL within conjugate gradient framework.

Publications

- Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693).

Concluding Remarks

Main Contributions

- **In my Ph.D. dissertation**, I have investigated biologically inspired techniques for learning useful representations for model-free reinforcement learning, as well as numerical optimization methods for improving learning.
- **Learning Sparse Representations in RL:** I have implemented efficient algorithms that incorporate a kind of lateral inhibition into artificial neural network layers, driving these machine learning systems to produce sparse conjunctive internal representations.
- **Learning Representations in Model-Free HRL:** I have implemented a novel model-free method for subgoal discovery using incremental unsupervised learning over a small memory of the most recent experiences of the agent. When combined with an intrinsic motivation learning mechanism, and temporal abstraction, this method learns subgoals and skills together, based on experiences in the environment.

Main Contributions

- **Optimization Methods in Deep RL:** I have contributed to the design of an efficient optimization method, based on the L-BFGS quasi-Newton method within line search strategy, offering it as an alternative to SGD methods.
- **Quasi-Newton Optimization in Deep Learning:** I have implemented efficient algorithms based on L-BFGS optimization method suitable for general deep learning applications to improve the quality of representation learning, as well as convergence properties. I have implemented L-BFGS optimization under both rust-region and line-search frameworks, and I have produced evidence that this approach is efficient for deep learning problems such as classification and regression from big data.
- **Improving L-BFGS Initialization for trust-region methods:** I have explored various initialization methods for the L-BFGS matrices, within a trust-region framework.

Future Work

- **Learning Representations in Model-Based RL:** In model-based approach, the agent can incorporate planning into learning process by learning a model of the environment. I will study methods for learning representations of the agent's state within model-based reinforcement learning framework.
- **Model selection in RL:** I will study different families of function approximators for the value function, and investigate their effects on learning representations of the agent's state.
- **Formal Convergence and Optimality Analysis:** Proofs on effectiveness of the proposed methods in this dissertation (as well as majority of the deep learning and the deep RL literature) rely on empirical results on some numerical simulations, which are very time consuming. I will attempt to study formal convergence analysis in order to compute upper-bounds, and lower-bounds of each proposed method in order to analyze the limits and power of each method.
- **Real world applications:** I will investigate the effectiveness of the proposed methods on real-world applications such as robotics, autonomous driving, etc.

Publications from Ph.D. Dissertation

1. Jacob Rafati, David C. Noelle. (2019). Unsupervised Subgoal Discovery Method for Learning Hierarchical Representations. In 7th International Conference on Learning Representations, ICLR 2019 Workshop on "Structure & Priors in Reinforcement Learning", New Orleans, LA, USA.
2. Jacob Rafati, and David C. Noelle (2019). Learning Representations in Model-Free Hierarchical Reinforcement Learning. (Extended version of AAAI 2019 abstract). arXiv e-print (arXiv:1810.10096).
3. Jacob Rafati, and Roummel F. Marcia (2019). Deep Reinforcement Learning via L-BFGS Optimization. arXiv e-print (arXiv:1811.02693).
4. Jacob Rafati, David C. Noelle. (2019). Unsupervised Methods For Subgoal Discovery During Intrinsic Motivation in Model-Free Hierarchical Reinforcement Learning. In 33rd AAAI Conference on Artificial Intelligence (AAAI-19). Workshop on Knowledge Extraction From Games. Honolulu, Hawaii. USA.
5. Jacob Rafati, and David C. Noelle (2019). Learning Representations in Model-Free Hierarchical Reinforcement Learning. In 33rd AAAI Conference on Artificial Intelligence (AAAI-19), Honolulu, Hawaii.
6. Jacob Rafati, Omar DeGuchy and Roummel F. Marcia (2018). Trust-Region Minimization Algorithm for Training Responses (TRMinATR): The Rise of Machine Learning Techniques. In 26th European Signal Processing Conference, Rome, Italy.
7. Jacob Rafati, and Roummel F. Marcia (2018). Improving L-BFGS Initialization For Trust-Region Methods In Deep Learning. In 17th IEEE International Conference on Machine Learning and Applications (ICMLA 2018), Orlando, Florida.
8. Jacob Rafati and David C. Noelle (2017). Sparse Coding of Learned State Representations in Reinforcement Learning. In Cognitive Computational Neuroscience Conference, New York City, NY.
9. Jacob Rafati and David C. Noelle (2015). Lateral Inhibition Overcomes Limits of Temporal Difference Learning. In 37th Annual Cognitive Science Society Meeting, Pasadena, CA, USA.

Acknowledgments

David C. Noelle

Roummel F. Marcia

Jeff Yoshimi

Shawn Newsam

Marcelo Kallmann

Acknowledgement

- The Graduate Dean's Dissertation Fellowship has supported the preparation of this dissertation.

- Collaboration with Roummel F. Marcia is supported by NSF Grants CMMI 1333326, and IIS 1741490.

- I used MERCED clusters for some of the numerical computations, supported by the NSF Grant No. ACI-1429783.

- Generous conference travel funds, and the EECS Bobcat Fellowships from the School of Engineering.

Acknowledgement

- The Graduate Dean, Marjorie Zatz, the Associate Dean, Chris Kello, The staff of the Graduate Division.
- EECS faculty members.
- Tomiko Hale, Tamika Hankston, Staff of School of Engineering.
- Members of the Computational Cognitive Neuroscience Laboratory (M. Ebrahmipour, Jeff Rodny, W.S.T Clair, Angelo Kyrilov, etc)
- Omar DeGuchy, Johannes Brust, Harish Bhat, Applied Math faculty members, School of Natural Science Staff.
- Sarvani Chadalapaka, and IT staff.

Acknowledgement

https://www.jacobrafati.com

Thank you

Paper, Code, Slides:

`http://rafati.net`