Meta-Learning

[Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks]
Chelsea Finn, Pieter Abbeel, Sergey Levine. ICML 2017

[RL2: Fast Reinforcement Learning via Slow Reinforcement Learning]
Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, Pieter Abbeel. ICLR 2017

Presented by Chen Jinfan

[Meta-Learning is to tell] agents to learn how to learn new tasks faster by reusing previous experience, rather than considering each new task in isolation.

–Chelsea Finn

Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks

Intuition

- Maximising the 'sensitivity' of the loss function of tasks w.r.t parameters
- By pre-training parameters for all tasks
- Sensitivity is high if small local changes lead to large improvement for tasks

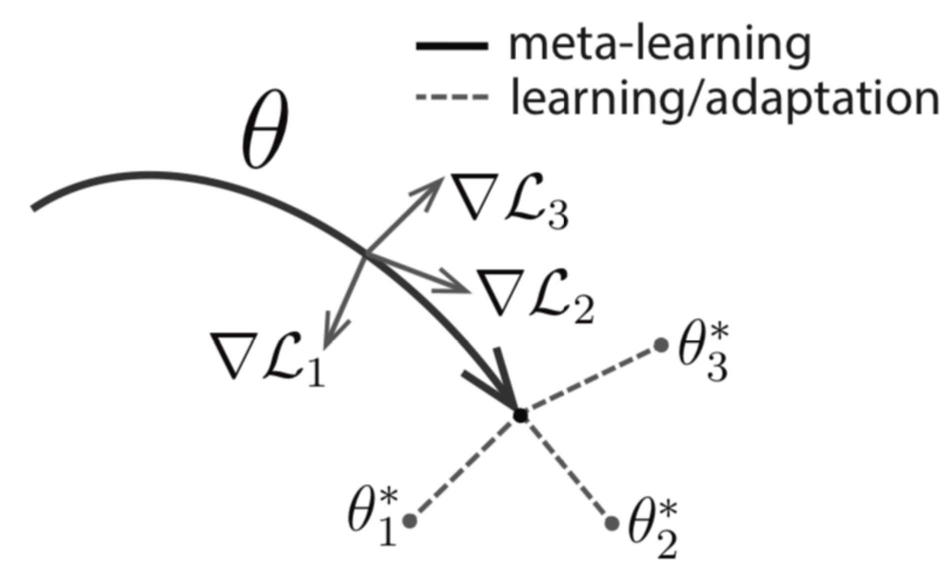


Figure 1. Diagram of our model-agnostic meta-learning algorithm (MAML), which optimizes for a representation θ that can quickly adapt to new tasks.

Algorithm

- The parameters after gradient decent updates on task i

$$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta).$$

- Our objective function (for a distribution of tasks) is

$$\min_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}) = \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)})$$

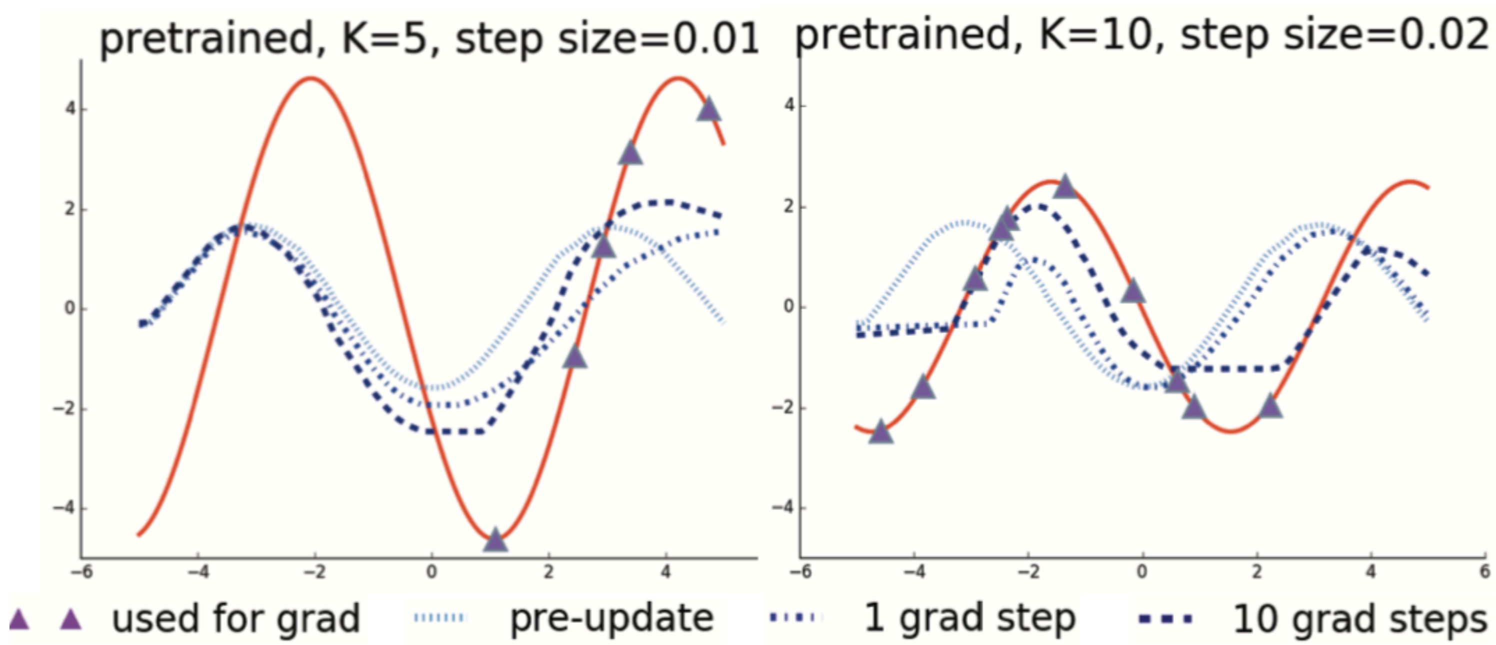
- So one gradient update w.r.t.our objective is

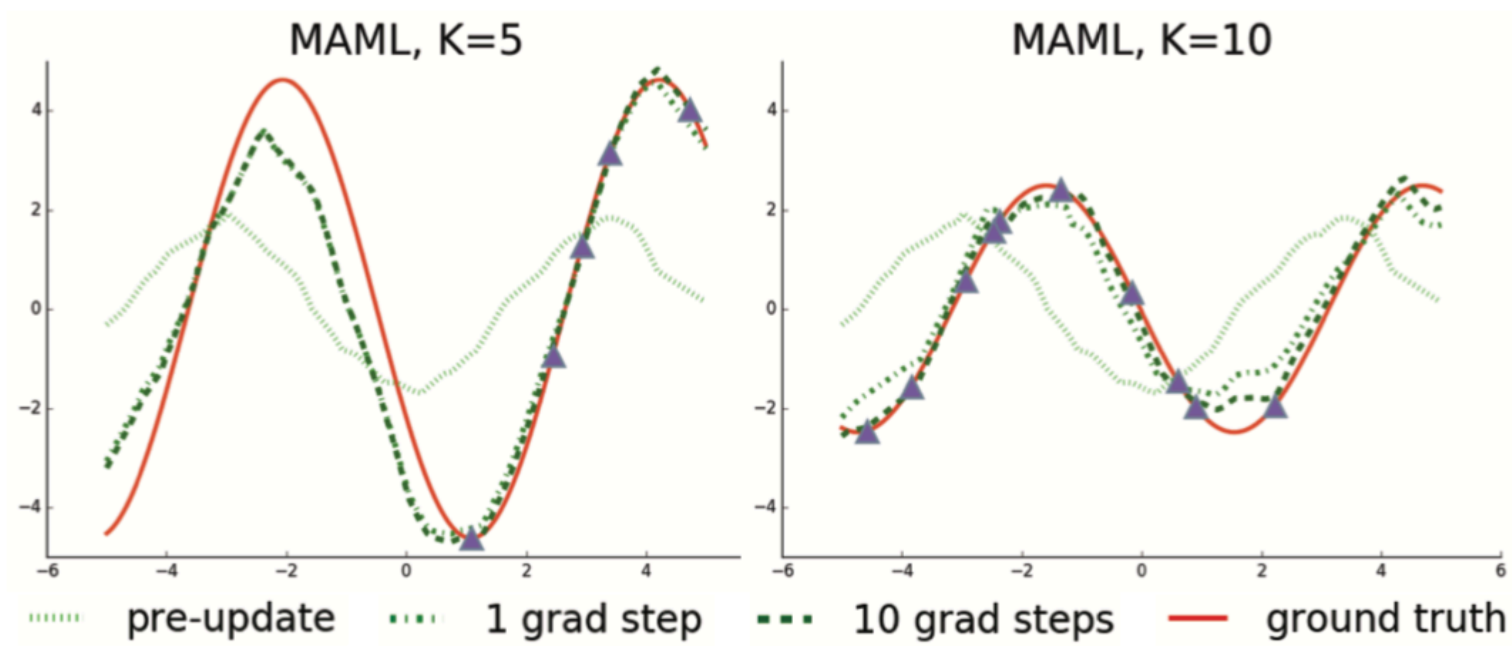
$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$$

Regression Experiment

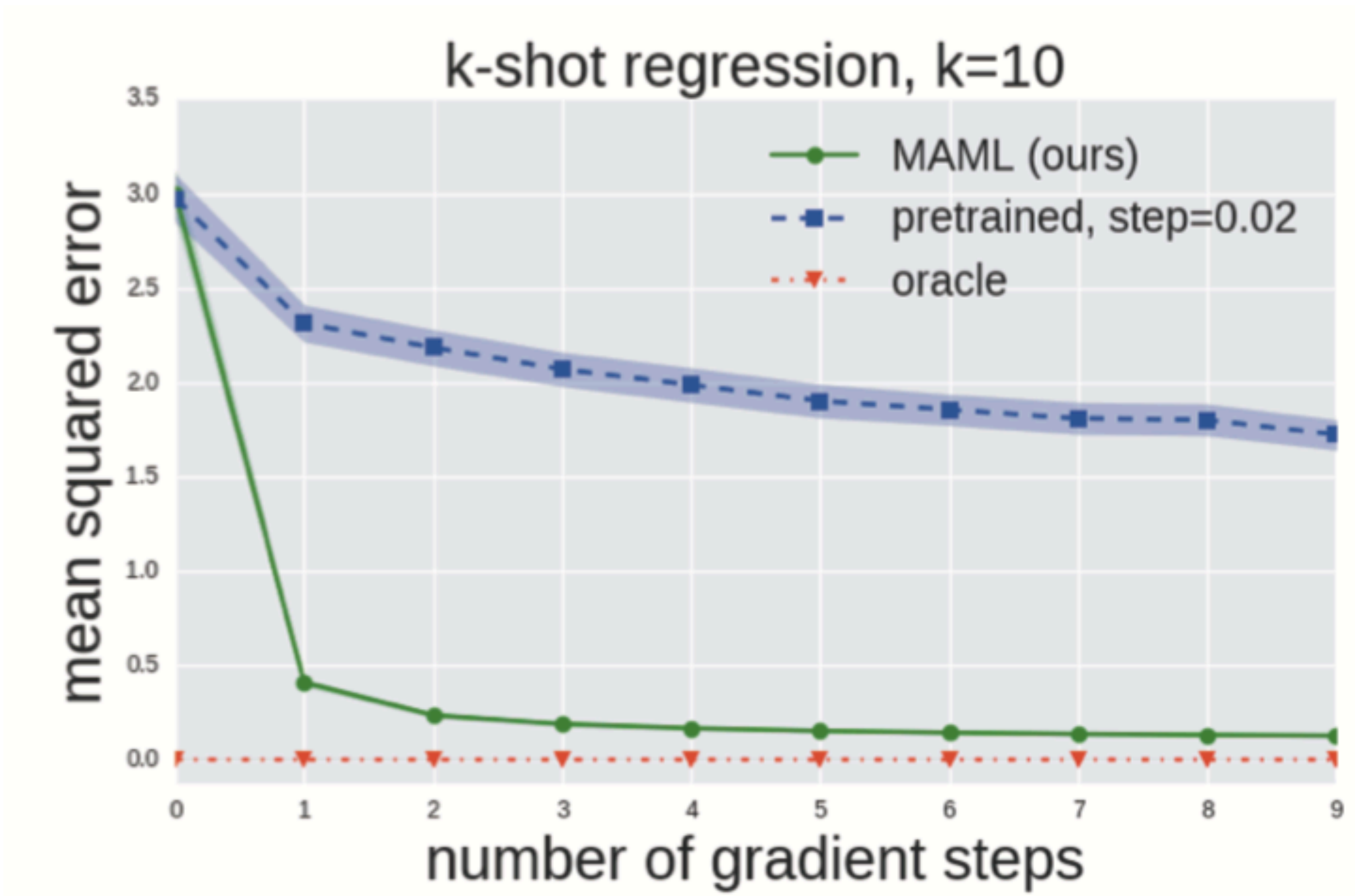
- Sinusoid Function with amplitude in [0.1, 0.5] and phase in [0, π]
- A model of 2 layers each with size 40 and ReLu-activation
- Compared with ground truth and model pre-trained on same metadata

Regression Experiment





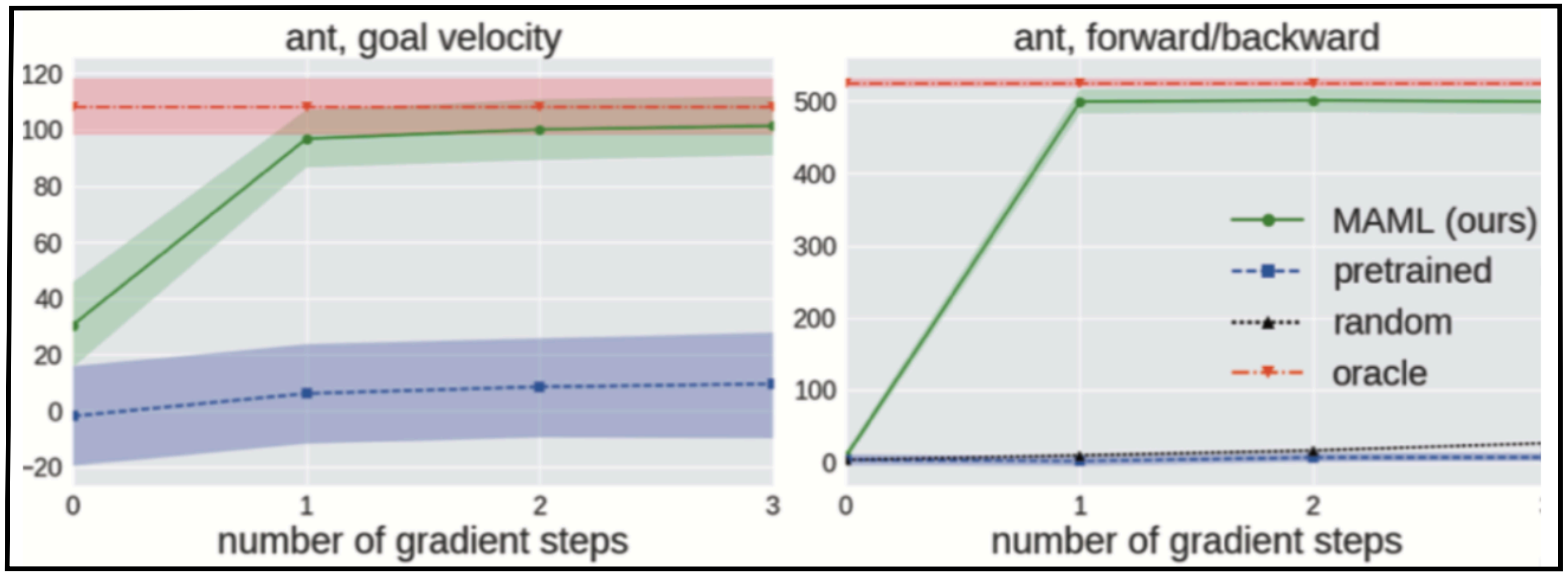
Regression Experiment

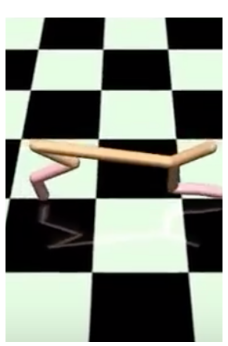


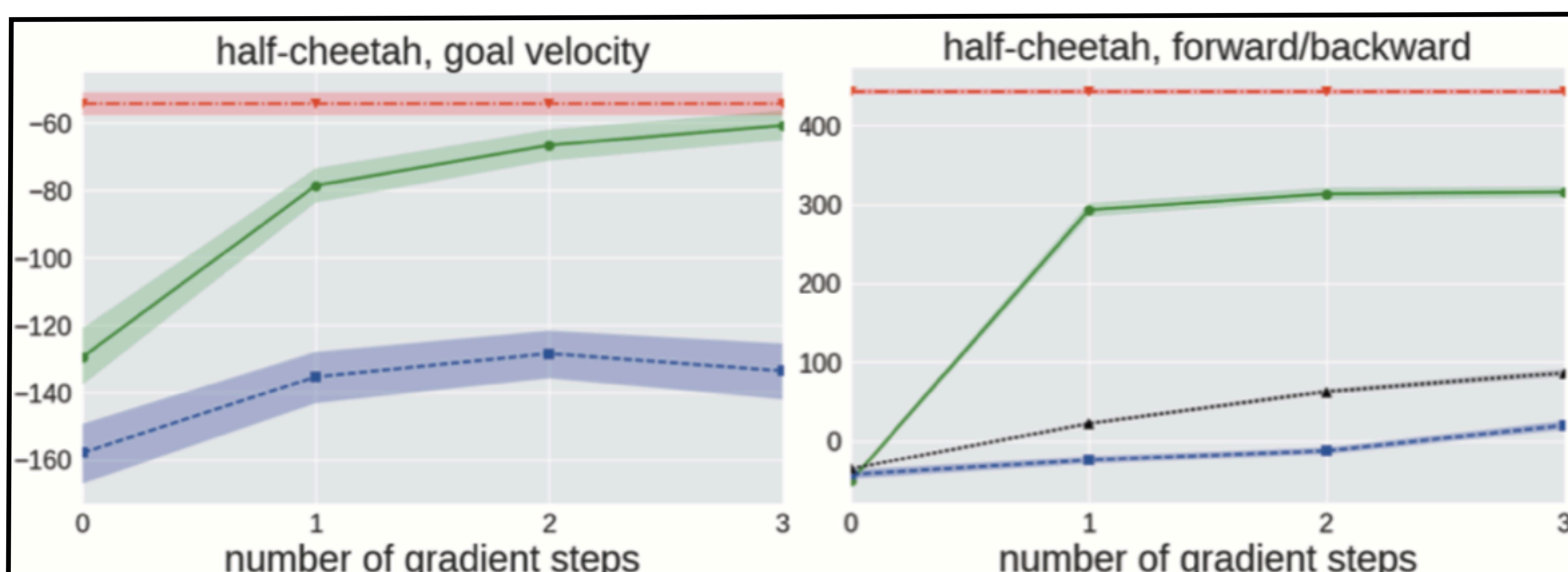
RL Experiment

- Continuous control as proposed in Duan et al. 2016
- 2 hidden layers of size 100 with ReLu activation
- TRPO as metaoptimizer and vanilla policy gradient as actual update
- Compared with ground truth and model pre-trained on same metadata

RL Experiment

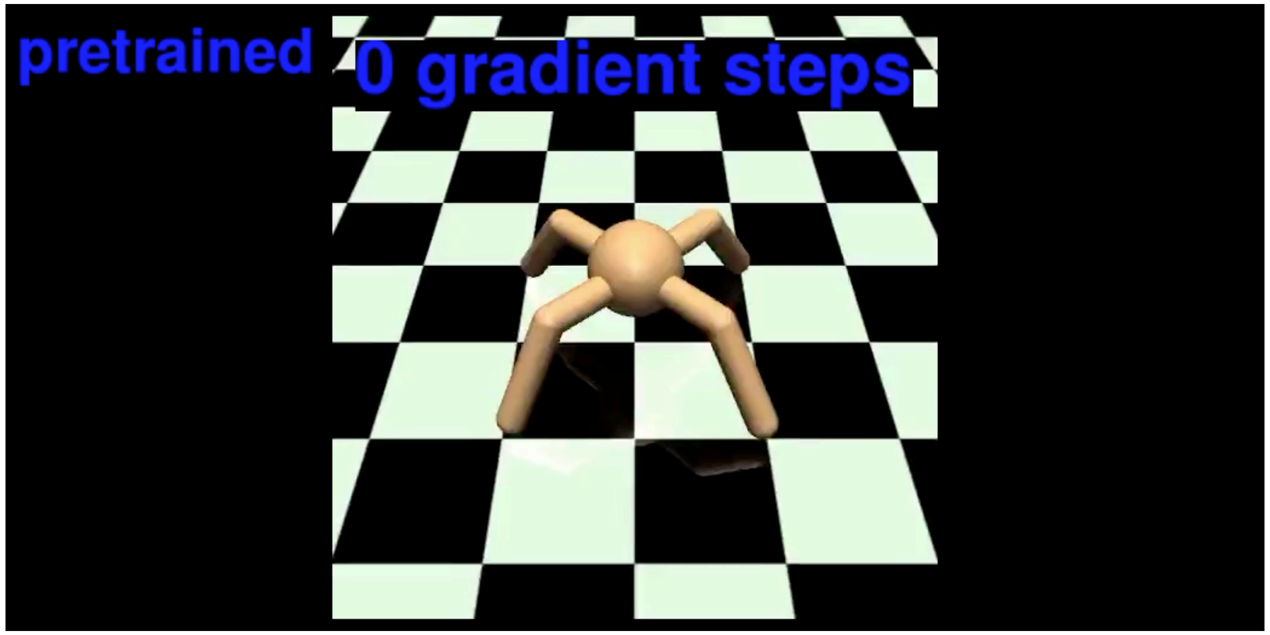


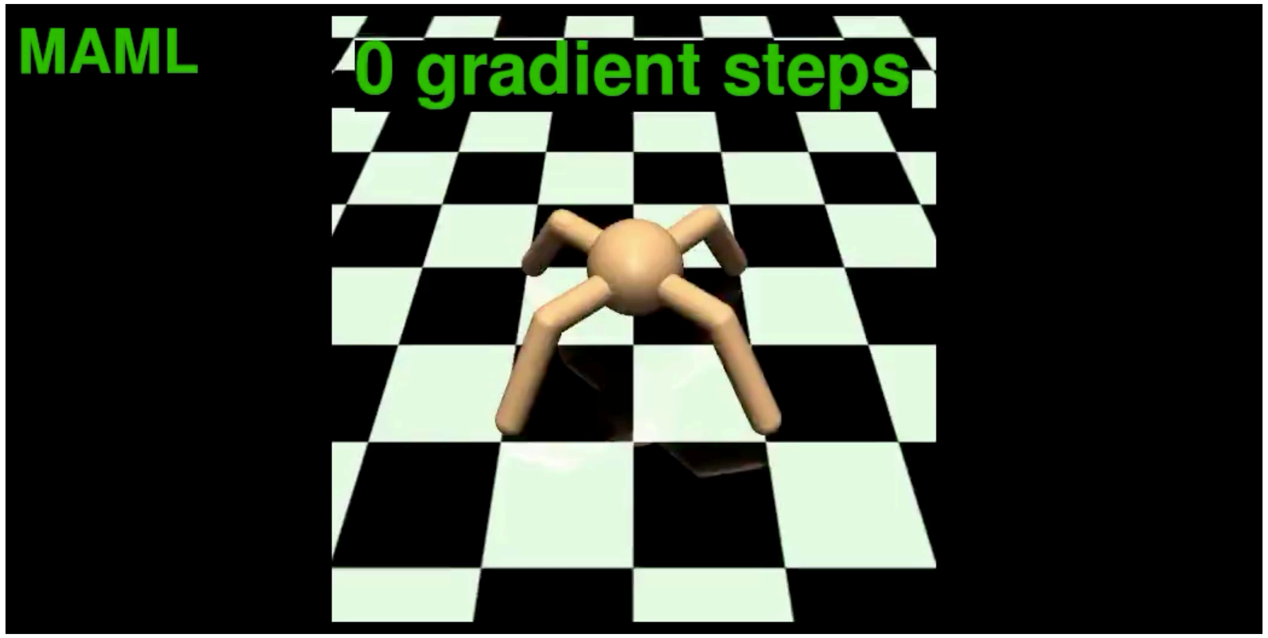




RL Experiment

More videos on: https://sites.google.com/view/maml



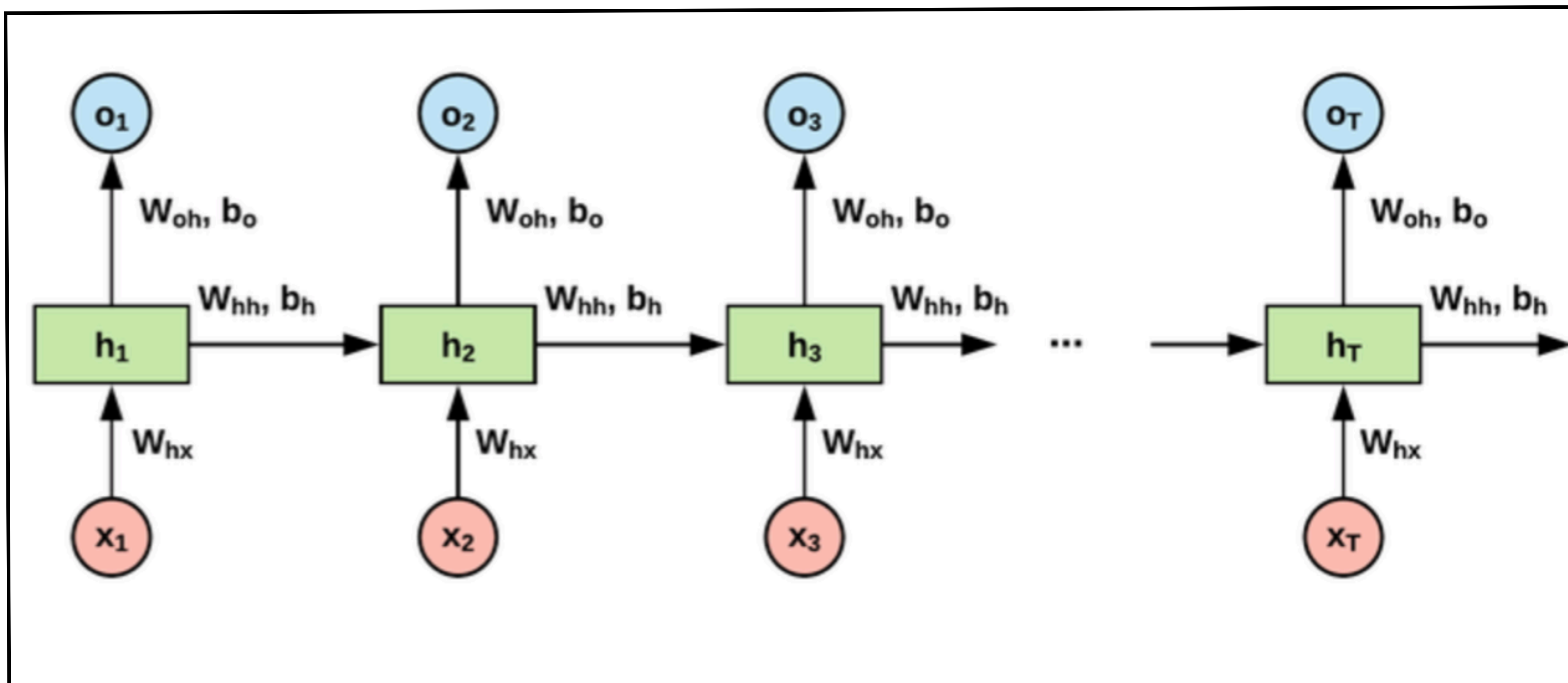


Wrap up MAML

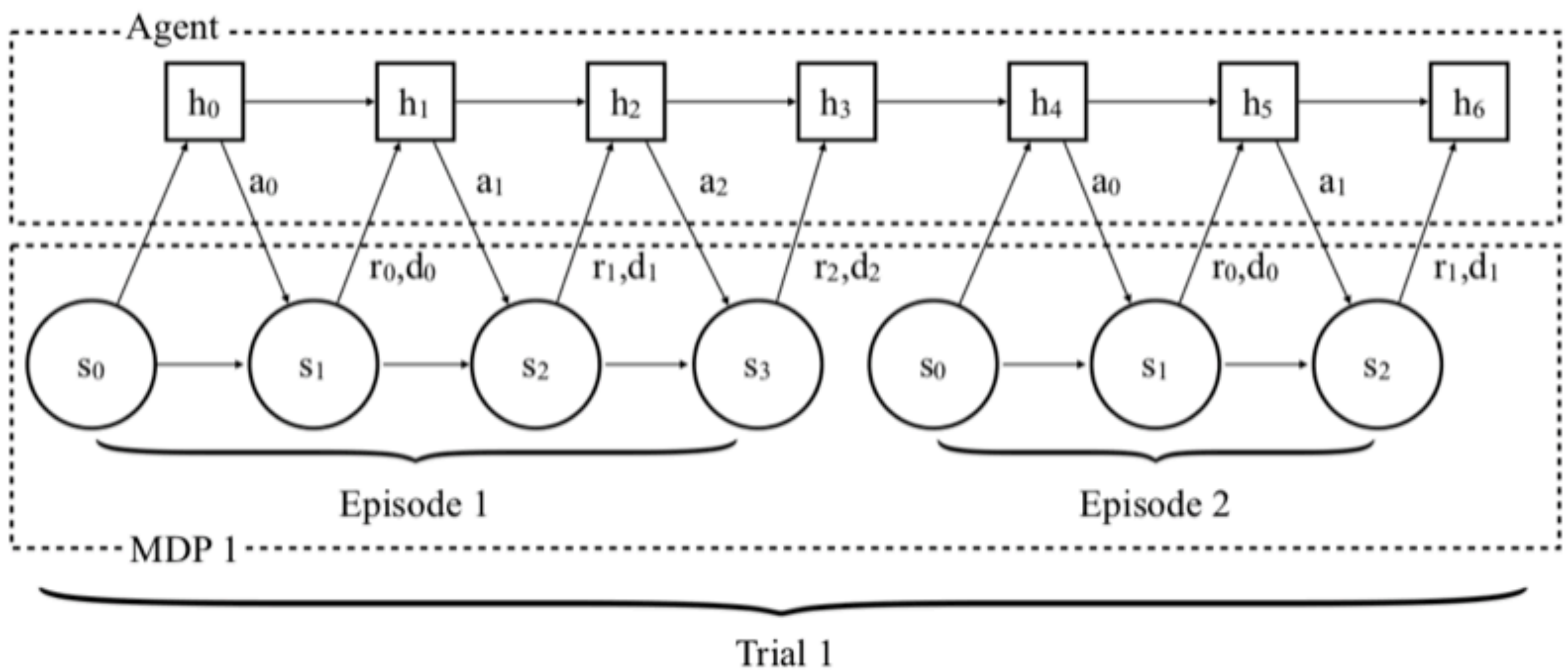
- Model-agnostic: compatible with any gradient trained model
- Flexible: take advantage of any amount of data with any number of gradient steps
- Simple: No additional parameters needed
- Disadvantage: need to compute higher order derivatives during meta-training

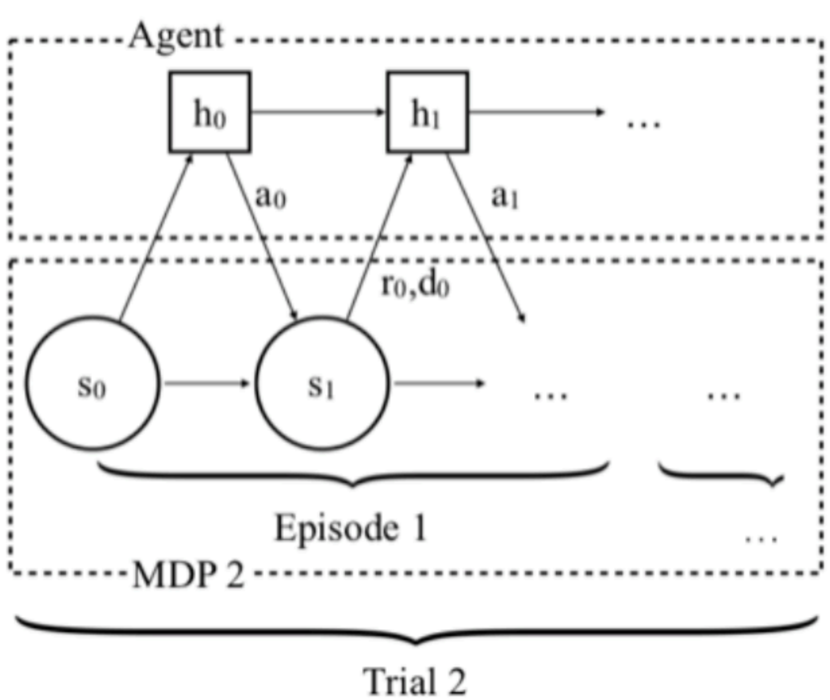
RL2

RNN


o_1
o_2
o_3
o_T
W_{oh}, b_o
W_{oh}, b_o
W_{oh}, b_o
W_{oh}, b_o
W_{hh}, b_h
W_{hh}, b_h
W_{hh}, b_h
W_{hh}, b_h
h_1
h_2
h_3
...
h_T
W_{hx}
W_{hx}
W_{hx}
W_{hx}
x_1
x_2
x_3
x_T

General Architecture

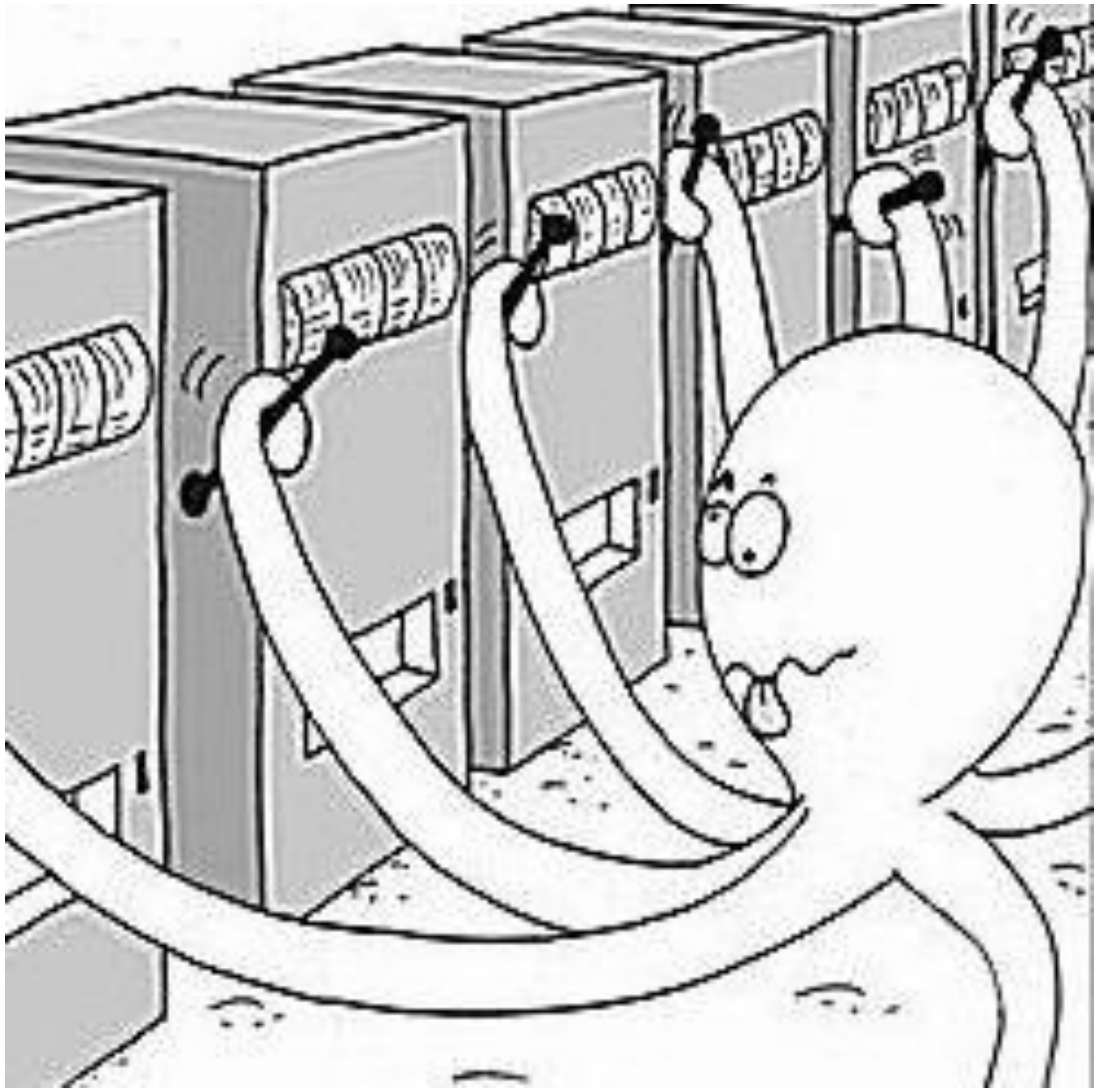




Implementation

- RL problems seen as MDPs or POMDPs
- RNN implemented by GRU network
- First-order TRPO as training algorithm
- GAE to further reduce variance

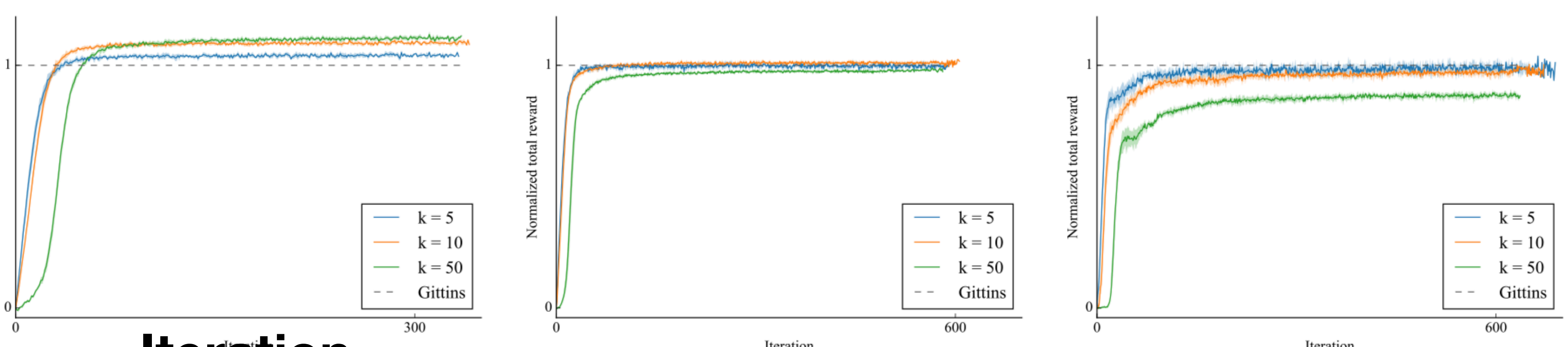
Multi-Armed Bandits



Multi-Armed Bandits

Setup	Random	Gittins	TS	OTS	UCB1	ϵ-Greedy	Greedy	RL^2
$n = 10, k = 5$	5.0	**6.6**	5.7	6.5	**6.7**	**6.6**	**6.6**	**6.7**
$n = 10, k = 10$	5.0	**6.6**	5.5	6.2	**6.7**	**6.6**	**6.6**	**6.7**
$n = 10, k = 50$	5.1	6.5	5.2	5.5	**6.6**	6.5	6.5	**6.8**
$n = 100, k = 5$	49.9	**78.3**	74.7	**77.9**	**78.0**	75.4	74.8	**78.7**
$n = 100, k = 10$	49.9	**82.8**	76.7	81.4	82.4	77.4	77.1	**83.5**
$n = 100, k = 50$	49.8	**85.2**	64.5	67.7	84.3	78.3	78.0	**84.9**
$n = 500, k = 5$	249.8	**405.8**	**402.0**	**406.7**	**405.8**	388.2	380.6	**401.6**
$n = 500, k = 10$	249.0	**437.8**	429.5	**438.9**	**437.1**	408.0	395.0	432.5
$n = 500, k = 50$	249.6	**463.7**	427.2	437.6	457.6	413.6	402.8	438.9

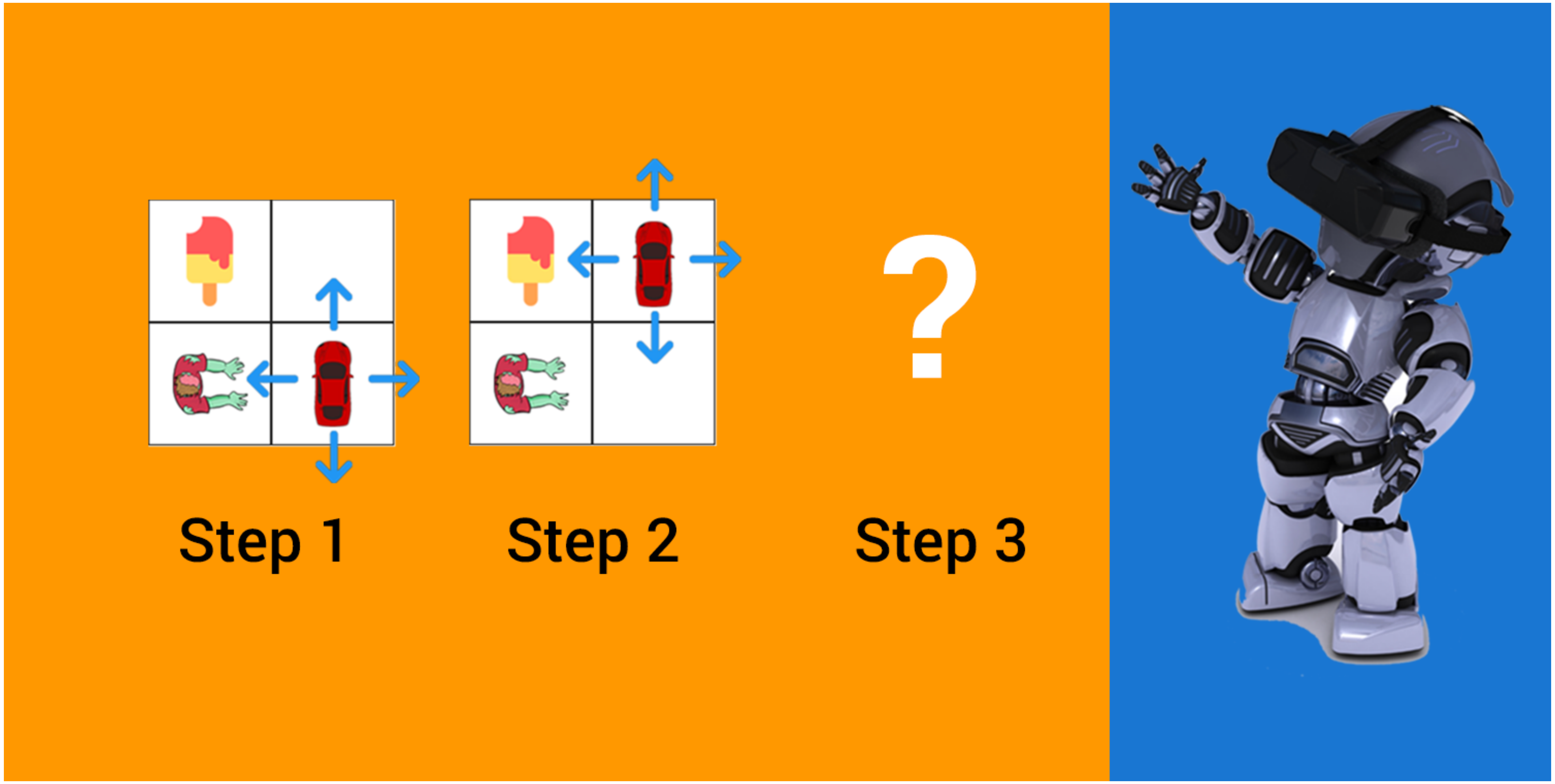
Normalised total reward



Iteration

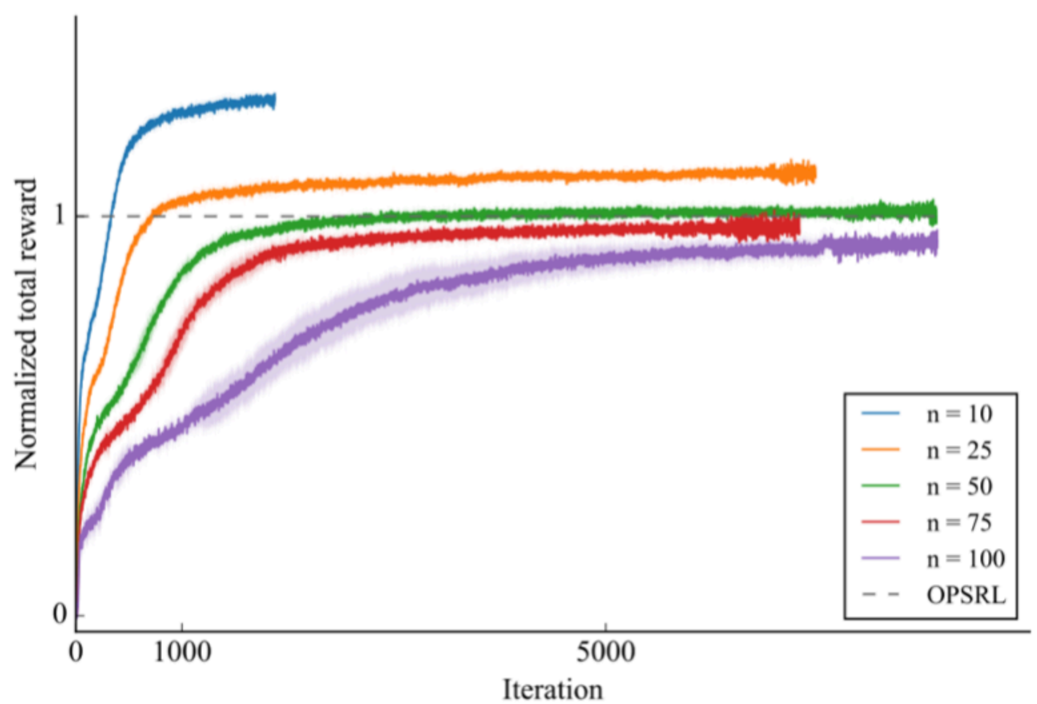
(a) $n = 10$ (b) $n = 100$ (c) $n = 500$

Tabular MDPs

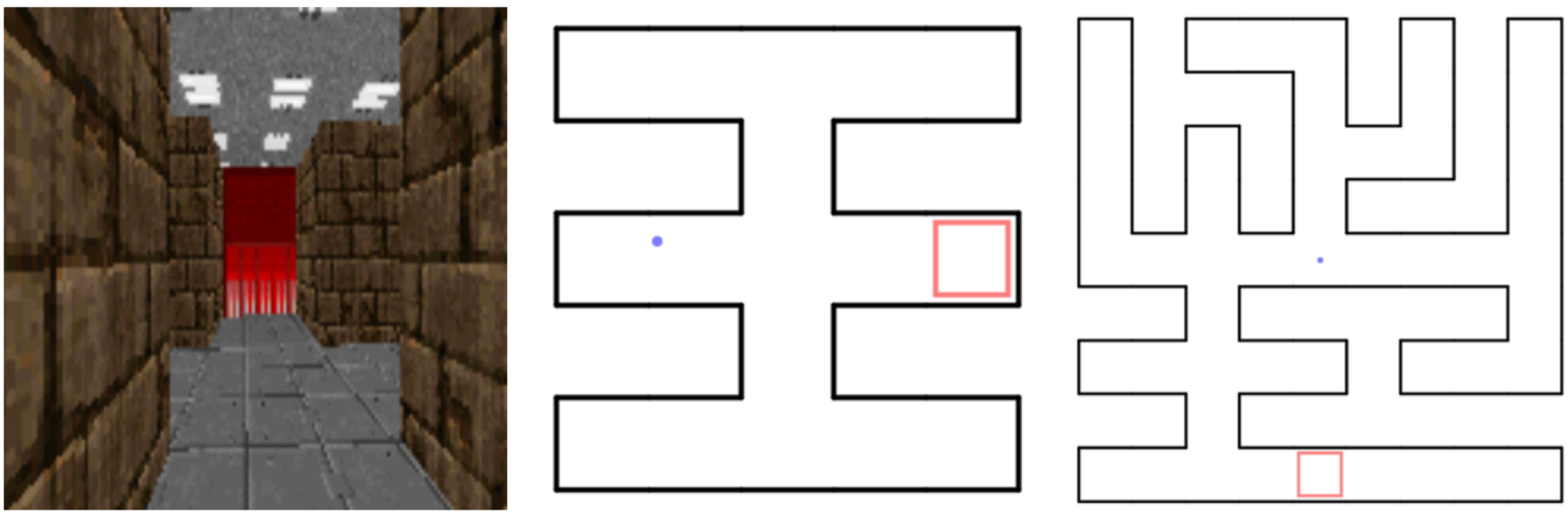


Tabular MDPs

Setup	Random	PSRL	OPSRL	UCRL2	BEB	ϵ-Greedy	Greedy	RL^2
$n = 10$	100.1	138.1	144.1	146.6	150.2	132.8	134.8	**156.2**
$n = 25$	250.2	408.8	425.2	424.1	427.8	377.3	368.8	**445.7**
$n = 50$	499.7	904.4	**930.7**	918.9	917.8	823.3	769.3	**936.1**
$n = 75$	749.9	1417.1	**1449.2**	1427.6	1422.6	1293.9	1172.9	1428.8
$n = 100$	999.4	1939.5	**1973.9**	1942.1	1935.1	1778.2	1578.5	1913.7



Visual navigation



(a) Sample observation (b) Layout of the 5×5 maze in (a) (c) Layout of a 9×9 maze

Visual navigation

(a) Average length of successful trajectories

Episode	Small	Large
1	52.4 ± 1.3	180.1 ± 6.0
2	39.1 ± 0.9	151.8 ± 5.9
3	42.6 ± 1.0	169.3 ± 6.3
4	43.5 ± 1.1	162.3 ± 6.4
5	43.9 ± 1.1	169.3 ± 6.5

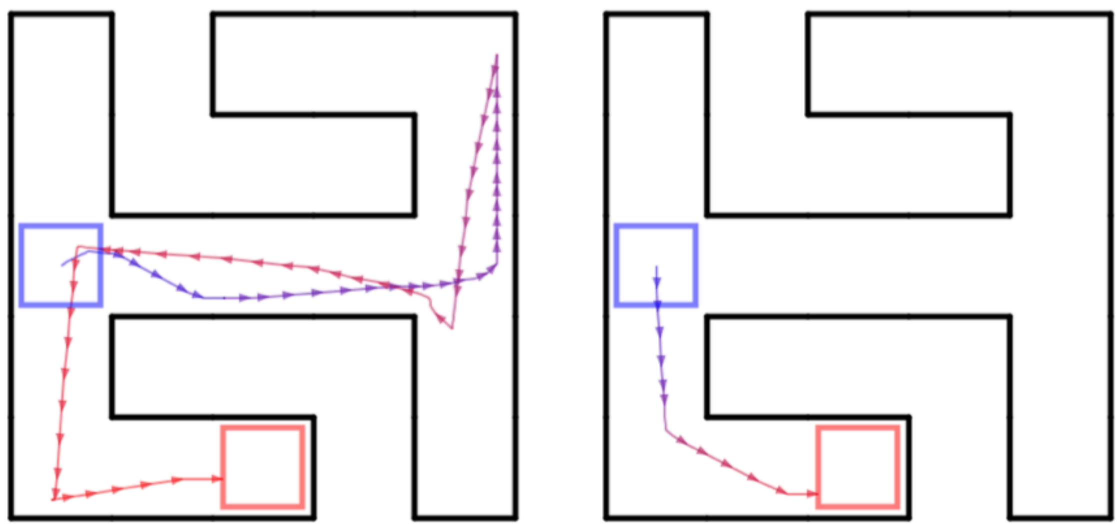
(b) %Success

Episode	Small	Large
1	99.3%	97.1%
2	99.6%	96.7%
3	99.7%	95.8%
4	99.4%	95.6%
5	99.6%	96.1%

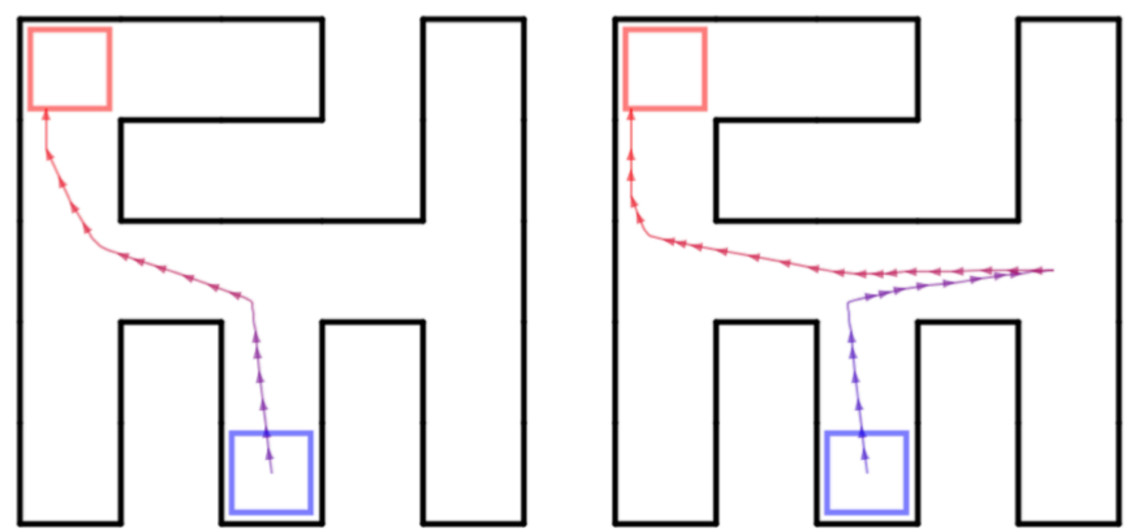
(c) %Improved

Small	Large
91.7%	71.4%

Visual navigation

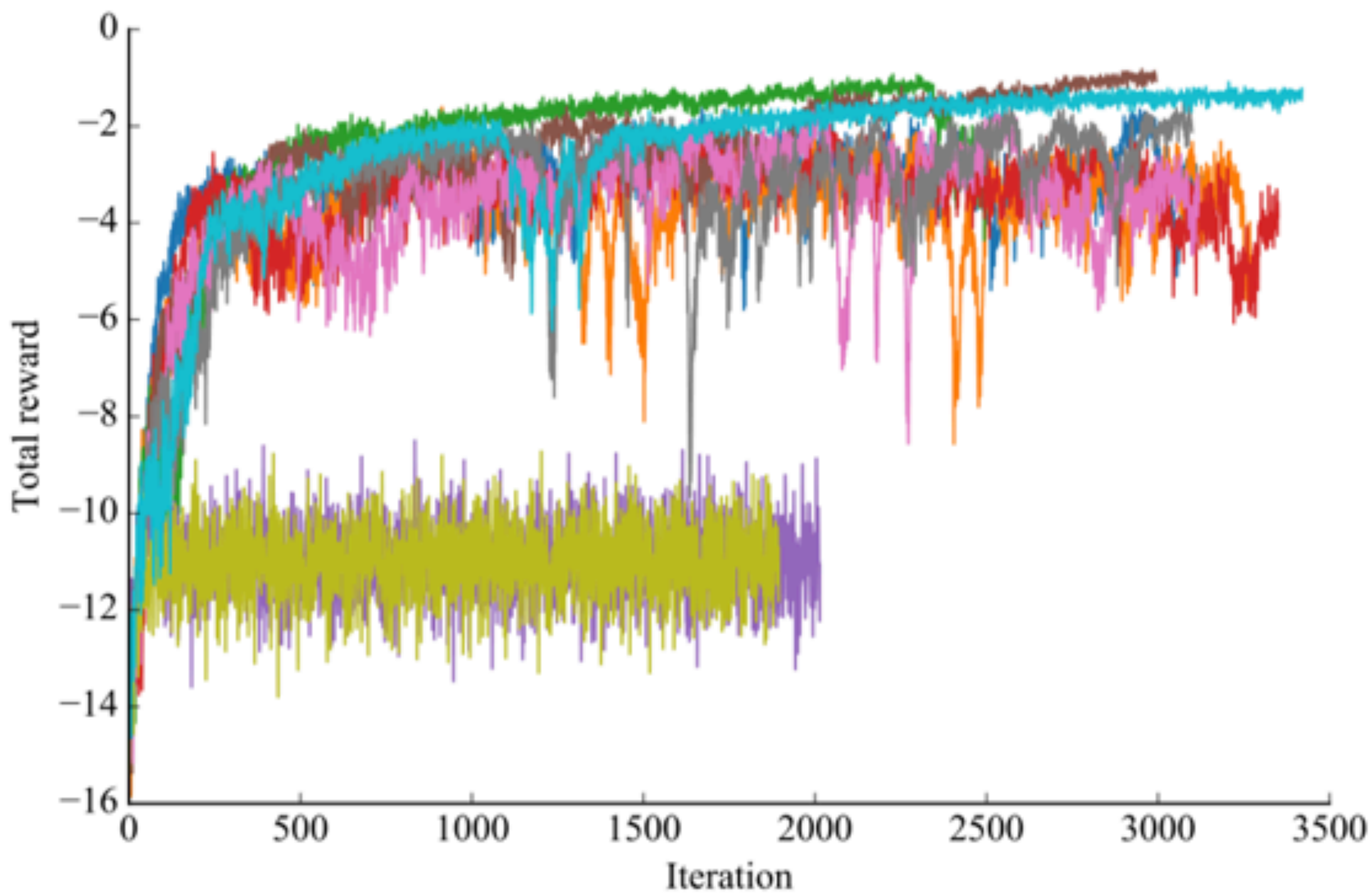


(a) Good behavior, 1st episode (b) Good behavior, 2nd episode



(c) Bad behavior, 1st episode (d) Bad behavior, 2nd episode

Visual navigation



Wrap up RL2

- Fast reinforcement learning via slow reinforcement learning using RNN states
- Comparable to theoretical optimum in small problem setting
- Scalable to complicated vision tasks
- Potential improvement for RL algorithm and network architecture

Summary

Learning
How to Learn

Thanks for listening

References

- Paper && Quotes:

  - Chelsea Finn, Pieter Abbeel, Sergey Levine: Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks arXiv:1703.03400
  - Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, Pieter Abbeel: RL2 Fast Reinforcement Learning via Slow Reinforcement Learning arXiv:1611.02779
  - Yan Duan, Xi Chen, Rein Houthooft, John Schulman, Pieter Abbeel: Benchmarking Deep Reinforcement Learning for Continuous Control arXiv:1604.06778
  - https://bair.berkeley.edu/blog/2017/07/18/learning-to-learn/

- Pictures:

  - https://paperswithcode.com/task/multi-armed-bandits
  - https://medium.com/@curiousily/solving-an-mdp-with-q-learning-from-scratch-deep-reinforcement-learning-for-hackers-part-1-45d1d360c120
  - https://www.coursera.org/learn/learning-how-to-learn