Deep Reinforcement Learning : Reliability and Multi-Agent Environments

Abhishek Naik (CS13B030)

Indian Institute of Technology, Madras

Guide : Professor B. Ravindran

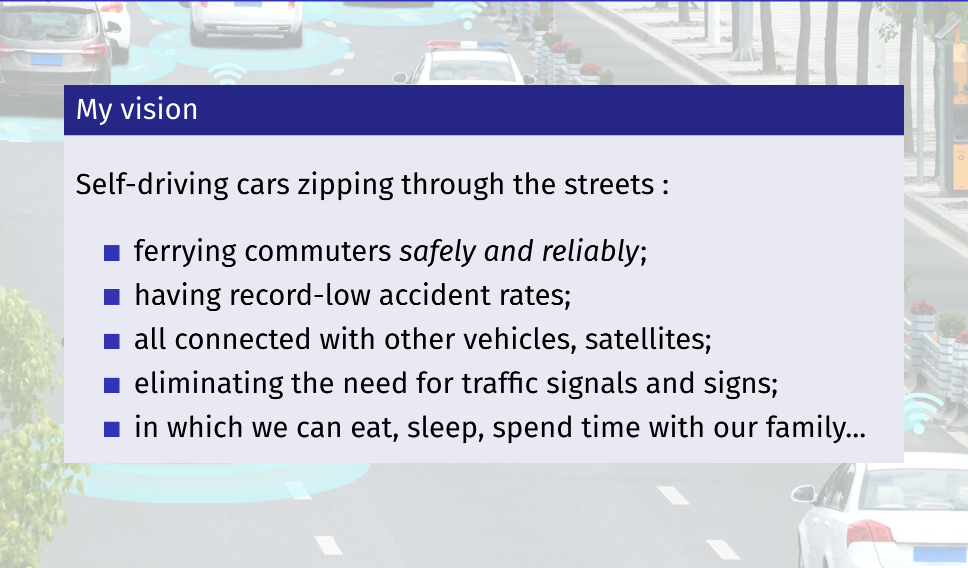
DDP Evaluation

Outline

Outline

Motivation

Motivation



My vision

Self-driving cars zipping through the streets :

- ferrying commuters *safely and reliably*;
- having record-low accident rates;
- all connected with other vehicles, satellites;
- eliminating the need for traffic signals and signs;
- in which we can eat, sleep, spend time with our family...

Motivation

Reinforcement Learning (RL) has achieved success at human-level or superhuman performance in :

- full-information games - Chess, Go [1, 2]
- control tasks - robotic navigation, helicopter-flying [3, 4]
- partial-information games - ATARI, DoTA, Poker [5, 6]

Motivation

Ongoing efforts in extremely challenging risk-sensitive applications like autonomous driving or robotic surgery to achieve:

1. human-level (expert) performance in these tasks
2. with appropriate guarantees of safety.

Motivation

Ongoing efforts in extremely challenging risk-sensitive applications like autonomous driving or robotic surgery to achieve:

1. human-level (expert) performance in these tasks
2. with appropriate guarantees of safety.

Specific to autonomous driving:

- Negotiating in the *multi-agent* game of traffic ...
- to get from source to destination *safely and reliably*.

Problem Statement(s)

A three-pronged strategy :

Problem Statement(s)

A three-pronged strategy :

1. Improving the reliability of the state-of-the-art imitation learning algorithms when learning from only a fixed set of expert trajectories for risk-sensitive applications. [Risk-Averse Imitation Learning]

Problem Statement(s)

A three-pronged strategy :

1. Improving the reliability of the state-of-the-art imitation learning algorithms when learning from only a fixed set of expert trajectories for risk-sensitive applications. [Risk-Averse Imitation Learning]
2. Setting up a simple framework for enabling multi-agent research for autonomous driving, and benchmarking multi-agent learning algorithms on the HFO RoboSoccer simulator. [Multi-Agent Learning]

Problem Statement(s)

A three-pronged strategy :

1. Improving the reliability of the state-of-the-art imitation learning algorithms when learning from only a fixed set of expert trajectories for risk-sensitive applications. [Risk-Averse Imitation Learning]
2. Setting up a simple framework for enabling multi-agent research for autonomous driving, and benchmarking multi-agent learning algorithms on the HFO RoboSoccer simulator. [Multi-Agent Learning]
3. Mastering the hard, sparse-reward task of RoboSoccer by learning a sequence of simpler sub-tasks in a principled manner. [Curriculum Learning]

Outline

Reinforcement Learning

- Discover the ‘right’ behaviour in the given context . . .
- to achieve the maximum reward . . .
- via trial-and-error.

Reinforcement Learning

- Discover the 'right' behaviour in the given context . . .
- to achieve the maximum reward . . .
- via trial-and-error.

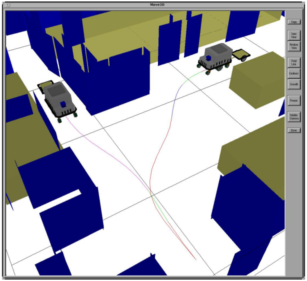
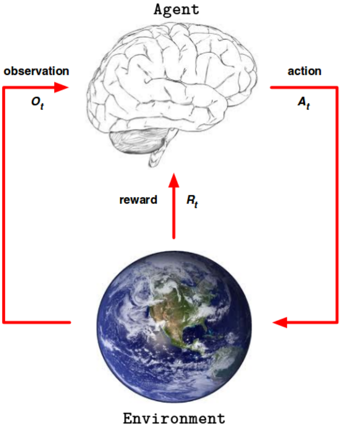


Image credits : TheSchoolRun and projects.laas.fr

Reinforcement Learning

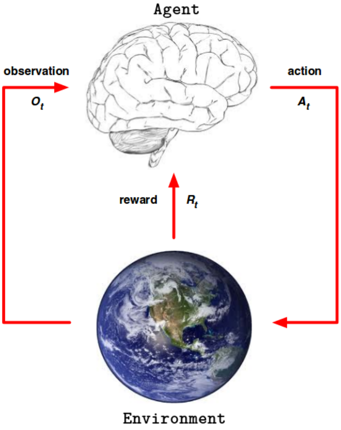
Mathematically, consider a Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$.



Reinforcement Learning

Mathematically, consider a Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. At each timestep t,

- the agent receives a state s_t (or observation o_t) in a state space $\mathcal{S}$,
- selects an action a_t from an action space $\mathcal{A}$ following a policy $\pi(a_t|s_t)$,
- receives a scalar reward r_t according to the reward function $\mathcal{R}(s, a)$,
- and transitions to the next state s_{t+1} with the state transition probability $\mathcal{P}(s_{t+1}|s_t, a_t)$
- where γ is the MDP's discount factor



Outline

Imitation Learning

The idea

Learns policies through imitation of an expert's behavior without the need of a handcrafted reward function. [7]

Imitation Learning : Paradigm 1

Behavioural Cloning

Uses supervised learning to fit a policy function to the state-action pairs from expert-demonstrated trajectories.

Imitation Learning : Paradigm 1

Behavioural Cloning

Uses supervised learning to fit a policy function to the state-action pairs from expert-demonstrated trajectories.

Notable applications:

- ALVINN - the first self-driving car (1989) [8]
- NVIDIA's recent self-driving efforts [9]

Imitation Learning : Paradigm 1

Behavioural Cloning

Uses supervised learning to fit a policy function to the state-action pairs from expert-demonstrated trajectories.

Notable applications:

- ALVINN - the first self-driving car (1989) [8]
- NVIDIA's recent self-driving efforts [9]

Main drawback: Compounding errors [10]

Assume observations are i.i.d.; learns to fit single time-step decisions.

Imitation Learning : Paradigm 1

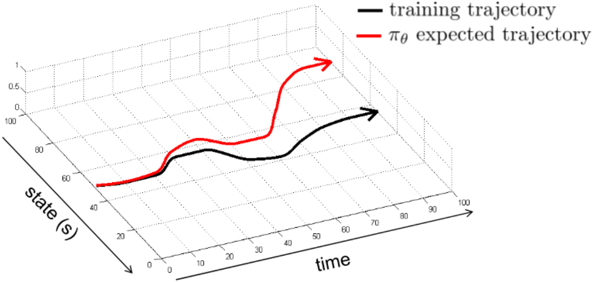


Figure: An illustration of the compounding error due to covariate shift (adapted from Sergey Levine's RL course slides).

Imitation Learning : Paradigm 1

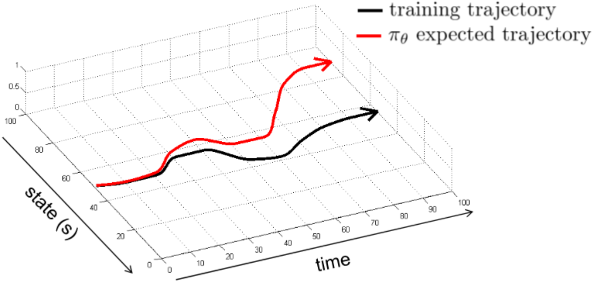


Figure: An illustration of the compounding error due to covariate shift (adapted from Sergey Levine's RL course slides).

Approaches like DAgger [11] ameliorate this problem, but require querying of expert in training.

Imitation Learning : Paradigm 2

Apperenticeship Learning [12]

Attempts to uncover the underlying reward function (IRL), then applies standard RL to learn a policy.

Imitation Learning : Paradigm 2

Apperenticeship Learning [12]

Attempts to uncover the underlying reward function (IRL), then applies standard RL to learn a policy.

- \+ Does not suffer from issue of compounding error.
- − Indirect; computationally expensive
- − Not scalable to large domains. [13]

Imitation Learning : State-of-the-art

Generative Adversarial Imitation Learning (GAIL) [14]

GAIL uses the generative-adversarial framework to generate state-action pairs similar to those generated by an 'expert'.

Imitation Learning : State-of-the-art

Generative Adversarial Imitation Learning (GAIL) [14]

GAIL uses the generative-adversarial framework to generate state-action pairs similar to those generated by an 'expert'.

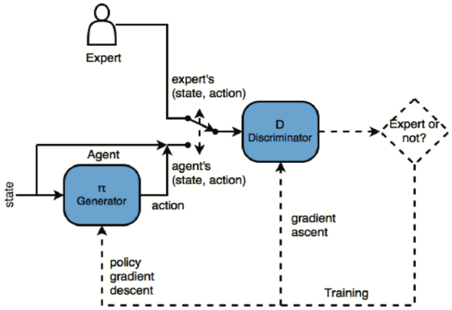


Figure: The GAIL framework

Imitation Learning : State-of-the-art

Generative Adversarial Imitation Learning (GAIL) [14]
Ho and Ermon, *NIPS 2016*

+ Does not suffer from issue of compounding error.
+ Scalable to large domains.
- But distributions of trajectory-costs are heavy-tailed.

Imitation Learning : State-of-the-art

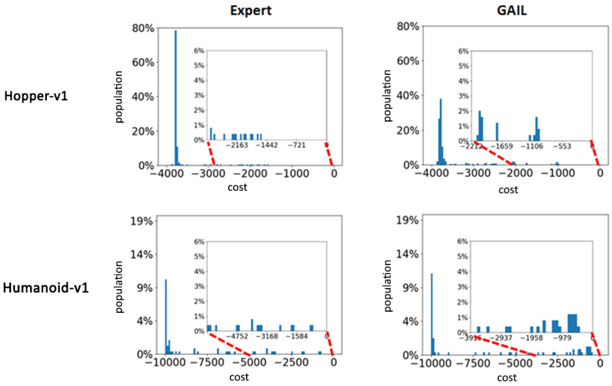


Figure: Histograms of the costs of 250 trajectories generated by the expert and GAIL agents at high-dimensional continuous control tasks

Risk-sensitivity

Two broad categories : [15]

1. constraining the agent to safe states during exploration.
2. modifying the optimality criterion of the agent to embed a term for minimizing risk.

Studies on risk-minimization are rather scarce in the imitation learning literature, and focus on average-case performance at the center, overlooking tail-end events.

Overview

Conditional-Value-at-Risk [16]

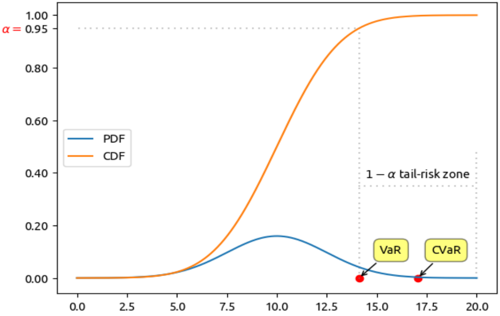


Figure: $VaR_{0.95}$ and $CVaR_{0.95}$ for a gaussian distribution

$$VaR_\alpha(Z) \triangleq \min(z \mid P(Z \leq z) \geq \alpha)$$

$$CVaR_\alpha(Z) \triangleq \mathbb{E}\left[Z \mid Z \geq VaR_\alpha(Z)\right]$$

Objective

To find a policy π^* ($\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$) which minimize the high-cost tail-end trajectories.

$$\min_{\pi,\nu} \max_{\mathcal{D} \in (0,1)^{\mathcal{S}\times\mathcal{A}}} \Big\{ \mathbb{E}_{\pi_E}[log(1 - \mathcal{D}(s,a))] \\ + \mathbb{E}_{\pi}[log(\mathcal{D}(s,a))] - H(\pi) \\ + \lambda_{CVaR}\, H_\alpha(\mathcal{R}^\pi(\xi | c(\mathcal{D})), \nu) \Big\}$$

Experiments

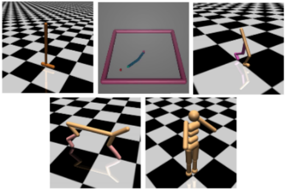
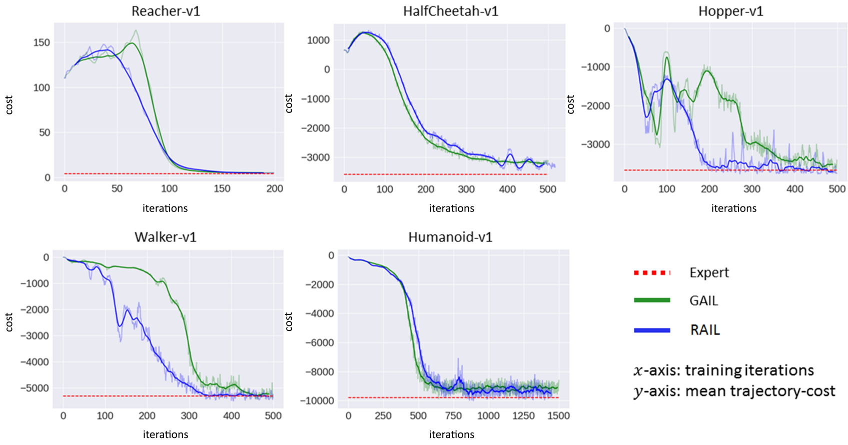


Figure: The continuous control environments

Environment	Dimensionality	
	State	**Action**
Reacher	11	2
Hopper	11	3
HalfCheetah	17	6
Walker	17	6
Humanoid	376	17

Table: Dimensionality of the environments

Results


Reacher-v1
HalfCheetah-v1
Hopper-v1
Walker-v1
Humanoid-v1
cost
iterations
Expert
GAIL
RAIL
x-axis: training iterations
y-axis: mean trajectory-cost

Results

Table: Values of percentage relative tail risk measures and gains in reliability on using RAIL over GAIL. RAIL shows a remarkable improvement over GAIL in both the metrics.

Environment	$VaR_{0.9}(A\|E)$**(%)**		**GR-VaR** (%)	$CVaR_{0.9}(A\|E)$ **(%)**		**GR-CVaR** (%)
	GAIL	**RAIL**		**GAIL**	**RAIL**	
Reacher	-62.41	-23.81	**38.61**	-108.99	-48.42	**60.57**
Hopper	-53.17	-0.23	**52.94**	-49.62	39.38	**89.00**
HalfCheetah	-21.66	-8.20	**13.46**	-33.84	-12.24	**21.60**
Walker	-1.64	0.03	**1.66**	45.39	70.52	**25.13**
Humanoid	-73.16	-5.97	**67.19**	-71.71	1.07	**72.78**

Conclusion

Conclusion

- RAIL obtains a superior performance at all the metrics across all the 5 continuous control tasks.

Conclusion

- RAIL obtains a superior performance at all the metrics across all the 5 continuous control tasks.
- RAIL converges atleast as fast as GAIL, and at times, even faster.

Conclusion

- RAIL obtains a superior performance at all the metrics across all the 5 continuous control tasks.
- RAIL converges atleast as fast as GAIL, and at times, even faster.
- RAIL is also scalable to complex environments with large state and action spaces.

Conclusion

- RAIL obtains a superior performance at all the metrics across all the 5 continuous control tasks.
- RAIL converges atleast as fast as GAIL, and at times, even faster.
- RAIL is also scalable to complex environments with large state and action spaces.
- RAIL works even in the absence of a heavy tail since minimization of *CVaR* also leads to minimization of mean and standard deviation. [16]

Conclusion

- RAIL obtains a superior performance at all the metrics across all the 5 continuous control tasks.
- RAIL converges atleast as fast as GAIL, and at times, even faster.
- RAIL is also scalable to complex environments with large state and action spaces.
- RAIL works even in the absence of a heavy tail since minimization of *CVaR* also leads to minimization of mean and standard deviation. [16]

Risk-Averse Imitation Learning
Santara, A.*, **Naik, A.***, Ravindran, B., and others.
To appear in the proceedings of AAMAS 2018; arxiv.org/abs/1707.06658

Outline

Motivation

In the real world, learning often happens in groups rather than individually, in silos.

Image credits : RealMadrid.com and FactorDaily

Motivation

In the real world, learning often happens in groups rather than individually, in silos.

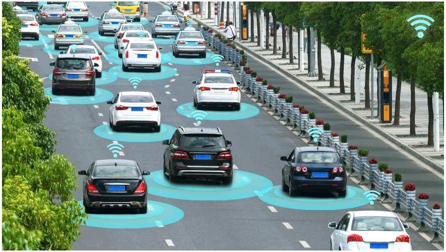


Image credits : RealMadrid.com and FactorDaily

Related Work

Related Work

- Classical approaches :
  - Independent Q-learning [17], Nash Q-learning [18], WoLF [19], etc.

Related Work

- Classical approaches :
  - Independent Q-learning [17], Nash Q-learning [18], WoLF [19], etc.
- Recent (and deep) approaches :
  - MA-DQN [20], Deep Hysteretic Q-learning [21], etc.

Related Work

- Classical approaches :
  - Independent Q-learning [17], Nash Q-learning [18], WoLF [19], etc.
- Recent (and deep) approaches :
  - MA-DQN [20], Deep Hysteretic Q-learning [21], etc.
- Issues :
  - Work only on small, discrete domains.
  - Not scalable to high-dimensional, continuous control tasks.

State-of-the-Art

State-of-the-Art

Multi-Agent DDPG (MADDPG) [22]

- DDPG algorithm extended for multiple agents.
- Relatively new, does not seem scalable.

State-of-the-Art

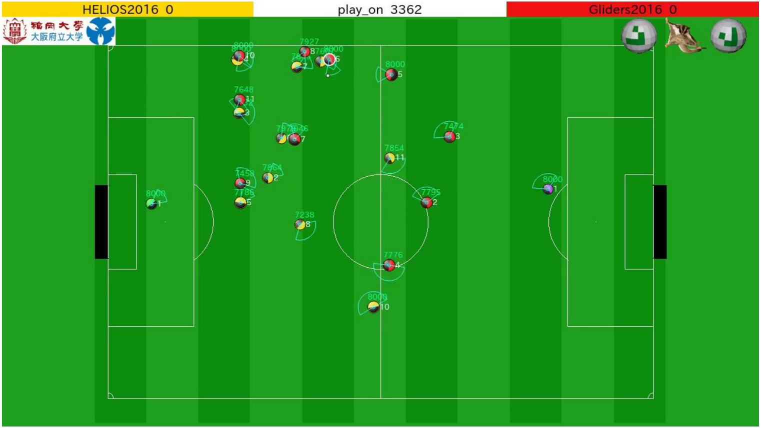
Multi-Agent DDPG (MADDPG) [22]

- DDPG algorithm extended for multiple agents.
- Relatively new, does not seem scalable.

PSMADDPG [23] claims scalability.

Overview

RoboSoccer

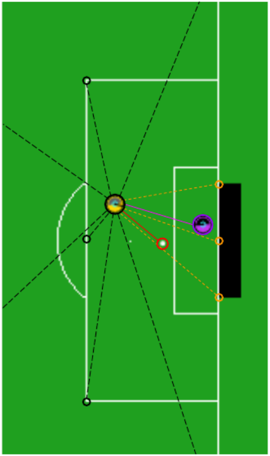


RoboSoccer

Challenges

- High-dimensional Spaces
- Parameterized Action Space
- Multi-agent Learning

Observation Space

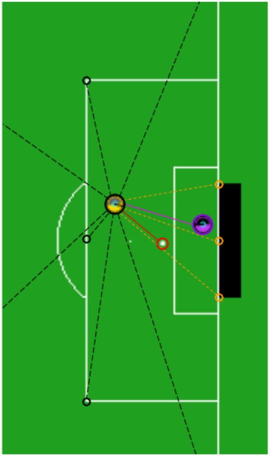


Notable features:

- Agent's position, velocity, orientation
- Distances and angles to ball, goal-posts, players, etc.

Total 58 continuous-valued features.

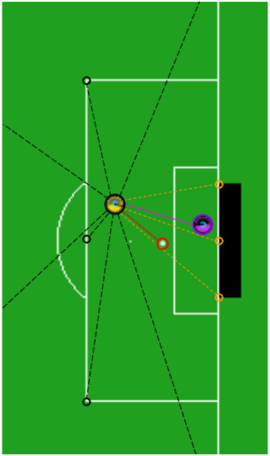
Action Space



- Kick(*power*, *direction*)
- Dash(*power*, *direction*)
- Turn(*direction*)
- Tackle(*direction*)

Total : **4** actions + **6** parameters

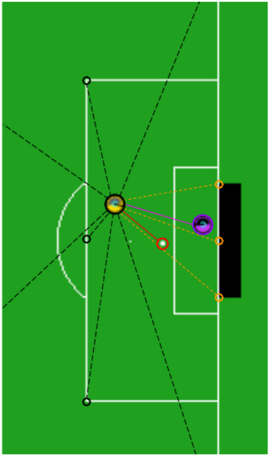
Reward Function



Components:

1. MoveToBall $[r_1(t)]$
2. FirstBallTouch $[r_2(t)]$
3. MoveToGoal $[r_3(t)]$
4. ScoreGoal $[r_4(t)]$

Reward Function



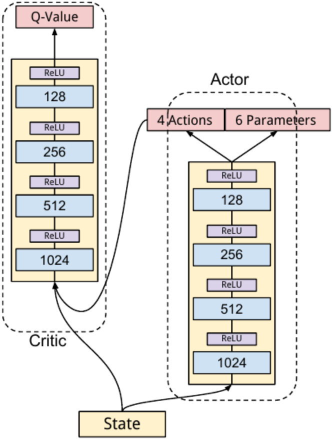
Components:

1. MoveToBall $[r_1(t)]$
2. FirstBallTouch $[r_2(t)]$
3. MoveToGoal $[r_3(t)]$
4. ScoreGoal $[r_4(t)]$

Total reward:

$$r(t) = r_1(t) + r_2(t) + \mathbf{3}r_3(t) + \mathbf{5}r_4(t)$$

Model



Courtesy [24]

An *actor-critic* model

Digression : Paradigms for solving RL

Digression : Paradigms for solving RL

1. Value-based : Solve for the optimal v^*

Digression : Paradigms for solving RL

1. Value-based : Solve for the optimal v^*
2. Policy-based : Solve for the optimal π^*

Digression : Paradigms for solving RL

1. Value-based : Solve for the optimal v^*
2. Policy-based : Solve for the optimal π^*
3. Actor-Critic : Solves for both.

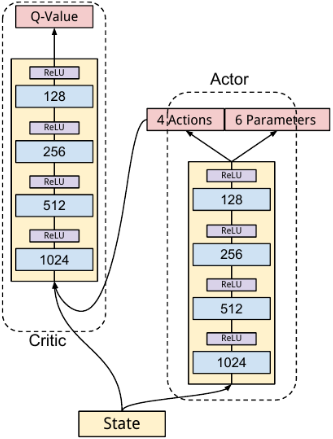
Digression : Paradigms for solving RL

1. Value-based : Solve for the optimal v^*
2. Policy-based : Solve for the optimal π^*
3. Actor-Critic : Solves for both.

Figure: Takes an action

Figure: Evaluates the action

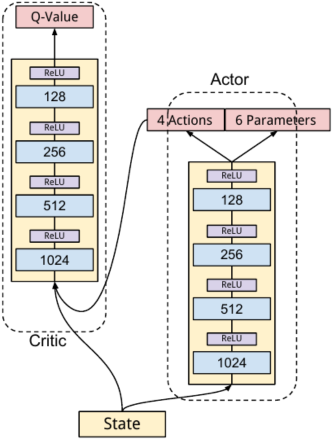
Model



Courtesy [24]

An *actor-critic* model

Model



Courtesy [24]

An *actor-critic* model

- Actor : 4 + 6 outputs
- Action chosen : *max*(Kick, Dash, Turn, Tackle)
- Parameters used : corresponding to chosen action
- Critic : 4 + 6 gradients

Experiments

Experiments

The following combination of scenarios were tested :

- one or more agents
- independent and shared network (lower) layers
- independent and shared replay buffers
- with and without a goalkeeper
- an expert or a naive goalkeeper

Results

Table: Some interesting results corresponding to some of the combinations of the aforementioned scenarios.

Scenario	Trials	Goals #	Goals %	Iterations	AvgFrame/Goal
1v0	275896	234031	**84.83**	250000	**126.4**
2v0 (indp)	247900	178995	**72.20**	250000	**116.9**
2v0 (memory)	307341	232201	**75.55**	∼300000	**116.3**
2v0 (layers)	241160	183751	**76.19**	250000	**120**
1v1 (expert)	646046	392	**0.06**	∼650000	**136.8**
1v1 (goalie)	236821	116909	**49.37**	250000	**130**
1v1 (goalie; noFreeze)	227804	119070	**52.27**	250000	**127.6**
2v1 (ind)	300127	197	**0.07**	300000	**135.7**
2v1 (memory)	250000	72	**0.029**	250000	-
2v1 (memory, pass)	198039	68	**0.03**	300000	**220**

Takeaways

- Multi-agent learning is hard.
- Problems of non-stationarity and scalability are real.
- Reward-engineering is extremely hard to get to work in complex environments.

Takeaways

- Multi-agent learning is hard.
- Problems of non-stationarity and scalability are real.
- Reward-engineering is extremely hard to get to work in complex environments.

And what about autonomous driving?

Overview

Motivation

The issues with existing driving simulators :

Motivation

The issues with existing driving simulators :

- **Lack of multi-agent control** :
  innately support only ego-centric control, have pre-programmed behaviors for the other agents.

Motivation

The issues with existing driving simulators :

- **Lack of multi-agent control** :
  innately support only ego-centric control, have
  pre-programmed behaviors for the other agents.
- **Lack of customizability of non-ego-control cars** :
  difficulty in introducing agents with custom behaviors
  restricts the diversity of real-world scenarios
  that can be simulated.

Motivation

The issues with existing driving simulators :

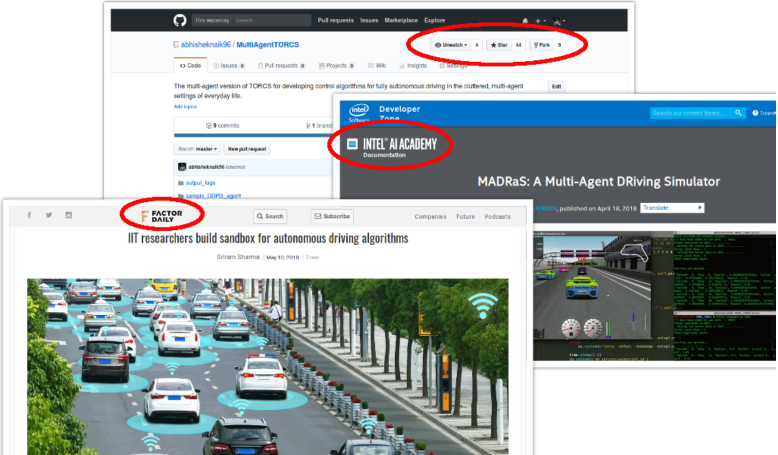
- **Lack of multi-agent control** :
  innately support only ego-centric control, have pre-programmed behaviors for the other agents.
- **Lack of customizability of non-ego-control cars** :
  difficulty in introducing agents with custom behaviors restricts the diversity of real-world scenarios that can be simulated.
- **Proprietary technology** :
  secrecy of players like Google and Uber add to the inaccessibility of autonomous driving research for researchers without (very) deep pockets.

Multi-Agent DRiving Simulator

Figure: Screenshot of MADRaS' interface

Multi-Agent DRiving Simulator

Encouraging response from the community



Planned work

1. Benchmarking multi-agent RL algorithms :
   - MADDPG [22], PSMADDPG [23], SOM [25], DIAL and RAIL [26]
2. Creating a dataset of traffic scenarios :
   - the aim to create a plethora of plug-and-play scenarios for ease of research
3. Simulation of classical multi-agent scenarios :
   - Platooning; Pooling knowledge, Leveraging intent, ...

Outline

Curriculum Learning

Humans inherently break problems down to a sequence of manageable stages and sub-goals that are of progressively greater complexity.

Curriculum Learning

Humans inherently break problems down to a sequence of manageable stages and sub-goals that are of progressively greater complexity.

- Dual Degree ‘Curriculum’

Curriculum Learning

Humans inherently break problems down to a sequence of manageable stages and sub-goals that are of progressively greater complexity.

- Dual Degree ‘Curriculum’
- Idea introduced in 1993 [27],
  made popular 2009 onwards [28]

Motivation

- Hand-engineered reward functions are too hard to get to work in real-world scenarios :

$$r(t) = r_1(t) + r_2(t) + \mathbf{3} r_3(t) + \mathbf{5} r_4(t)$$

Motivation

- Hand-engineered reward functions are too hard to get to work in real-world scenarios :

$$r(t) = r_1(t) + r_2(t) + \mathbf{3} r_3(t) + \mathbf{5} r_4(t)$$

- For driving?

Motivation

- Hand-engineered reward functions are too hard to get to work in real-world scenarios :

$$r(t) = r_1(t) + r_2(t) + \mathbf{3}r_3(t) + \mathbf{5}r_4(t)$$

- For driving?

Instead, let the agent learn how important each task is, along with learning the optimal policy for the same.

Related Work

Related Work

- Classical usage
  - multi-stage learning for language and vision tasks [28]

Related Work

- Classical usage
  - multi-stage learning for language and vision tasks [28]
- Task generation
  - from hand-coded [29] to learned tasks [30]

Related Work

- Classical usage
  - multi-stage learning for language and vision tasks [28]
- Task generation
  - from hand-coded [29] to learned tasks [30]
- Task sequencing
  - manual ordering to automatic sequencing [31]
  - catastrophic forgetting of older tasks [32]

Related Work

- Classical usage
  - multi-stage learning for language and vision tasks [28]
- Task generation
  - from hand-coded [29] to learned tasks [30]
- Task sequencing
  - manual ordering to automatic sequencing [31]
  - catastrophic forgetting of older tasks [32]
- Task encoding
  - naïve one-hot to principled approaches [33]

Task Generation

Domain knowledge is used to design to following sub-tasks in order to teach the agent to score goals :

1. Go to ball - the basic skill of approaching the ball
2. Dribble to goal - requires knowledge of (1)
3. Shoot - attempting to score a goal

Task Sequencing

A heuristic approach to cycle between the sub-tasks :

Task Sequencing

A heuristic approach to cycle between the sub-tasks :

Algorithm 2 Sequential Ordering

procedure LEARN
    current task index $i = EvaluateTasks()$
    while $iter < maxIter$ **do**
        $PlayEpisode(T_i)$ ▷ Play and learn on current task
        if $iter \,\%\, 10000 == 0$ **then**
            $i = EvaluateTasks()$ ▷ Update the task to be evaluated

function EVALUATETASKS
    for $i \in 1 \ldots |T|$ **do** ▷ Follow the ordering of tasks
        average return $R_i^{avg} = Evaluate(T_i)$
        if $R_i^{avg} < 0.8 \times R_i^{max}$ **then**
            return i ▷ Task T_i needs more training
    return $|T|$

Task Embeddings

$$\mathcal{T} = W^{emb} i$$

where vector i represents the one-hot encoding of the sub-task

Task Embeddings

$$\mathcal{T} = W^{emb} i$$

where vector i represents the one-hot encoding of the sub-task

1. State embedding - task embedding concatenated with agent's state representation vector

Task Embeddings

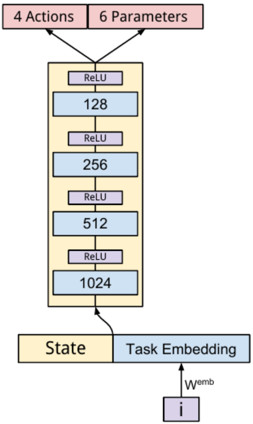
$$\mathcal{T} = W^{emb} i$$

where vector i represents the one-hot encoding of the sub-task

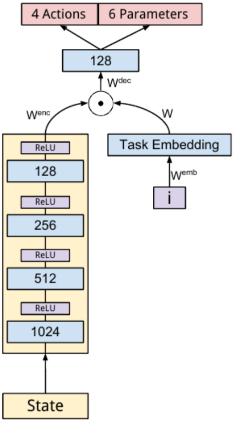
1. State embedding - task embedding concatenated with agent's state representation vector
2. Weight embedding - task embedding vector interacts multiplicatively with activations of agent's network

$$o = W^{dec}(W\mathcal{T} \odot W^{enc} h) + b$$

Task Embeddings



(a) State Embedding

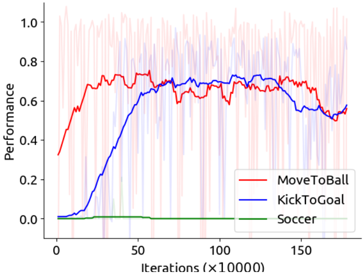


(b) Weight Embedding

Courtesy [24]

Overview

Results



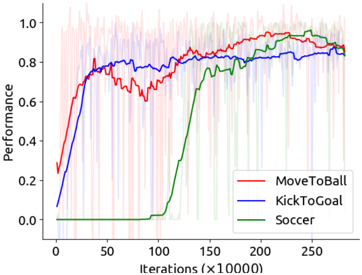


Figure: Performance on the three tasks of the two types of embeddings of size 128, using the sequential ordering

Results - Ablative Analysis

Importance of task embedding

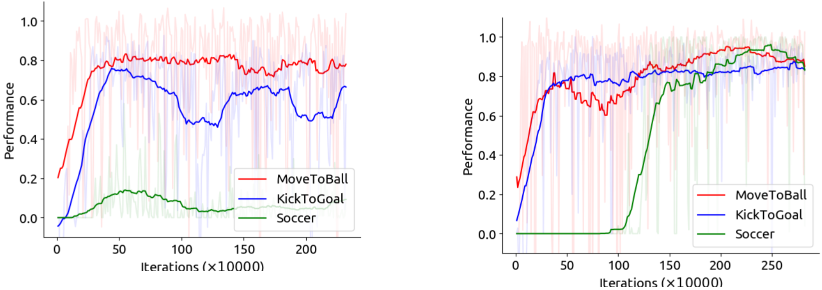
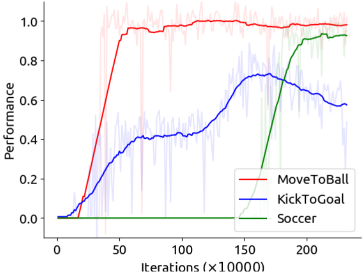


Figure: Performance of the agent trained naïvely with no embeddings versus the one trained with the weight embedding architecture (with the sequential ordering and embedding size 128)

Results - Ablative Analysis

Importance of task ordering



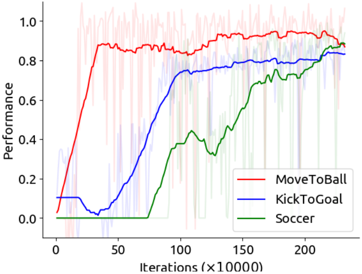


Figure: Performance of the agent trained with the sequential ordering and the lack of it using a weight embedding of size 8

Results - Additional Analysis

Size of embedding

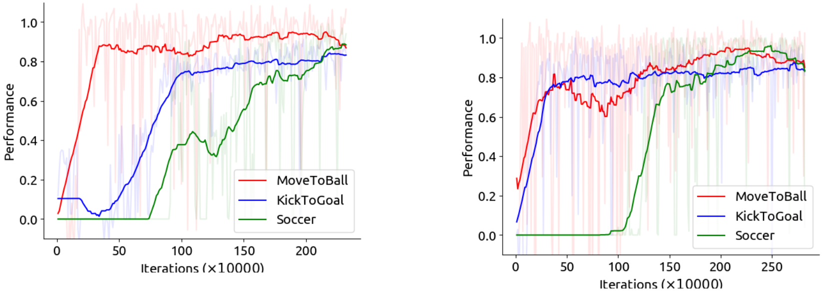


Figure: Performance of the agent trained using different sizes of embeddings of the weight embedding architecture - sizes 8 and 128

Takeaways

Takeaways

1. Tasks embeddings indeed help in discerning between different sub-tasks that have been designed to make the target task easier

Takeaways

1. Tasks embeddings indeed help in discerning between different sub-tasks that have been designed to make the target task easier
2. The order in which the sub-tasks are presented to the agent is critical in enabling stable learning as well as catastrophic forgetting of the tasks-at-hand

Takeaways

1. Tasks embeddings indeed help in discerning between different sub-tasks that have been designed to make the target task easier
2. The order in which the sub-tasks are presented to the agent is critical in enabling stable learning as well as catastrophic forgetting of the tasks-at-hand
3. The weight embedding architecture is fairly robust to the size of the embeddings used, with larger sizes encoding more and sufficient information.

Outline

Summary

Summary

1. **Risk-Averse Imitation Learning** - identified a drawback with the existing SOTA algorithm for learning a behavioral policy from a fixed set of expert trajectories, and proposed a viable alternative for application in risk-sensitive applications.

Summary

1. **Risk-Averse Imitation Learning** - identified a drawback with the existing SOTA algorithm for learning a behavioral policy from a fixed set of expert trajectories, and proposed a viable alternative for application in risk-sensitive applications.
2. **Multi-Agent Learning** - developed the first open-source, fully-controllable Multi-Agent DRiving Simulator, and identified problems of non-stationarity and reward-engineering in the multi-agent domain.

Summary

1. **Risk-Averse Imitation Learning** - identified a drawback with the existing SOTA algorithm for learning a behavioral policy from a fixed set of expert trajectories, and proposed a viable alternative for application in risk-sensitive applications.
2. **Multi-Agent Learning** - developed the first open-source, fully-controllable Multi-Agent DRiving Simulator, and identified problems of non-stationarity and reward-engineering in the multi-agent domain.
3. **Curriculum Learning** - broke down the sparse reward goal-scoring task of RoboSoccer into smaller, individual sub-tasks and demonstrated the importance of each proposed module.

Ultimate Goal

Revolutionizing the transportation industry by safely and reliably deploying a homogeneous set of connected self-driving vehicles on our roads.

Ultimate Goal

Self-driving cars zipping through the streets,

- ferrying commuters from place-to-place *safely and reliably*
- having record-low accident rates
- eliminating the need for traffic signals and signs
- in which we can eat, sleep, spend time with our family
- running on renewable sources of energy
- available at the tap of an app.

Ultimate Goal

Self-driving cars zipping through the streets of India,

- ferrying commuters from place-to-place *safely and reliably*
- having record-low accident rates
- eliminating the need for traffic signals and signs
- in which we can eat, sleep, spend time with our family
- running on renewable sources of energy
- available at the tap of an app.

References I

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al.
Mastering the game of go with deep neural networks and tree search.
nature, 529(7587):484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al.
Mastering the game of go without human knowledge.
Nature, 550(7676):354, 2017.

Gary Yen and Travis Hickey.
Reinforcement learning algorithms for robotic navigation in dynamic environments.
In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, volume 2, pages 1444–1449. IEEE, 2002.

Andrew Y Ng, Adam Coates, Mark Diel, Varun Ganapathi, Jamie Schulte, Ben Tse, Eric Berger, and Eric Liang.
Autonomous inverted helicopter flight via reinforcement learning.
In *Experimental Robotics IX*, pages 363–372. Springer, 2006.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al.
Human-level control through deep reinforcement learning.
Nature, 518(7540):529–533, 2015.

References II

Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling.
Deepstack: Expert-level artificial intelligence in heads-up no-limit poker.
Science, 356(6337):508–513, 2017.

Stefan Schaal.
Learning from demonstration.
In *Advances in Neural Information Processing Systems*, pages 1040–1046, 1997.

Dean A Pomerleau.
Alvinn: An autonomous land vehicle in a neural network.
In *Advances in Neural Information Processing Systems*, pages 305–313, 1989.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al.
End to end learning for self-driving cars.
arXiv preprint arXiv:1604.07316, 2016.

Stéphane Ross and Drew Bagnell.
Efficient reductions for imitation learning.
In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 661–668, 2010.

Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell.
A reduction of imitation learning and structured prediction to no-regret online learning.
In *International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

References III

Pieter Abbeel and Andrew Y Ng.
Apprenticeship learning via inverse reinforcement learning.
In *Proceedings of the 21st International Conference on Machine Learning*, page 1. ACM, 2004.

Sergey Levine and Vladlen Koltun.
Continuous inverse optimal control with locally optimal examples.
arXiv preprint arXiv:1206.4617, 2012.

Jonathan Ho and Stefano Ermon.
Generative adversarial imitation learning.
In *Advances in Neural Information Processing Systems*, pages 4565–4573, 2016.

Javier Garcıa and Fernando Fernández.
A comprehensive survey on safe reinforcement learning.
Journal of Machine Learning Research, 16(1):1437–1480, 2015.

R Tyrrell Rockafellar and Stanislav Uryasev.
Optimization of conditional value-at-risk.
Journal of risk, 2:21–42, 2000.

Ming Tan.
Multi-agent reinforcement learning: Independent vs. cooperative agents.
In *Proceedings of the tenth international conference on machine learning*, pages 330–337, 1993.

References IV

Junling Hu and Michael P Wellman.
Nash q-learning for general-sum stochastic games.
Journal of machine learning research, 4(Nov):1039–1069, 2003.

Michael Bowling and Manuela Veloso.
Multiagent learning using a variable learning rate.
Artificial Intelligence, 136(2):215–250, 2002.

Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente.
Multiagent cooperation and competition with deep reinforcement learning.
PloS one, 12(4):e0172395, 2017.

Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P How, and John Vian.
Deep decentralized multi-task multi-agent reinforcement learning under partial observability.
In *International Conference on Machine Learning*, pages 2681–2690, 2017.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch.
Multi-agent actor-critic for mixed cooperative-competitive environments.
In *Advances in Neural Information Processing Systems*, 2017.

Xiangxiang Chu and Hangjun Ye.
Parameter sharing deep deterministic policy gradient for cooperative multi-agent reinforcement learning.
arXiv preprint arXiv:1710.00336, 2017.

References V

M. Hausknecht and P. Stone.
Deep Reinforcement Learning in Parameterized Action Space.
In *Proceedings of the 4th International Conference on Learning Representations (ICLR-16)*, 2016.

Roberta Raileanu, Emily Denton, Arthur Szlam, and Rob Fergus.
Modeling others using oneself in multi-agent reinforcement learning.
arXiv preprint arXiv:1802.09640, 2018.

Jakob Foerster, Yannis Assael, Nando de Freitas, and Shimon Whiteson.
Learning to communicate with deep multi-agent reinforcement learning.
In *Advances in Neural Information Processing Systems*, pages 2137–2145, 2016.

Jeffrey L Elman.
Learning and development in neural networks: The importance of starting small.
Cognition, 48(1):71–99, 1993.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston.
Curriculum learning.
In *Proceedings of the 26th annual International Conference on Machine Learning*, pages 41–48. ACM, 2009.

Andrej Karpathy and Michiel Van De Panne.
Curriculum learning for motor skills.
Advances in Artificial Intelligence, pages 325–330, 2012.

References VI

Sanmit Narvekar, Jivko Sinapov, Matteo Leonetti, and Peter Stone.
Source task creation for curriculum learning.
In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 566–574. International Foundation for Autonomous Agents and Multiagent Systems, 2016.

Sanmit Narvekar, Jivko Sinapov, and Peter Stone.
Autonomous task sequencing for customized curriculum design in reinforcement learning.
In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, August 2017.

Robert M French.
Catastrophic forgetting in connectionist networks.
Trends in cognitive sciences, 3(4):128–135, 1999.

Matthew John Hausknecht.
Cooperation and communication in multiagent deep reinforcement learning.
PhD thesis, The University of Austin, 2016.

Thank You.

Questions?