King Abdullah University of Science and Technology

KAUST

Computer Vision meets Robotics

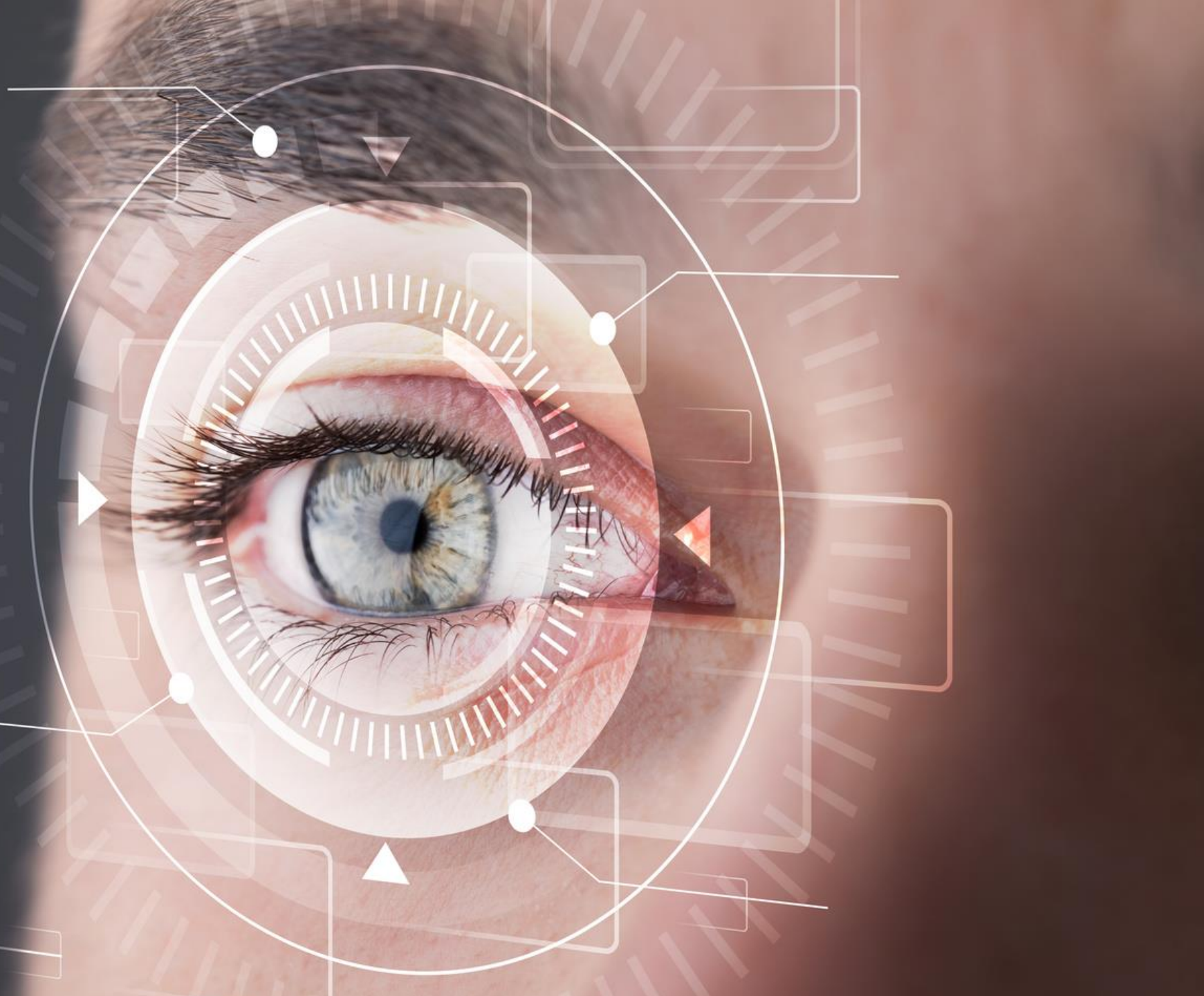
DEEP REINFORCEMENT FOR SENSORIMOTOR CONTROL

GROUP MEETING – 30.03.2017

by
Matthias Mueller

80%

- Cold
- 5 flavors
- Red Wine
- Music
- Partner



Overview

- Why Deep Learning?
- Reinforcement Learning (Recap)
- Deep Reinforcement Learning (Recap)
  - Deep Q-Network (DQN)
  - Policy Gradients (PG)
- Direct Future Prediction (DFP)

Deep Learning Use Cases

General use case	Industry
Sound	
Voice recognition	UX/UI, Automotive, Security, IoT
Voice search	Handset maker, Telecoms
Sentiment analysis	CRM
Flaw detection (engine noise)	Automotive, Aviation
Fraud detection (latent audio artifacts)	Finance, Credit Cards
Time Series	
Log analysis/Risk detection	Data centers, Security, Finance
Enterprise resource planning	Manufacturing, Auto., Supply chain
Predictive analysis using sensor data	IoT, Smart home, Hardware manufact.
Business and Economic analytics	Finance, Accounting, Government
Recommendation engine	E-commerce, Media, Social Networks

Text	
Sentiment Analysis	CRM, Social media, Reputation mgt.
Augmented search, Theme detection	Finance
Threat detection	Social media, Govt.
Fraud detection	Insurance, Finance
Image	
Facial recognition	
Image search	Social media
Machine vision	Automotive, aviation
Photo clustering	Telecom, Handset makers
Video	
Motion detection	Gaming, UX, UI
Real-time threat detection	Security, Airports

https://deeplearning4j.org/use_cases.html

Deep Learning as a bridge between fields

- Earth Science
- Marine Science
- Medicine
- Engineering
- Computer Science
- Robotics

Sergey Levine

Assistant Professor, UC Berkeley, EECS

Address:
754 Sutardja Dai Hall
UC Berkeley
Berkeley, CA 94720-1758

News and Announcements

News

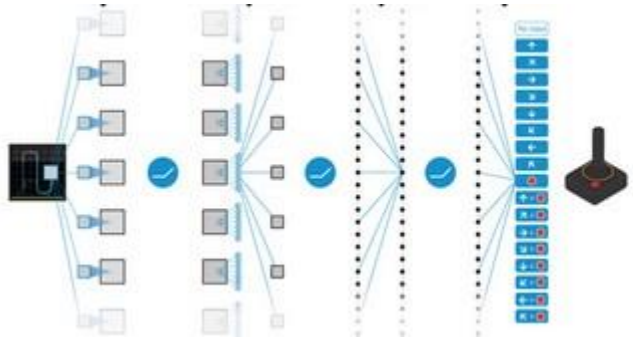
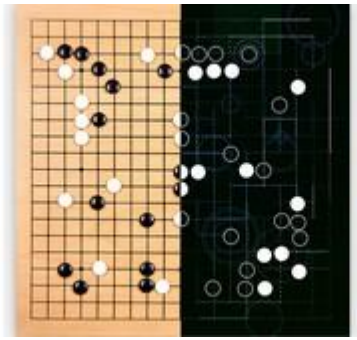
March 14, 2017	Six new papers on deep reinforcement learning posted: Uncertainty-Aware Reinforcement Learning, Model-Agnostic Meta-Learning, Reinforcement Learning with Deep Energy-Based Policies, Exploration with Exemplar Models, Combining Model-Based and Model-Free Updates, and Learning Invariant Feature Spaces (accepted to ICLR 2017).
March 1, 2017	Two new papers on deep robotic learning posted: Cognitive Mapping, accepted at CVPR 2017, and Rope Manipulation, accepted at ICRA 2017.
February 17, 2017	Videos of the lectures from our NIPS 2016 Workshop on Deep Learning for Action and Interaction are posted here.
February 10, 2017	Five papers accepted at the International Conference on Learning Representations (ICLR), including one oral presentation! Three are available below, the rest are coming soon.
January 21, 2017	Nine papers accepted at the International Conference on Robotics and Automation (ICRA). Eight available below, the ninth is coming soon.
January 14, 2017	Two new preprints on deep reinforcement learning posted!
November 25, 2016	Four new preprints posted and one technical report on GANs, IRL, and EBMs posted!
November 18, 2016	I will be co-organizing a new conference in November 2017: CoRL (Conference on Robotic Learning) will take place in mid-November in Mountain View, California.
October 4, 2016	Six new preprints on deep reinforcement learning and deep robotic learning posted!
October 3, 2016	Google Research Blog post written with Timothy Lillicrap and Mrinal Kalakrishnan on our latest work at Google is now posted! Nice summary from MIT Technology Review here.
September 29, 2016	Three new preprints on deep robotic learning and deep reinforcement learning posted!

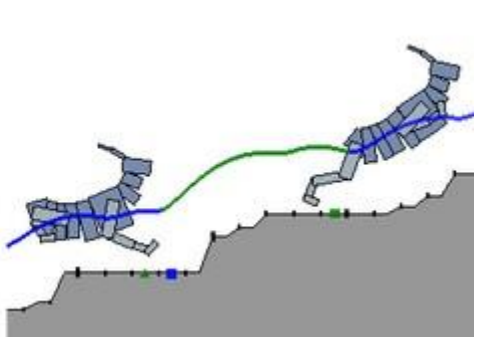
Reinforcement Learning Approaches

- Monte Carlo
- TD learning: TD(0) and TD(λ)
- Q-Learning: Q(0) and Q(λ)
- SARSA

Deep Reinforcement Learning (Recap)

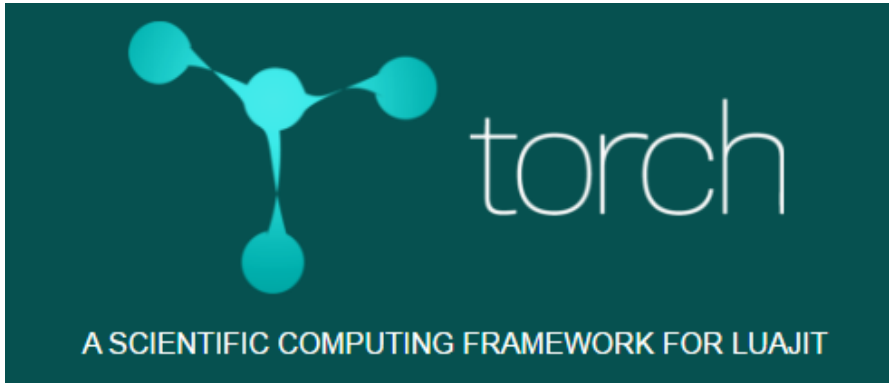
- Automatically learn to play ATARI games (from raw game pixels!)
- Beat world champions at Go
- Robots are learning how to perform complex manipulation tasks that defy explicit programming
- Simulated quadrupeds are learning to run and leap



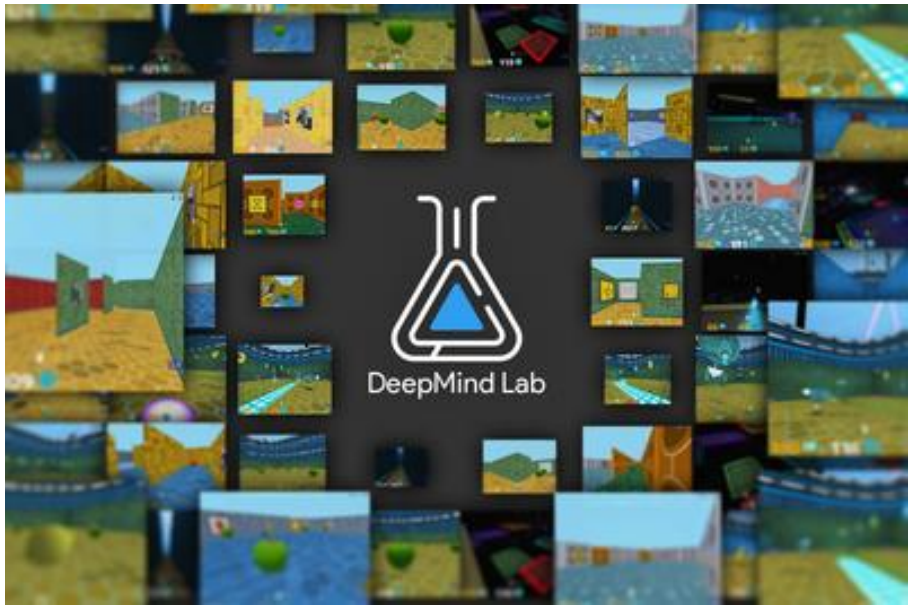
TorchCraft (Facebook)

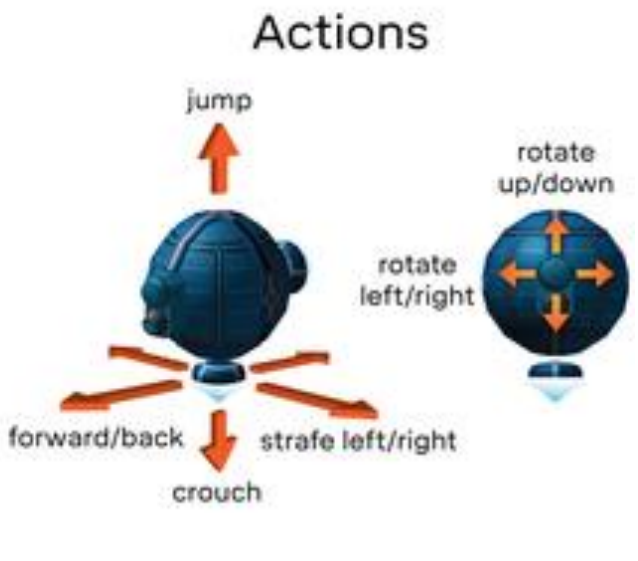
- https://github.com/TorchCraft/TorchCraft



DeepMind Lab (DeepMind)

- https://deepmind.com/blog/open-sourcing-deepmind-lab/





OpenAI Universe (OpenAI)

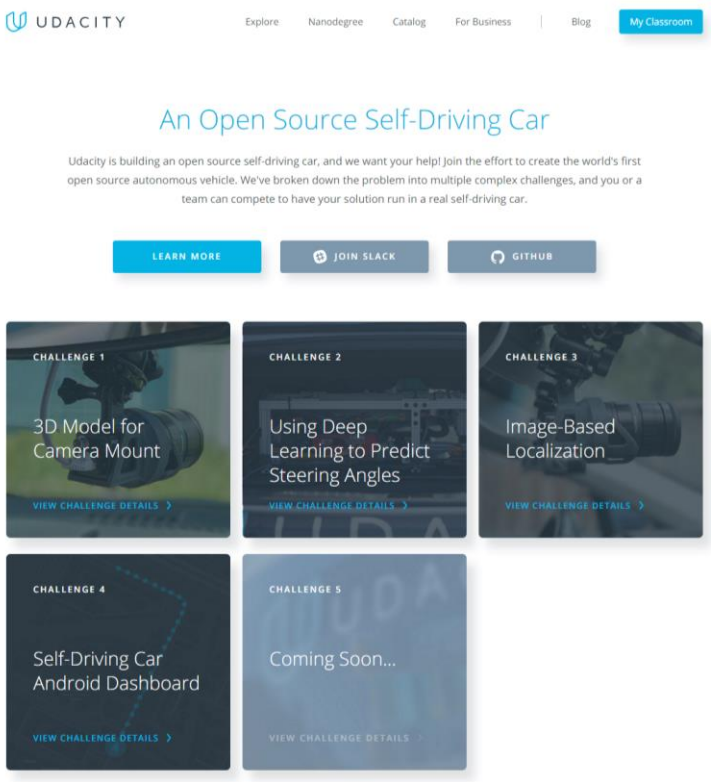
- https://universe.openai.com/

Human-like interface

Agents use the same senses and controls as humans: seeing pixels and using a keyboard and mouse. Universe makes it possible to train a single agent on any task a human can complete with a computer.

Self-Driving Car Sim (Udacity)

- https://github.com/udacity/self-driving-car-sim



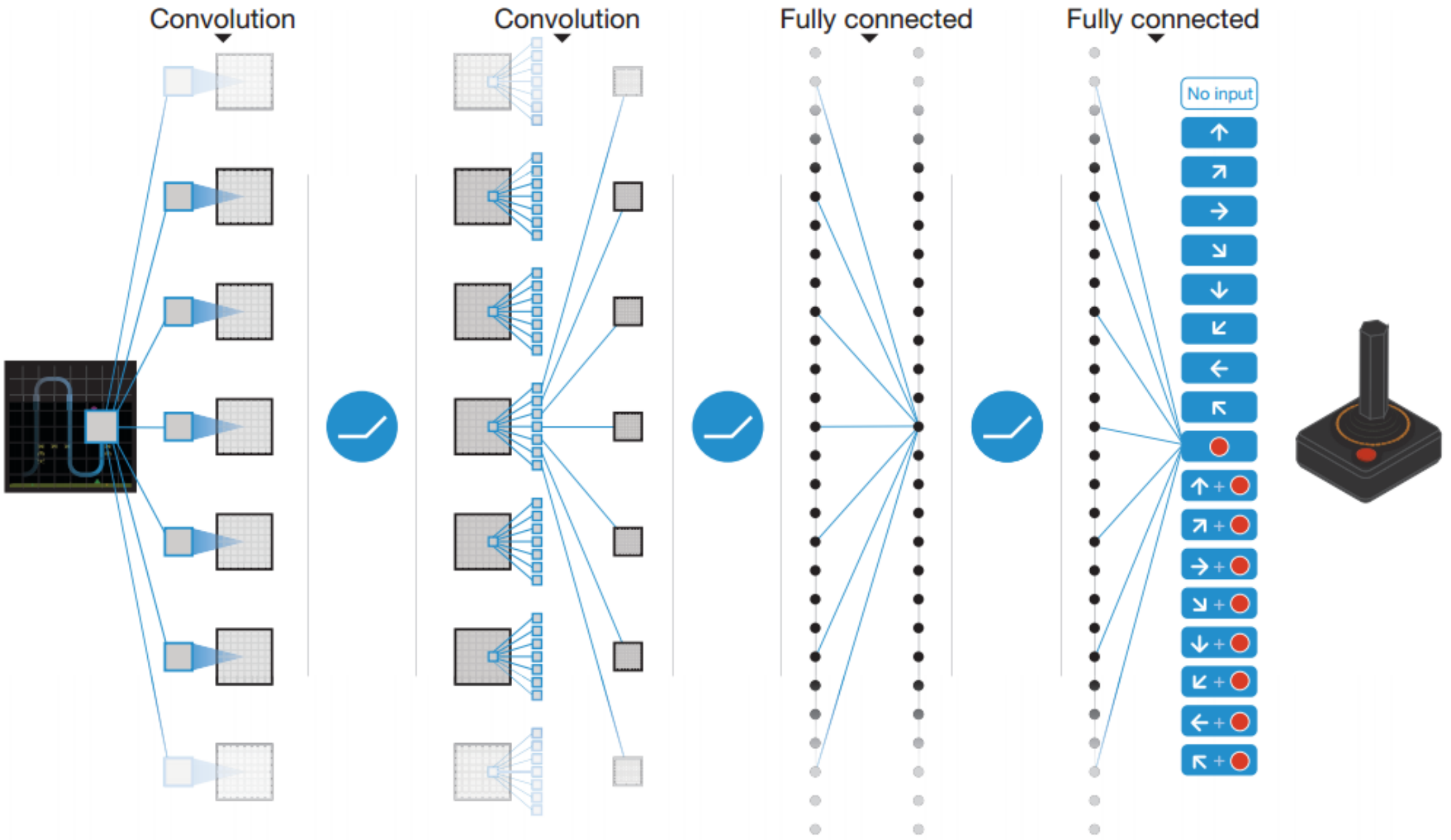
Deep Q-Network

- Use deep neural network to approximate

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots \middle| s_t = s,\ a_t = a,\ \pi\right]$$

- The Q-learning update at iteration i uses the following loss function

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s') \sim \mathrm{U}(D)} \left[\left(r + \gamma \max_{a'} Q(s',a';\theta_i^-) - Q(s,a;\theta_i) \right)^2 \right]$$


Convolution
Convolution
Fully connected
Fully connected
No input

Algorithm 1: deep Q-learning with experience replay.

Initialize replay memory D to capacity N

Initialize action-value function Q with random weights θ

Initialize target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$

For episode = 1, M **do**

Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$

For t = 1,T **do**

With probability ε select a random action a_t

otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$

Execute action a_t in emulator and observe reward r_t and image x_{t+1}

Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$

Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D

Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D

Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-}) & \text{otherwise} \end{cases}$

Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters θ

Every C steps reset $\hat{Q} = Q$

End For

End For

Policy Gradients (PG)

- Recently, most people prefer to use Policy Gradients, including the authors of the original DQN
- PG is preferred because it is end-to-end
- There's an explicit policy and a principled approach that directly optimizes the expected reward
- Example: learn to play an ATARI game (Pong!) with PG, from scratch, from pixels, with a deep neural network

Policy Gradient Methods: Overview

Problem:

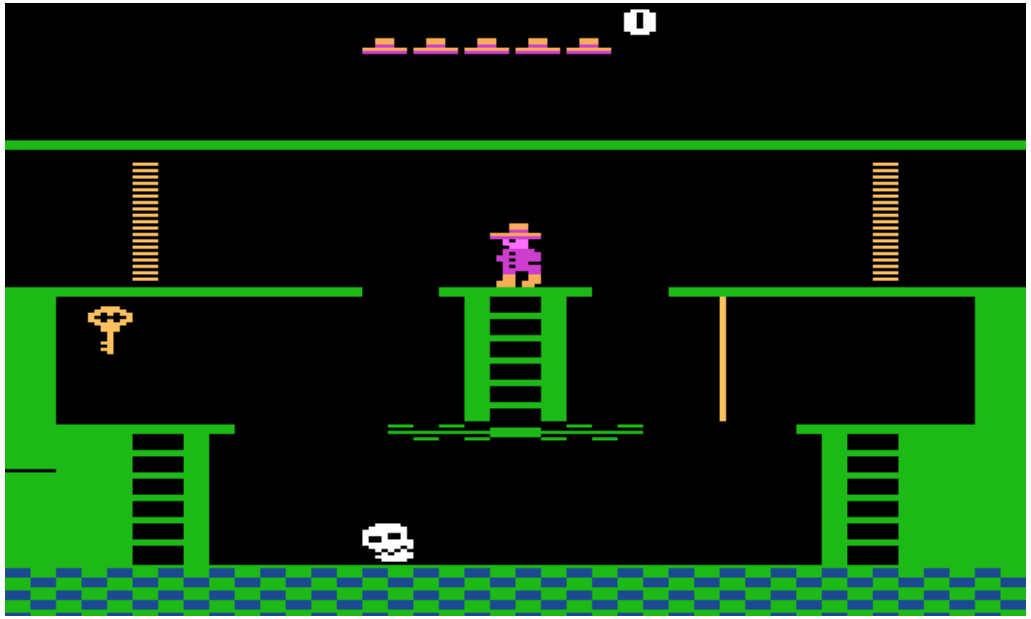
$$\text{maximize } E[R \mid \pi_\theta]$$

Intuitions: collect a bunch of trajectories, and ...

1. Make the good trajectories more probable
2. Make the good actions more probable (actor-critic, GAE)
3. Push the actions towards good actions (DPG, SVG)

What are RL machines good at?

- RL machines excel in games with frequent reward signals, that requires precise play, fast reflexes, and not too much long-term planning
- RL machines fail in games where a rich, abstract model of the game is necessary



Human vs. Machine

- Human understands the objective of the game and infers the reward.
- Human brings in a huge amount of prior knowledge, such as intuitive physics and intuitive psychology.
- Humans build a rich, abstract model and plan within it. Humans can figure out what is likely to give rewards without ever actually experiencing the rewarding or unrewarding transition.

- RL machine assumes an arbitrary reward function and updates it through environment interactions.
- RL machine starts from scratch and policy gradients are a brute force solution, where the correct actions are eventually discovered and internalized into a policy.
- RL machine actually has to experience a positive reward, and experience it very often in order to eventually and slowly shift the policy parameters towards repeating moves that give high rewards.

Direct Future Prediction (DFP)

- Approach to sensorimotor control in immersive environments
- Semi-supervised learning while learning from raw experience
- Given present sensory input, measurements, and goal, the agent can be trained to predict the effect of different actions on future measurements

Direct Future Prediction (DFP)

- High-dimensional sensory stream $\{s_t\}$
  - e.g. raw visual, auditory, and tactile input
- Lower-dimensional measurement stream $\{m_t\}$
  - e.g. physical system: attitude, supply levels, and structural integrity
  - e.g. computer game: health, ammunition levels, number of killed adversaries

→ Rich and temporally dense supervision

Advantages over Deep RL

- Learning without a fixed goal at training time
- Pursue dynamically changing goals at test time
- Outperforms state-of-the-art deep RL models, particularly on complex tasks (Winner of Visual Doom AI Competition)
- Models generalize across environments and goals
- Use of frequent multi-dimensional measurements instead of a scalar reward is beneficial

Model

- At each time step t, the agent receives an observation o_t and executes an action a_t based on this observation
- Observations have the following structure: $o_t = \{s_t, m_t\}$ where s and m denote sensory input and measurement vector respectively
- The agent aims to predict future measurements
- Goals of the agent can be defined in terms of future measurements

Model

- Let $\tau_1, \ldots, \tau_n$ be a set of temporal offsets
- Let $\mathbf{f} = \{\mathbf{m}_{t+\tau_1} - \mathbf{m}_t, \ldots, \mathbf{m}_{t+\tau_n} - \mathbf{m}_t\}$ be the corresponding differences of future and present measurements
- Assume any goal that the agent will pursue can be defined as maximization of some function $u(\mathbf{f}; \mathbf{g})$, where the vector $\mathbf{g}$ parameterizes the goal
- For goals that can be expressed as linear combinations of future measurements: $u(\mathbf{f}; \mathbf{g}) = \mathbf{g}^T \mathbf{f}$

Model

- Future measurements are predicted with a parameterized function approximator, denoted by F:

- $\mathbf{p}_t^a = F(\mathbf{o}_t, a, \mathbf{g}; \boldsymbol{\theta})$, where $a \in A$ is an action, $\boldsymbol{\theta}$ contains the learned parameters of F, and $\mathbf{p}_a^t$ is the resulting prediction

- At test time, given learned parameters $\boldsymbol{\theta}$, the agent can choose the action that yields the best predicted outcome:

- $a_t = \underset{a \in A}{\operatorname{argmax}}\ \mathbf{g}^T F(\mathbf{o}_t, a, \mathbf{g}; \boldsymbol{\theta})$

Training

- Consider a set of experiences collected by the agent, yielding a set D of training examples: $D = \{\mathbf{o}_i, a_i, \mathbf{g}_i, \mathbf{f}_i\}_{i=1}^{N}$ where $\{\mathbf{o}_i, a_i, \mathbf{g}_i, \mathbf{f}_i\}$ is the input and $\mathbf{f}_i$ is the output of example i.

- The predictor is trained using a regression loss:
$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} \|F(\mathbf{o}_i, a_i, \mathbf{g}_i; \boldsymbol{\theta}) - \mathbf{f}_i\|^2$

- The parameters of the predictor used by the agent are updated after every k new experiences

- Agent follows an ϵ-greedy policy: it acts greedily according to the current goal with probability 1 ϵ, and selects a random action with probability ϵ

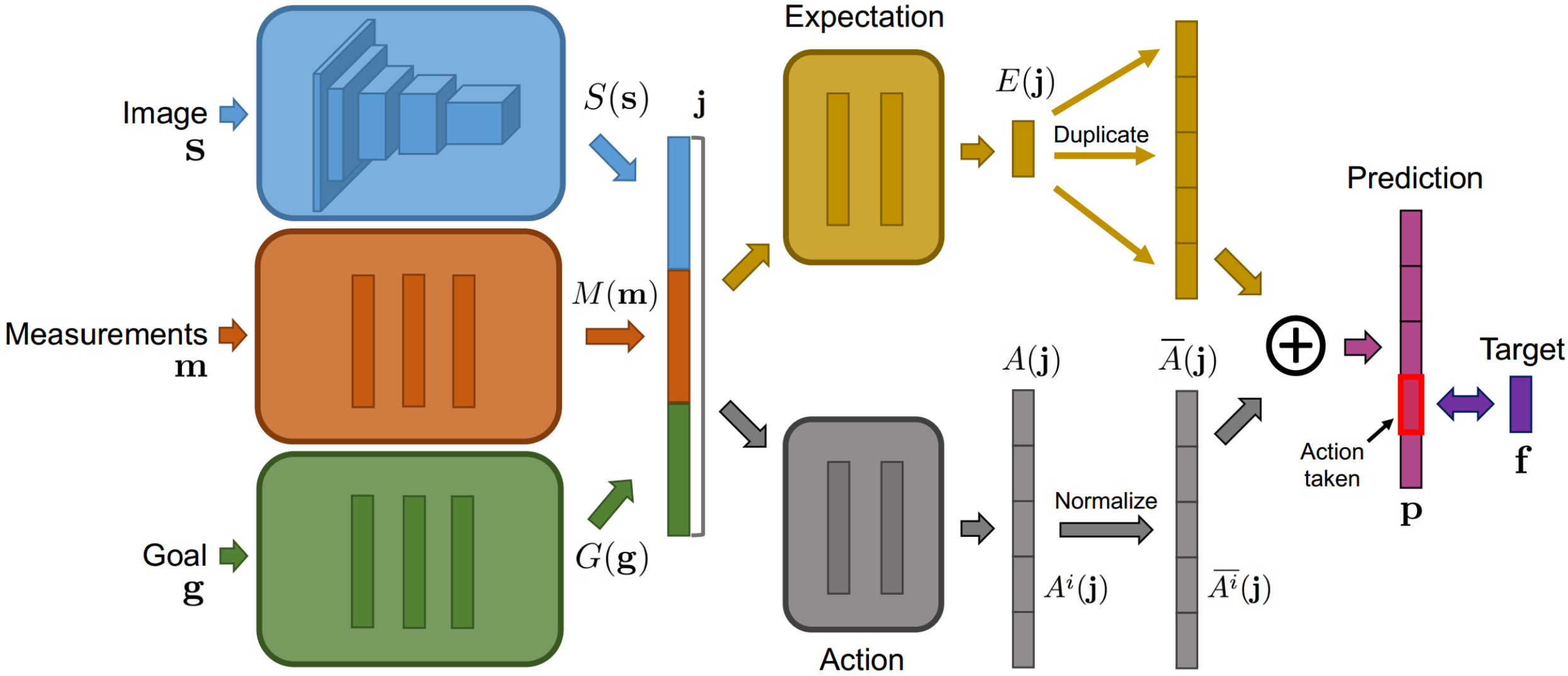
Architechture

- The predictor F is a deep network parameterized by $\boldsymbol{\theta}$
- The network has three input modules: a perception module $S(\mathbf{s})$, a measurement module $M(\mathbf{m})$ and a goal module $G(\mathbf{g})$
- The outputs of the three input modules are concatenated, forming the joint input representation used for subsequent processing:

$$\mathbf{j} = J(\mathbf{s}, \mathbf{m}, \mathbf{g}) = \{S(\mathbf{s}), M(\mathbf{m}), G(\mathbf{g})\}$$

Architechture

- Prediction module is split into two streams: an expectation stream $E(\mathbf{j})$ and an action stream $A(\mathbf{j})$
- The expectation stream predicts the average of the future measurements over all potential actions
- The action stream concentrates on the fine differences between actions: $A(\mathbf{j} = \{A^1(\mathbf{j}), \ldots, A^w(\mathbf{j})\}$, where $w = |A|$ is the number of actions



Learning to Act by Predicting the Future, Alexey Dosovitskiy and Vladlen Koltun, International Conference on Learning Representations (ICLR), 2017 (Oral)

Experiments

- Comparison to prior work

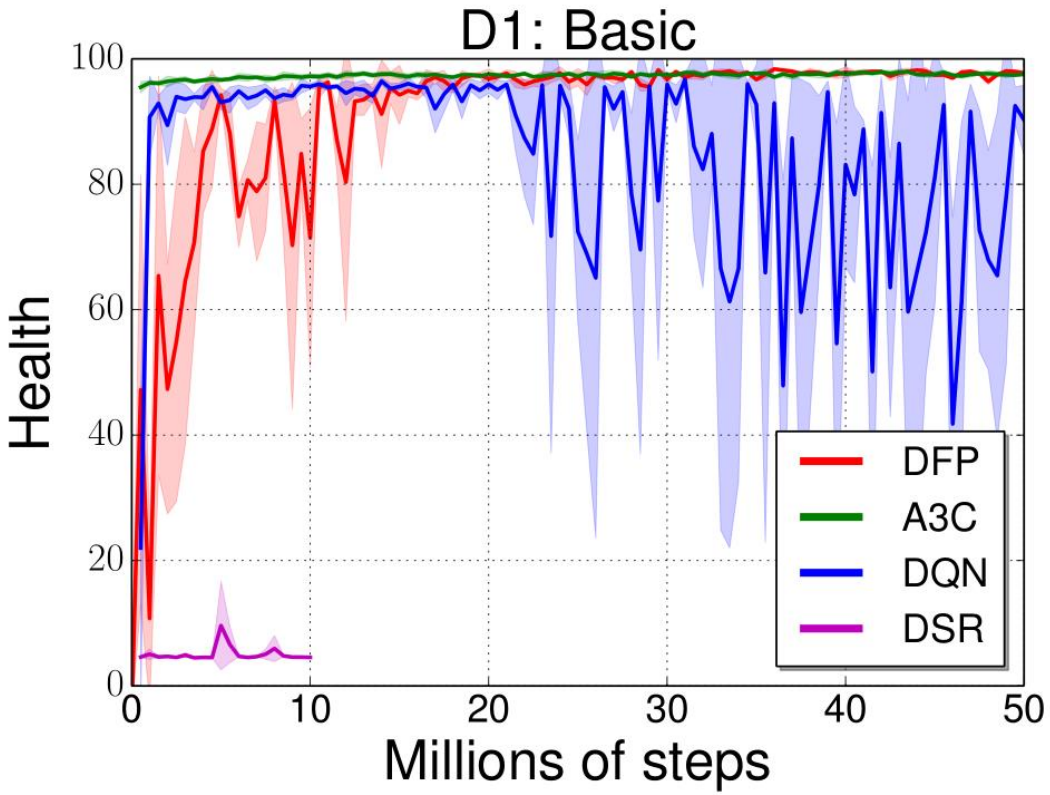
	D1 (health)	D2 (health)	D3 (frags)	D4 (frags)	steps/day
DQN	89.1 ± 6.4	25.4 ± 7.8	1.2 ± 0.8	0.4 ± 0.2	7M
A3C	$\mathbf{97.5 \pm 0.1}$	59.3 ± 2.0	5.6 ± 0.2	6.7 ± 2.9	80M
DSR	4.6 ± 0.1	–	–	–	1M
DFP	$\mathbf{97.7 \pm 0.4}$	$\mathbf{84.1 \pm 0.6}$	$\mathbf{33.5 \pm 0.4}$	$\mathbf{16.5 \pm 1.1}$	70M

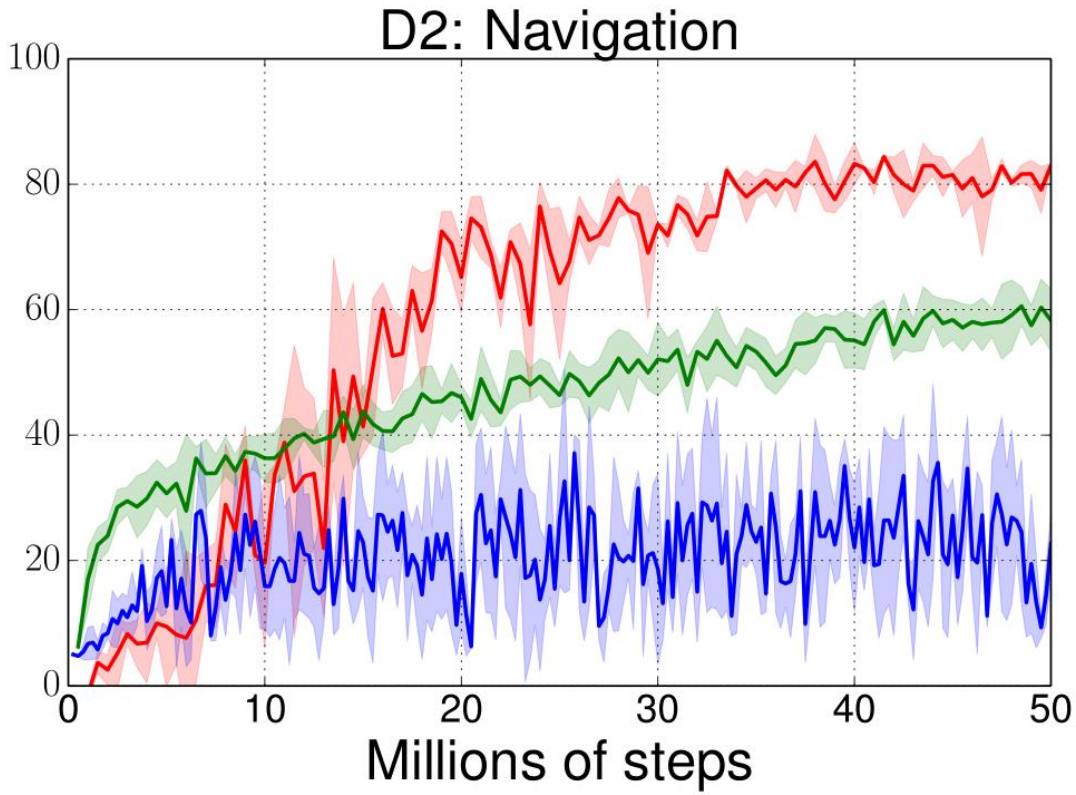
Experiments

D1: Basic

D2: Navigation

Experiments





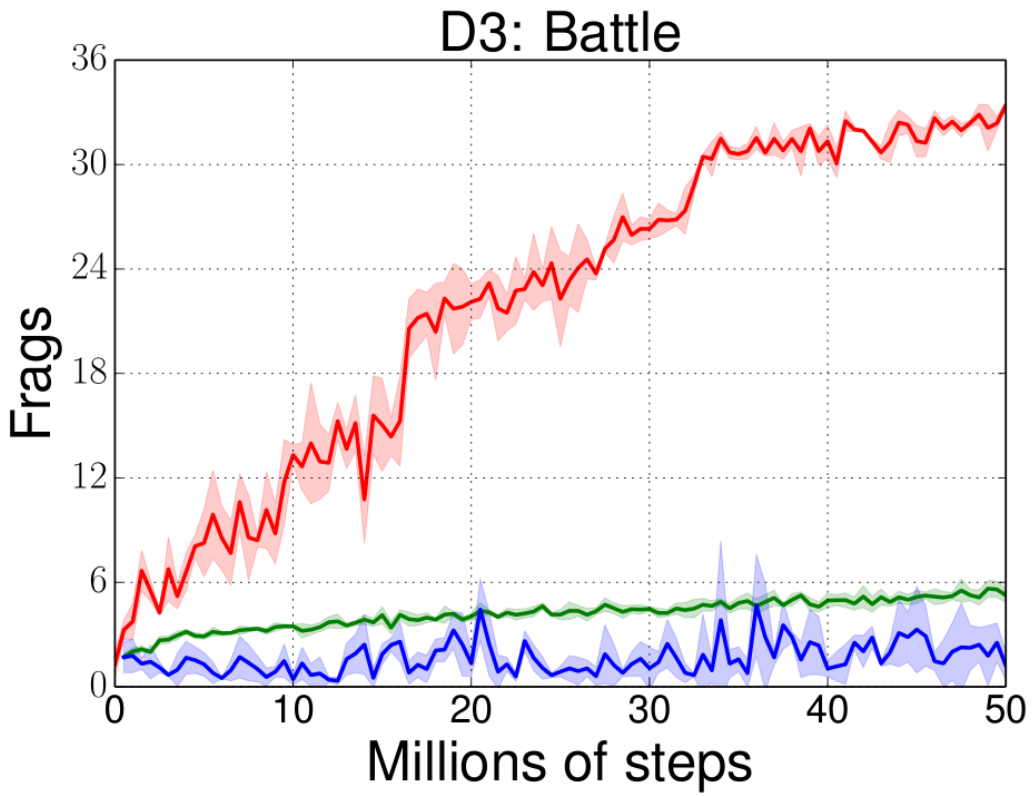
Learning to Act by Predicting the Future, Alexey Dosovitskiy and Vladlen Koltun, International Conference on Learning Representations (ICLR), 2017 (Oral)

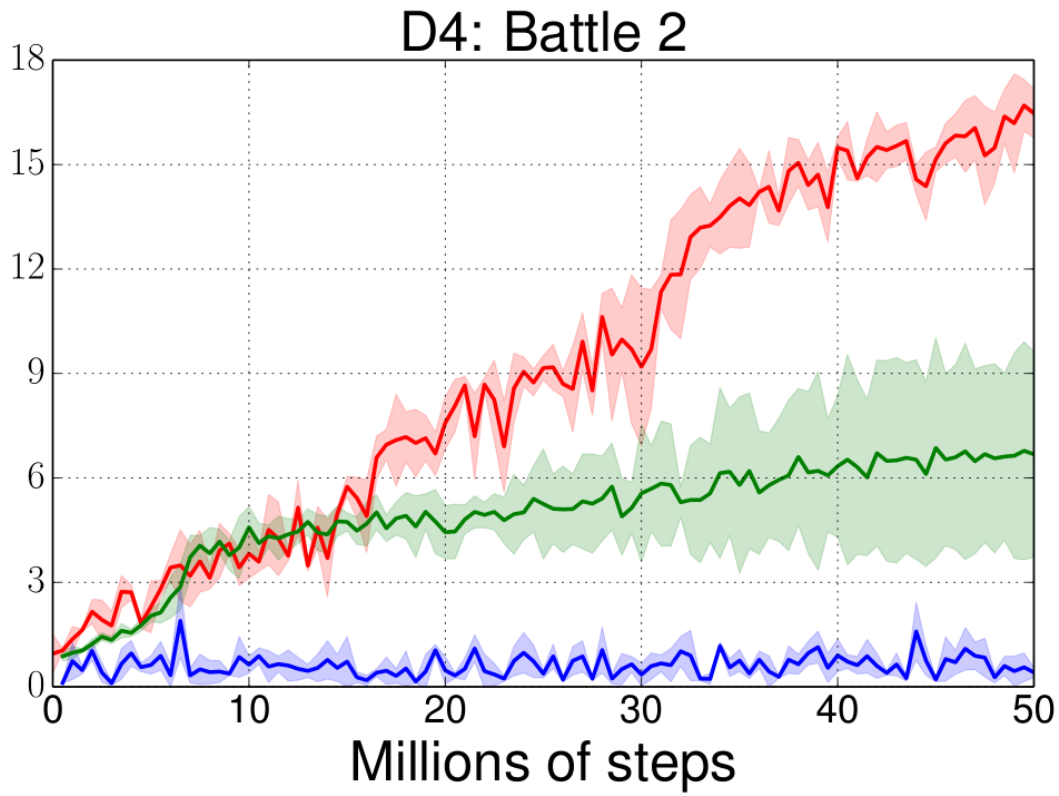
Experiments

D3: Battle

D4: Battle 2

Experiments






Our approach
AMMO
HEALTH
ARMS
ARMOR

Experiments

- Generalization across environments

		Train				
		D3	D4	D3-tx	D4-tx	D4-tx-L
Test	D3	33.6	17.8	29.8	20.9	22.0
	D4	1.6	17.1	5.4	10.8	12.4
	D3-tx	3.9	8.1	22.6	15.6	19.4
	D4-tx	1.7	5.1	6.2	10.2	12.7

Experiments

- Goal-agnostic training

	(a) fixed goal $(0.5, 0.5, 1)$			(b) random goals $[0, 1]$			(c) random goals $[-1, 1]$		
test goal	ammo	health	frags	ammo	health	frags	ammo	health	frags
$(0.5, 0.5, 1)$	83.4	97.0	33.6	92.3	96.9	31.5	49.3	94.3	28.9
$(0, 0, 1)$	0.3	-3.7	11.5	4.3	30.0	20.6	21.8	70.9	24.6
$(1, 1, -1)$	28.6	-2.0	0.0	22.1	4.4	0.2	89.4	83.6	0.0
$(-1, 0, 0)$	1.0	-8.3	1.7	1.9	-7.5	1.2	0.9	-8.6	1.7
$(0, 1, 0)$	0.7	2.7	2.6	9.0	77.8	6.6	3.0	69.6	7.9

Experiments

- Ablation study

		frags
all measurements	all offsets	22.6
all measurements	one offset	17.2
frags only	all offsets	10.3
frags only	one offset	5.0

References & Resources

- Learning to Act by Predicting the Future (ICLR, 2017)
- Playing FPS Games with Deep Reinforcement Learning (AIII, 2017)
- Deep Reinforcement Learning - An Overview (Archive, 2017)
- Loss is its own Reward: Self-Supervision for Reinforcement Learning (Archive, 2017)
- One-Shot Imitation Learning (Archive, 2017)
- Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World (Archive, 2017)
- Cognitive Mapping and Planning for Visual Navigation (CVPR, 2017)
- Playing Doom with SLAM-Augmented Deep Reinforcement Learning (Archive, 2016)

References & Resources

- Playing Atari with Deep Reinforcement Learning (Archive, 2013)
- High-Dimensional Continuous Control Using Generalized Advantage Estimation (Archive, 2015)
- Human-level control through deep reinforcement learning (Nature, 2015)
- Gradient Estimation Using Stochastic Computation Graphs (NIPS, 2015)
- Asynchronous Methods for Deep Reinforcement Learning (Archive, 2016)
- http://rll.berkeley.edu/deeprlcourse/
- http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html
- https://www.nervanasys.com/demystifying-deep-reinforcement-learning/
- http://karpathy.github.io/2016/05/31/rl/