

Playing Atari with Deep Reinforcement Learning (13.12)

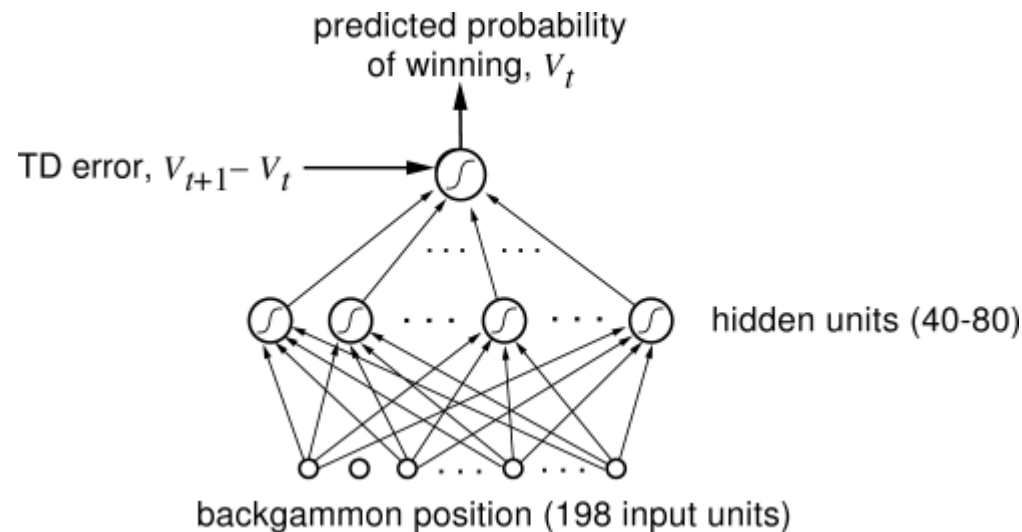
Seungjae Ryan Lee

Previous Applications of RL

- Linear value functions or policy representations
 - Rely on hand-crafted features
 - Feature representation determines performance
- Can diverge with model-free RL, nonlinear approximation, off-policy

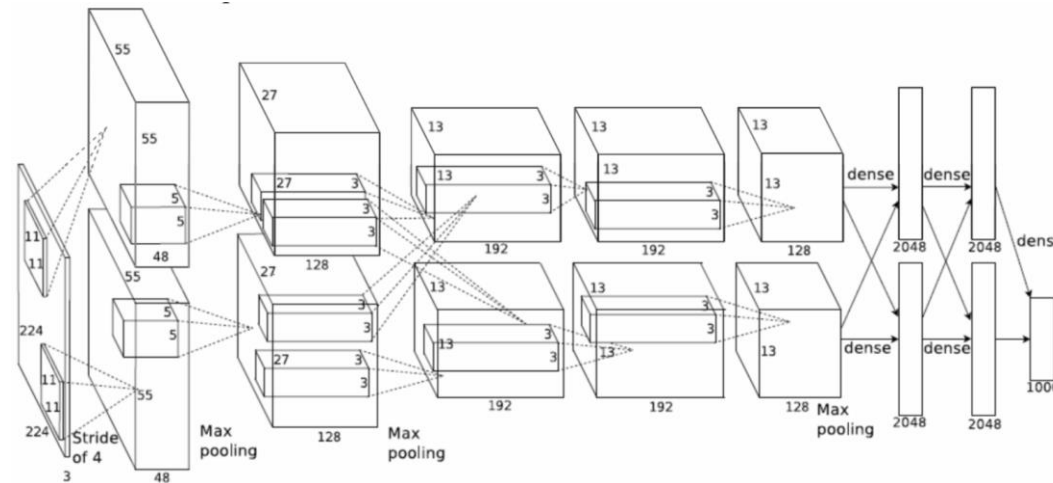
TD-gammon

- Superhuman-level Backgammon playing RL agent
- Model-free algorithm with one-hidden-layer MLP
- Considered a “special case”



Meanwhile, in Deep Learning...

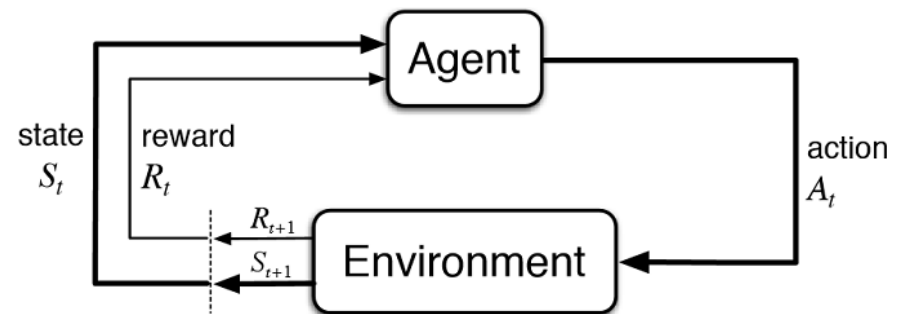
- CNN, MLP, RNN
 - Extract high-level features from raw data
 - Works in both supervised and unsupervised learning



- Will it work on Reinforcement Learning?

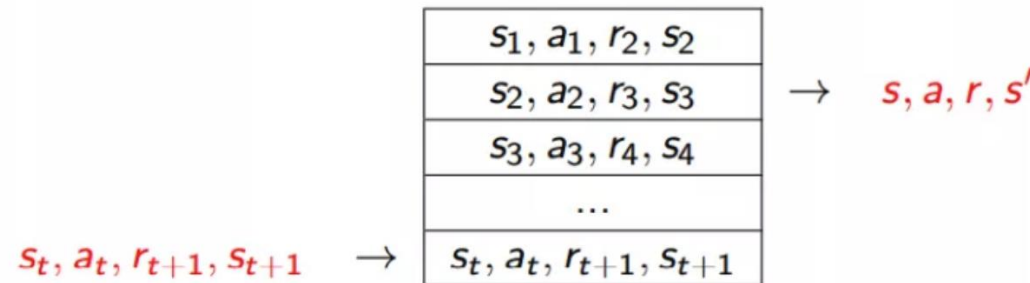
Problems of combining DL and RL

- No labelled data: only sparse, noisy, delayed rewards
- Breaks most underlying assumptions of deep learning algorithms
 - Highly correlated data
 - Moving distribution of data



Proposal: Experience Replay

- Idea originally by Long-Ji Lin
- Save transitions in a *replay memory* D
- Randomly sample previous transitions to update parameters



Advantages of Experience Replay

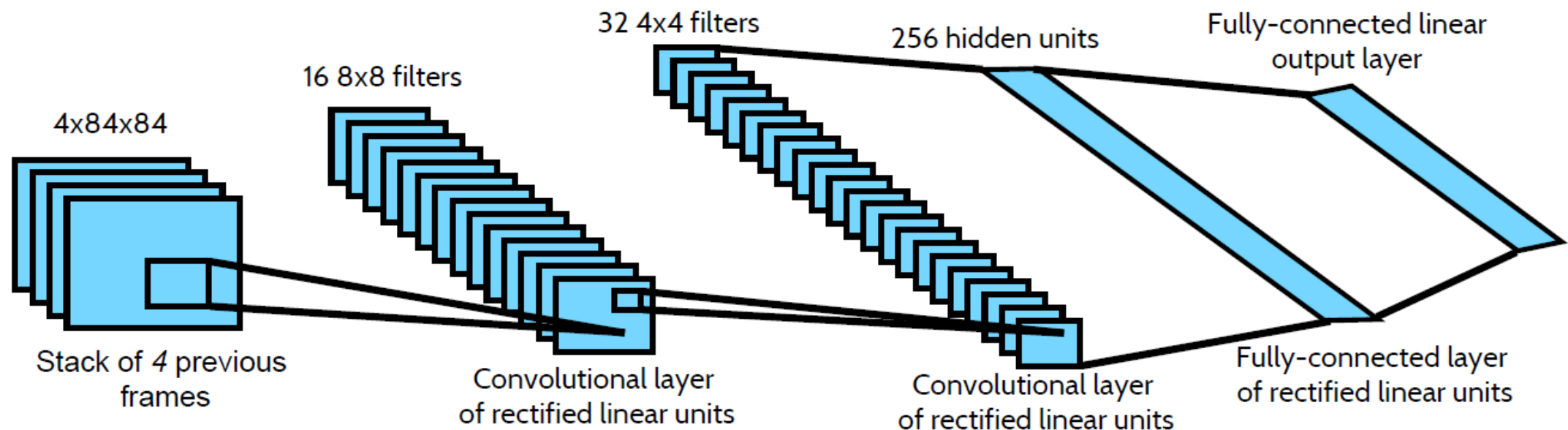
1. Achieve greater data efficiency
 - Each experience is used multiple times
2. Break correlation between samples
 - Randomly sampled experience is nonconsecutive
3. Average the behavior distribution
 - Current parameters does not affect incoming data distribution

Preprocessing

- Originally 210×160 image with 128 color palette
- Gray-scale and downsampled to 110×84
- Cropped to 84×84
 - Only to fit particular GPU implementation
- *Frame stacking*
 - Stack 4 preprocessed frames as input
 - Need multiple frames for velocity, etc.

Model Architecture: Deep Q-Networks (DQN)

- Return Q values for all 10 actions



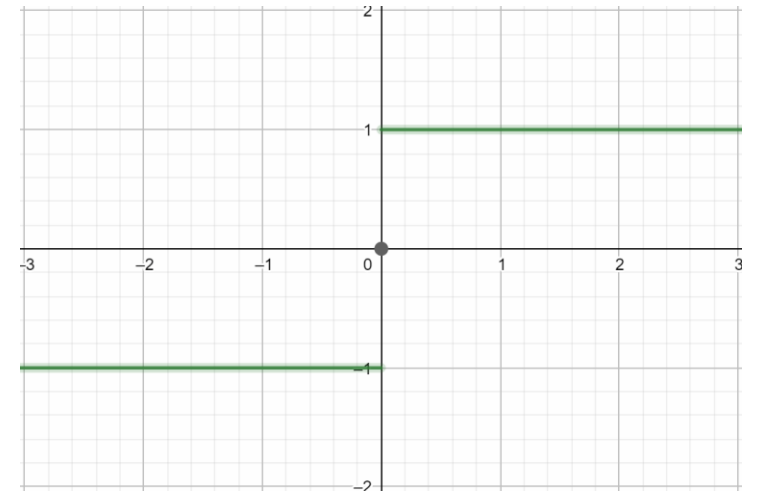
Environment: Atari 2600

- 7 games selected
- Create a **single agent** that can play as many games possible
 - No game-specific information
 - No hand-designed features
 - Same network architecture and hyperparameters



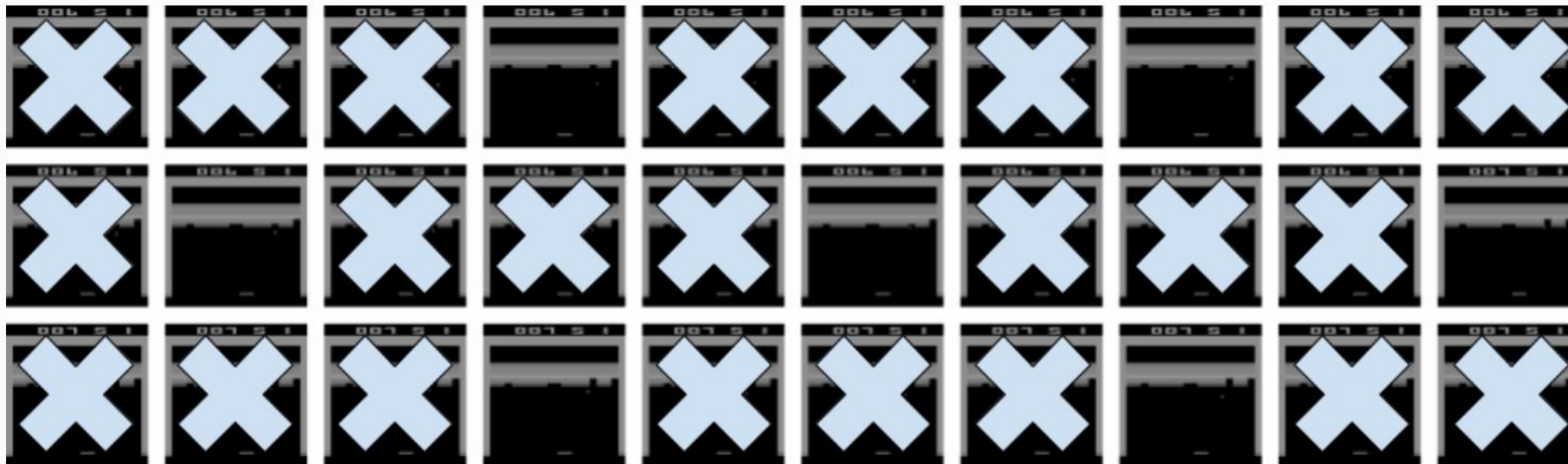
Reward Clipping

- Clip all rewards to $\{-1, 0, 1\}$
 - Fix all positive reward as $+1$
 - Fix all negative rewards as -1
 - Leave zero rewards unchanged
- Limits scale of error derivatives for different games
- Cannot differentiate between rewards of different magnitude



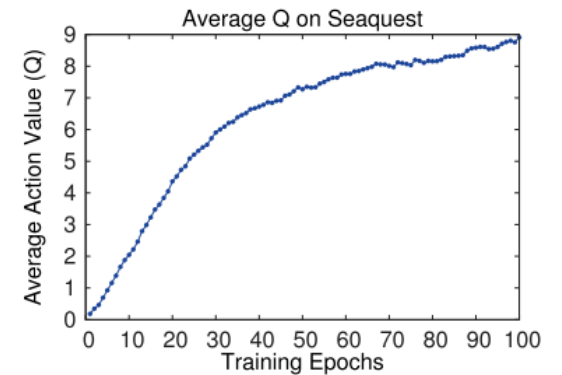
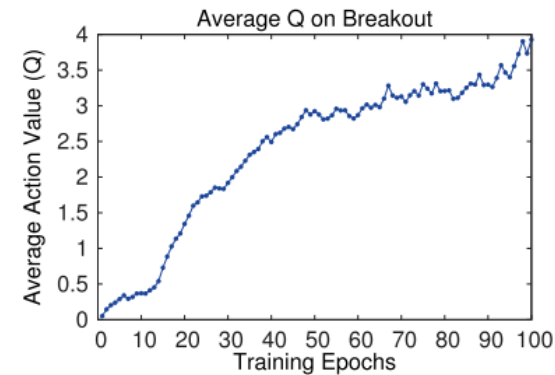
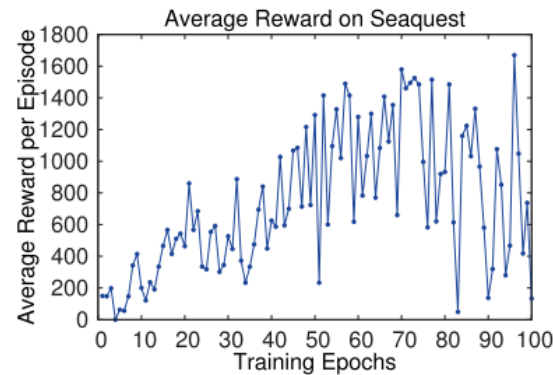
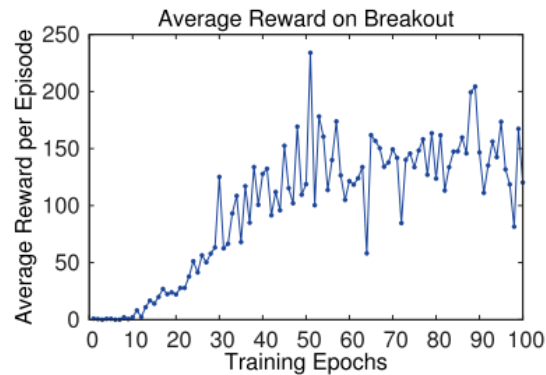
Frame Skipping

- Agent sees and selects actions on every k^{th} frame
- Action is repeated on skipped frames
- Agent can play roughly k times more games



Training and Stability

- How to evaluate agent during training?
 - Total episodic reward: very noisy
 - Policy's estimated Q function: smooth



- Did not encounter any divergence issue in practice

Result on Atari 2600

- Outperform all previous RL algorithms in 6 games
- Surpass expert human player in 3 games

	B. Rider	Breakout	Enduro	Pong	Q*bert	Seaquest	S. Invaders
Random	354	1.2	0	-20.4	157	110	179
Sarsa [3]	996	5.2	129	-19	614	665	271
Contingency [4]	1743	6	159	-17	960	723	268
DQN	4092	168	470	20	1952	1705	581
Human	7456	31	368	-3	18900	28010	3690
HNeat Best [8]	3616	52	106	19	1800	920	1720
HNeat Pixel [8]	1332	4	91	-16	1325	800	1145
DQN Best	5184	225	661	21	4500	1740	1075

Thank you!

Original Paper: <https://arxiv.org/abs/1312.5602>

Paper Recommendations:

- [\[1502\] Human-level control through Deep Reinforcement Learning](#)
- [\[1511.05952\] Prioritized Experience Replay](#)
- [\[1712.01275\] A Deeper Look at Experience Replay](#)

You can find more content in www.endtoend.ai/slides