University of Toronto
Faculty of Arts and Science
Department of Computer Science

State of the Art Control of Atari Games Using Shallow Reinforcement Learning

Yitao Liang, Marlos C. Machado,
Erik Talvitie, and Michael Bowling

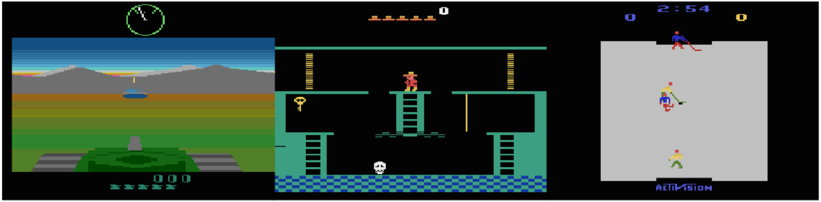
(presented by Rodrigo Toro Icarte)

December 01, 2016

Acknowledgment

Some of the slides used in this presentation are modifications of Yitao Liang's AAMAS 2016 presentation. I would like to Thank Marlos Machado and Michael Bowling for sharing Liang's slides with me.

Context



Picture was taken from Liang et al. (2016)

Figure : Atari game examples.

Context: Sarsa(λ)

The Arcade Learning Environment:
An Evaluation Platform for General Agents

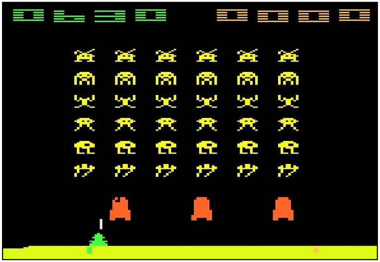
Marc G. Bellemare MG17@CS.UALBERTA.CA
University of Alberta, Edmonton, Alberta, Canada

Yavar Naddaf YAVAR@EMPIRICALRESULTS.CA
Empirical Results Inc., Vancouver,
British Columbia, Canada

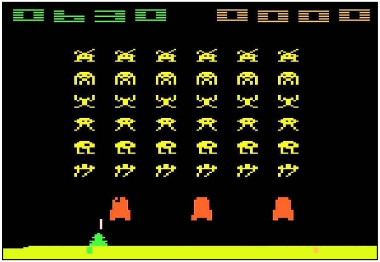
Joel Veness VENESS@CS.UALBERTA.CA
Michael Bowling BOWLING@CS.UALBERTA.CA
University of Alberta, Edmonton, Alberta, Canada

Figure : Sarsa(λ) + Linear value function approximation.

Context: Sarsa(λ)

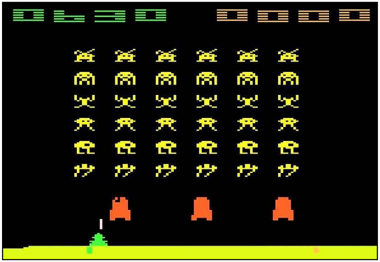


Context: Sarsa(λ)



`extract_features(I)`

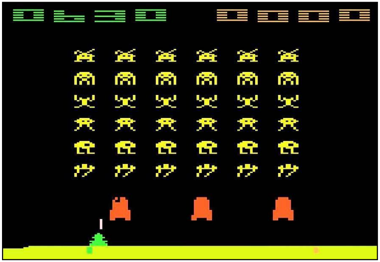
Context: Sarsa(λ)



extract_features(I)

$$\begin{bmatrix} 1.0 \\ 0.6 \\ 0.7 \\ (...) \\ 0.3 \\ 1.0 \\ 0.9 \end{bmatrix}^T$$

Context: Sarsa(λ)



extract_features(I)

$$\begin{bmatrix} 1.0 \\ 0.6 \\ 0.7 \\ (...) \\ 0.3 \\ 1.0 \\ 0.9 \end{bmatrix}^{\mathrm{T}} w \approx Q^*(s,a)$$

Context: Sarsa(λ)

A Few Useful Things to Know about Machine Learning

Pedro Domingos
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
pedrod@cs.washington.edu

Context: Sarsa(λ)

A Few Useful Things to Know about Machine Learning

Pedro Domingos
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
pedrod@cs.washington.edu

The basic argument is remarkably simple [5]. Let's say a classifier is bad if its true error rate is greater than ϵ. Then the probability that a bad classifier is consistent with n random, independent training examples is less than $(1-\epsilon)^n$. Let b be the number of bad classifiers in the learner's hypothesis space H. The probability that at least one of them is consistent is less than $b(1-\epsilon)^n$, by the union bound. Assuming the learner always returns a consistent classifier, the probability that this classifier is bad is then less than $|H|(1-\epsilon)^n$, where we have used the fact that $b \leq |H|$. So if we want this probability to be less than δ, it suffices to make $n > \ln(\delta/|H|)/\ln(1-\epsilon) \geq \frac{1}{\epsilon}\left(\ln|H| + \ln\frac{1}{\delta}\right)$.

Unfortunately, guarantees of this type have to be taken with a large grain of salt. This is because the bounds obtained in this way are usually extremely loose. The wonderful feature of the bound above is that the required number of examples only grows logarithmically with $|H|$ and $1/\delta$. Unfortunately, most interesting hypothesis spaces are *doubly* exponential in the number of features d, which still leaves us needing a number of examples exponential in d. For example, consider the space of Boolean functions of d Boolean variables. If there are e possible different examples, there are 2^e possible different functions, so since there are 2^d possible examples, the

capacity, they are quite useful; indeed, the close interplay of theory and practice is one of the main reasons machine learning has made so much progress over the years. But *caveat emptor*: learning is a complex phenomenon, and just because a learner has a theoretical justification and works in practice doesn't mean the former is the reason for the latter.

8. FEATURE ENGINEERING IS THE KEY

At the end of the day, some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used. If you have many independent features that each correlate well with the class, learning is easy. On the other hand, if the class is a very complex function of the features, you may not be able to learn it. Often, the raw data is not in a form that is amenable to learning, but you can construct features from it that are. This is typically where most of the effort in a machine learning project goes. It is often also one of the most interesting parts, where intuition, creativity and "black art" are as important as the technical stuff.

First-timers are often surprised by how little time in a machine learning project is spent actually doing machine learning. But it makes sense if you consider how time-consuming

Context: Sarsa(λ)

	Basic	BASS	DISCO	LSH	RAM
Times Best	6	17	1	8	8

Table : Results Sarsa(λ)

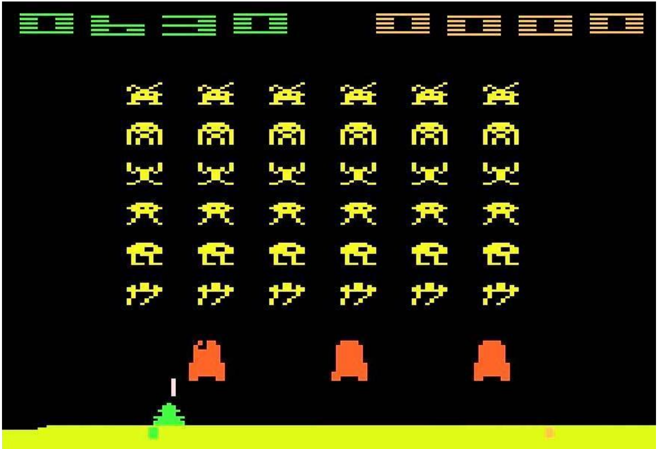
Context: Sarsa(λ)

	Basic	BASS	DISCO	LSH	RAM
Times Best	6	17	1	8	8

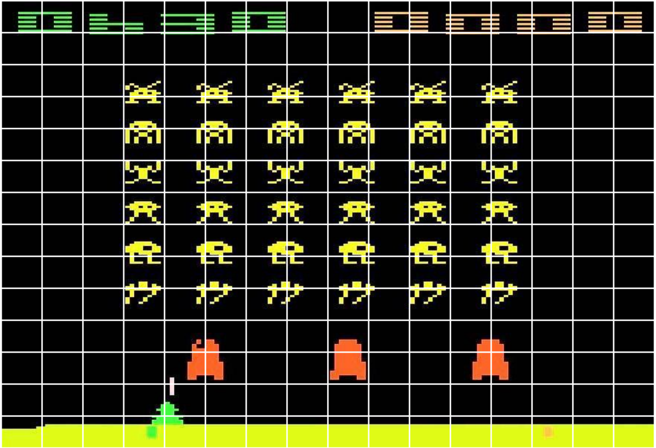
Table : Results Sarsa(λ)

BASS: Basic Features + Pairwise combinations of them.

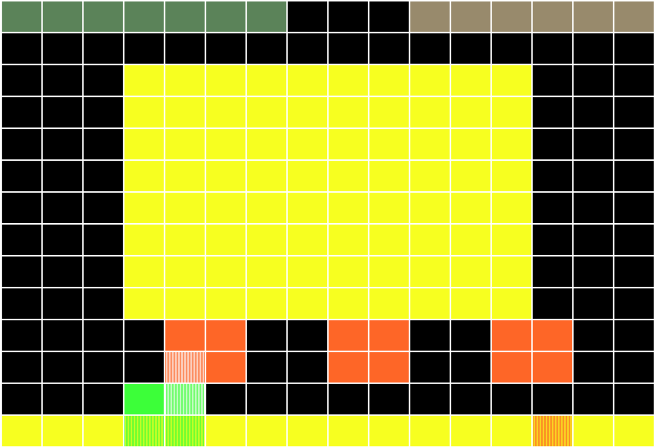
Context: Basic Features



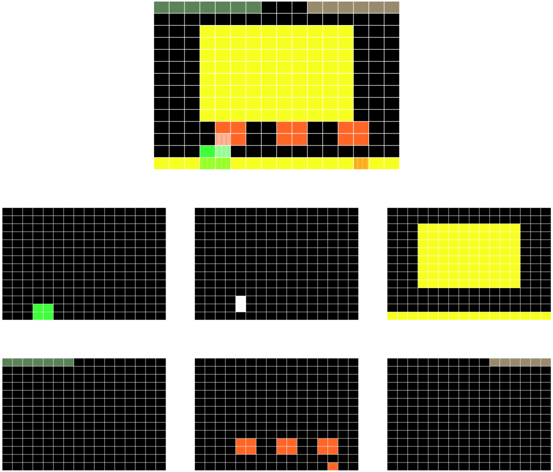
Context: Basic Features



Context: Basic Features



Context: Basic Features



Context: Pairwise Combinations

Basic Features

$\phi_b(c, r, k) = 1$ iff color k is present within tile (c, r).

Context: Pairwise Combinations

Basic Features

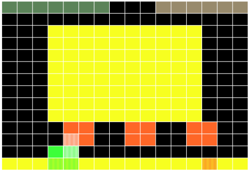
$\phi_b(c, r, k) = 1$ iff color k is present within tile (c, r).

Pairwise combinations

$\phi_p(c_1, r_1, k_1, c_2, r_2, k_2) = 1$ iff $\phi_b(c_1, r_1, k_1) = \phi_b(c_2, r_2, k_2) = 1$

Context: Pairwise Combinations

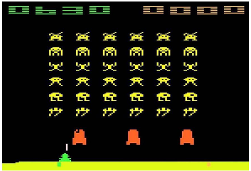
Context: Pairwise Combinations

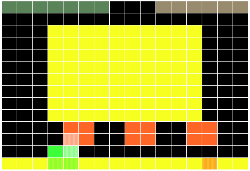


Basic features:

- $(\phi_b(5, 12, \mathrm{W}) = 1$ and $\phi_b(4, 10, \mathrm{Y}) = 1) \rightarrow$ Reward!

Context: Pairwise Combinations





Basic features:

- $(\phi_b(5, 12, \mathrm{W}) = 1$ and $\phi_b(4, 10, \mathrm{Y}) = 1) \rightarrow$ Reward!

BASS:

- $\phi_p(5, 12, \mathrm{W}, 4, 10, \mathrm{Y}) = 1 \rightarrow$ Reward!

Context: DQN

Playing Atari with Deep Reinforcement Learning

Volodymyr Mnih Koray Kavukcuoglu David Silver Alex Graves Ioannis Antonoglou

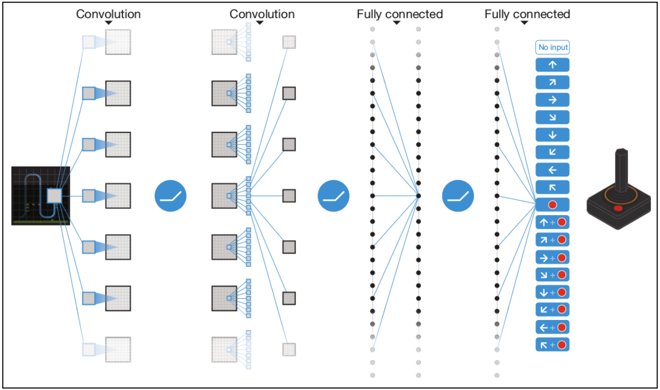
Daan Wierstra Martin Riedmiller

DeepMind Technologies

{vlad,koray,david,alex.graves,ioannis,daan,martin.riedmiller} @ deepmind.com

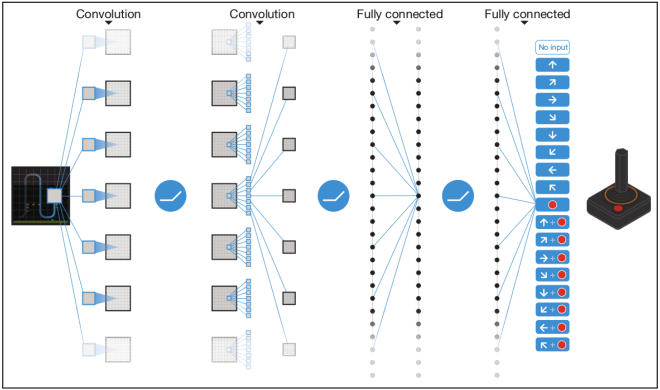
Figure : Deep Q-Learning.

Context: DQN



Picture was taken from Mnih et al. (2015)

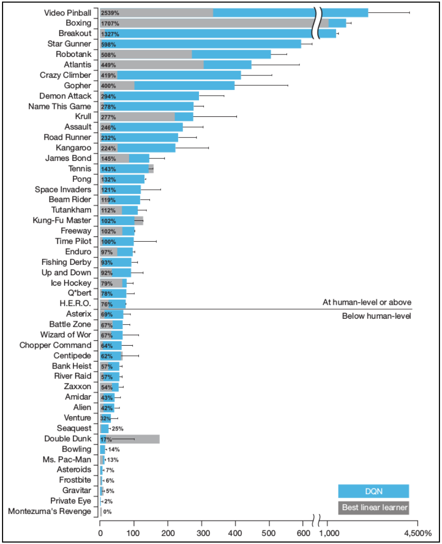
Context: DQN



Picture was taken from Mnih et al. (2015)

Where is the feature vector?

Context: DQN



Picture was taken from Mnih et al. (2015)

Motivation

State of the Art Control of Atari Games Using Shallow Reinforcement Learning

Yitao Liang[†], Marlos C. Machado[‡], Erik Talvitie[†], and Michael Bowling[‡]

[†]Franklin & Marshall College
Lancaster, PA, USA
{yliang, erik.talvitie}@fandm.edu

[‡]University of Alberta
Edmonton, AB, Canada
{machado, mbowling}@ualberta.ca

Motivation

State of the Art Control of Atari Games Using Shallow Reinforcement Learning

Yitao Liang[†], Marlos C. Machado[‡], Erik Talvitie[†], and Michael Bowling[‡]
[†]Franklin & Marshall College
Lancaster, PA, USA
{yliang, erik.talvitie}@fandm.edu
[‡]University of Alberta
Edmonton, AB, Canada
{machado, mbowling}@ualberta.ca

DQN's comparison with Sarsa(λ) was unfair.

Motivation

State of the Art Control of Atari Games Using Shallow Reinforcement Learning

Yitao Liang[†], Marlos C. Machado[‡], Erik Talvitie[†], and Michael Bowling[‡]
[†]Franklin & Marshall College
Lancaster, PA, USA
{yliang, erik.talvitie}@fandm.edu
[‡]University of Alberta
Edmonton, AB, Canada
{machado, mbowling}@ualberta.ca

DQN's comparison with Sarsa(λ) was unfair.

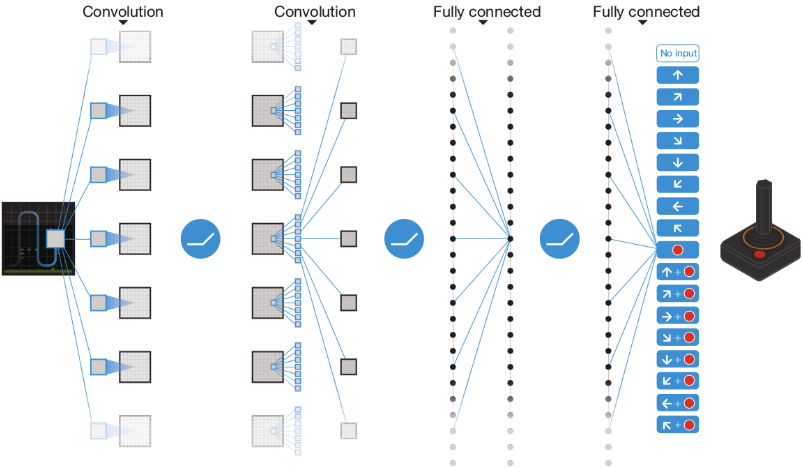
- Sarsa(λ) was trained with far less training data.
- DQN uses 4 frames as input.
- DQN has representational biases that Sarsa(λ) doesn't.

Motivation

Methodology:

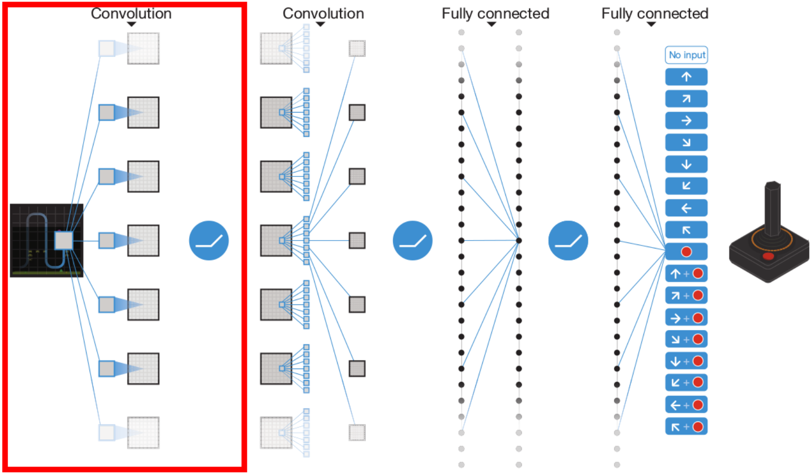
- Identify representational biases in DQN.
- Incorporate identified biases into feature vector.
- Evaluate Sarsa(λ) using the new feature vector.
- Repeat.

Basic Features



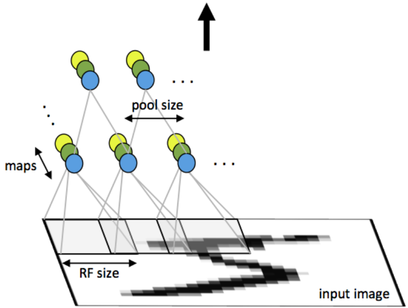
Picture was taken from Mnih et al. (2015)

Basic Features



Picture was taken and modified from Mnih et al. (2015)

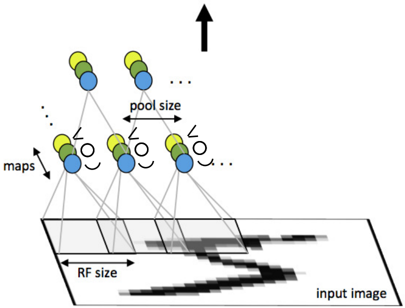
Basic Features



Picture was retrieved and modified from

http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

Basic Features



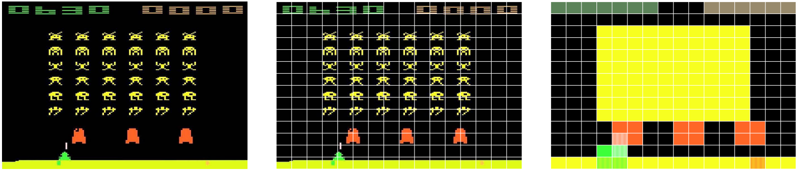
Picture was retrieved and modified from

http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

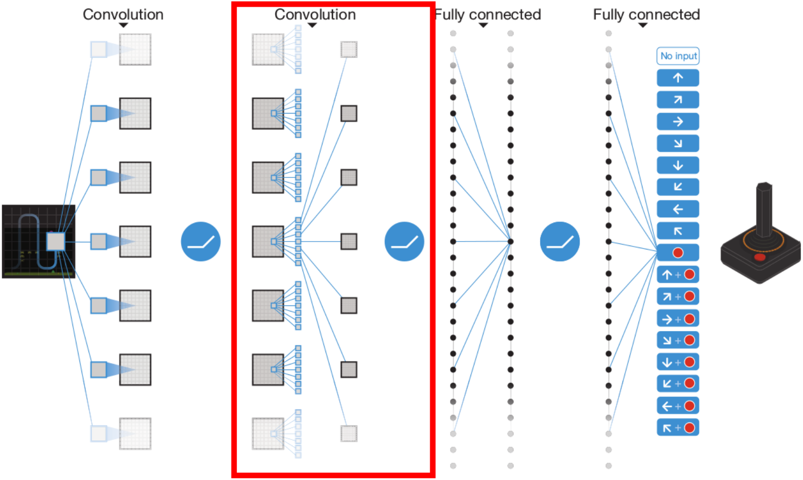
Basic Features

Basic Features

$\phi_b(c, r, k) = 1$ iff color k is present within tile (c, r).

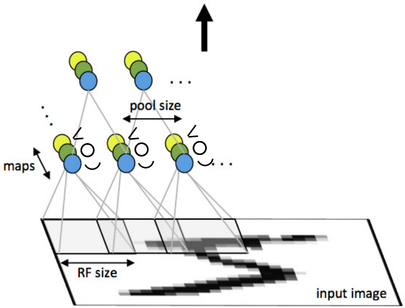


Spatial Invariance



Picture was taken and modified from Mnih et al. (2015)

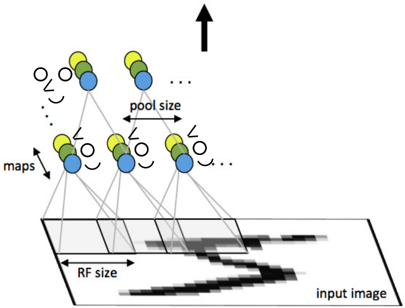
Spatial Invariance



Picture was retrieved and modified from

http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

Spatial Invariance



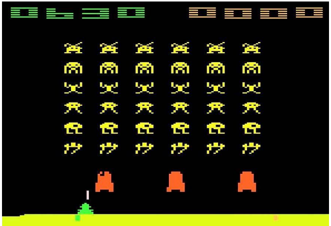
Picture was retrieved and modified from

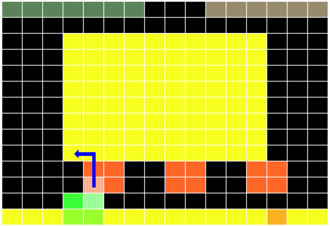
http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

Spatial Invariance

Pairwise combinations

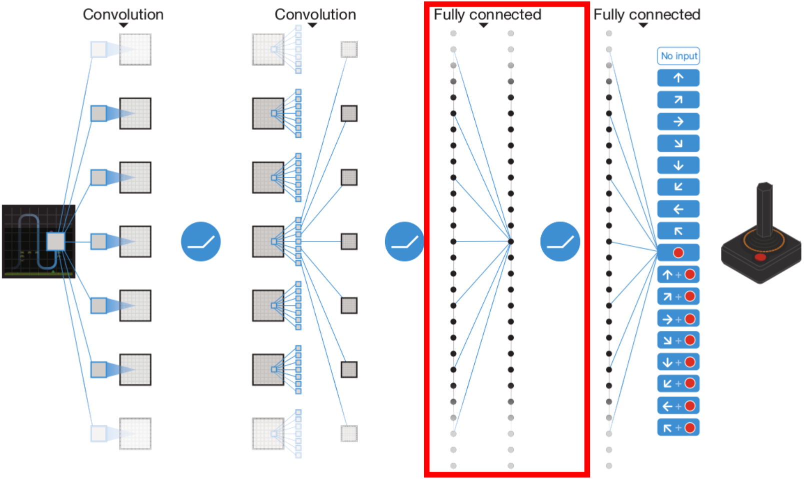
$\phi_p(c_1, r_1, k_1, c_2, r_2, k_2) = 1$ iff $\phi_b(c_1, r_1, k_1) = \phi_b(c_2, r_2, k_2) = 1$





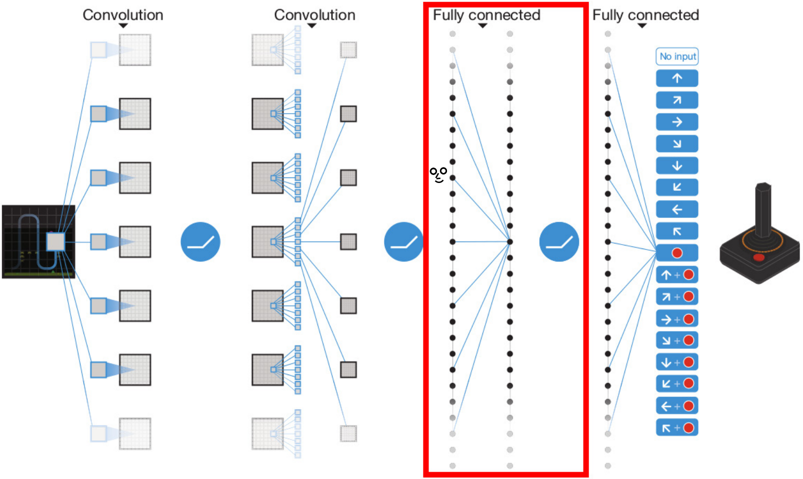
$$\phi_p(5, 12, \mathrm{W}, 4, 10, \mathrm{Y}) = 1$$

Spatial Invariance



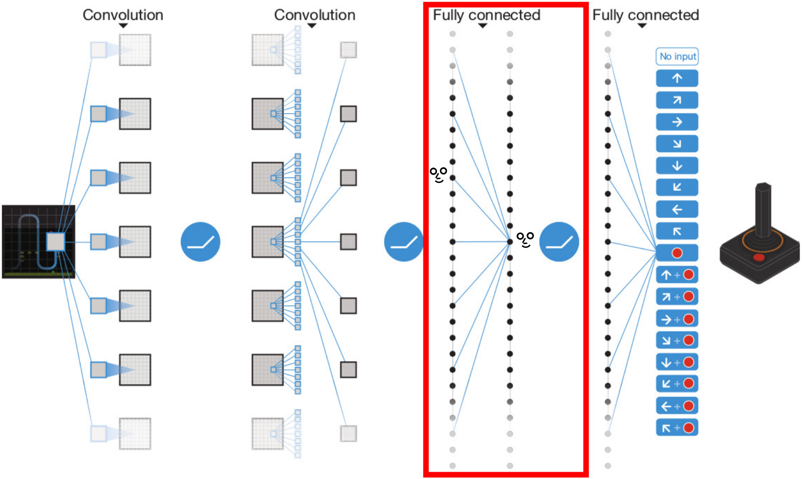
Picture was taken and modified from Mnih et al. (2015)

Spatial Invariance



Picture was taken and modified from Mnih et al. (2015)

Spatial Invariance

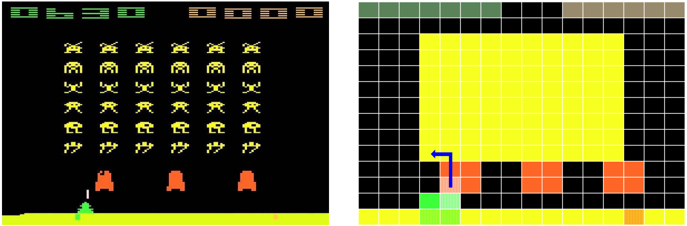


Picture was taken and modified from Mnih et al. (2015)

Spatial Invariance

Pairwise combinations

$\phi_p(c_1, r_1, k_1, c_2, r_2, k_2) = 1$ iff $\phi_b(c_1, r_1, k_1) = \phi_b(c_2, r_2, k_2) = 1$



$$\phi_p(5, 12, \mathrm{W}, 4, 10, \mathrm{Y}) = 1$$

Spatial Invariance

Idea: Use relative positions instead of absolute positions[1].

[1] $\approx$ Take the max of BASS features over absolute position.

Spatial Invariance

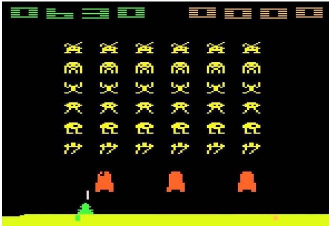
Idea: Use relative positions instead of absolute positions[1].

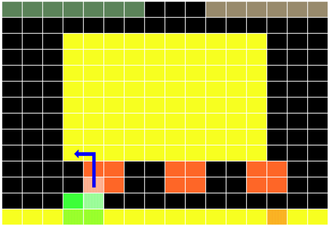
B-PROS (Basic Pairwise Relative Offsets in Space)

- $\phi_b(c, r, k)$.
- $\phi_s(k_1, k_2, i, j) = 1$ iff exists c and r such that $\phi_b(c, r, k_1) = 1$ and $\phi_b(c + i, r + j, k_2) = 1$.

[1] $\approx$ Take the max of BASS features over absolute position.

Spatial Invariance

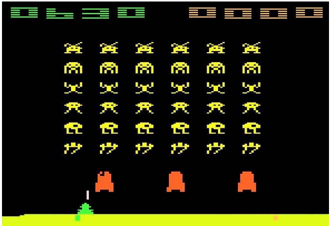


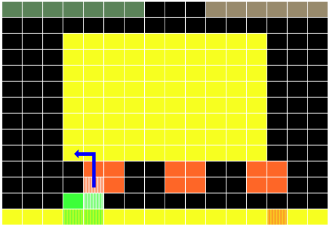


BASS:

- $\phi_p(5, 12, W, 4, 10, Y) = 1 \rightarrow$ Reward!

Spatial Invariance





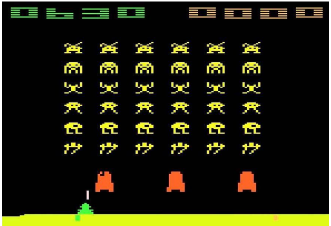
BASS:

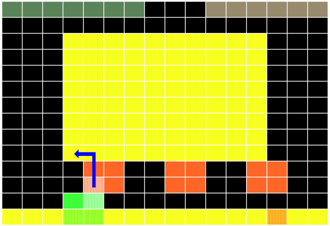
- $\phi_p(5, 12, W, 4, 10, Y) = 1 \rightarrow$ Reward!

B-PROS:

- $\phi_s(-2, -1, W, Y) = 1 \rightarrow$ Reward!

Spatial Invariance





BASS:

- $\phi_p(5, 12, W, 4, 10, Y) = 1 \rightarrow$ Reward!

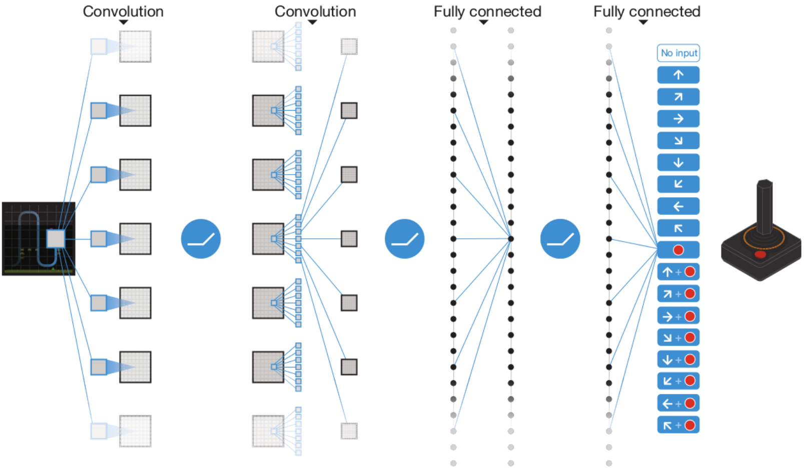
B-PROS:

- $\phi_s(-2, -1, W, Y) = 1 \rightarrow$ Reward!

Results:

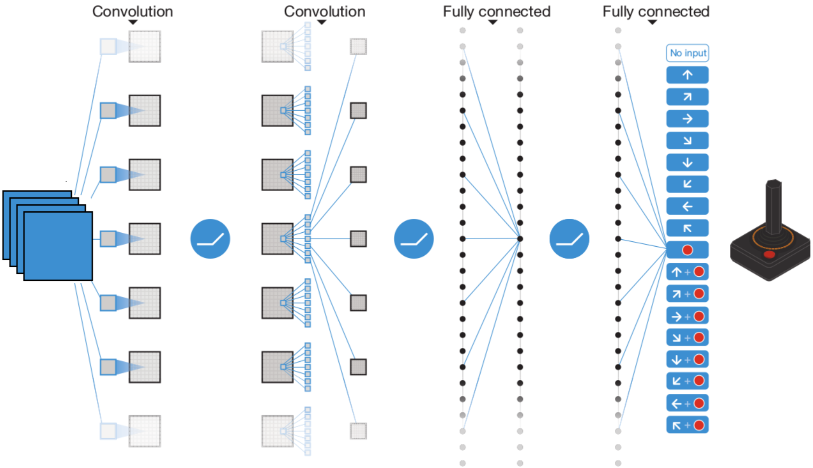
- B-PROS 41 vs 12 Basic, BASS, DISCO, LSH.

Non-Markovian Features



Picture was taken from Mnih et al. (2015)

Non-Markovian Features



Picture was taken and modified from Mnih et al. (2015)

Non-Markovian Features

Idea: Compare basic features between the current screen and the screen 5 frames in the past.

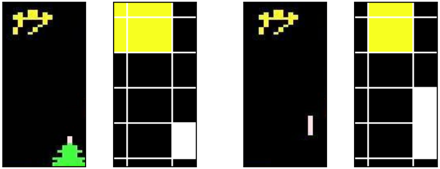
Non-Markovian Features

Idea: Compare basic features between the current screen and the screen 5 frames in the past.

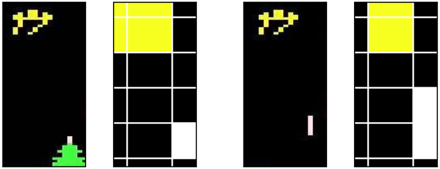
B-PROST (... Time)

- $\phi_b(c, r, k)$.
- $\phi_s(k_1, k_2, i, j)$.
- $\phi_t(k_1, k_2, i, j) = 1$ iff exists c and r such that $\phi_b^{t_c - 5}(c, r, k_1) = 1$ and $\phi_b^{t_c}(c + i, r + j, k_2) = 1$.

Non-Markovian Features



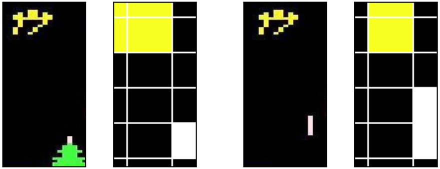
Non-Markovian Features



B-PROS:

- $\phi_s(-2, -1, W, Y) = 1 \rightarrow$ Reward... I guess...

Non-Markovian Features



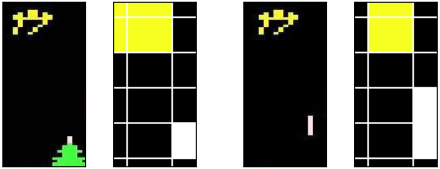
B-PROS:

- $\phi_s(-2, -1, W, Y) = 1 \rightarrow$ Reward... I guess...

B-PROST:

- $(\phi_s(-2, -1, W, Y) = 1$ and $\phi_t(2, 2, Y, W) = 1) \rightarrow$ Reward!

Non-Markovian Features



B-PROS:

- $\phi_s(-2,-1,W,Y)=1 \rightarrow$ Reward... I guess...

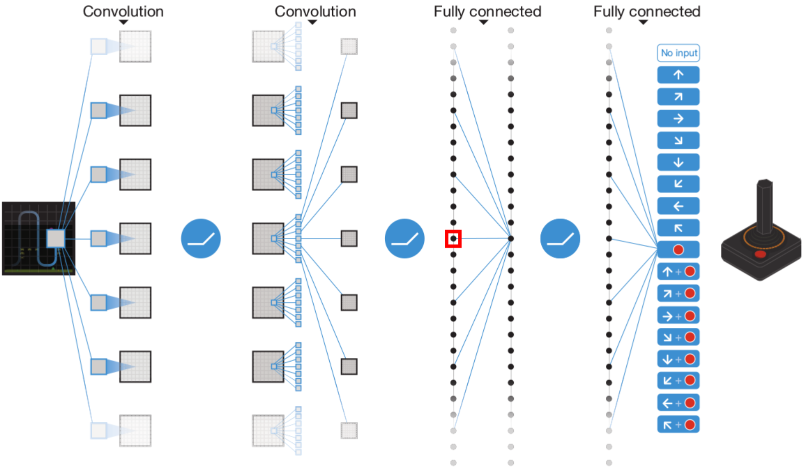
B-PROST:

- $(\phi_s(-2,-1,W,Y)=1$ and $\phi_t(2,2,Y,W)=1) \rightarrow$ Reward!

Results:

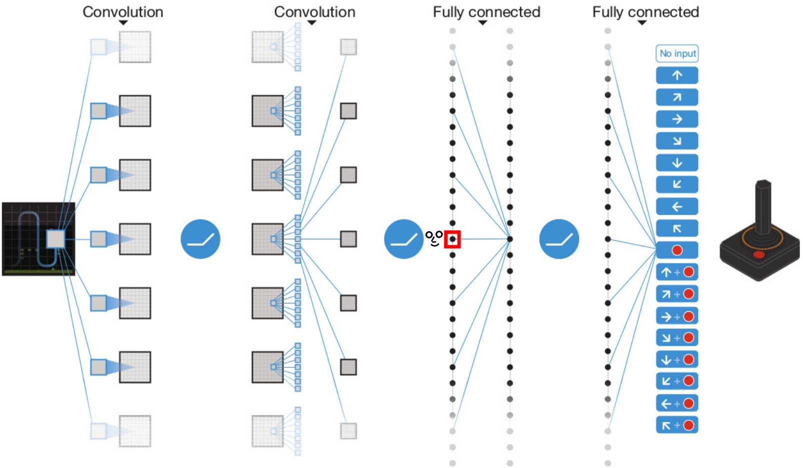
- B-PROST 40 vs 9 B-PROS.

Object Detection



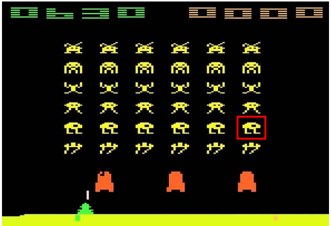
Picture was taken and modified from Mnih et al. (2015)

Object Detection

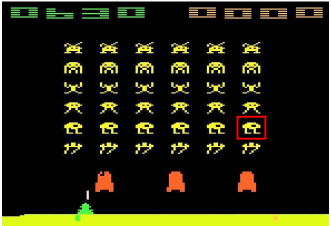


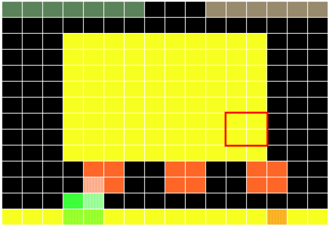
Picture was taken and modified from Mnih et al. (2015)

Object Detection



Object Detection





Object Detection

Idea: Approximate object detection by grouping contiguous pixels of the same color (*blobs*).

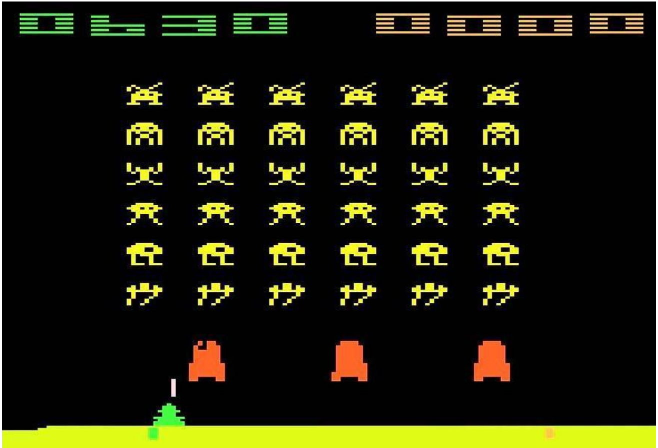
Object Detection

Idea: Approximate object detection by grouping contiguous pixels of the same color (*blobs*).

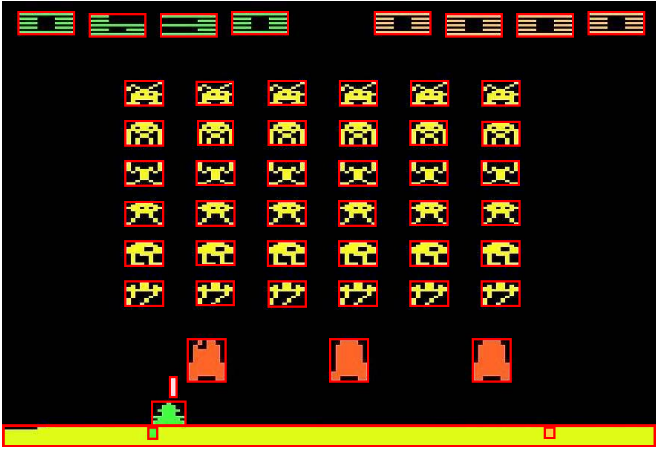
Blob-PROST

- Compute blobs.
- Define $\phi_b(c, r, k)$ over blobs.
- $\phi_s(k_1, k_2, i, j)$.
- $\phi_t(k_1, k_2, i, j)$.

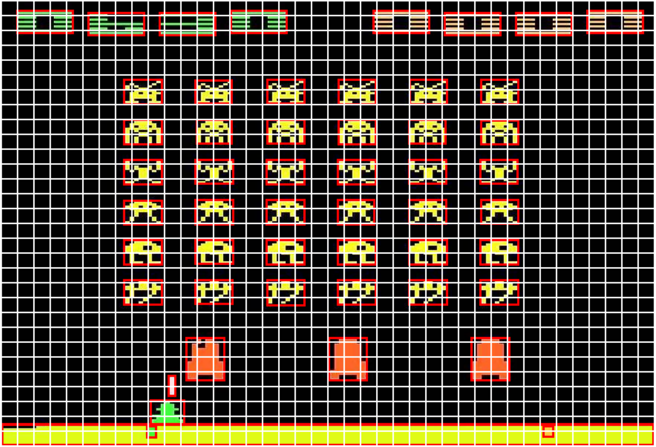
Object Detection



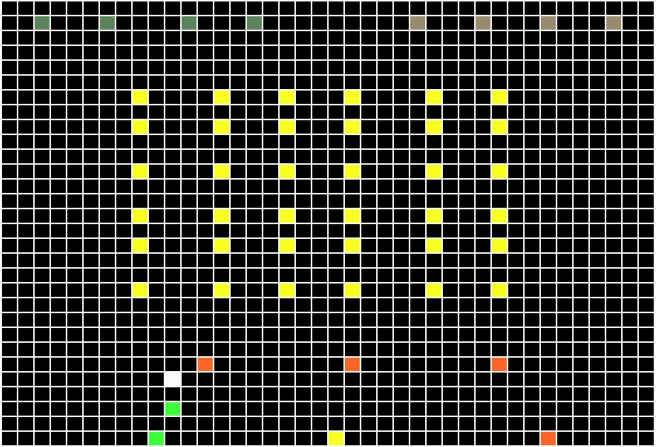
Object Detection



Object Detection



Object Detection



Object Detection

Results:

- Blob-PROST 29 vs 20 B-PROS and B-PROST.

Comparison with DQN (methodology)

Comparing Blob-PROST and DQN:

- Train for $200,000,000$ frames.
- Run 24 independent trials.
- Evaluate using 499 episodes (at the end of training).
- Start with a random number of *no-op* actions.
- Use the minimal action set.

Comparison with DQN (computational cost)

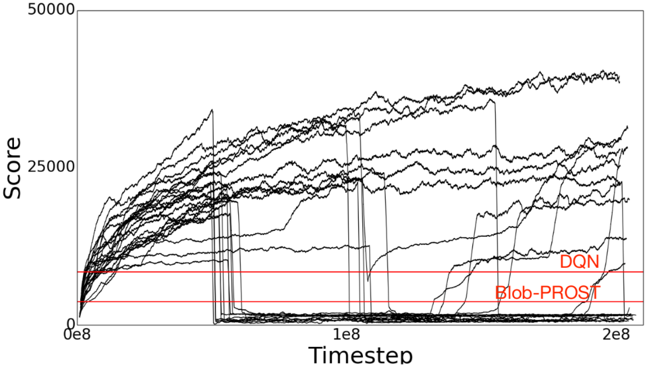
	Blob-PROST	DQN
Memory	50MB-3.7, 9GB (in most game, 1GB)	9.8GB
Running speed	56-300 decisions/second (in most games, 150)	5 (83 when using GPU)

Comparison with DQN (performance)

They report results over 24 trials. DQN only reports 1 trial:

- Blob-PROST 20 vs 29 DQN (average)
- Blob-PROST 21 vs 28 DQN (median)
- Blob-PROST 32 vs 17 DQN (best trial)

Comparison with DQN (performance)



Picture was retrieved from Liang et al. (2016)

Conclusions

- Blob-PROST is a strong, light-weight, alternative to DQN in ALE.

Conclusions

- Blob-PROST is a strong, light-weight, alternative to DQN in ALE.
- Blob-PROST is better than DQN when:
  - It is fairly easy to die (e.g. `Montezuma's Revenge`).
  - Reward is sparse (e.g. `Tennis`).

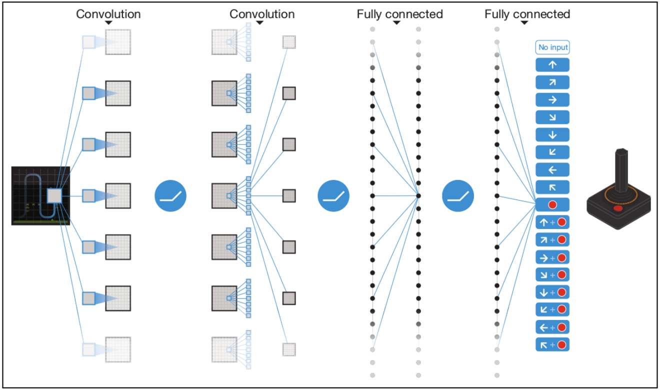
Conclusions

- Blob-PROST is a strong, light-weight, alternative to DQN in ALE.
- Blob-PROST is better than DQN when:
  - It is fairly easy to die (e.g. `Montezuma's Revenge`).
  - Reward is sparse (e.g. `Tennis`).
- DQN is better than Blob-PROST when:
  - Object velocities are important (e.g. shooting games).
  - Holistic information is important (e.g. `Breakout`, `Space Invaders`).

Conclusions

"*We saw progressive and dramatic improvements by respectively incorporating relative distances between objects, non-Markov features, and more sophisticated object detection. This illuminates some important representational issues that likely underly DQN's success. It also suggests that the general properties of the representations learned by DQN may be more important to its success in ALE than the specific features it learns in each game.*"

Conclusions



Picture was taken from Mnih et al. (2015)

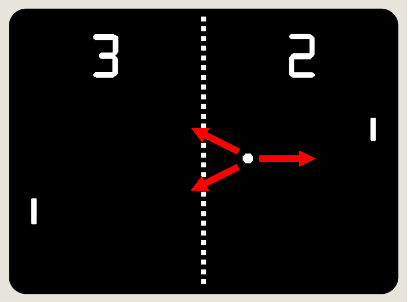
Questions?

Questions?

Q1) Why is a single Atari screenshot inadequate, and how does comparing between screenshots that are 5 frames apart help? Explain these using Pong as an example.

Questions?

Q1) Why is a single Atari screenshot inadequate, and how does comparing between screenshots that are 5 frames apart help? Explain these using Pong as an example.

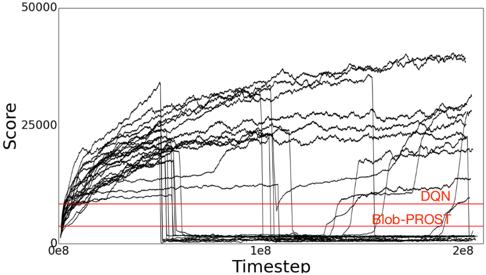


Questions?

Q2) One of the issues identified in the DQN experimentation is that it uses the best performing weights found at different points during the training phase. Why is this problematic?

Questions?

Q2) One of the issues identified in the DQN experimentation is that it uses the best performing weights found at different points during the training phase. Why is this problematic?



Picture was taken from Liang et al. (2016)

References I

Liang, Y., Machado, M. C., Talvitie, E., & Bowling, M. (2016). State of the art control of atari games using shallow reinforcement learning. In *Proceedings of the 2016 international conference on autonomous agents & multiagent systems* (pp. 485–493).

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., … others (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.