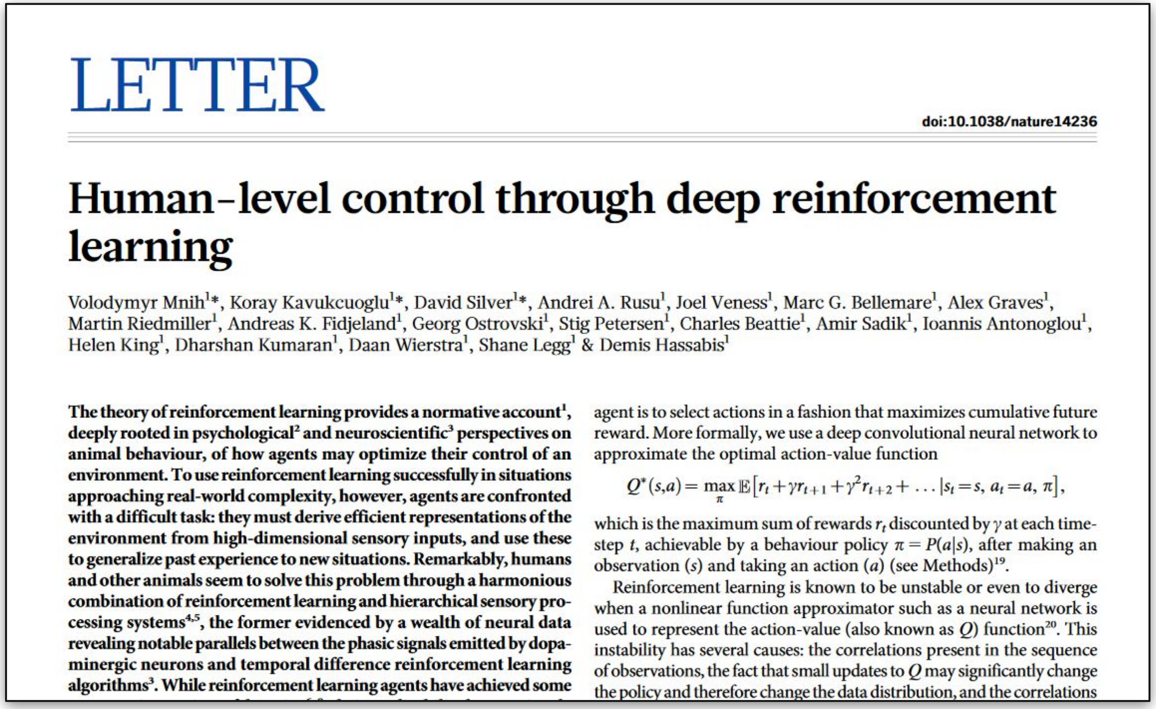
SEM3
Sample-Efficient, Multi-Modal Model

Noah Rousell, Adithya Sriram, Julian Beaudry, & Ryan Eng

Deep Q Networks (DQN)



LETTER

doi:10.1038/nature14236

Human-level control through deep reinforcement learning

Volodymyr Mnih[1]*, Koray Kavukcuoglu[1]*, David Silver[1]*, Andrei A. Rusu[1], Joel Veness[1], Marc G. Bellemare[1], Alex Graves[1], Martin Riedmiller[1], Andreas K. Fidjeland[1], Georg Ostrovski[1], Stig Petersen[1], Charles Beattie[1], Amir Sadik[1], Ioannis Antonoglou[1], Helen King[1], Dharshan Kumaran[1], Daan Wierstra[1], Shane Legg[1] & Demis Hassabis[1]

The theory of reinforcement learning provides a normative account[1], deeply rooted in psychological[2] and neuroscientific[3] perspectives on animal behaviour, of how agents may optimize their control of an environment. To use reinforcement learning successfully in situations approaching real-world complexity, however, agents are confronted with a difficult task: they must derive efficient representations of the environment from high-dimensional sensory inputs, and use these to generalize past experience to new situations. Remarkably, humans and other animals seem to solve this problem through a harmonious combination of reinforcement learning and hierarchical sensory processing systems[4,5], the former evidenced by a wealth of neural data revealing notable parallels between the phasic signals emitted by dopaminergic neurons and temporal difference reinforcement learning algorithms[3]. While reinforcement learning agents have achieved some

agent is to select actions in a fashion that maximizes cumulative future reward. More formally, we use a deep convolutional neural network to approximate the optimal action-value function

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

which is the maximum sum of rewards r_t discounted by γ at each time-step t, achievable by a behaviour policy $\pi = P(a|s)$, after making an observation (s) and taking an action (a) (see Methods)[19].

Reinforcement learning is known to be unstable or even to diverge when a nonlinear function approximator such as a neural network is used to represent the action-value (also known as Q) function[20]. This instability has several causes: the correlations present in the sequence of observations, the fact that small updates to Q may significantly change the policy and therefore change the data distribution, and the correlations

- Comparable/Superhuman on the majority of Atari 2600 games
- But:
  - Requires **38 days** of data
  - Single modality

Mnih et al. 2015

Deep Q-Learning (DQN)

- Train a neural network to estimate the Q value of each action
- Store experience (state, action, reward) in replay buffer and sample in mini-batches when performing gradient descent

$$\mathcal{L}_\theta^{DQN} = \left(Q_\theta(s_t, a_t) - (r_t + \gamma \max_a Q_\xi(s_{t+1}, a))\right)^2$$

- Where Q_ϵ is a version of the network with “stale” parameters
    - This stabilizes training

Mnih et al. 2015

DQN Improvements

N-step learning

Idea: Look further ahead at ground-truth rewards to improve training

$$\mathcal{L}_\theta^{DQN} = \left(Q_\theta(s_t, a_t) - (r_t + \gamma \max_a Q_\xi(s_{t+1}, a))\right)^2$$

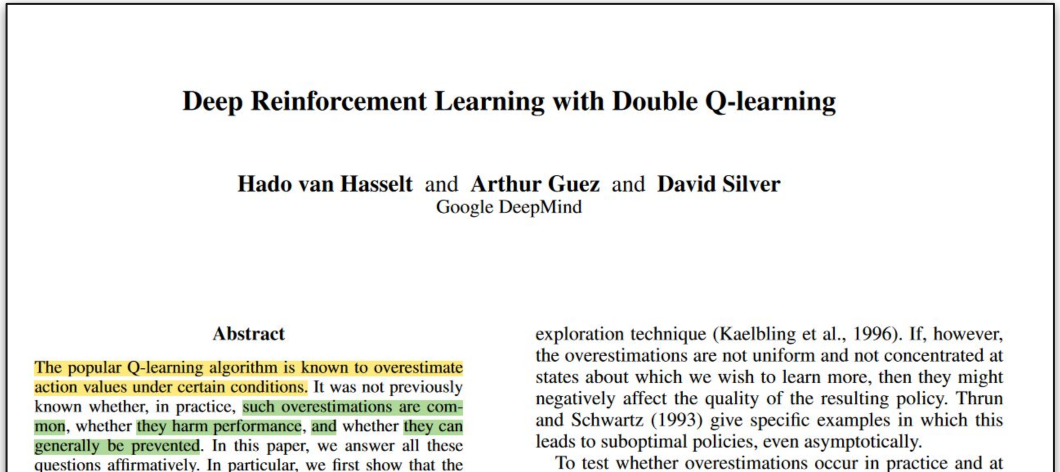
$$R_n = r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \ldots + \gamma^{n-1} \cdot r_{t+n-1}$$

Double DQN

[Hasselt et al. 2015](#)

Idea: reduce over-estimation of Q-values by using the online network for action selection of the Q-learning target

$$Y_t^{\text{DoubleQ}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \operatorname{argmax}_a Q(S_{t+1}, a; \boldsymbol{\theta}_t); \boldsymbol{\theta}'_t)$$



Deep Reinforcement Learning with Double Q-learning

Hado van Hasselt and **Arthur Guez** and **David Silver**
Google DeepMind

Abstract

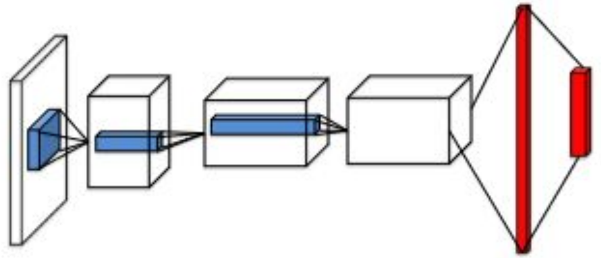
The popular Q-learning algorithm is known to overestimate action values under certain conditions. It was not previously known whether, in practice, such overestimations are common, whether they harm performance, and whether they can generally be prevented. In this paper, we answer all these questions affirmatively. In particular, we first show that the

exploration technique (Kaelbling et al., 1996). If, however, the overestimations are not uniform and not concentrated at states about which we wish to learn more, then they might negatively affect the quality of the resulting policy. Thrun and Schwartz (1993) give specific examples in which this leads to suboptimal policies, even asymptotically.
To test whether overestimations occur in practice and at

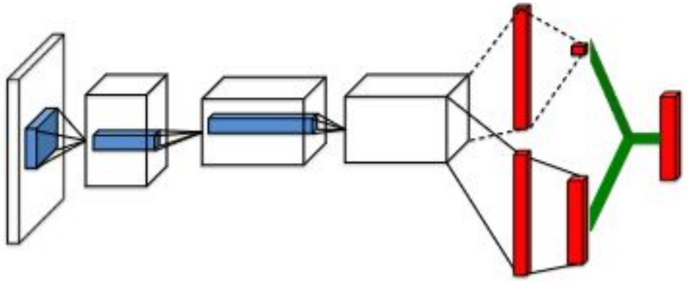
Wang et al. 2016

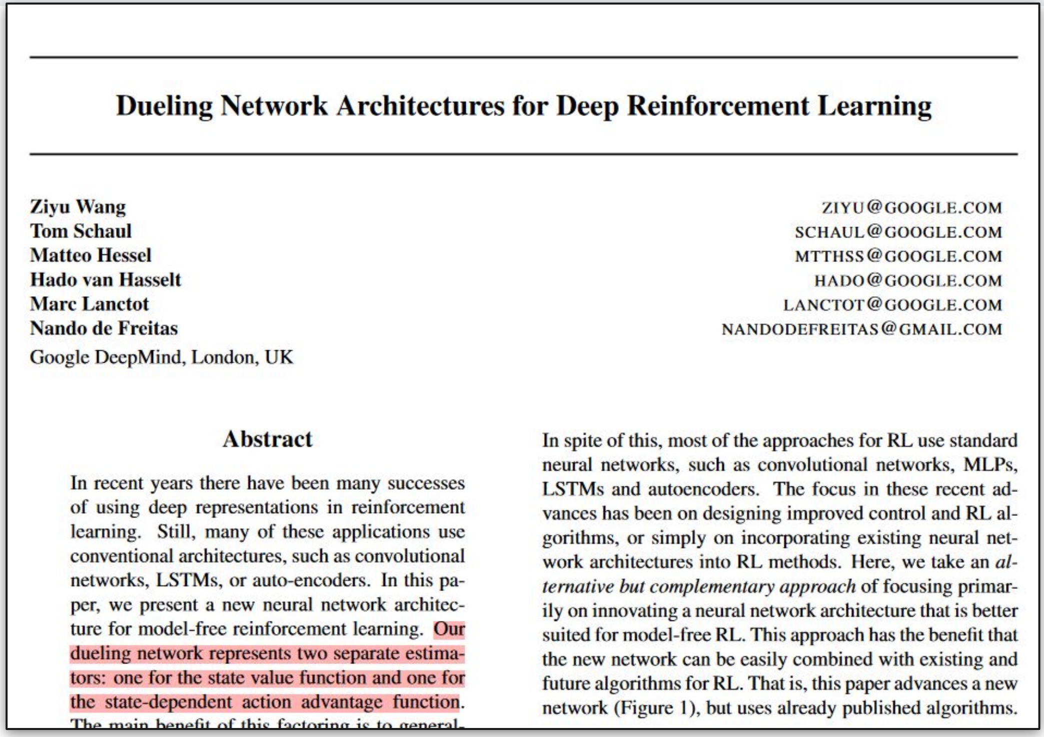
Dueling DQN

Regular Q-network



Dueling Q-network





Dueling Network Architectures for Deep Reinforcement Learning

Ziyu Wang ZIYU@GOOGLE.COM
Tom Schaul SCHAUL@GOOGLE.COM
Matteo Hessel MTTHSS@GOOGLE.COM
Hado van Hasselt HADO@GOOGLE.COM
Marc Lanctot LANCTOT@GOOGLE.COM
Nando de Freitas NANDODEFREITAS@GMAIL.COM
Google DeepMind, London, UK

Abstract

In recent years there have been many successes of using deep representations in reinforcement learning. Still, many of these applications use conventional architectures, such as convolutional networks, LSTMs, or auto-encoders. In this paper, we present a new neural network architecture for model-free reinforcement learning. Our dueling network represents two separate estimators: one for the state value function and one for the state-dependent action advantage function. The main benefit of this factoring is to general-

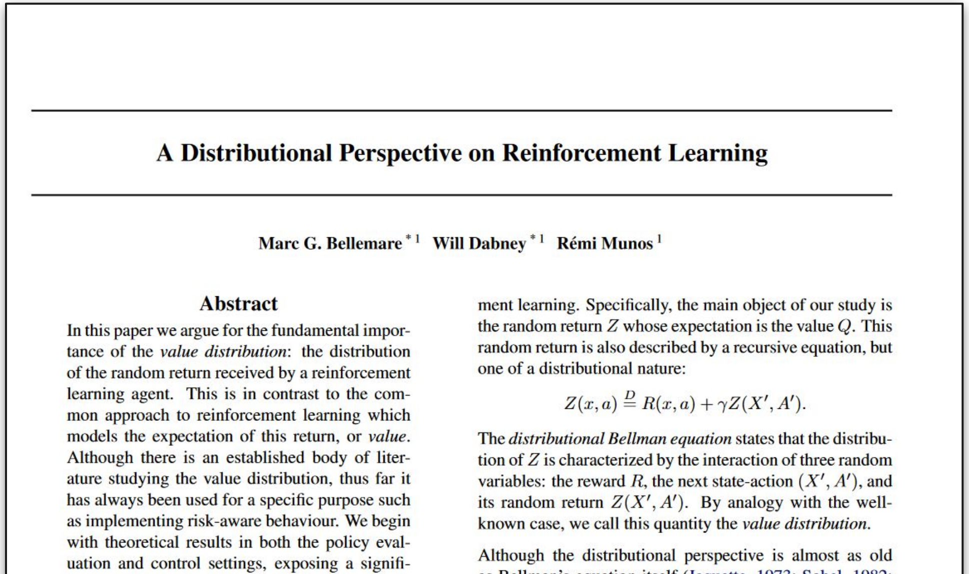
In spite of this, most of the approaches for RL use standard neural networks, such as convolutional networks, MLPs, LSTMs and autoencoders. The focus in these recent advances has been on designing improved control and RL algorithms, or simply on incorporating existing neural network architectures into RL methods. Here, we take an *alternative but complementary approach* of focusing primarily on innovating a neural network architecture that is better suited for model-free RL. This approach has the benefit that the new network can be easily combined with existing and future algorithms for RL. That is, this paper advances a new network (Figure 1), but uses already published algorithms.

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left(A(s,a;\theta,\alpha) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a';\theta,\alpha)\right)$$

Distributional DQN

Bellemare et al. 2017

Idea: Approximate the distribution of each Q value instead of the expected value



A Distributional Perspective on Reinforcement Learning

Marc G. Bellemare [*1] Will Dabney [*1] Rémi Munos [1]

Abstract

In this paper we argue for the fundamental importance of the *value distribution*: the distribution of the random return received by a reinforcement learning agent. This is in contrast to the common approach to reinforcement learning which models the expectation of this return, or *value*. Although there is an established body of literature studying the value distribution, thus far it has always been used for a specific purpose such as implementing risk-aware behaviour. We begin with theoretical results in both the policy evaluation and control settings, exposing a signifi-

ment learning. Specifically, the main object of our study is the random return Z whose expectation is the value Q. This random return is also described by a recursive equation, but one of a distributional nature:

$$Z(x,a) \overset{D}{=} R(x,a) + \gamma Z(X', A').$$

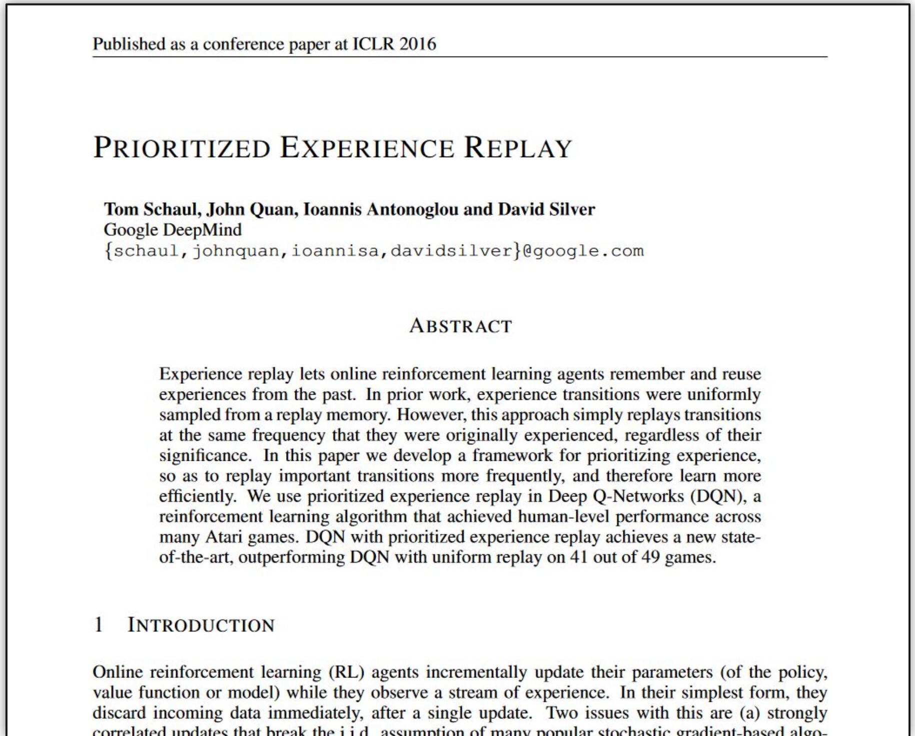
The *distributional Bellman equation* states that the distribution of Z is characterized by the interaction of three random variables: the reward R, the next state-action (X', A'), and its random return $Z(X', A')$. By analogy with the well-known case, we call this quantity the *value distribution*.

Although the distributional perspective is almost as old

Prioritized Experience

Idea: Sample experiences from replay buffer in proportion to their corresponding RL loss

Schaul et al. 2016


Published as a conference paper at ICLR 2016

PRIORITIZED EXPERIENCE REPLAY

Tom Schaul, John Quan, Ioannis Antonoglou and David Silver
Google DeepMind
{schaul,johnquan,ioannisa,davidsilver}@google.com

ABSTRACT

Experience replay lets online reinforcement learning agents remember and reuse experiences from the past. In prior work, experience transitions were uniformly sampled from a replay memory. However, this approach simply replays transitions at the same frequency that they were originally experienced, regardless of their significance. In this paper we develop a framework for prioritizing experience, so as to replay important transitions more frequently, and therefore learn more efficiently. We use prioritized experience replay in Deep Q-Networks (DQN), a reinforcement learning algorithm that achieved human-level performance across many Atari games. DQN with prioritized experience replay achieves a new state-of-the-art, outperforming DQN with uniform replay on 41 out of 49 games.

1 INTRODUCTION

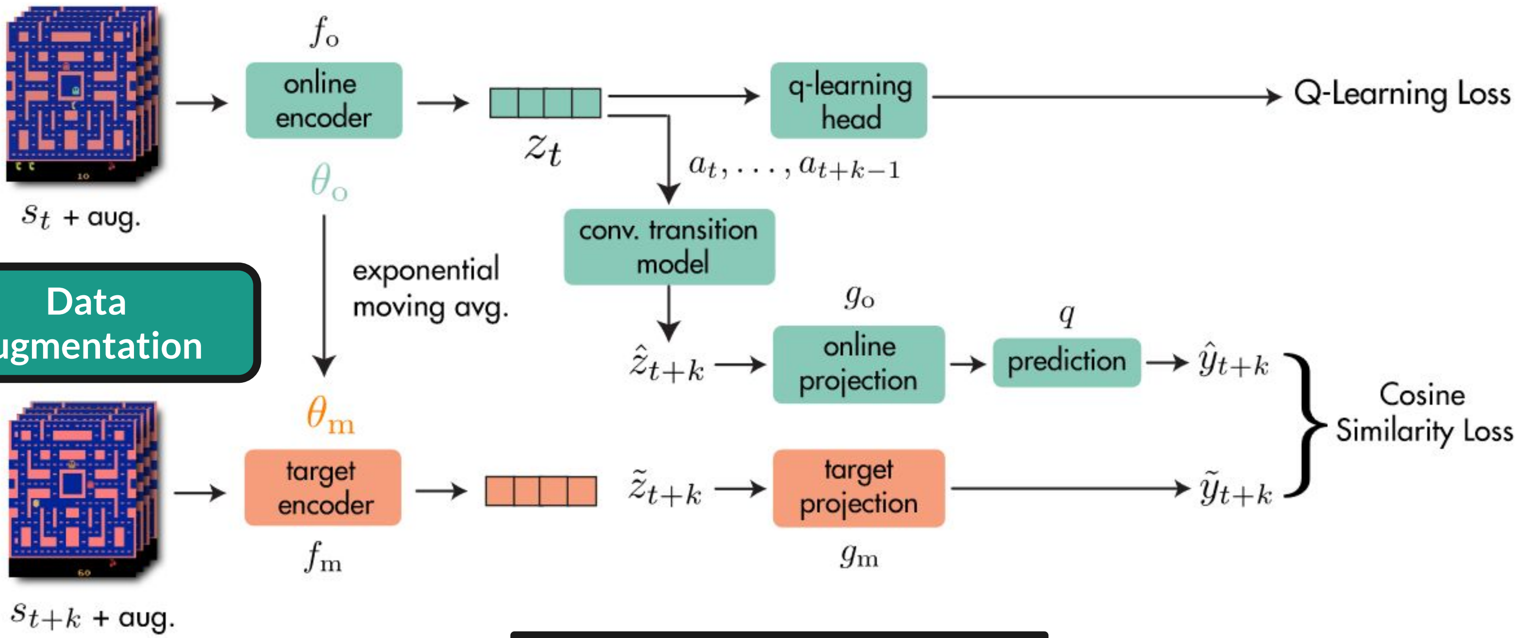
Online reinforcement learning (RL) agents incrementally update their parameters (of the policy, value function or model) while they observe a stream of experience. In their simplest form, they discard incoming data immediately, after a single update. Two issues with this are (a) strongly correlated updates that break the i.i.d. assumption of many popular stochastic gradient-based algo-

But… 38 days?

Training Sample-Efficient Models is hard!

1. Reward signal is sparse
2. Re-using experiences from replay buffer leads to massive overfitting

Self-Predictive Representations (SPR)



Data augmentation

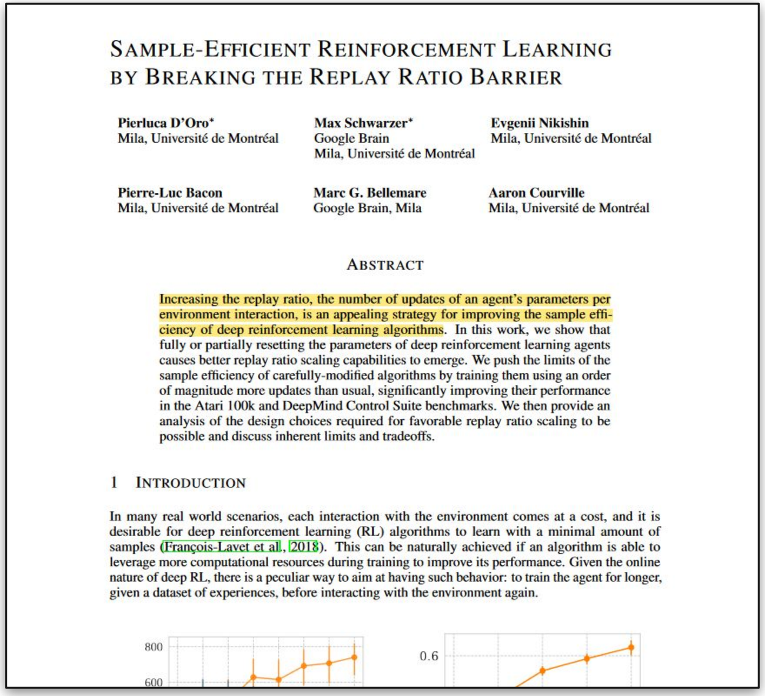
$$\boxed{\mathcal{L}_\theta^{\text{total}} = \mathcal{L}_\theta^{\text{RL}} + \lambda \mathcal{L}_\theta^{\text{SPR}}}$$

Schwarzer et. al. 2021

Making Experiences Go Farther

- **Idea:** reset / partially-reset parameters periodically

- **Shrink-and-perturb**: interpolate weights partway to random values (they move weights 20% to random)



SAMPLE-EFFICIENT REINFORCEMENT LEARNING BY BREAKING THE REPLAY RATIO BARRIER

Pierluca D'Oro* Mila, Université de Montréal

Max Schwarzer* Google Brain Mila, Université de Montréal

Evgenii Nikishin Mila, Université de Montréal

Pierre-Luc Bacon Mila, Université de Montréal

Marc G. Bellemare Google Brain, Mila

Aaron Courville Mila, Université de Montréal

ABSTRACT

Increasing the replay ratio, the number of updates of an agent's parameters per environment interaction, is an appealing strategy for improving the sample efficiency of deep reinforcement learning algorithms. In this work, we show that fully or partially resetting the parameters of deep reinforcement learning agents causes better replay ratio scaling capabilities to emerge. We push the limits of the sample efficiency of carefully-modified algorithms by training them using an order of magnitude more updates than usual, significantly improving their performance in the Atari 100k and DeepMind Control Suite benchmarks. We then provide an analysis of the design choices required for favorable replay ratio scaling to be possible and discuss inherent limits and tradeoffs.

1 INTRODUCTION

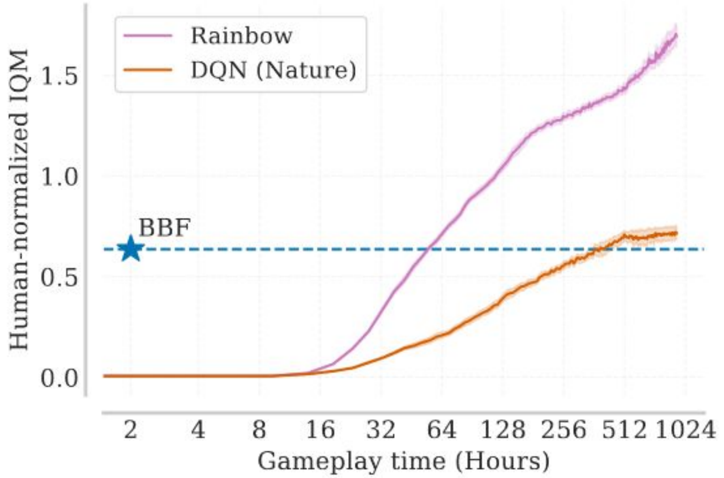
In many real world scenarios, each interaction with the environment comes at a cost, and it is desirable for deep reinforcement learning (RL) algorithms to learn with a minimal amount of samples (François-Lavet et al., 2018). This can be naturally achieved if an algorithm is able to leverage more computational resources during training to improve its performance. Given the online nature of deep RL, there is a peculiar way to aim at having such behavior: to train the agent for longer, given a dataset of experiences, before interacting with the environment again.

D'Oro et al. 2023

Bigger, Better, Faster (BBF)

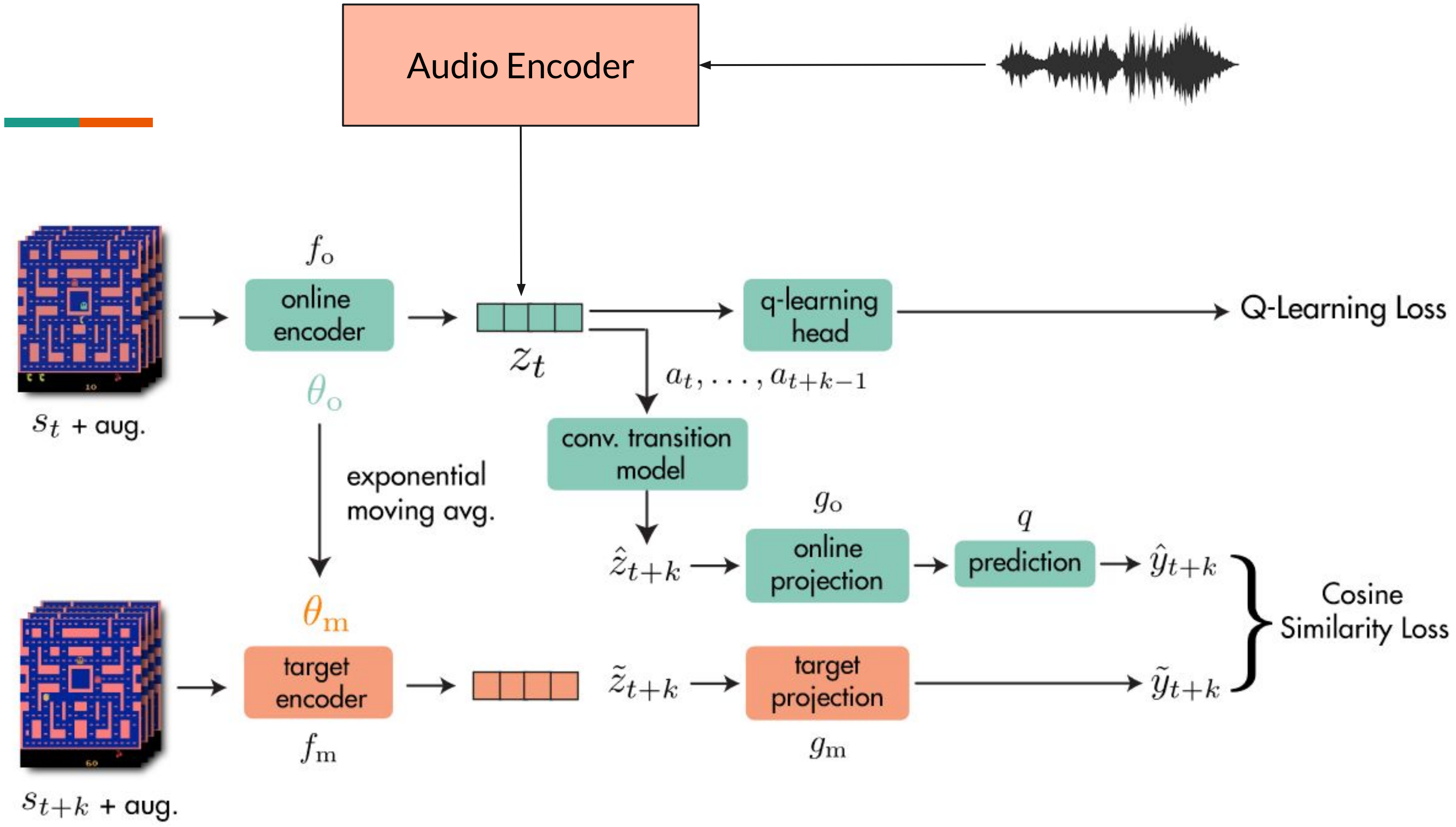
Everything so far, with:

- Wider/deeper network
- Carefully-designed Regularization
  - **Shrink-and-Perturb**: interpolate 50% to random values
  - **Weight decay**
  - **Exponential schedule** for update horizon (n) and discount factor (γ)



Schwarzer et. al. 2023

Our Model

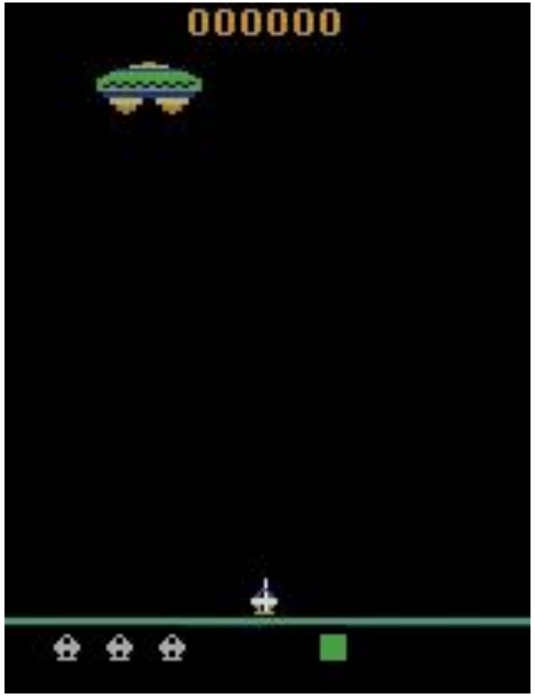


Schwarzer et. al. 2021

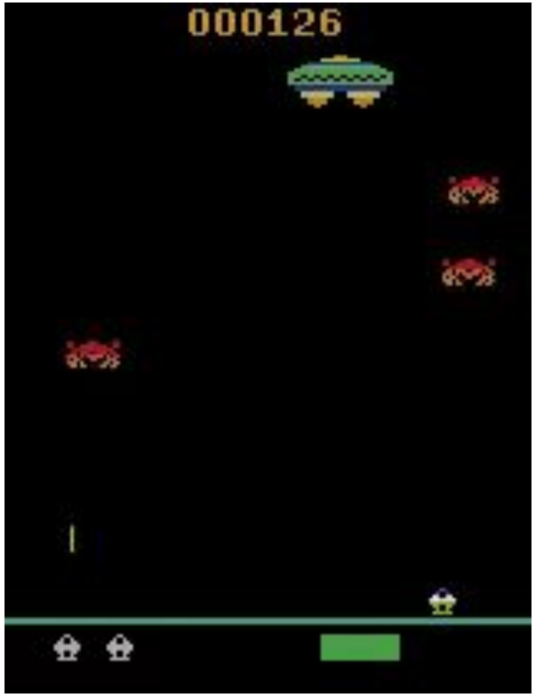
Results (!)

*Replay Ratio: 2

10k steps (~12 minutes)



25k steps (~25 minutes)



Questions?

Papers & Code

- Bigger, Better, Faster 2023- Github
- Self-Predictive Representations- Github
- Threads
- Audio

- Cross-modal Attentive Skill Learner- Github
- A Distributional Perspective on Reinforcement Learning
- Dueling Network Architectures for Deep Reinforcement Learning- Github
- Deep Reinforcement Learning with Double Q-learning