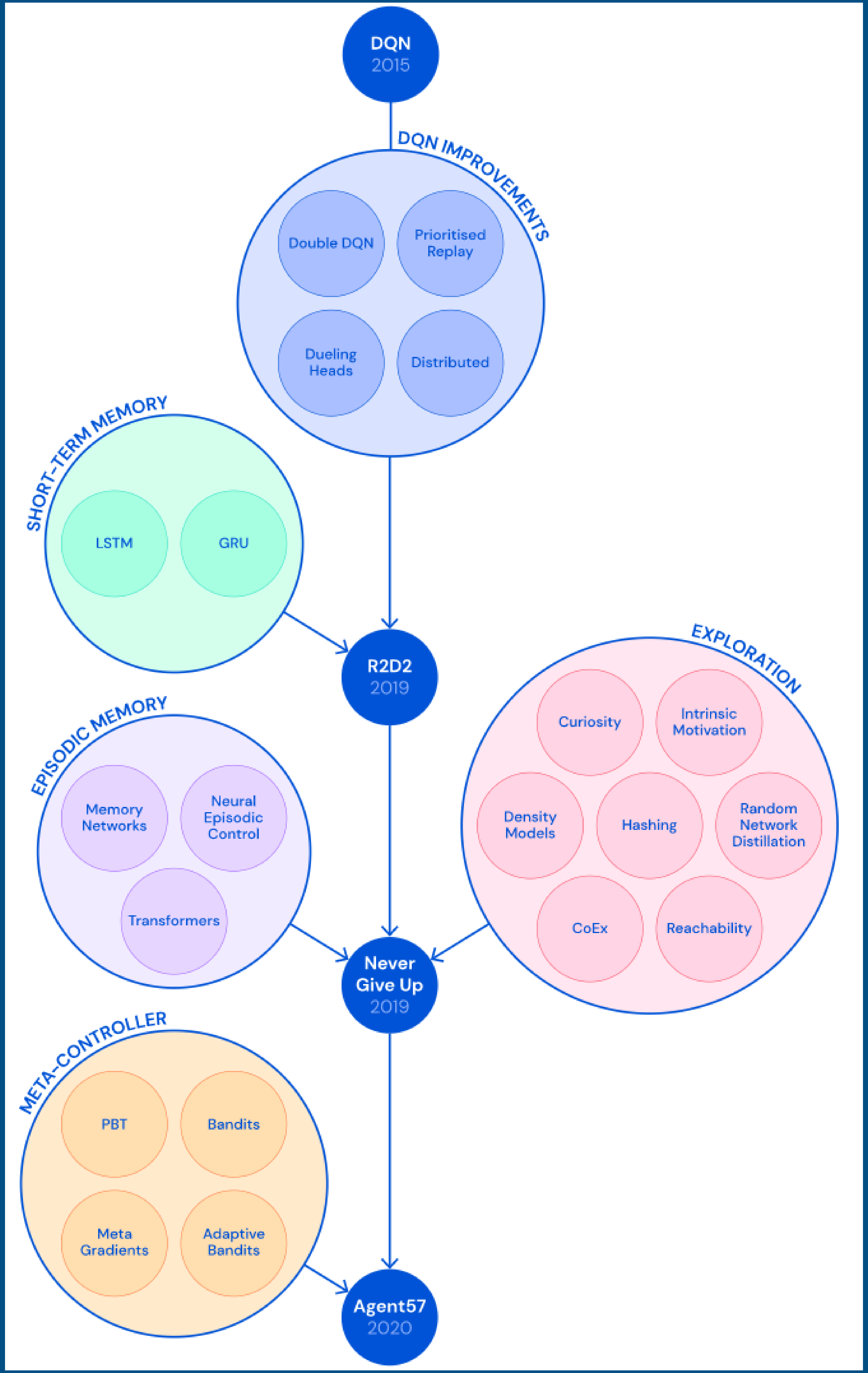
Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

옥찬호
utilForever@gmail.com


DQN
2015
DQN IMPROVEMENTS
Double DQN
Prioritised Replay
Dueling Heads
Distributed
SHORT-TERM MEMORY
LSTM
GRU
R2D2
2019
EPISODIC MEMORY
Memory Networks
Neural Episodic Control
Transformers
EXPLORATION
Curiosity
Intrinsic Motivation
Density Models
Hashing
Random Network Distillation
CoEx
Reachability
Never Give Up
2019
META-CONTROLLER
PBT
Bandits
Meta Gradients
Adaptive Bandits
Agent57
2020

Introduction

- Arcade Learning Environment (ALE)
  - An interface to a diverse set of Atari 2600 game environments designed to be engaging and challenging for human players
  - The Atari 2600 games are well suited for evaluating general competency in AI agents for three main reasons
    - 1) Varied enough to claim generality
    - 2) Each interesting enough to be representative of settings that might be faced in practice
    - 3) Each created by an independent party to be free of experimenter's bias

Introduction

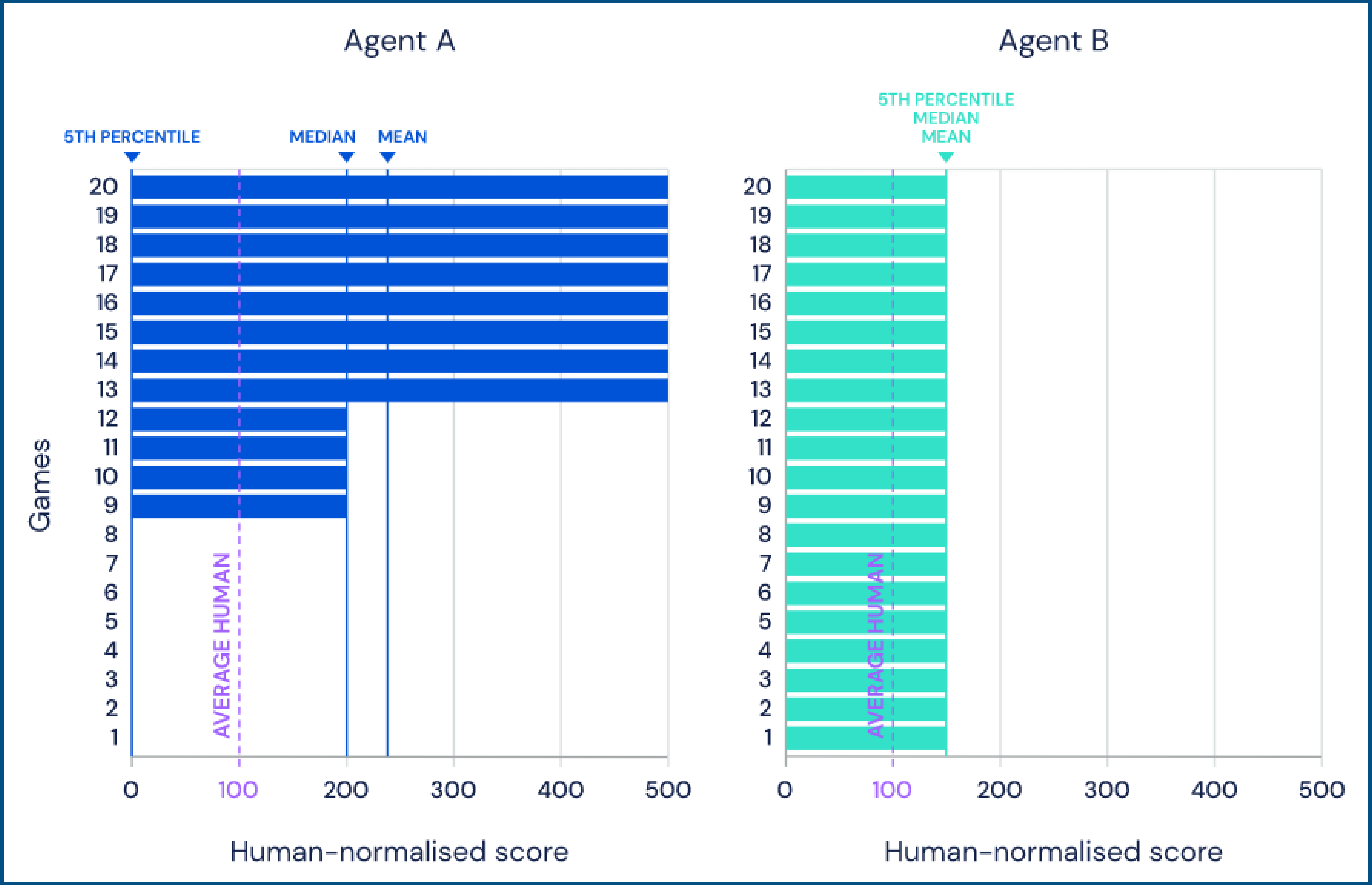
- Previous achievement of Deep RL in ALE
  - DQN (Minh et al., 2015) : 100% HNS on 23 games
  - Rainbow (M. Hessel et al., 2017) : 100% HNS on 53 games
  - R2D2 (Kapturowski et al., 2018) : 100% HNS on 52 games
  - MuZero (Schrittwieser et al., 2019) : 100% HNS on 51 games

→ Despite all efforts, no single RL algorithm has been able to achieve over 100% HNS on all 57 Atari games with one set of hyperparameters.

Introduction

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

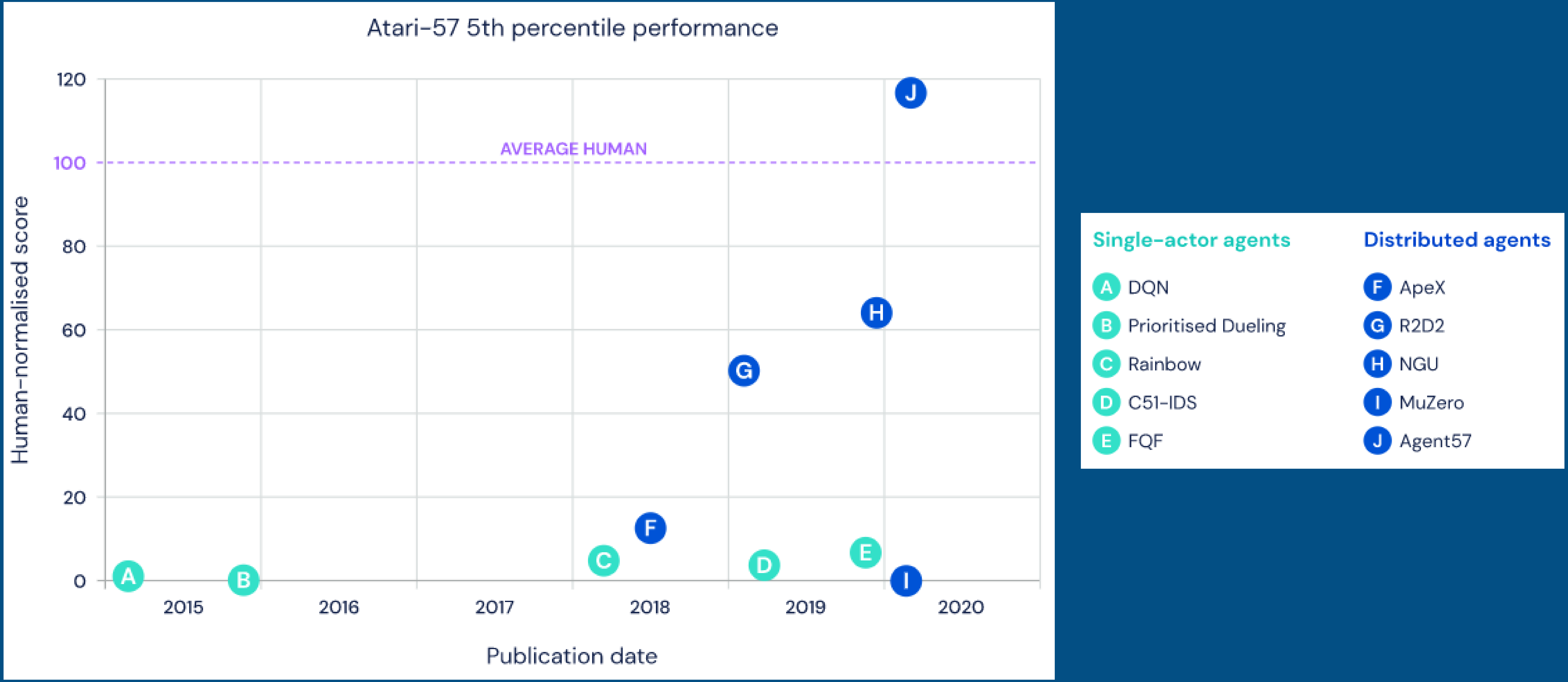
- How to measure Artificial General Intelligence?



Introduction

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- Atari-57 5th percentile performance



Introduction

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- Challenging issues of Deep RL in ALE
  - 1) Long-term credit assignment
    - This problem is particularly hard **when rewards are delayed and credit needs to be assigned over long sequences of actions**.
    - The game of Skiing is a canonical example due to its peculiar reward structure.
    - The goal of the game is to run downhill through all gates as fast as possible.
    - A penalty of five seconds is given for each missed gate.
    - The reward, given only at the end, is proportional to the time elapsed.
    - long-term credit assignment is needed to understand why an action taken early in the game (e.g. missing a gate) has a negative impact in the obtained reward.

Introduction

- Challenging issues of Deep RL in ALE
  - 2) Exploration in large high dimensional state spaces
    - Games like Private Eye, Montezuma's Revenge, Pitfall! or Venture are widely considered **hard exploration games as hundreds of actions may be required before a first positive reward is seen**.
    - In order to succeed, the agents need to keep exploring the environment despite the apparent impossibility of finding positive rewards.

Introduction

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- Challenging issues of Deep RL in ALE
  - Exploration algorithms in deep RL generally fall into three categories:
    - Randomize value function
    - Unsupervised policy learning
    - Intrinsic motivation: NGU
  - Other work combines handcrafted features, domain-specific knowledge or privileged pre-training to side-step the exploration problem.

Introduction

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- In summary, our contributions are as follows:
  - 1) **A new parameterization of the state-action value function** that decomposes the contributions of the intrinsic and extrinsic rewards. As a result, we significantly increase the training stability over a large range of intrinsic reward scales.
  - 2) **A meta-controller**: an adaptive mechanism to select which of the policies (parameterized by exploration rate and discount factors) to prioritize throughout the training process. This allows the agent to control the exploration/exploitation trade-off by dedicating more resources to one or the other.

Introduction

- In summary, our contributions are as follows:
  - 3) Finally, **we demonstrate for the first time performance that is above the human baseline across all Atari 57 games**. As part of these experiments, we also find that simply re-tuning the backprop through time window to be twice the previously published window for R2D2 led to superior long-term credit assignment (e.g., in Solaris) while still maintaining or improving overall performance on the remaining games.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- Our work builds on top of the **Never Give Up (NGU)** agent, which combines two ideas:
  - The curiosity-driven exploration
  - Distributed deep RL agents, in particular R2D2
- NGU computes an intrinsic reward in order to encourage exploration. This reward is defined by combining
  - Per-episode novelty
  - Life-long novelty

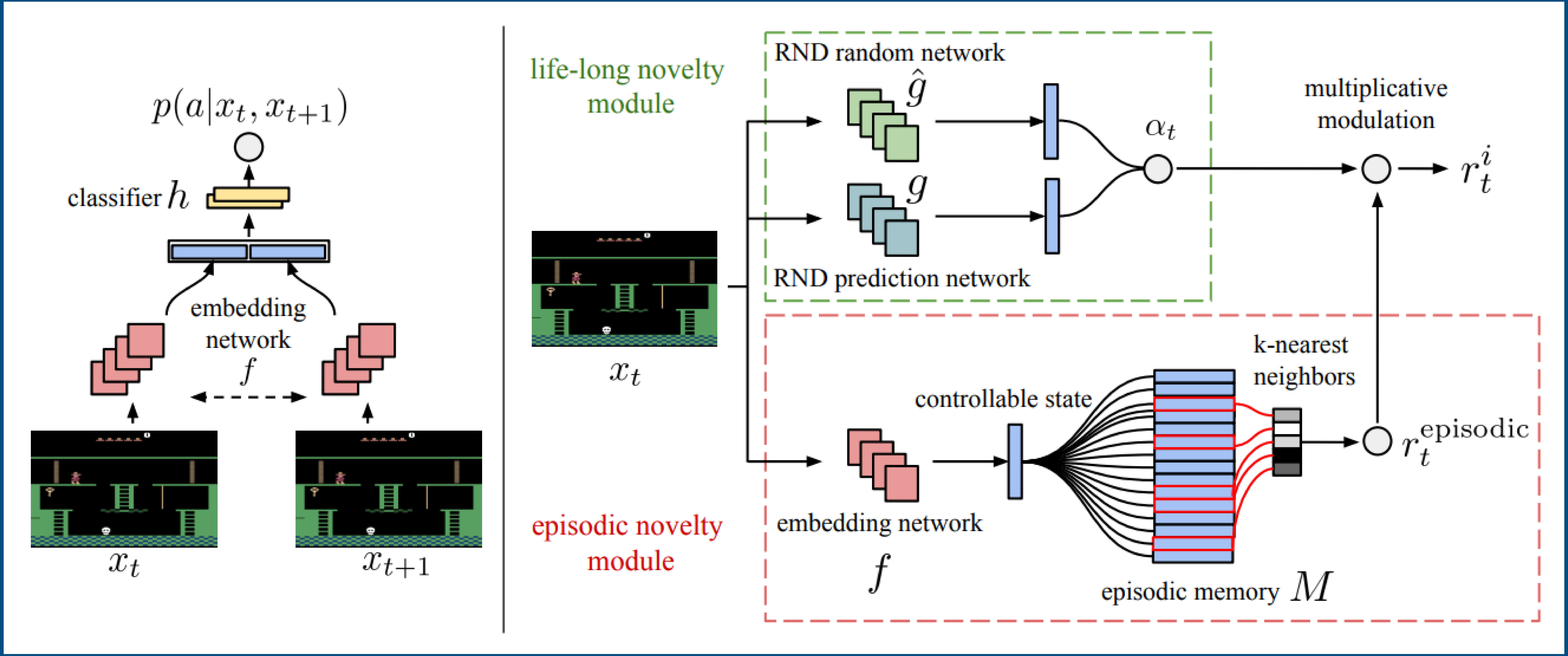
Background: NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- **Per-episode novelty**, r_t^{episodic}, **rapidly vanishes over the course of an episode**, and it is computed by comparing observations to the contents of an episodic memory.
- **The life-long novelty**, α_t, **slowly vanishes throughout training**, and it is computed by using a parametric model.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020



Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- With this, the **<u>intrinsic reward r_i^t</u>** is defined as follows:

$$r_t^i = r_t^{\text{episodic}} \cdot \min\{\max\{\alpha_t, 1\}, L\}$$

  where $L = 5$ is a chosen maximum reward scaling.
- This leverages the long-term novelty provided by α_t, while r_t^{episodic} continues to encourage the agent to explore within an episode.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- At time t, NGU adds N different scales of the same intrinsic reward $\beta_j r_t^i$ ($\beta_j \in \mathbb{R}^+, j \in 0, \ldots, N-1$) to the extrinsic reward provided by the environment, r_t^e, to form N potential total rewards $r_{j,t} = r_t^e + \beta_j r_t^i$.
- Consequently, NGU aims to learn the N different associated optimal state-action value functions $Q_{r_j}^*$ associated with each reward function $r_{j,t}$.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- The exploration rates β_j are parameters that control the degree of exploration.
  - Higher values will encourage exploratory policies.
  - Smaller values will encourage exploitative policies.
- Additionally, for purposes of learning long-term credit assignment, each $Q^*_{r_j}$ has its own associated discount factor γ_j.

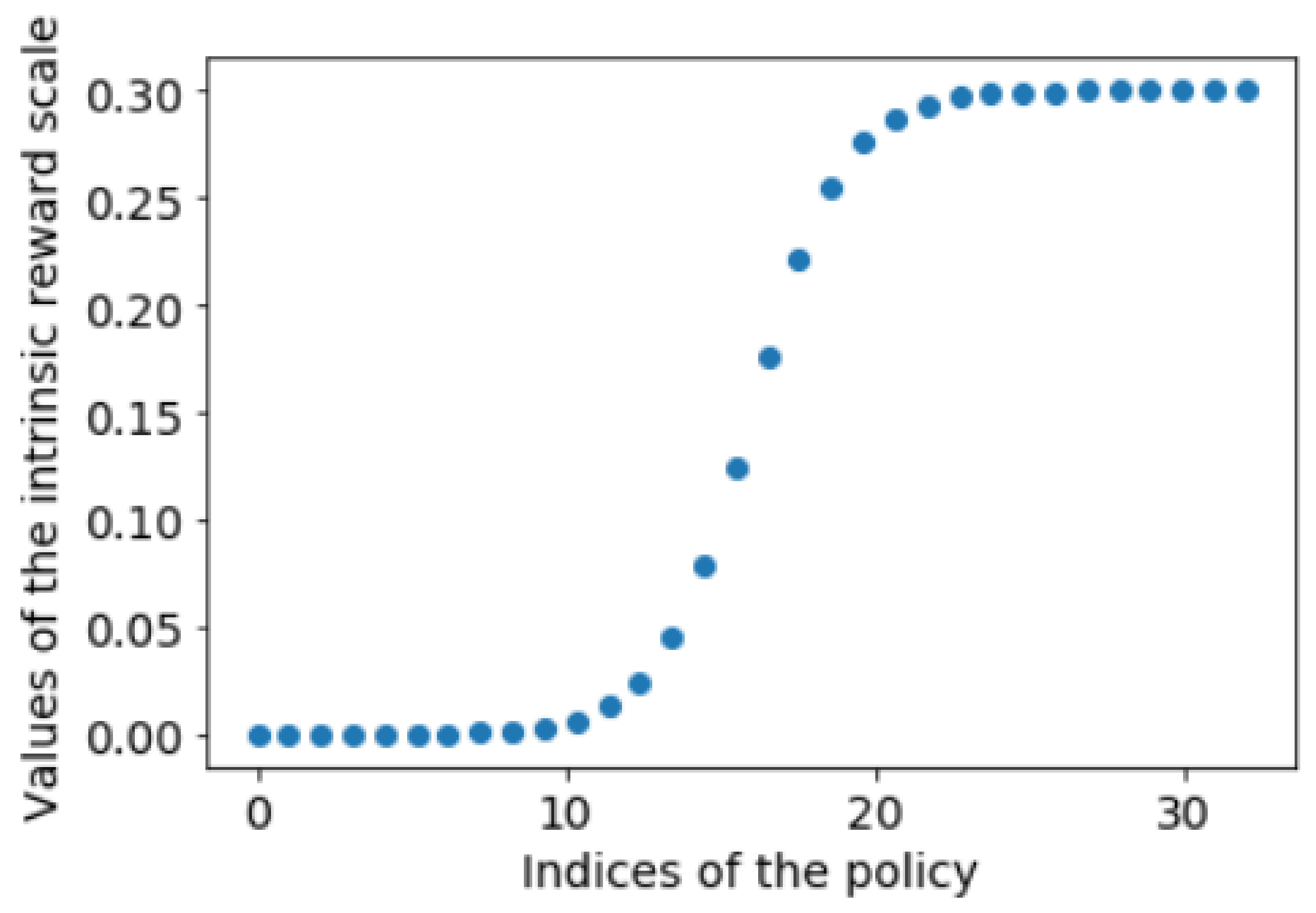
Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

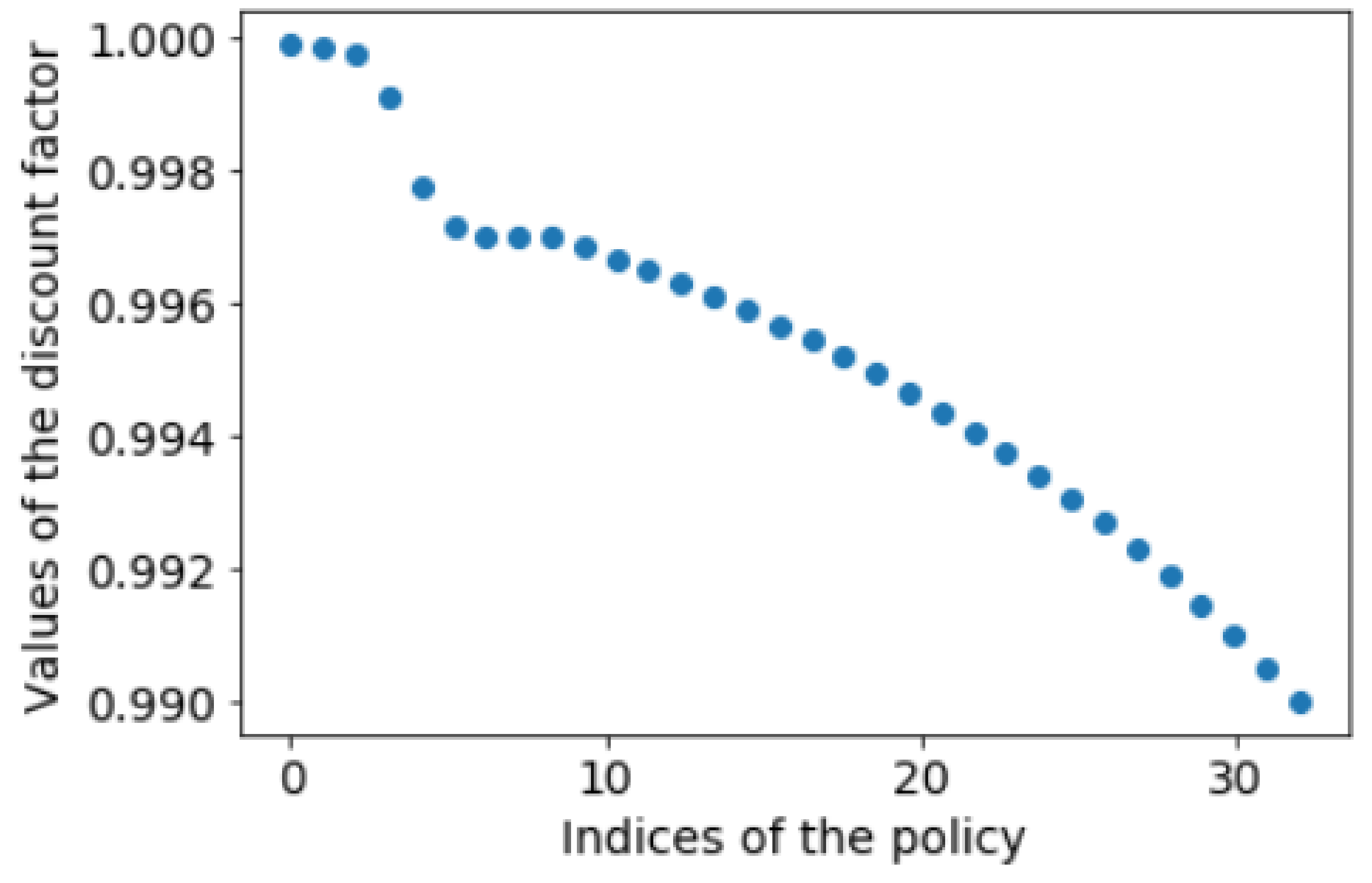
- Since the intrinsic reward is typically much more dense than the extrinsic reward, $\{(\beta_j, \gamma_j)\}_{j=0}^{N-1}$ are chosen so as to allow for long term horizons (high values of γ_j) for exploitative policies (small values of β_j) and small term horizons (low values of γ_j) for exploratory policies (high values of β_j).

Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020



(a) Values taken by the $\{\beta_i\}_{i=0}^{N-1}$



(b) Values taken by the $\{\gamma_i\}_{i=0}^{N-1}$

Figure 11. Values taken by the $\{\beta_i\}_{i=0}^{N-1}$ and the $\{\gamma_i\}_{i=0}^{N-1}$ for $N = 32$ and $\beta = 0.3$.

$$\beta_j = \begin{cases} 0 & \text{if } j = 0 \\ \beta = 0.3 & \text{if } j = N-1 \\ \beta \cdot \sigma(10 \frac{2j-(N-2)}{N-2}) & otherwise \end{cases}, \quad \gamma_j = \begin{cases} \gamma_0 & \text{if } j = 0 \\ \gamma_1 + (\gamma_0 - \gamma_1)\sigma(10 \frac{2i-6}{6}) & \text{if } j \in \{1, \ldots, 6\} \\ \gamma_1 & \text{if } j = 7 \\ 1 - \exp\left(\frac{(N-9)\log(1-\gamma_1) + (j-8)\log(1-\gamma_2)}{N-9}\right) & otherwise \end{cases}$$

where $N = 32$, $\gamma_0 = 0.9999$, $\gamma_1 = 0.997$ and $\gamma_2 = 0.99$.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- To learn the state-action value function $Q^*_{r_j}$, NGU trains a recurrent neural network $Q(x, a, j; \theta)$, where j is a one-hot vector indexing one of N implied MDPs (in particular (β_j, γ_j)), x is the current observation, a is an action, and θ are the parameters of the network (including the recurrent state).
- In practice, NGU can be unstable and fail to learn an appropriate approximation of $Q^*_{r_j}$ for all the state-action value functions in the family, even in simple environments.

Background: NGU

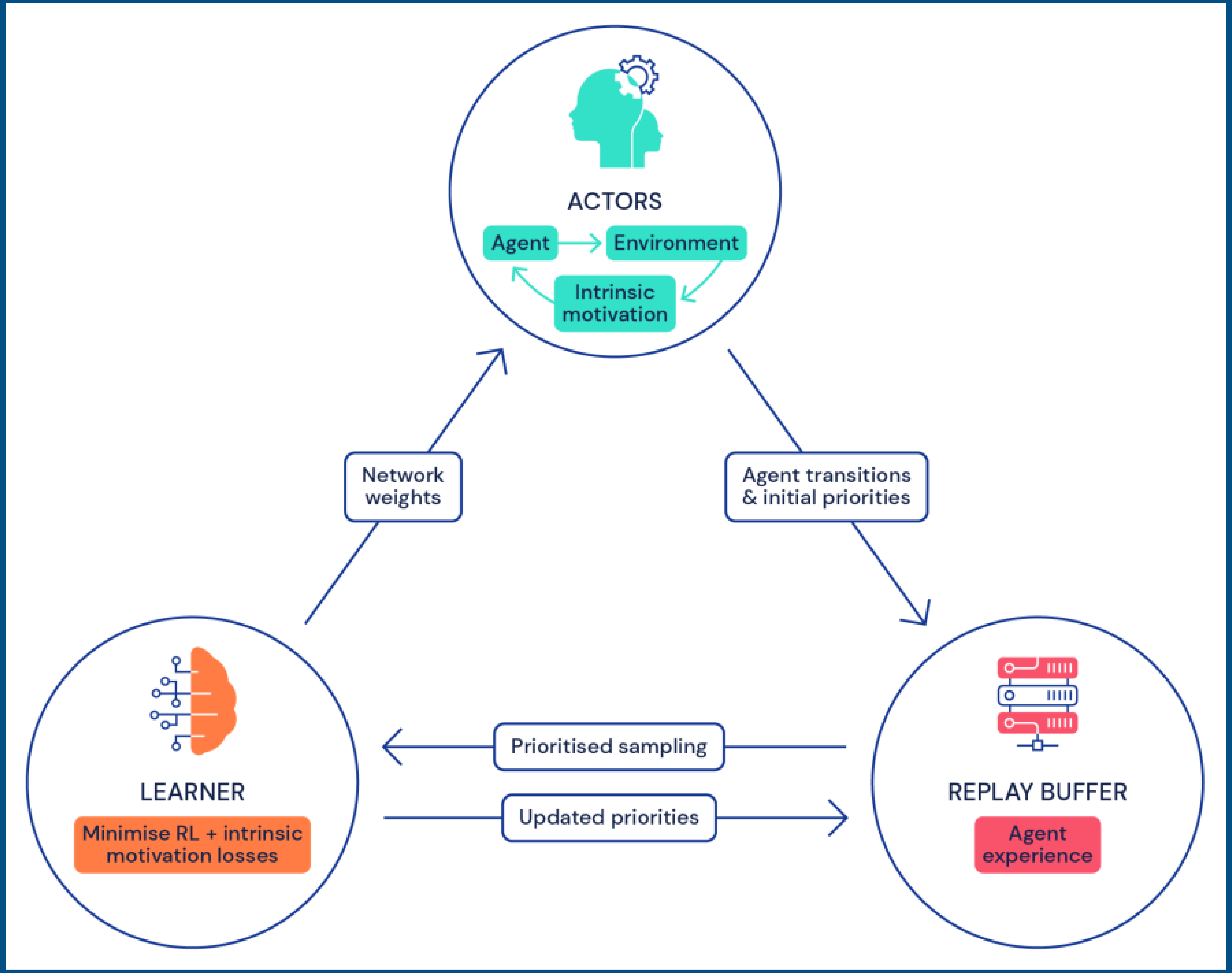
Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- This is especially the case when the scale and sparseness of r_t^e and r_t^i are both different, or when one reward is more noisy than the other.
- We conjecture that learning a common state-action value function for a mix of rewards is difficult when the rewards are very different in nature.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- Our agent is a deep distributed RL agent, in the lineage of R2D2 and NGU.



Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- It decouples the data collection and the learning processes by having many actors feed data to a central prioritized replay buffer.
- A learner can then sample training data from this buffer.
- More precisely, the replay buffer contains sequences of transitions that are removed regularly in a FIFO-manner.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- These sequences come from actor processes that interact with independent copies of the environment, and they are prioritized based on temporal differences errors.
- The priorities are initialized by the actors and updated by the learner with the updated state-action value function $Q(x, a, j; \theta)$.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- According to those priorities, the learner samples sequences of transitions from the replay buffer to construct an RL loss.
- Then, it updates the parameters of the neural network $Q(x, a, j; \theta)$ by minimizing the RL loss to approximate the optimal state-action value function.
- Finally, each actor shares the same network architecture as the learner but with different weights.

Background: NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- We refer as θ_l to the parameters of the l−th actor. The learner weights θ are sent to the actor frequently, which allows it to update its own weights θ_l.
- Each actor uses different values ϵ_l, which are employed to follow an ϵ_l−greedy policy based on the current estimate of the state−action value function $Q(x, a, j; \theta_l)$.
- In particular, at the beginning of each episode and in each actor, NGU uniformly selects a pair (β_j, γ_j).

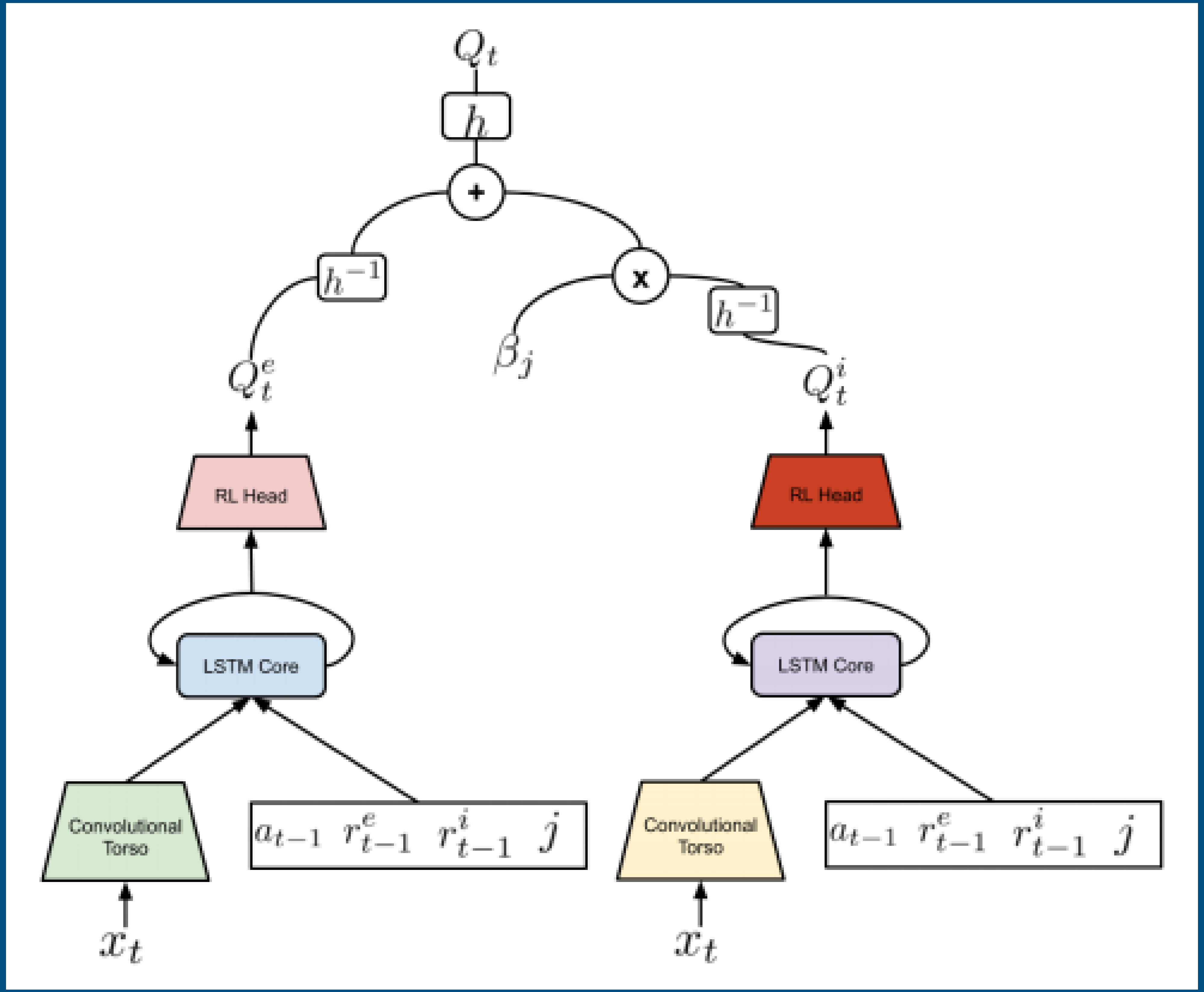
Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- State-Action Value Function Parameterization

$$Q(x,a,j;\theta) = Q(x,a,j;\theta^e) + \beta_j Q(x,a,j;\theta^i)$$

- $Q(x,a,j;\theta^e)$: Extrinsic component
- $Q(x,a,j;\theta^i)$: Intrinsic component
- Weight: θ^e and θ^i (separately)
  with identical architecture and $\theta = \theta^i \cup \theta^e$
- Reward: r^e and r^i (separately)
  But, same target policy
  $\pi = \mathrm{argmax}_{a\in\mathcal{A}} Q(x,a,j;\theta^e)$



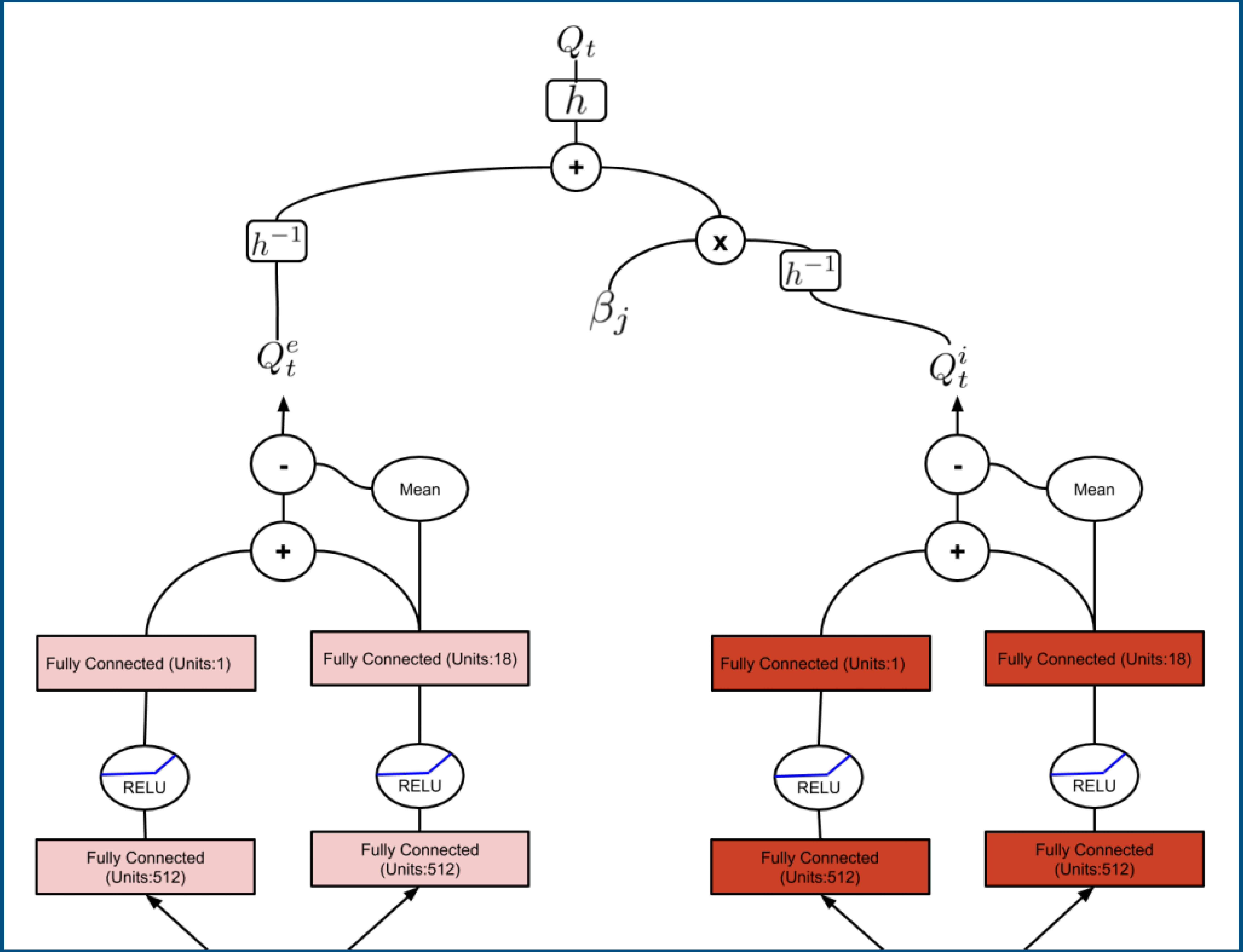
Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- State-Action Value Function Parameterization
  - We show that **this optimization of separate state-action values is equivalent to the optimization of the original single state-action value function with reward $r^e + \beta_j r^i$** in Appendix B.
  - Even though the theoretical objective being optimized is the same, the parameterization is different: we **use two different neural networks to approximate each one of these state-action values**.

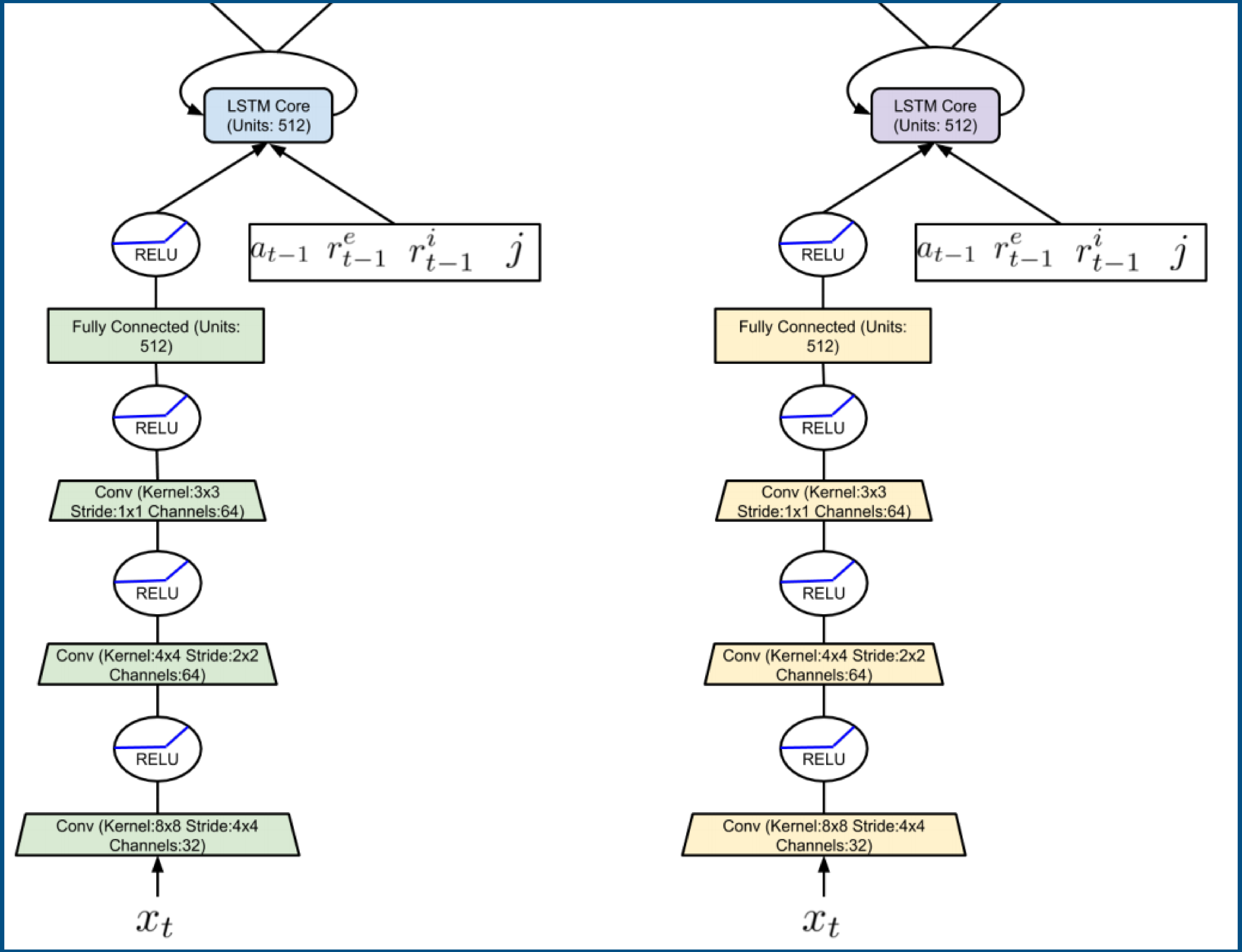
Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020


Q_t
h
+
h^{-1}
x
h^{-1}
β_j
Q_t^e
Q_t^i
-
Mean
+
-
Mean
+
Fully Connected (Units:1)
Fully Connected (Units:18)
Fully Connected (Units:1)
Fully Connected (Units:18)
RELU
RELU
RELU
RELU
Fully Connected (Units:512)
Fully Connected (Units:512)
Fully Connected (Units:512)
Fully Connected (Units:512)

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020



LSTM Core (Units: 512)
a_{t-1} r^e_{t-1} r^i_{t-1} j
RELU
Fully Connected (Units: 512)
RELU
Conv (Kernel:3x3 Stride:1x1 Channels:64)
RELU
Conv (Kernel:4x4 Stride:2x2 Channels:64)
RELU
Conv (Kernel:8x8 Stride:4x4 Channels:32)
x_t
LSTM Core (Units: 512)
a_{t-1} r^e_{t-1} r^i_{t-1} j
RELU
Fully Connected (Units: 512)
RELU
Conv (Kernel:3x3 Stride:1x1 Channels:64)
RELU
Conv (Kernel:4x4 Stride:2x2 Channels:64)
RELU
Conv (Kernel:8x8 Stride:4x4 Channels:32)
x_t

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- Adaptive Exploration over a Family of Policies
  - The core idea of NGU is to jointly train a family of policies with different degrees of exploratory behavior using <u>a single network architecture</u>.
  - In this way, training these exploratory policies plays the role of a set of auxiliary tasks that can help train the shared architecture even in the absence of extrinsic rewards.
  - A major limitation of this approach is that <u>**all policies are trained equally, regardless of their contribution to the learning progress**</u>.

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- We propose to incorporate a **meta-controller** that can adaptively select which policies to use both at training and evaluation time. This carries two important consequences.

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- Meta-controller with adaptively select
  - 1) By selecting which policies to prioritize during training, we can allocate more of the capacity of the network to better represent the state-action value function of the policies that are most relevant for the task at hand.
  - Note that this is likely to change throughout the training process, naturally building a curriculum to facilitate training.

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- Meta−controller with adaptively select
  - Policies are represented by pairs of exploration rate and discount factor, (β_j, γ_j), which determine the discounted cumulative rewards to maximize.
  - It is natural to expect policies with higher β_j and lower γ_j to make more progress early in training, while the opposite would be expected as training progresses.

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- Meta-controller with adaptively select
  - 2) This mechanism also provides a natural way of choosing the best policy in the family to use at evaluation time.
  - Considering a wide range of values of γ_j with $\beta_j \approx 0$, provides a way of automatically adjusting the discount factor on a per-task basis.
  - This significantly increases the generality of the approach.

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- We propose to implement the meta-controller using **a nonstationary multi-arm bandit algorithm** running independently on each actor.
- The reason for this choice, as opposed to a global meta-controller, is that each actor follows a different ϵ_l-greedy policy which may alter the choice of the optimal arm.

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- **The multi-armed bandit problem** is a problem in which a fixed limited set of resources must be allocated between competing (alternative) choices in a way that maximizes their expected gain, when each choice's properties are only partially known at the time of allocation, and may become better understood as time passes or by allocating resources to the choice.

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- Nonstationary multi-arm bandit algorithm
  - Each arm j from the N-arm bandit is linked to a policy in the family and corresponds to a pair (β_j, γ_j). At the beginning of each episode, say, the kth episode, the meta-controller chooses an arm J_k setting which policy will be executed. (J_k: random variable)
  - Then the l-th actor acts l-greedily with respect to the corresponding state-action value function, $Q(x, a, J_k; \theta_l)$, for the whole episode.
  - The undiscounted extrinsic episode returns, noted $R_k^e(J_k)$, are used as a reward signal to train the multi-arm bandit algorithm of the meta-controller.

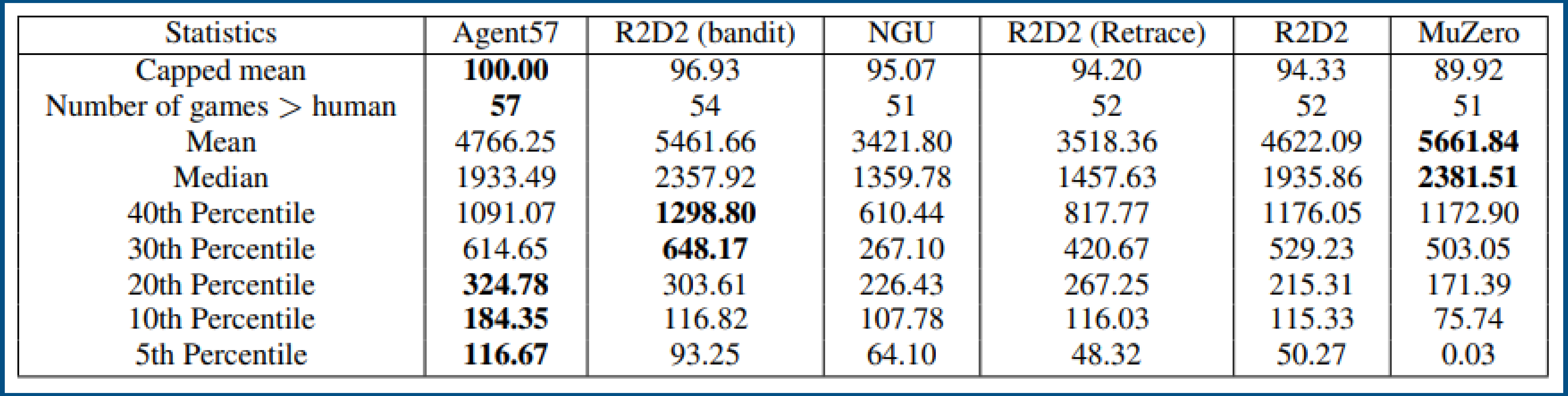
Improvements to NGU

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

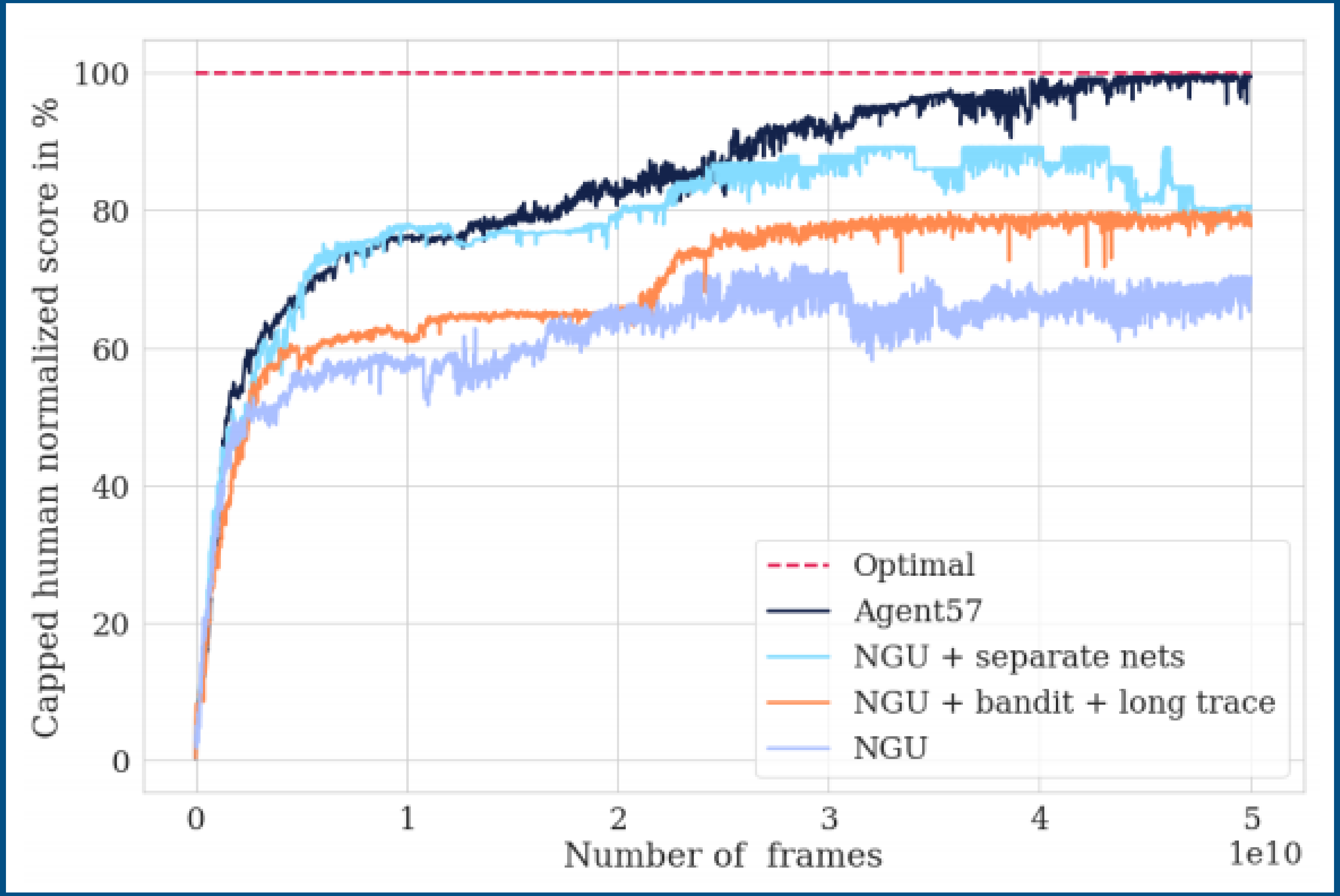
- Nonstationary multi-arm bandit algorithm
  - The reward signal $R_k^e(J_k)$ is non-stationary, as the agent changes throughout training. Thus, a classical bandit algorithm such as Upper Confidence Bound (UCB) will not be able to adapt to the changes of the reward through time.
  - Therefore, we employ **a simplified sliding-window UCB (SW-UCB) with ϵ_{UCB}-greedy exploration**.
  - With probability $1 - \epsilon_{\mathrm{UCB}}$, this algorithm runs a slight modification of classic UCB on a sliding window of size τ and selects a random arm with probability ϵ_{UCB}.

Experiments

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

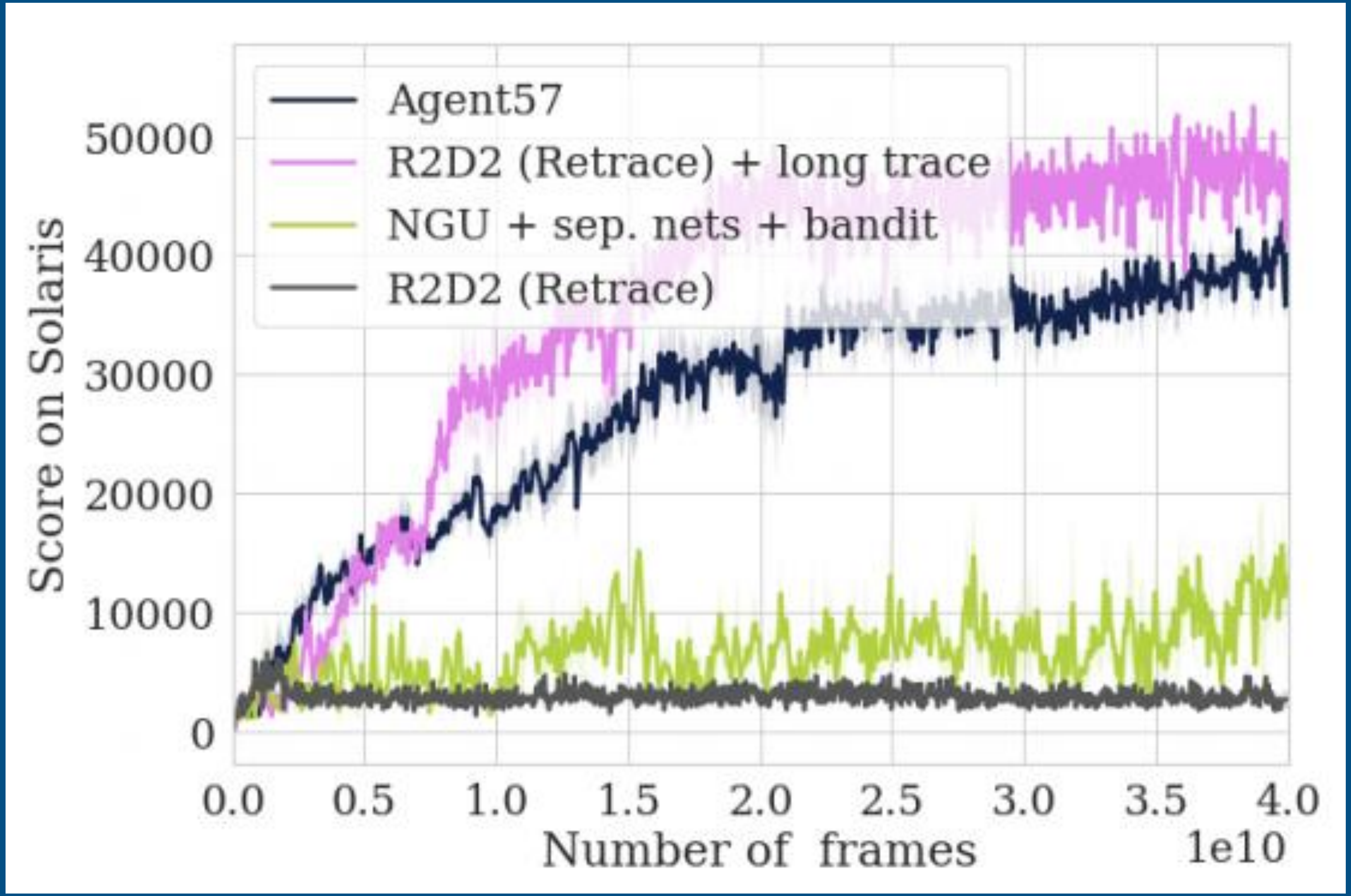


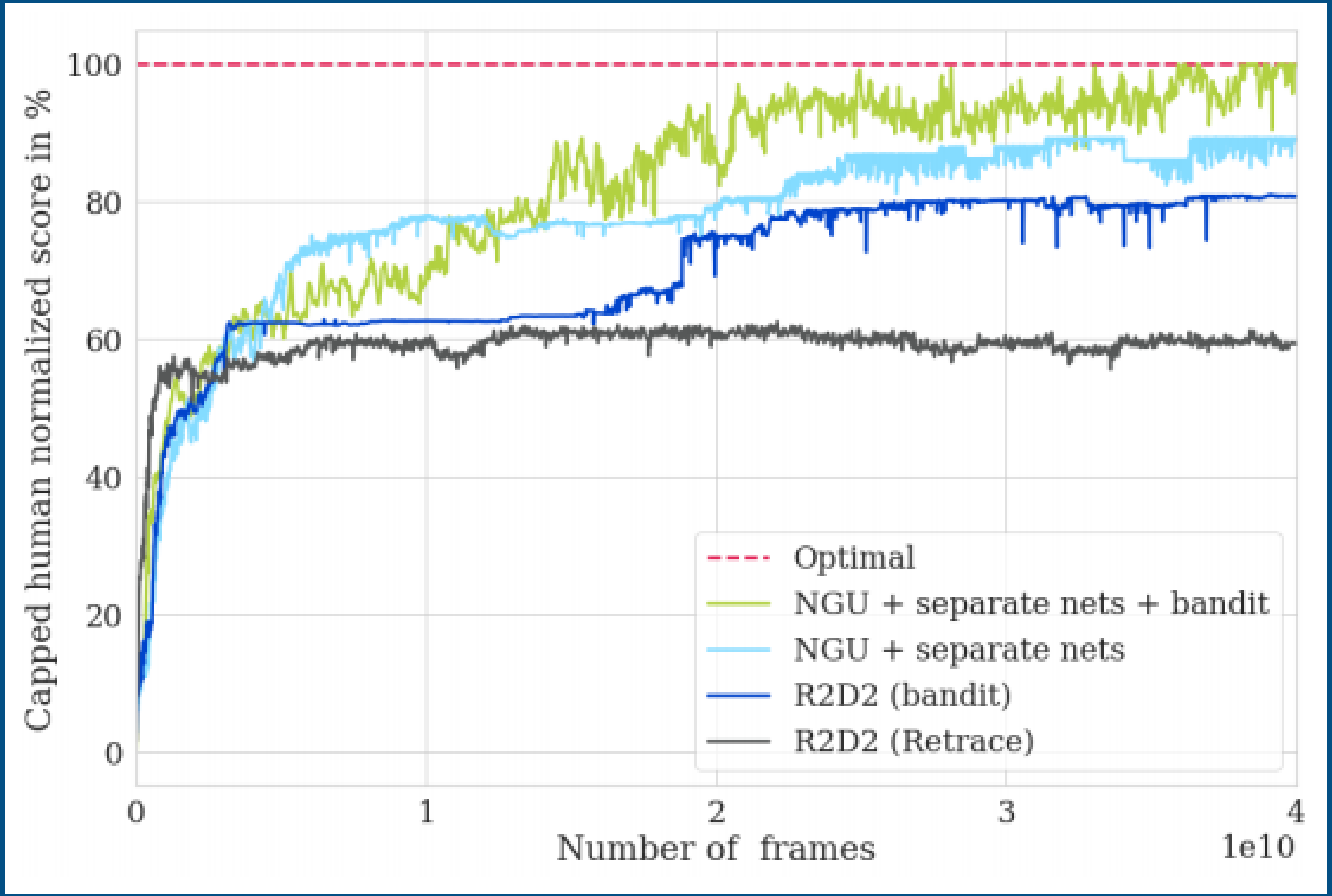
Statistics	Agent57	R2D2 (bandit)	NGU	R2D2 (Retrace)	R2D2	MuZero
Capped mean	**100.00**	96.93	95.07	94.20	94.33	89.92
Number of games > human	**57**	54	51	52	52	51
Mean	4766.25	5461.66	3421.80	3518.36	4622.09	**5661.84**
Median	1933.49	2357.92	1359.78	1457.63	1935.86	**2381.51**
40th Percentile	1091.07	**1298.80**	610.44	817.77	1176.05	1172.90
30th Percentile	614.65	**648.17**	267.10	420.67	529.23	503.05
20th Percentile	**324.78**	303.61	226.43	267.25	215.31	171.39
10th Percentile	**184.35**	116.82	107.78	116.03	115.33	75.74
5th Percentile	**116.67**	93.25	64.10	48.32	50.27	0.03



Experiments

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020





Conclusions

Agent57: Outperforming the Atari Human Benchmark, Badia, A. P. et al, 2020

- We present the first deep reinforcement learning agent with performance above the human benchmark on all 57 Atari games.
- The agent is able to balance the learning of different skills that are required to be performant on such diverse set of games:
  - Exploration
  - Exploitation
  - Long-term credit assignment

Conclusions

Agent57: Outperforming the Atari Human Benchmark,
Badia, A. P. et al, 2020

- To do that, we propose simple improvements to an existing agent, Never Give Up, which has good performance on hard-exploration games, but in itself does not have strong overall performance across all 57 games.
  - 1) Using a different parameterization of the state-action value function
  - 2) Using a meta-controller to dynamically adapt the novelty preference and discount
  - 3) The use of longer backprop-through time window to learn from using the Retrace algorithm

References

- https://deepmind.com/blog/article/Agent57-Outperforming-the-human-Atari-benchmark
- https://seungeonbaek.tistory.com/4
- https://seolhokim.github.io/deeplearning/2020/04/08/NGU-review/
- https://openreview.net/pdf?id=Sye57xStvB
- https://en.wikipedia.org/wiki/Multi-armed_bandit
- https://arxiv.org/abs/0805.3415

Thank you!