# Bio-Plausible Reinforcement Learning Systems Learn to Play Atari From Human

**Sepehr Abbasi Zadeh**
Department of Computer Science
University of Toronto
sepehr@cs.toronto.edu

## Abstract

We explore a biologically plausible deep reinforcement learning system by feeding it the human observations of the experiment world. The main hypothesis is that the more similar our learning model with the actual human learning model is, the better the performance should be. We examine this idea by using the AuGMEnT deep neural network which is a bio-plausible reinforcement system with a focus on attention and show that we can instruct our agent the general policies of the environment with just a few episodes of human actions in that world. In addition, we experiment one non-bio-plausible learning system and show that it cannot earn the abilities that our bio-plausible method earns under the same settings.

## 1 Introduction

Recent developments in the deep learning have enabled reinforcement learning (RL) methods to achieve human level expertise or even surpass their performance in many tasks. New successful methods have combined deep learning with value function approximation, by using a deep convolutional neural network to represent the action-value (Q) function [5]. Specifically, a new method for training such deep Q-networks, known as DQN, has enabled RL to learn control policies in complex environments with high dimensional images as inputs. Subsequent follow-ups such as DDQN method also use the same logic [9]. In order to evaluate these methods, they usually use a package of Atari 2600 games which has established a unified and reasonable platform that feeds raw pixels to the learning agents and rewards them for each action regarding the observed environment. This method is exactly the same as what human agents use to play each game. It means that we can compare these agents' learning performance with human.

One of the main goals of these agents is to learn a generalized set of policies for acting in the same environment, however, in some environments the starting point is deterministic. This determinism dictates in a way to the agents to learn the sequences and perform well without generalization. To cope with this issue, they use the human trajectories to sample starting points for feeding to the agents. This method helps the agents to learn more generalized and robust policies in the deterministic environments with the cost of loosing some accuracy.

Despite all of the recent achievements in RL, still there are many games that agents cannot beat human. Figure 1 (based on [8]) depicts one of these games that an agent cannot receive more than 200 score after 920 hours of training, while a human can achieve higher than 6000 in less than 15 minutes. One natural question that comes to mind is, how our agents perform if we train them with human actions during their actual learning phase instead of just using them as a randomization method? Another important question is, what happens if our agents use a bio-plausible learning method? In this paper, we use a bio-plausible RL method to test the aforementioned questions and we show that we can instruct our agent the general policies of the environment using few human actions and choosing a right bio-plausible model. This experiment also suggests that we can compare the

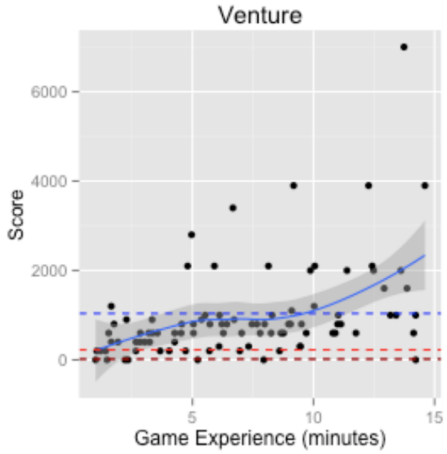

Figure 1: Human learning curves for Venture Atari game. Black horizontal line: random play. Blue horizontal line: 'expert' play. Red horizontal lines: DDQN after 10, 25, and 200 million frames of game-play experience (46, 115, and 920 hours, respectively)

performance of different bio-plausible LR methods by feeding them human actions. This comparison is in the sense that which one of them are more compatible with the actual human learning procedure.

## 2 Related Works

The use of the Atari 2600 emulator as a reinforcement learning platform was introduced by [1], who applied standard reinforcement learning algorithms with linear function approximation and generic visual features. Subsequently, results were improved by using a larger number of features and deepened neural networks. There is a thorough survey on the usages of the deep neural networks and recurrent neural networks in the reinforcement learning systems in [7] and their different learning strategies. Q-Learning has been one of the most successful and widely used methods in RL in the past few years[5, 9, 7]. However, the literature on bio-plausible RL methods is not that vast which has always been a critical issue from the neuroscientists point of view. In [2] they define more biologically-plausible versions of deep representation learning, but they fail to address the direct use of RL in their method.

## 3 Bio-Plausible Reinforcement Learning Model

We have used the *AuGMEnT* (Attention-Gated MEmory Tagging) model suggested by [6] which focuses on the role of attention on creating synaptic tags for the learning of working memories in sequential tasks. Here we quote from their paper the benefits of the AuGMEnT model and how it is bio-plausible:

> This method explains the formation of working memories during trial-and-error learning and that is inspired by the role of attention and neuromodulatory systems in the gating of neuronal plasticity. AuGMEnT addresses two well-known problems in learning theory: temporal and structural credit-assignment. The temporal credit-assignment problem arises if an agent has to learn actions that are only rewarded after a sequence of intervening actions, so that it is difficult to assign credit to the appropriate ones. AuGMEnT solves this problem like previous temporal-difference reinforcement learning (RL) theories. It learns action-values (known as Q-values), i.e. the amount of reward that is predicted for a particular action when executed in a particular state of the world. If the outcome deviates from the reward-prediction, a neuromodulatory signal that codes the global reward-prediction error (RPE) gates

> synaptic plasticity in order to change the Q-value, in accordance with experimental findings. The key new property of AuGMEnT is that it can also learn tasks that require working memory, thus going beyond standard RL models.
>
> AuGMEnT also solves the structural credit-assignment problem of networks with multiple layers. Which synapses should change to improve performance? AuGMEnT solves this problem with an "attentional" feedback mechanism. The output layer has feedback connections to units at earlier levels that provide feedback to those units that were responsible for the action that was selected. We propose that this feedback signal tags relevant synapses and that the persistence of tags (known as eligibility traces) permits learning if time passes between the action and the RPE. We will here demonstrate the neuroscientific plausibility of AuGMEnT. A preliminary and more technical version of these results has been presented at a conference.

Now we discuss the architecture of this network:

### 3.1 Input Layer

We show the sensory stimuli at time $t$ with $x(t)$ and represent each input with three different sensory units as follows:

$$\begin{aligned} &x(t) \quad \text{(regular input units)} \\ x^+(t) &= ReLU\left(x(t) - x(t-1)\right) \quad \text{(transient + input units)} \\ x^-(t) &= ReLU\left(x(t-1) - x(t)\right) \quad \text{(transient - input units)} \end{aligned}$$

where $ReLU$ is the Rectified Linear Unit. This encoding helps us to interpret the amount of change in each stimuli than before.

### 3.2 Association Layer

This layer models the association cortex and contains regular units which are connected to the regular input units as well as some memory units. Memory units are fully connected to the transient input units and get activated using a sigmoidal activation function.

### 3.3 Q-value Layer

This layer receives input from the association layer through plastic connections. Its task is to compute action-values (i.e. Q-values) for every possible action. Specifically, a Q-value unit aims to represent the (discounted) expected reward for the remainder of a trial if the network selects an action a in the current state.

### 3.4 Action Layer

This final layer represents the actions and it is activating using the winner-gets-all method.

### 3.5 Learning

Learning in the network is controlled by two factors that gate plasticity: a global neuromodulatory signal and the attentional feedback signal. Once an action is selected, the unit that codes the winning action feeds back to earlier processing levels to create synaptic tags, also known as eligibility traces on the responsible synapses. Tagging of connections from the association layer to the motor layer follows a form of Hebbian plasticity: the tag strength depends on presynaptic activity and postsynaptic activity after action selection and tags thus only form at synapses onto the winning unit. For the exact mathematical formulation of the learning method please refer to the original AuGMEnT paper [6].

## 4 Experiments

In this section we describe our experiment settings and results.

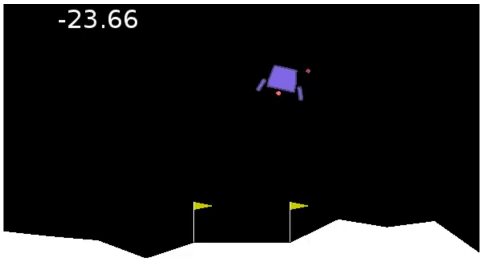

Figure 2: The environment of the Lunar Lander game.

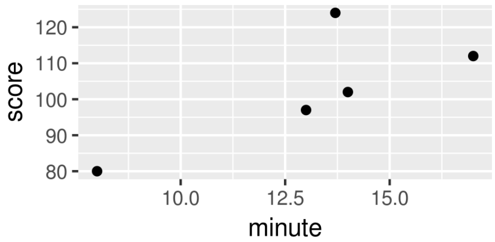

Figure 3: Best score of each human player (and its corresponding time) in the Lunar Lander game.

### 4.1 Baselines

In order to test our ideas we used the AuGMEnT system that has been introduced in section 3 as our bio-plausible reinforcement learning system. Also we used one non-bio-plausible implementation, *CMA* (Covariance Matrix Adaptation)[4] as our baseline. The CMA method uses a randomized black box optimization method to optimize the parameters of a neural network using the rewards that achieves for running each episode. The parameters of the networks of both architectures were chosen carefully so that we had the same number of parameters in each network to be learned.

### 4.2 Environment

The task that we have used for evaluating our reinforcement learning systems is the *Lunar Lander* game from the OpenAI gym platform[3] (see Figure2). The goal of this game is to land a lander on a segmented surface between two fixed flags. This lander has three engines consisting of a main engine and two side engines. The side engines can move you right or left by their force and the main engine can help you flight upward and this is the only engine that costs you. This means that you have four possible actions in each time-step: to move right, left, upward, or to do nothing which causes the gravity to move the lander downward.

### 4.3 Dataset

One of the crucial demands of our experiment was the actions of the real players who were learning how to play Lunar Lander. We asked five people to play for at least 20 minutes and they were only instructed the four possible actions of the game. They could see their total reward after each action in the same window that they were playing. After playing a few episodes, they could learn the physics of the lander movement and the rewarding strategy of the game. For example they understood that they can get more rewards by keeping the lander balanced and also by staying in the landing area. In addition, they understood that using the main engine reduces from their reward.

We recorded all the environment and the player responses so that we could simulate them again while we were training our neural networks. We collected more than 36000 frames from each player. Figure 3 shows the best score that each player achieved during his play.

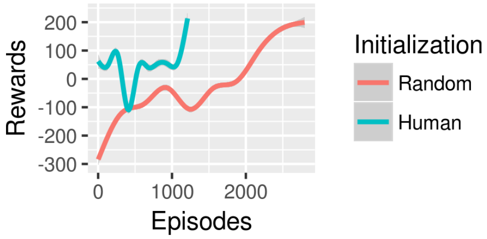

Figure 4: Comparison between the random initialization of the AuGMEnT network and initializing it with human actions.

### 4.4 Results

Now we are ready to study the performance of our bio-plausible agent under different assumptions.

#### 4.4.1 Policy Learning

In this experiment we want to show that our AuGMEnT agent can learn the general policies of the environment just by using the human data. One of the main policies was not to land with high speed. To test this ability, we changed the gravity force of the environment and used the human data of only one person for training to see whether the agent decides to use its main engine or not. We found out that our agent uses its main engine to reduce its speed when it approaches the ground. However, it collided with ground with a high speed as this task was not what it was trained for. We should mention that the CMA method could not solve this task.

In another experiment, we changed the forces of the side engines to test the balance ability of our agent. Our agent responded quickly to this change and used its side engines more carefully to keep itself balanced.

One of the tasks that our agent could not handle was the landing area experiment. The aim of this task was to test the agents' consciousness of the environment. The agent was not sensitive to the landing area and it lost the possible rewards that it could get by changing its position toward the landing area.

#### 4.4.2 Solving the Game

Following the OpenAI gym platform, we consider Lunar Lander as solved when the agent obtains an average reward of at least 200 over 100 consecutive episodes. Now we want to compare the solving time of the AuGMEnT when it uses its random initialization and when it uses human data. It worth mentioning that none of our human observers could achieve this high standard of 200 score as it needs high frequency responses when the lander approaches the ground, however, the our computer agents can satisfy this need after a while of playing.

Figure 4 illustrates the difference between the random initialization of the neural network and human initialization of it. The human initialization accuracy is high in its 200 starting episodes because we have simulated the exact results of a single human player evaluations. Although the human initialization method traps in a local minima for a while, it solves the problem faster. It should be mentioned that this result was not consistent with initializing by other human players since they could not solve as fast as this player. Also, this player was the one who ranked fourth among all the players. It might be justified by the fact that he was a good observer of the environment (please see Figure 3 again to see that he has achieved his high score in the last episodes of his playing).

The performance of the CMA method can be seen in Figure 5 which shows that it needs 58344 episodes to solve the game. This figure is the smoothed version of the real performance which means it hides the fluctuations that happen even in the final episodes (as an example, we had some records around -100 even in the final 1000 episodes). One thing that both CMA and AuGMEnT share when they start with random initialization is the negative initial rewards. This is pretty normal as they should observe the environment in order to learn how they can maximize their reward. Even though our agent needs to observe when it uses the human data, we can see that the negative rewards are

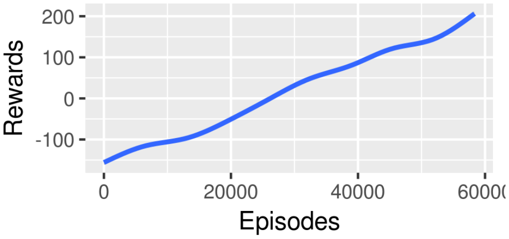

Figure 5: CMA solving progress.

much less than what we faced in random initialization which is justifiable by the performance of the human learner.

## 5 Conclusion

In this paper we study the learning strategies of a biologically-plausible agent and show that we can get acceptable results such as learning the general policies of the environment and faster solving of the games by initializing our agents with human actual actions in similar situations. Although these studies show a defensible method for training our bio-plausible networks, they cannot be used to rule out the possibility that these methods can be beneficial for non-bio-plausible methods as we have only tested them against one method (CMA). On the same side, we should test these studies with more bio-plausible agents in various tasks with different demands which seems like a promising direction for future studies.

## References

[1] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 2012.

[2] Y. Bengio, D.-H. Lee, J. Bornschein, T. Mesnard, and Z. Lin. Towards biologically plausible deep learning. *arXiv preprint arXiv:1502.04156*, 2015.

[3] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

[4] N. Hansen. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[6] J. O. Rombouts, S. M. Bohte, and P. R. Roelfsema. How attention can create synaptic tags for the learning of working memories in sequential tasks. *PLoS Comput Biol*, 11(3):e1004060, 2015.

[7] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[8] P. A. Tsividis, T. Pouncy, J. L. Xu, J. B. Tenenbaum, and S. J. Gershman. Human learning in atari. 2017.

[9] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI*, pages 2094–2100, 2016.