Knowledge Transfer in Deep Reinforcement Learning

Ieshan Vaidya (iav225@nyu.edu)
Jatin Khilnani (jk6373@nyu.edu)
Sarthak Agarwal (sa5154@nyu.edu)
Center for Data Science, New York University, NY

Abstract

Humans have always showed the capacity of generalization in the real world. Applying prior knowledge, understanding and experience to new situations comes naturally to us. These characteristics display a higher level of intelligence that is desirable in artificial agents. Reinforcement learning has achieved incredible progress in building intelligent agents, particularly to play games on a super-human level. The Deep Q-learning algorithm gained wide attention by learning to play a variety of Atari games using raw image input. However, such approaches require extensive training on a single task which may not be feasible in many real-world cases. Oftentimes, we don't have a simulator that can generate arbitrarily many experiences to learn from. Moreover, they require retraining the agent for every new task. Trying to learn with minimal experience is thus valuable to generalize quicker. In this work, we address this problem by exploring transfer learning techniques with a test-bed of Atari games. We evaluate whether an agent trained to play one game has learned abstract knowledge about game properties that will help generalize faster to other games. [1]

Keywords: Transfer Learning; Reinforcement Learning; DQN; CNN

Introduction

Reinforcement learning provides a computational framework to build a reward based cognitive model. An agent interacts with the environment to gain a scalar reward which allows it to improve itself. The reward can be delayed and actions taken at one time step can have ramifications much later in the future. Reinforcement learning thus provides an efficient approach to solve such temporal planning problems. This is quite similar to some aspects of decision making seen in humans. Humans are able to model high-dimensional environments with sensory input to solve complex problems and reinforcement learning provides a framework to model this behavior.

Early successes in reinforcement learning were naturally seen on low-dimensional tractable problems. With the advances in computation along with breakthroughs in deep learning techniques, problems with higher complexity have become accessible. Although the state of the art of reinforcement learning is still evolving, techniques such as the Deep Q-Network (DQN) (Mnih et al., 2015) have demonstrated its power in learning how to play video games on a super-human level. The world's best Go player (Silver et al., 2017) was created based solely on reinforcement learning and without human data, guidance, or domain knowledge beyond game rules.

Despite these advances, these agents are often computationally expensive and lack generalization ability. To illustrate, DQN required human equivalent training time of 38 days to beat a professional human tester who was allowed to practice the game for 2 hours (Sutton & Barto, 2018). Studies have been done to understand human priors while playing games (Dubey, Agrawal, Pathak, Griffiths, & Efros, 2018) that suggest that agents lack such priors. These issues raise the need to explore avenues which lead to efficient agents.

Humans are adept at using knowledge gained in one domain and reusing it to solve problems in another. In most learning environments, humans do not draw inferences and form hypotheses from scratch but rather rely on past experiences. This ability of transferring knowledge is a key aspect of human cognition that enables us to learn new skills rapidly and adapt to new situations with minimal data. Signs of transfer learning has been seen in humans even at an early age of 3 (Brown & Kane, 1988). Transfer learning has also been an active area of research in machine learning as it enables models to transfer knowledge learned in one domain to another quickly.

Transfer learning in machine learning assumes that the model is able to learn some characteristics of one task which are transferable to another. We focus our work to explore if the current reinforcement learning algorithms benefit from such knowledge transfer. Concretely, we use the DQN algorithm on Atari games to evaluate whether an agent can learn to play a game faster if provided with knowledge of a different game, along with analysis of conditions in which this may happen. We also evaluate whether the type of game plays a role in determining knowledge transfer. This experiment is akin to cognitive science study by Bavelier, Bediou, and Green (2018) on action video games as a tool to enhance generalization in humans.

Related Work

Application of transfer learning to reinforcement learning has been explored extensively in the past decade. The main insight behind transfer learning comes from the psychological literature (Woodworth & Thorndike, 1901; Skinner, 1953). It is based on the fact that generalization occurs not only within

[1] Code available at: https://github.com/ieshanvaidya/DQN-Atari-Transfer-Learning

tasks, but also across tasks. Consequently, transfer learning in the reinforcement learning domain has seen a lot of success stories (Da Silva & Costa, 2019; Gamrian & Goldberg, 2018).

Taylor and Stone (2009) present a framework to classify transfer learning methods in terms of their capabilities and goals whilst suggesting future directions. Generalized policy improvement (GPI) (Barreto, Munos, Schaul, & Silver, 2016) is another framework that seamlessly integrates transfer learning into the reinforcement learning domain. It is a generalized approach to perform classic dynamic programming operation with successor features (SFs), an scheme that makes evaluation faster and easier across multiple tasks. Moreover, a method to learn policies specialized to new tasks is also presented.

There have been approaches to transfer knowledge learned from multiple environments to new situations (Parisotto, Ba, & Salakhutdinov, 2016). Using task-specific expert agents as guides, a single multi-task Deep Q-Network is trained to obtain a pre-trained agent. By altering the final softmax layer, this pre-trained agent is shown to learn a new task significantly faster than random initialization thus demonstrating knowledge transfer. Our approach is along these lines although we focus more on the capability of an agent to capture generic features of the environment it experiences rather than trying looking at multiple tasks at the same time.

Apart from transfer learning, the field of meta learning is also seeing prominence. The objective of meta learning is to train a model on a variety of tasks in order that the model can learn a new task rapidly. Approaches like meta-agnostic machine learning (Finn, Abbeel, & Levine, 2017) and learning to learn (Wang et al., 2016) (again, inspired from cognitive science literature (Harlow, 1949; Kemp, Goodman, & Tenenbaum, 2010)) have been shown accelerate fine-tuning of reinforcement learning policies.

Methodology

The idea of transfer learning in reinforcement learning domain is that the agent captures some aspect of decision making during the pre-training phase that is useful for the fine-tuning task. This has a plethora of uses, the main one being that the agent can quickly generalize to the fine-tuning task, demonstrating 'positive' transfer and thus saving training time. Additionally, this validates the ability of an agent to learn generalizable properties and aligns with the observation by Woodworth and Thorndike (1901) - "It might be that improvement in one function might fail to give in another improved ability, but succeed in giving ability to improve faster" (p. 248). This naturally depends on the nature of the two tasks and whether there are any shared properties that the agent can pick up. To identify if this indeed plays a role in learning, we use two fine-tuning tasks, one which is similar to the pre-training task, and one which is not. We expect that the agent can learn a similar task quicker but has to relearn for a completely different task. Transfer learning broadly consists of two steps

1. Pre-train the agent on one task and
2. Transfer knowledge by fine-tuning on a different task.

Environment

The Atari-2600 platform consists of 49 games in various categories like shooter, exploratory, racing among others that are typically challenging for humans. Given this diversity in tasks, it serves as an ideal test-bed to evaluate our strategy. We choose `Demon Attack`, a shooter game as the task to pre-train on. For fine-tuning, we consider two games: (a) `Assault`, a shooter game and (b) `RoadRunner`, a racing game.

Reinforcement Learning

Algorithm 1: deep Q-learning with experience replay

Initialize replay memory D to capacity N
Initialize action-value function Q with random weights θ
Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
for episode $= 1, M$ **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  for $t = 1, T$ **do**
    With probability ε select a random action a_t otherwise select $a_t = \arg\max_a Q(\phi(s_t), a; \theta)$
    Execute action a_t and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
    Set $y_j =$
    $$\begin{cases} r_j & \text{episode ends at } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ with respect to network parameters θ
    Every C steps reset $\hat{Q} = Q$
  end
end

We use the Deep Q-Network (DQN) algorithm that is able to achieve super-human performance on several Atari games. It builds upon the standard Q-learning algorithm (Watkins, 1989; Sutton & Barto, 2018) by introducing a neural network as the Q-function approximator. Q-learning is an off-policy TD learning algorithm defined by

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Standard Q-learning is limited to discrete state-action environments. An improvement on it is to use Q-function approximators that output Q-values for a given state-action input. Non-linear function approximators have been known to be unstable when used as a representation for the Q-function (Tsitsiklis & Roy, 1996) and neural networks suffer from it as well. One cause of this instability is the correlations present in sequential observations. Experience replay and target network are two novel ideas introduced to overcome this. The replay keeps a history of observed (state, action, reward, next-state) tuple in memory. During training, a batch of examples are randomly sampled from this memory, thus ensuring that adjacent examples are not temporally correlated. The target network is a copy of the Q-function estimator that is updated every few steps. It's used to generate the targets in the Q-learning update and improves the stability of the method.

DQN uses a convolutional neural network that takes in an input state (pixels of image) and outputs Q-values corresponding to all possible actions. It takes in the current state as input and gives out the Q-values for all different possible action in that state. This has the advantage of requiring a single forward pass to compute all possible Q-values. The network is trained by calculating the loss between $r_{t+1} + \gamma \max_a Q(s_{t+1}, a)$ and $Q(s_t, a_t)$.

Figure 1 shows the raw pixel state of Atari games. To reduce computational requirement and to make the agent more realistic, some pre-processing is done, which is summarized below

1. Single frame is encoded by taking the pixel-wise maximum of the frame being encoded and previous frame. This prevents flickering issues seen in games.

2. Color frame is rescaled to 84×84 grayscale frame.

3. m such rescaled frames are stacked together to form the input state. We use $m = 4$ in all the experiments.

4. Frame-skipping is used wherein the agent sees and acts on every k^{th} frame and actions repeated on skipped frames. We use $k = 4$ in all the experiments.

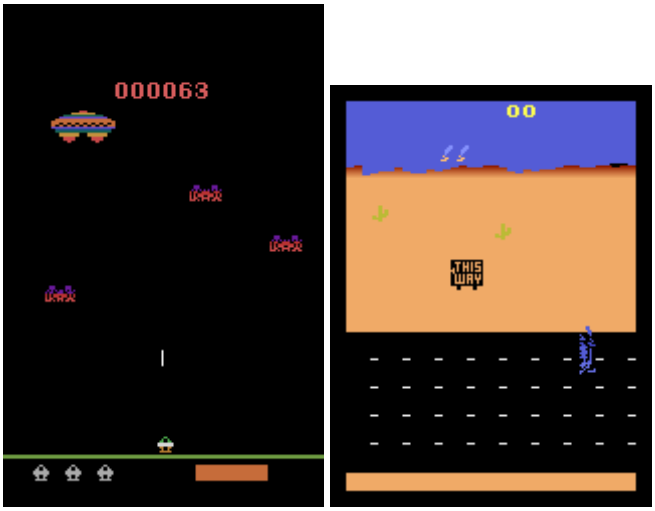

Figure 1: Raw image input of (a) `Assault` and (b) `RoadRunner`

Figure 2 shows a sample preprocessed input state from `DemonAttack` that is fed into the network.

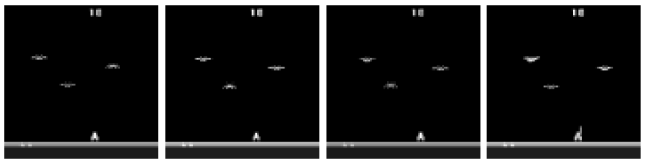

Figure 2: Stack of processed frames that form the input state

Algorithm 1 outlines the training steps. There are a few other additional implementation details of which the relevant ones are discussed in subsequent sections.

Transfer Learning

The objective of transfer learning is to reduce the training time on new tasks by pre-training on a base task. The metric we use to evaluate training time is the number of network updates / gradient descent steps. During training, the rewards are clipped to $[-1, 1]$ and thus agent performance is measured using these clipped rewards rather than raw score. We assume training is completed when the episode rewards saturate. Due to stochastic nature of episodes, we track mean rewards over 100 episodes rather than reward over a single episode. To summarize, we measure mean rewards against network updates to evaluate the efficacy of different training schemes.

As is common for human beings and in reinforcement learning, there is a balance between exploration and exploitation (Daw, O'Doherty, Dayan, Seymour, & Dolan, 2006). Exploration is incorporated in the agent using an ε-greedy strategy with ε (probability of taking a random action) being linearly annealed to a small value. During training, the replay memory is initialized with 50,000 random experiences.

We keep this constant throughout all experiments. Thus, there is always some exploration no matter the ε strategy. In pre-training phase, we linearly anneal ε from 1 to 0.1 over 1,000,000 network updates. During fine-tuning, it is not straight-forward what the standard technique is. We thus follow 3 schemes for fine-tuning:

1. Full exploration: Linearly anneal ε over 1,000,000 network updates

2. Small exploration: Linearly anneal ε over 100,000 network updates

3. No exploration: Fix $\varepsilon = 0.1$

Similarly, to evaluate feature learning capability of the estimator, we experiment with different transfer learning schemes. In convolutional networks, the initial layers capture crude features of pixels and the final layers capture task specific information (Zeiler & Fergus, 2014). This suggests that the initial layers should show generalization properties and strengthens the proposal that development of automatization as expertise sets in may affect generalization in humans (Bavelier et al., 2018). We thus consider the following schemes for our DQN:

1. Freeze first $n \in \{0, 1, 2\}$ layers.

2. Differential learning rate with initial layers having a smaller rate than the final. We set the learning rate for the first 3 layers to be 0.000025 and for the final layers to be 0.0001.

These strategies are evaluated over `Assault` to determine the optimal one. To wrap up, we evaluate the optimal strategy on `RoadRunner` to determine if the knowledge learned by the agent is not as effective or even detrimental ('negative' transfer) to a different game.

Results and Discussions

While we use most of the default settings of the DQN algorithm, we experiment with different optimizers as well as batch normalization (Ioffe & Szegedy, 2015). Adding batch normalization to the hidden layers in convolution networks has been demonstrated to improve the performance of the model. Given the large computational cost of training these agents, we evaluate whether we observe similar benefits for DQN. We evaluate two optimizers - Adam and RMSprop. For reference, vanilla DQN uses RMSprop without batch normalization. Figure 3 shows the performance of agents trained on `DemonAttack` with these settings.

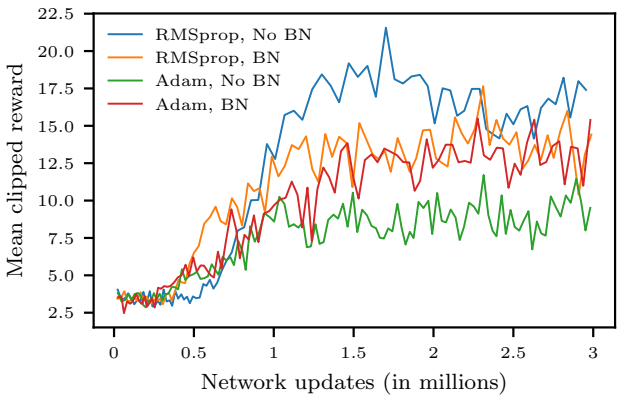

Figure 3: Performance on `DemonAttack` with different parameter settings

As described before, we evaluate the mean clipped reward over the last 100 episodes. We observe that RMSprop without batch normalization is consistently better. The fact that batch normalization did not improve performance of the agent is a bit surprising but can be attributed to the stochastic nature of sampling from the replay memory.

We use the agent trained on `DemonAttack` with RMSprop and no batch normalization as the initialization of fine-tuning. To transfer knowledge of this agent, we initialize weights of the Q-function estimator with learned weights and modify the final layer of the estimator to now project onto the action space of `Assault`.

As mentioned in the methodology, we consider 3 exploration strategies along with 4 fine-tuning strategies. To keep the results concise, we discuss the exploration strategies for a single fine-tuning one. Figure 4 shows the dependence of initial exploration for freeze-0 strategy.

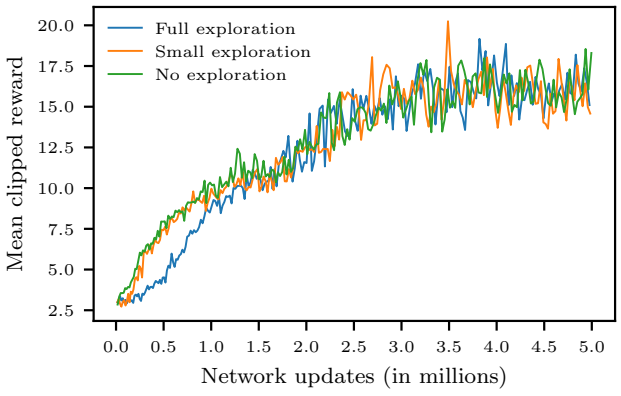

Figure 4: Comparison of exploration strategies with freeze-0 setting on `Assault`

We observe that the reward increases faster when ε is ini-

tialized with a low value as compared to linear annealing. But eventually, the agent is able to achieve similar rewards in all the three cases. It should be noted that for full exploration, the rewards are dampened by the fact that the agent is taking more random actions. The strategies are comparable beyond 1 million network updates. Nevertheless, not exploring during fine-tuning is not detrimental to the success of the agent. We observe this behavior for all the other fine-tuning strategies.

To evaluate different fine-tuning strategies, we set the agent in no exploration mode ($\varepsilon = 0.1$). Figure 5 shows the performance of the agent trained from scratch with different transfer learning strategies.

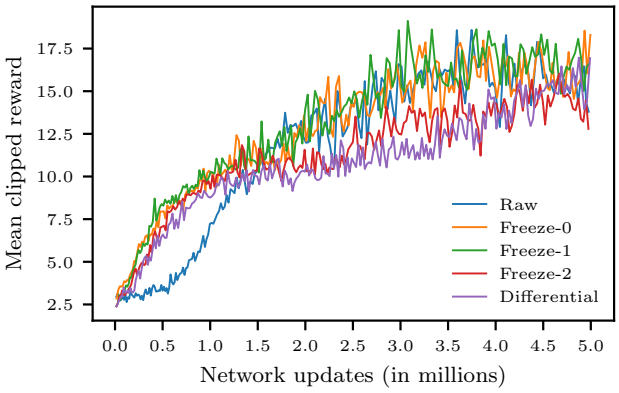

Figure 5: Comparison of fine-tuning strategies with fixed ε on `Assault`

While one would expect that pre-training on a similar task would imply that the agent will reach competitive performance faster, the results do not show this. All the agents saturate on similar rewards while requiring similar number of network updates. As noted before, the poor initial performance of the agent trained from scratch is due to the damping effect of random actions. There is slight difference in the initial performance of the other agents caused due to limiting the model capacity. The agent trained with two layers frozen has generally poor performance comparatively which suggests that the initial features learned for one game are not directly transferable to other games.

We evaluate the freeze-0 strategy on `RoadRunner` which has different characteristics than `DemonHunter`. Figure 6 shows the performance of the fine-tuned agent compared to an agent trained from scratch.

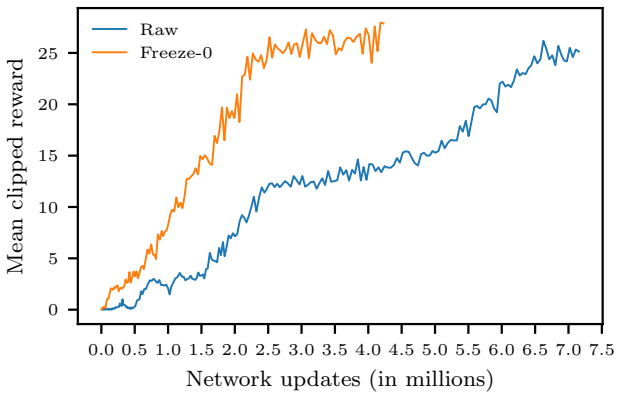

Figure 6: Comparison between raw and fine-tuning strategies on `RoadRunner`

The performance is completely opposite to our assumption in that the agent supposedly does learn game agnostic characteristics. Despite being a racing game, the fine-tuned agent reaches competitive performance significantly faster than the raw one. This is equivalent to attaining enhanced attentional control and cognitive flexibility in humans through training that fosters 'far' generalization (Bavelier et al., 2018).

Conclusions

Generalization from experiences is quite common for humans and an agent that displays such qualities is desirable. Although transfer learning on paper promises such behavior, we observe that it is not so straight-forward to translate that to reinforcement learning. A possible reason as to why the performance is poor is that we pre-train a model on a specific environment and that limits its capacity to generalize.

The results also suggest that there are complex underlying characteristics of these games that determine 'near' transfer learning performance. It is important to note that the evaluation was performed over three games and thus it is difficult to glean generalization capability of transfer learning when it comes to Atari games. It does confirm that it is not simple as the games being shooter or racing that determine if it is easy to transfer knowledge and attributes to the lack of theoretical framework to systematically identify the relevant level of overlap between any two tasks for predicting generalization (Bavelier et al., 2018). It also raises the question of whether an agent should actually show these generalization capabilities as it does in the racing game.

We believe that training on multiple tasks simultaneously (Parisotto et al., 2016) would lead to the expected results from transfer learning as it would then mimic how humans learn in a multi-environment setting and overcome this problem. We also deem it beneficial to exploit the computational benefits of a simulated cognitive environment, by analyzing the representations learned by the agent to determine similar processing components between two environments and

gauge potential generalization. Nevertheless, Human intuition suggest that someone who is trained to play shooting games should find it easy to play other shooting games. Thus, enforcing such generalization would be an interesting avenue to explore.

Acknowledgments

We would like to thank the instructors and the teaching staff for their guidance and support. We are grateful to Professor Todd Gureckis for his feedback and discussion about the project.

References

Barreto, A., Munos, R., Schaul, T., & Silver, D. (2016). Successor features for transfer in reinforcement learning. *CoRR*, *abs/1606.05312*. Retrieved from http://arxiv.org/abs/1606.05312

Bavelier, D., Bediou, B., & Green, C. S. (2018). Expertise and generalization: lessons from action video games. current opinion in behavioral sciences. *Current Opinion in Behavioral Sciences*, *20*, 169-173. Retrieved from https://doi:10.1016/j.cobeha.2018.01.012

Brown, A., & Kane, M. (1988, October). Preschool children can learn to transfer: Learning to learn and learning from example. *Cognitive Psychology*, *20*(4), 493–523. doi: 10.1016/0010-0285(88)90014-X

Da Silva, F. L., & Costa, A. H. R. (2019). A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research*, *64*, 645–703.

Daw, N., O'Doherty, J., Dayan, P., Seymour, B., & Dolan, R. (2006, 07). Cortical substrates for exploratory decision in humans. *Nature*, *441*, 876-9. doi: 10.1038/nature04766

Dubey, R., Agrawal, P., Pathak, D., Griffiths, T. L., & Efros, A. A. (2018). Investigating human priors for playing video games. In *Icml.*

Finn, C., Abbeel, P., & Levine, S. (2017). *Model-agnostic meta-learning for fast adaptation of deep networks.*

Gamrian, S., & Goldberg, Y. (2018). *Transfer learning for related reinforcement learning tasks via image-to-image translation.*

Harlow, H. F. (1949). The formation of learning sets. *Psychological Review*, *56(1)*, 51–65. Retrieved from https://doi.org/10.1037/h0062474

Ioffe, S., & Szegedy, C. (2015). *Batch normalization: Accelerating deep network training by reducing internal covariate shift.*

Kemp, C., Goodman, N. D., & Tenenbaum, J. B. (2010). Learning to learn causal models. *Cognitive Science*, *34*(7), 1185-1243. Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1551-6709.2010.01128.x doi: 10.1111/j.1551-6709.2010.01128.x

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015, February). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. Retrieved from http://dx.doi.org/10.1038/nature14236

Parisotto, E., Ba, J., & Salakhutdinov, R. (2016, 11). Actor-mimic: Deep multitask and transfer reinforcement learning..

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... Hassabis, D. (2017, Oct 01). Mastering the game of go without human knowledge. *Nature*, *550*(7676), 354-359. Retrieved from https://doi.org/10.1038/nature24270 doi: 10.1038/nature24270

Skinner, B. (1953). Science and human behavior: Simon and schuster.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Taylor, M. E., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, *10*(1), 1633-1685. Retrieved from http://www.cs.utexas.edu/users/ai-lab?taylor:jmlr09

Tsitsiklis, J. N., & Roy, B. V. (1996). Analysis of temporal-diffference learning with function approximation. In *Nips.*

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., ... Botvinick, M. (2016). *Learning to reinforcement learn.*

Watkins, C. (1989, 01). Learning from delayed rewards.

Woodworth, R. S., & Thorndike, E. (1901). The influence of improvement in one mental function upon the efficiency of other functions.(i). *Psychological review*, *8*(3), 247.

Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science*, 818–833. Retrieved from http://dx.doi.org/10.1007/978-3-319-10590-1_53 doi: 10.1007/978-3-319-10590-1_53