General Intelligence in Game Playing

Ben Dahlgren, Sanjana Gundala, Gabe Hyun, Sean Nesbit
Cal Poly,
San Luis Obispo, United States
{bdahlgre,gundala,gohyun,sfnesbit}@calpoly.edu

***Abstract*—Excelling at video games often require a great amount of knowledge and experience. To develop an agent to come near the playing ability of a human, developers need to ensure that this skill is imparted to the autonomous agent. With the help of game experts, agents have superseded human ability in game playing and surpassed expectations of artificial intelligence around the world. The hyper-specialization required to develop these agents is not scalable, however, as they have required the careful tuning to play specific games. We hope to present an agent that can develop super-human game playing ability without intervention and rapid performance development. Using only the screen as input and fixed controls as output, our agent will be able to quickly train to play a number of games.**

I. INTRODUCTION

Deep learning methods have seen great success in computer vision and problem solving fields of study over the past few years. Many fail, however, in their ability to transfer predictive power outside of a strictly defined environment. The problem of creating an autonomous agent that performs well regardless of its environment is known as general intelligence. With varied tasks and environments and a relatively simple control scheme, video games are a perfect simulation environment to conduct our research. We will use OpenAI's Gym platform[1] to facilitate our simulations. We hope to build on existing research by presenting a system that can quickly learn and adapt to a new game with transfer learning and reinforcement learning. Though we hope to further develop the field of machine learning, this can benefit games by offering new developments to intelligent NPCs or this research could be used to develop autonomous game testers for game developers.

Reinforcement learning presents different problems from other Deep Learning approaches. Whereas many other Deep Learning approaches rely on large amounts of hand labeled data, reinforcement learning is based on reward functions that can be sparse and/or delayed. In this paper we use Deep Q-Learning and Transfer Learning to address these problems when training agents to play Atari games. One agent is initially trained on a single game using Deep Q-Learning. Then we take this first model and apply it to another game using Transfer learning, hopefully leading to a decrease in training time for newer agents.

Our results showed that through Deep Q-Learning, agents were able to achieve game play results that were much better than random. We believe that with more time to train the agent and more preferment training hardware our agents would be able to achieve even better results. However, we found that Transfer Learning did not provide much of an advantage over training each model individually from scratch. The model was not able to overcome the vast differences in the various games to be able to much a difference in training efficiency. Our code can be found on GitHub at https://github.com/dahlgreb/570FinalProject.

II. SPECIFICATION

To play each game, the key inputs and outputs are the game screen and game actions respectively. The game screen is a 210 by 160 RGB image that shows the game. The game actions are the total set of possible actions on the controller. This includes 18 actions such as right, left, noop, and fire. The model reads game screen data as downsampled tensor data and predicts an action for the agent to take. The Deep Q Network also fits in by reinforcement learning the optimal action for the agent to take through many game trials. This can then be used to train the convolution network to predict the optimal game action for a screen input.

The three games we chose to apply our algorithms to were Breakout, Pong, and Space Invaders. Since these three are all Atari games, they are played using the same controller, so they all have the same action space: no-op, up, down, left, right, and fire, however not every action can be used in every game.

Breakout is a single-player game consisting of a multiple layers of bricks near the top of the screen. The goal of the same is to break all of the bricks by bouncing a ball off of a paddle and into the bricks. The possible actions available in Breakout are: no-op, left, and right.

Pong is a game that mimics table tennis, consisting of two paddles controlled by opposing players. The goal of the game is to get the ball passed the opposing player's paddle. Pong can be single-player when one human player competes against the computer, or it can be two-player when two different human players control each of the two paddles. The possible actions available in Pong are: no-op, up, and down.

Space Invaders is a single-player game in which a player-controlled laser cannon on the bottom of the screen destroys aliens on the top of the screen. The goal of the game is to eliminate all of the aliens. The possible actions available in Space Invaders are: no-op, left, right, and fire.

We initially trained our model on breakout, and then tried to transfer the learned behavior to pong and space invaders.

III. RELATED WORK

A. OpenAI Gym

Reinforcement learning is a very promising branch of machine learning that focuses around sequences of decisions that are chosen based on observations and rewards from an environment. This means that having easily accessible environments is a very necessary part of reinforcement learning research. OpenAI Gym was created to address this need for standardized environments for reinforcement learning. OpenAI Gym is a collection of tools that was created to make reinforcement learning research easier and more accessible. It helps researchers develop and compare different reinforcement learning algorithms by presenting many different benchmark problems that can all be accessed through a single interface. These different problems are separated into 'tasks' which are various environments for researchers to test and compare their algorithms. OpenAI Gym allows researchers to train agents in a variety of different benchmark environments from walking, to Atari games, to pinball. OpenAI Gym also fosters collaboration and community between reinforcement learning researchers by maintaining a website where researchers can upload the results of their algorithms in different environments, and these results can be seen in global leader boards. Researchers are also encouraged to post their results along with the source code that achieved these results so that other researchers can make use of and build off of their findings[1].

B. TD-Gammon, A Self-Teaching Backgammon Program, Achieves Master-Level Play

TD-Gammon is a neural network created by Gerald Tesauro which is able to teach itself to play backgammon only by playing against itself and learning from results. This paper presents a case study in which the TD Reinforcement learning algorithm developed by Sutton in 1988 was applied to training a multilayer neural network on a complex task. TD-Gammon achieves a surprisingly strong level of play despite starting from random initial weights. With no knowledge built in at the start of learning, the network learns to play at a strong intermediate level. However, when manually extracted features are added to the representation input for the neural network, the level of performance increases. The latest version of TD-Gammon, is estimated to play at a level that is close to the world's best human players [3].

C. Deep Reinforcement Learning with Double Q-Learning

Q-learning is one of the most popular reinforcement learning algorithms, but sometimes it is known to learn unrealistically high action values. This is because the algorithm includes a maximization step over estimated action values which tends to prefer overestimated to underestimated values. To test whether over estimations occur in practice and at scale, the authors of this research investigated the performance of the DQN algorithm and Double Q-learning algorithm. DQN combines Q-learning with a flexible deep neural network and was tested on a varied and large set of deterministic Atari 2600 games, reaching human-level performance on many games. A deep Q network (DQN) is a multi-layered neural network that for a given state s outputs a vector of action values Q(s,·;), where are the parameters of the network. Double Q-learning reduces over estimations by decomposing the max operation in the target into action selection and action evaluation. The results of this research show that Double DQN improves over DQN both in terms of value accuracy and policy quality. Overall this paper shows why Q-learning can be overoptimistic in large-scale problems, that these over estimations are more common and severe in practice than previously acknowledged, that Double Q-learning can be used at scale to successfully reduce this overoptimism, and that Double DQN finds better policies, obtaining new state-of-the-art results on the Atari 2600 domain. This research also proposes the use of Double DQN, which uses the existing architecture and deep neural network of the DQN algorithm without requiring additional networks or parameters[6].

D. Game Playing with Deep Q-Learning OpenAIGym

In this research authors Robert Chuchro and Deepak Gupta, explore the use of general AI techniques such as reinforcement learning and neural networks on a variety of OpenAI Gym environments to create a model which can be trained on more than one game. The input into the model was a sequence of pixel images as arrays generated by a particular OpenAI Gym environment. Then a Deep Q-network was used to output an action from the action space of the game. The neural network consisted of 3 convolution layers with ReLU non-linearity activation functions and followed by 2 fully connected layers. The final output layer dimension is equal to the number of valid actions allowed in the game. Ultimately the authors were able to train a model using Depp Q-network to learn Flappy Bird with superhuman performance and the model was able to transfer over to another game, Pixel Copter [2].

E. A Survey of Deep Reinforcement Learning in Video Games

Generally, Deep Reinforcement Learning (DRL) agents receive high-dimensional inputs at each step and make actions according to deep neural network based policies. This research surveys the process of and compares DRL methods, DRL combines Deep Learning (DL) and Reinforcement Learning (RL). There are various DRL methods including value-based, policy gradient, and model-based algorithms.The most famous DRL value-based model which learns policies directly from high-dimensional inputs is the Deep Q-Network. Policy gradient DRL optimizes the parameterized policy directly, there are several options for implementing this including the trust region method, deterministic policy, and entropy-regularized policy gradient. As for model-based DRL methods, TreeQN was used, it is a differentiable, recursive, tree-structured model. Each of these DRL agents achieved human-level or superhuman performances in various games. However, there are still some issues when applying DRL methods to this field, especially in 3D imperfect information multi-agent video games [4].

F. Playing Atari with Deep Reinforcement Learning

This paper explains some of the different problems with reinforcement learning as opposed to other deep learning methods. These problems include relying on reward functions that can be sparse and/or delayed, highly correlated inputs, and changing data distributions as a model learns new behaviors. To address these problems the authors implement a convolutional neural network that uses Q-learning with the goal of making an agent that can play Atari games. The agent takes in only the pixels of the game play, the reward, and the terminal game state. The authors use the same agent architecture to train agents for six different Atari games, showing that their approach can be generalized to multiple different games. Their results showed that the architecture produced agents that achieved superhuman ability in three of the games, and average human ability on the other three games.

IV. Implementation

We intend to apply our game playing agent to multiple games provided by the OpenAI Gym platform. From our research, a neural network that attempts computer vision and reinforcement learning has proved successful in this problem. We intend to use Tensorflow to build and train a neural network that incorporates convolutional layers to interpret frames of the game and predict an optimal game input. When trained across multiple games, we hope to create a powerful deep reinforcement learning agent that can play with superhuman ability. We hope to follow proposed methods in[5] for our base model and further develop methods from there.

To approach general intelligence, we will attempt transfer learning to tailor the neural network to other games. We assume that this approach will allow the model to quickly pick up new games rather than relearning from scratch.

A. Deep Q-Learning

The Deep Q-Learning method implemented here follows the proposed method in Mnih et. al.[5]. Screen captures are input to the model and the game action space is the predicted output. A Deep Q-Learning model is used to train a convolutional network over a sample batch from the experience replays. Frames are saved as experiences with which loss is calculated and weights are updated. The loss function used is mean squared error. Where many traditional Q-learning reinforcement models calculate loss with respect to a Bellman equation derived optimal action space, the Deep Q-Learning approach learns to optimize the action space with a neural network. This ”optimal” action space is then used to supervise train the convolutional network. Much of the core Deep Q-Learning code was adapted from Jodi Torres' code for Deep Q-Learning with the OpenAi Gym framework[7]. The architecture of our DQN model can be seen in its entirety in Figure 1.

```
DQN(
  (conv): Sequential(
    (0): Conv2d(4, 32, kernel_size=(8, 8), stride=(4, 4))
    (1): ReLU()
    (2): Conv2d(32, 64, kernel_size=(4, 4), stride=(2,
2))
    (3): ReLU()
    (4): Conv2d(64, 64, kernel_size=(3, 3), stride=(1,
1))
    (5): ReLU()
  )
  (fc): Sequential(
    (0): Linear(in_features=3136, out_features=512,
bias=True)
    (1): ReLU()
    (2): Linear(in_features=512, out_features=18,
bias=True)
  )
)
```

Fig. 1. The DQN Model Architecture

V. Presentation of Results

A. Transfer Learning

We hypothesized that once resources are invested to develop a Deep Q agent, the learned weights can be adapted to quickly learn to play a new game. If there are any similarities in game experiences, the trained model can be reused without unnecessary retraining. Transfer learning was implemented with several attempts to initialize and locking weights from a saved model from one game and applied to another. We invested training resources to develop one robust ”Breakout” model to attempt to transfer to both ”Pong” and ”Space Invaders.” Because the games are all built for the Atari console, they have a common action space. This makes models fully interoperable with different games. We tried retraining models with the ”Breakout” model as initial weights as well as for initializing the convolutional weights and locking to train just the linear output layers.

VI. Evaluation Methods

To determine the results of our general purpose AI, we analyze the reward generated by the model while training. Each OpenAI Gym game can return a reward at every game step. The reward is negative if the action is not beneficial or positive if the action is beneficial. This allows us to determine the real-time reward of our agent as its playing a match, and how it performs across the hundreds of thousands of epochs it is trained over.

The reward for each game is then compared to previous research attempts to measure whether our Deep Q-Learning with Transfer Learning approach outperforms the previously proposed models.

To determine whether the transfer learning approach produced stronger results than traditionally training a new network, we test the transfer learning method against a newly trained model with the same architecture. This will give us

insight as to whether there is inherent patterns and information each Atari game is based on that produces strong and consistent rewards with less training required.

	Breakout	Pong	Space Invaders
Random	1.2	-20.4	179
Sarsa	5.2	-19	271
Contingency	6	-17	268
DeepMind DQN	168	20	581
Human	31	-3	3690
Our DQN	11.7	8	500

TABLE I
OUR BEST MODELS WERE END-TO-END TRAINED FROM SCRATCH OVER MANY HOURS. RESULTS DO NOT SURPASS THOSE OF GOOGLE DEEPMIND BUT ARE MUCH BETTER THAN RANDOM.

A. Deep Q-Learning

We first trained a model for each game for a few hours each. Training time was severely limited by computation resource availability and access. These models are expected to represent the full potential of our model architecture on each game.

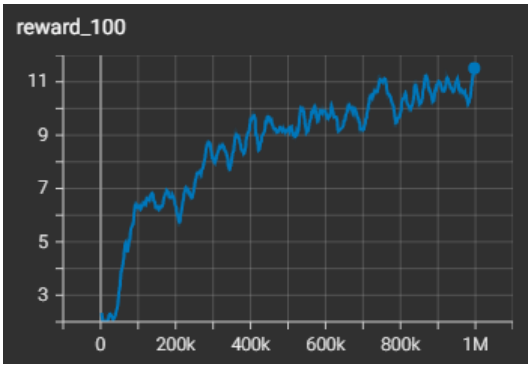

Fig. 2. Breakout Training Reward

The stumbling path that the DQN model takes to success is visible in Figure 2. This is a plot of the training reward of our breakout model over a period of 5 hours and 1 million frames.

B. Transfer Learning

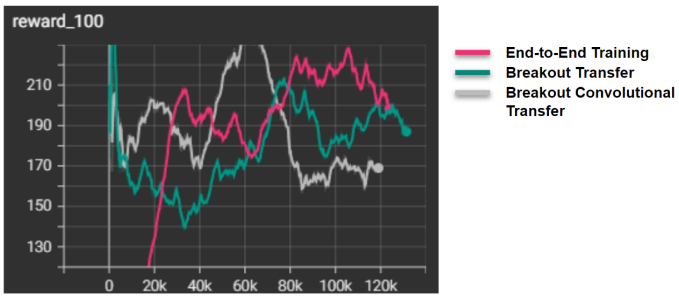

Fig. 3. Space Invaders Training Reward

To test our hypothesis of a faster model training due to transfer learning, we trained each model for about half an hour and plot reward over game frames. Training was thus ended earlier on transfer learned models than end-to-end trained models. The results of the transfer learning can be seen in Figure 3 and Figure 4.

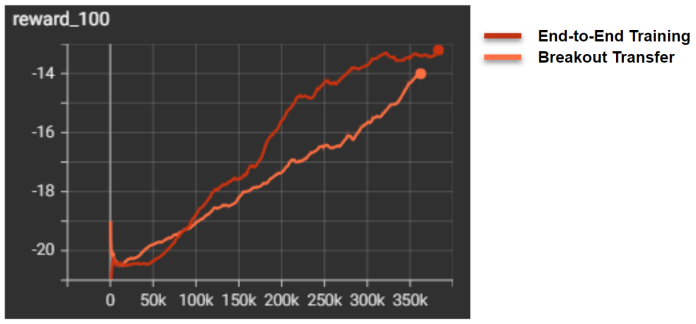

Fig. 4. Pong Training Reward

VII. ANALYSIS OF RESULTS

A. Deep Q-Learning

Though our results were not competitive with the Google Deepmind DQN results, as seen in Table I, our reward results are well above random. With a more exhaustive training process, including more time and system memory, we expect to see better results using the same model architecture. The stochastic nature of the training process also leaves room for poor performance in the case that our models failed to find game exploits. The DeepMind Breakout model found an exploit to dig a hole through the bricks and gain many points by trapping the ball above. Because our model failed to find this pattern does not mean that further trials would fail to find this pattern and other like it.

B. Transfer Learning

Our testing shows evidence that there is little advantage to transfer learning over training each model from scratch. Though the games have identical input and output types, the differences in game playing negate any potential benefit from using learned weights. From the test of locking the convolutional layers after copying weights, we are able to see that initial game attempts have higher reward in Figure 3. The uninitialized models reach similar reward scores after only a few minutes where both models need hours to reach a successful agent. The fact that convolution-layer-locked models are able to train does show that the convolution layer weights can translate between games, but the process of learning the optimal action space and translating convolution output to predicted action space far outweighs the time to train the convolutional layers. The model trained on Breakout performed surprisingly well on Space Invaders with absolutely no modifications. This was to the point that the training process would fail to yield better results than the unmodified model when transfer trained. In Figure 2, we see the fully transfer trained model performs the best at the start of training and quickly falls in line with the other two models in reward. The

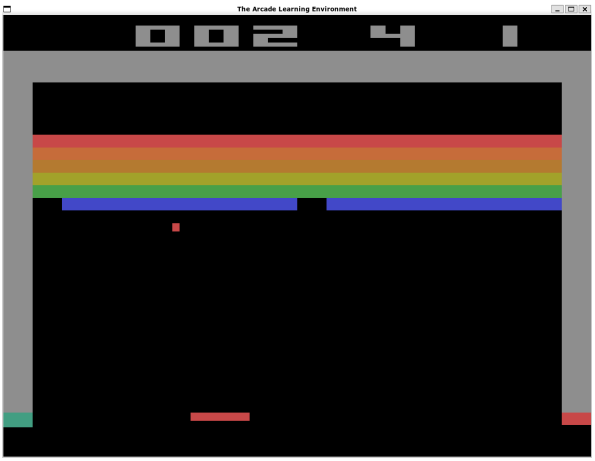

Fig. 5. Our Agent Playing Atari Breakout

model with only the convolutoinal layer transfer-initialized and locked performs slightly worse before evening out. All three models train with similar reward metrics within a few minutes of training.

VIII. CONCLUSIONS

A. Summary

In this project we developed a novel general purpose AI trained using the reinforcement learning methods of Deep Q-Learning and Transfer Learning. Our model was trained to play Breakout, Pong, and Space Invaders using the OpenAI Gym library. The results produced were stronger than various other approaches, scoring a reward of 11.7 for Breakout, 8 for Pong, and 312 for Space Invaders. Our model was not as strong as Google's DeepMind DQN, however we feel we had great success in such a short time frame. A screenshot of our model playing Breakout can be seen in Figure 5.

This general purpose game-playing AI and the methods described in this paper can be used to further autonomous testing of video games for their completion ability or generate opponents. Deep Q-Learning with Transfer Learning proved to be a useful and novel approach to general purpose game-playing AI. We believe this study validates this exciting methodology and opens the door to new research opportunities to expand upon Deep Q-Learning with Transfer Learning.

B. Limitations

Originally, we aimed for our model to be able to play a wider sample of games more efficiently. This would have utilized the hypothesized benefit of transfer learning to quickly train the model on a new game with minimal training time to have a competent agent. Because of the shortened time frame for our research and the lack of realized benefit from transfer learning, we were only able to get our model to play Breakout, Pong, and Space Invaders.

Ideally, our general purpose game-playing AI would be able to quickly pic up new Atari games using OpenAI Gym and compete with previous research.

C. Future Work

Our research used one architecture consisting of convolutional neural network to take as input the pixels of the Atari game. We would like to see this expanded to other architecture types like LSTMs to generate a "memory" component for the network to use. This may boost performance as models are able to store game state information from previous frames. We believe this could provide interesting applications for various games such as Pong where storing previous frames allows for trajectory information to be more easily computed.

Additionally, we believe there should be continued research into larger architectures that attempt to use transfer learning to capture more games. With a larger network and more weights, we reason that more fundamental game concepts can be stored which may result in the hypothesized gains in training when applying the transfer learning approach.

Overall, we are extremely satisfied with our models results and find the breadth of future work regarding reinforcement learning game-playing AI incredibly exciting. We hope future researchers continue to expand upon the work done in this paper.

IX. REFERENCES

[1] Brockman, Greg, et al. "Openai gym." arXiv preprint arXiv:1606.01540 (2016).

[2] Chuchro, Robert. "Game Playing with Deep Q-Learning using OpenAI Gym." (2017).

[3] Gerald Tesauro; TD-Gammon, a Self-Teaching Backgammon Program, Achieves Master-Level Play. Neural Comput 1994; 6 (2): 215–219. doi: https://doi.org/10.1162/neco.1994.6.2.215

[4] Kun Shao et. al; A Survey of Deep Reinforcement Learning in Video Games. Dec. 2019 doi: https://doi.org/10.48550/arXiv.1912.10944

[5] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).

[6] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-Learning", AAAI, vol. 30, no. 1, Mar. 2016.

[7] Torres, Jordi. "Deep Reinforcement Learning Explained." Jordi TORRES.AI, 11 Dec. 2021, torres.ai/deep-reinforcement-learning-explained-series.