Project Report : CS 4644 Spring 2022

Jim James
Georgia Institute of Technology
801 Atlantic Dr NW, Atlanta, GA 30332
jimjames@gatech.edu

Richy Meas
Georgia Institute of Technology
801 Atlantic Dr NW, Atlanta, GA 30332
richymeas@gatech.edu

Nan Zheng
Georgia Institute of Technology
801 Atlantic Dr NW, Atlanta, GA 30332
nzheng32@gatech.edu

Alon Baruch
Georgia Institute of Technology
801 Atlantic Dr NW, Atlanta, GA 30332
abaruch3@gatech.edu

Abstract

MineRL is a competition where competitors utilize reinforcement learning on human priors to obtain a diamond using only four days of training time. An agent is spawns in a random location in Minecraft without any items and receives rewards for obtaining items that are prerequisites for a diamond. Our approach utilized two networks: all layers until block two of ResNet-18 and a Dueling DQN. The ResNet-18 architecture was concatenated with a feed-forward network that outputted the action prediction. We did behavioral cloning with the ResNet-18 model. While, using the Dueling DQN for Soft Q Imitation learning, we are able to achieve a highest reward, 10.73, among all candidate models. Both Dueling DQN along and soft imitation learning out perform the baseline model and the behavior cloning model.

1. Introduction/Background/Motivation

Minecraft is a game where a player tries to gather resources and survive in a three dimensional, infinitely-large, and randomly generated world. One important resource is diamonds. Obtaining diamonds requires a player to first obtain other resources such as iron and can take up to an hour, even for an experienced player.

In this project, we aim to train a machine learning agent to play Minecraft and obtain a diamond within 15 minutes of gameplay using behavioral cloning, deep reinforcement learning, and imitation learning.

The current state of the art for this problem is a structured approach with both Deep Q Learning and Imitation Learning. Reinforcement learning with Minecraft usually involves a convolutional neural network to extract some importance from the current/previous frame(s) and having fully connected layers extract the state action pair. The winners in 2019 utilized multiple architectures (DQN, PreDQN, etc) to obtain the diamond with the given action space and reward system.

This is a fairly complex problem as it goes from pixels to both small and large actions, such as crafting items or building structures. Moreover, Minecraft has an incredibly large upper bound on its state space (as a result of its infinite world size). Therefore, solving it will demonstrate deep reinforcement learning's capabilities and high performance in a generalized manner that isn't possible via classical AI or tabular RL algorithms.

2. The MineRL Dataset and Environment

For our project we used the MineRL Dataset [1] which contains a set of videos of human beings performing specific tasks in Minecraft as well as a modified Minecraft gym-compatible environment to be used for reinforcement learning. Specifically, this dataset contains over 60 million state-action pairs of human demonstrations. The motivation behind this dataset was to combine the scale and freedom in actions of Minecraft with a large set of sample data to use for imitation learning. The MineRL dataset was created by William H. Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov at Carnegie Mellon University.

The environments provided by MineRL are tailored for specific tasks such as chopping trees or navigating complex landscapes. These environments are a 64x64 frame and a

Item	Reward	Use case
Log	1	Craft wood planks
Wood Planks	2	Craft sticks, crafting table
Sticks	4	Craft tools
Crafting Table	4	Craft tools, furnace
Wooden Pickaxe	8	Mine Cobblestone
Cobblestone	16	Craft Stone Pickaxe, Furnace
Furnace	32	Smelt Iron Ingot
Stone Pickaxe	32	Mine Iron Ore
Iron Ore	64	Smelt to Iron Ingot
Iron Ingot	128	Craft Iron Pickaxe
Iron Pickaxe	256	Mine Diamond
Diamond	1024	Final goal

Table 1. MineRL Rewards for each item and its purpose in the game.

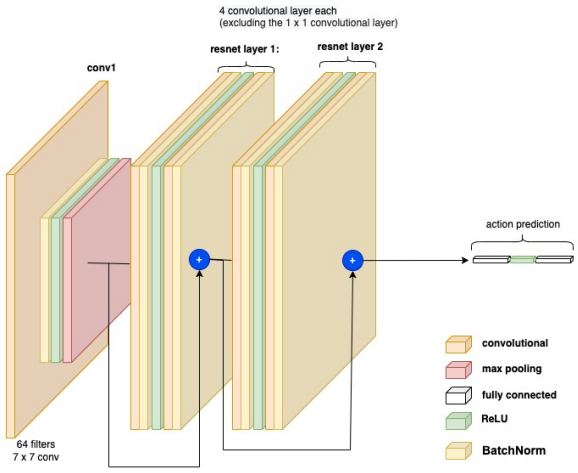

Figure 1. architecture of modified resnet

pair of observation spaces and action spaces. In this project, we specifically used the MineRLObtainDiamondVectorObf environment. This environment yields observations in the form of a 64 ×64 pixel RGB frames and an obfuscated vector of dimension 64 encoding metadata such as inventory items and previously obtained milestones. The actionspace is likewise an obfuscated vector of dimension 64. Rewards are given when the agent obtains items useful in the pursuit of a diamond for the first time in a world. These are shown in Table 1.

In terms of the videos contained in the dataset, they are collected through a plugin installed on player's computers while they play on the MineRL minecraft server. Each player is given a task to complete and is rewarded for completing the task. Throughout this task a packet of information is sent to the data collection pipeline where the current state of the players character is encoded. This state includes the player's point-of-view, player inventory, item collection events, distances to objectives, player attributes (health, level, achievements), and details about the current GUI the player has open. Furthermore the actions that a player takes are also encoded. These actions include all of the keyboard presses on the client, the change in view pitch and yaw (caused by mouse movement), all player GUI click and interaction events, any crafting or smelting performed. Lastly the state-action pairs also include certain metrics to represent the quality of the actions. These metrics include timestamped rewards, number of no-ops, number of deaths, and total score.

3. Approach

Since the environment uses actions in the form of encoded vectors, we used KMeans clustering to discretize the action space. We ran this with 150 cluster centers on 50% of the actions in the human training data to obtain the set of actions that our models would use. With these discrete actions, we tried three different approaches - pure behaviorial cloning with a CNN, deep reinforcement learning, and Soft Q Imitation learning.

For behaviorial cloning, we first used an architecture inspired by a ResNet18, which architecture is shown in Figure 1. One aspect we considered was emphasis on pixels in the center of the image. In Minecraft, the player's camera always is at the center of the screen, emphasizing the importance of those features. Thus, full translation-equivariance is not desired in our architecture, meaning that we do not want our encoder to output a 1 ×1 feature. Combined with the limited spatial resolution of our images, we decided not to have the full depth of ResNet18 and therefore our new model only reused the first convolutional layer to block 2. This was followed by a set of feedforward layers to produce an action vector prediction. Our implementation used pre-trained weights from Torchvision's Imagenet-trained ResNet18 to help with faster convergence.

For our reinforcement learning approach, we utilized the DuelingDQN algorithm [3]. Instead of having a Deep Q Network predict Q values for action-state pairs, it instead estimates *value* of a state and *advantage* of an action-state pair by splitting off the network into two outputs. In essence, value is an estimate of the expected reward from the state, while advantage indicates how much better a given action is compared to all of the others for a given state. To implement this in practice, we used an architecture with a CNN-based feature encoder, followed by concatenation with the obfuscated observation vector. This is then fed into two feedforward networks, one for predicting the value of the input state, and one for predicting the advantage of each action. This architecture can be seen in Figure 2.

Our final candidate model utilizes Soft Q Imitation learning [2]. In this paradigm, instead of relying on MineRL's rewards, rewards are set to 1 for states seen in human action-

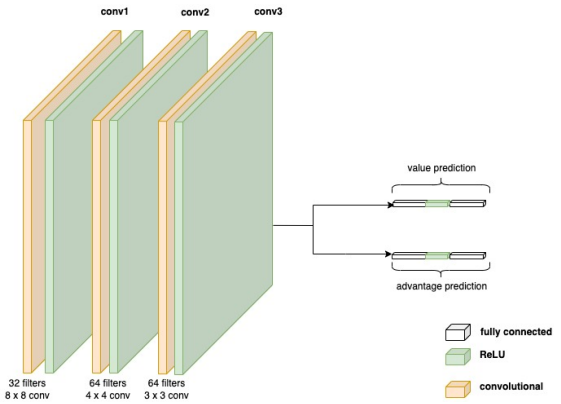

Figure 2. architecture of DQN

state pairs, and 0 for algorithm explored states. For training, half of the states in the replay buffer are randomly sampled from the human action-state dataset, and half are explored by the agent. This utilizes the same Bellman equation based loss as reinforcement learning, in essence incentivizing the network to exploit learned actions from human behavior, and to take actions that lead it to human action-state pairs if it is in an unfamiliar state. For our Deep Q Network, we again utilized the DuelingDQN. We pursued this method as we believed reinforcement learning by itself would be unlikely to converge to a good optimum, since the amount of exploration required to achieve the early milestones would likely be difficult.

For this project, we leveraged several repositories. We used the MineRL 2021 Baseline for the K-Means clustering code and the basic train loop. We modified this to support our ResNet-18 model and adjusted the train loop to add validation. We also used this PFRL MineRL example, and adjusted it to use standard DuelingDQN instead of Prioritized Dueling Double DQN. We later modified this to support Soft Q Imitation Learning by changing how experiences are sampled and how rewards are given.

We compared all of these models to the baseline behavioral cloning model from the 2021 Baseline, which also utilized KMeans clustering for action discretization and a simple CNN-based encoder.

4. Experiments and Results

The primary metric that we used for success was average reward on a set of 1000 randomly generated worlds. Additionally, we recorded the maximum obtained milestone, which is effectively the furthest along the rewards that the agent was able to receive over all 1000 worlds.

To train the two behavioral cloning models (baseline and ResNet-18), we used a simple training strategy with a random 70-30 train-validation data split, Adam as the optimizer, and no augmentation. Adam was chosen as we did not have enough compute time to do a large hyperparameter sweep, and no augmentation was done as there aren't clear ways to map a change in the input frames to a change in obfuscated vectors. We picked a learning rate of 0.0001 and a batch size of 32 by experimenting with these hyperparameters by hand, aiming to minimize the validation loss. The loss function chosen was CrossEntropy without any weights.

For the Imitation Learning and Reinforcement Learning method, we picked a learning rate of 0.0001 with Adam and an update interval (how frequently we run step the optimizer compared to simulation steps) of 10000. We also used 4 input frames at a time, as this would allow the networks to account for information such as velocities. Finally, we used a discount factor of 0.99. These hyperparameters were arbitrarily chosen as a result of the excessive compute time required for a large sweep. Each model took several days to train on fairly powerful hardware (24 CPU cores + GTX 980 Ti). As a result, we likely ended up with a very suboptimal set of hyperparameters. The loss used for these models was the typical loss used for Deep Q Learning (mean squared error between reward + previous maximum Q value and the predicted Q value), where predicted Q values were calculated by summing the predicted value and predicted advantage.

None of the techniques or models were able to successfully obtain a diamond, with the highest average reward of 10.73 being obtained by the Soft Q Imitation Learning DuelingDQN. Both Soft Q Imitation Learning and Reinforcement Learning with the DuelingDQN outperformed the baseline. Metrics can be found in Table 2.

We observed that both the baseline and our ResNet-18 for behavioral cloning would almost always choose to jump while running forwards. This likely is the result of dataset imbalance. Since the vast majority of gameplay in Minecraft involves running around, the actions being taken most of the time in human data would be running and jumping, resulting in this poor optimum. As a result, we believe that our behavioral cloning models underfit.

As we hypothesized, Soft Q Imitation Learning outperformed Reinforcement Learning using DuelingDQN. We believe this has to do with the starting point for optimization; since Imitation Learning learns from human examples over a long period of time, it is more informed than a randomly-initialized reinforcment learning model. This is indicated by the maximum obtained milestone of the Dueling DQN being Crafting Table, implying that the agent was never able to progress past the beginning of the game to larger rewards, hindering its learning. On the other hand, Soft Q Imitation Learning was able to progress all the way to a Stone Pickaxe, which is a fairly complex milestone to achieve. This suggests the importance of leveraging human

data for progressing past the early game.

Overall, despite the failure of all models, these results still indicate the potential for success of Deep Reinforcement Learning techniques, including Imitation Learning, while highlighting the potential issues with pure behavioral cloning.

Algorithm	Average Reward	Maximum Obtained Milestone
Soft Q Imitation Learning	10.73	Stone Pickaxe
Dueling DQN	0.58	Crafting Table
Behavioral Cloning CNN	0.00	None
Baseline CNN	0.00	None

Table 2. Results

5. Future Improvements

There are a variety of avenues to explore to improve these results. All of our techniques and networks relied on discretized actions using KMeans clustering. Other clustering techniques, like DBSCAN or GMM, could be used instead.

Additionally, there may be merit to combining Soft Q Imitation Learning and standard reinforcement learning. By first performing imitation learning, the network may effectively have a stronger initialization for reinforcement learning, allowing it to converge to a more optimal policy. Furthermore, there may be architectural improvements in the convolutional encoder used for the DuelingDQN. Adding in human-inspired biases, like a bias towards center pixels, may improve performance.

References

[1] William H. Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov. MineRL: A large-scale dataset of Minecraft demonstrations. *Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019. 1

[2] Siddharth Reddy, Anca D. Dragan, and Sergey Levine. Sqil: Imitation learning via reinforcement learning with sparse rewards, 2019. 2

[3] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning, 2015. 2