Learning to Play Pinball with MuZero

Eric Fan
Hatem Mohamed
Shivam Hingorani

Introduction

In this project, we aim to build a reinforcement learning agent capable of playing digital pinball autonomously. Pinball presents a uniquely challenging environment because it requires both fine motor control and long-term strategic planning in the face of chaotic, unpredictable physics. We approach this task using the MuZero algorithm, a state-of-the-art reinforcement learning technique developed by DeepMind. Unlike traditional model-based methods, MuZero learns to predict the optimal value, reward, and policy directly from observation sequences, without needing access to the true environment dynamics. We selected MuZero for its groundbreaking performance on a range of complex tasks, such as Atari, Go, and Chess, and for its potential to excel in real-time, physics-driven games like pinball.

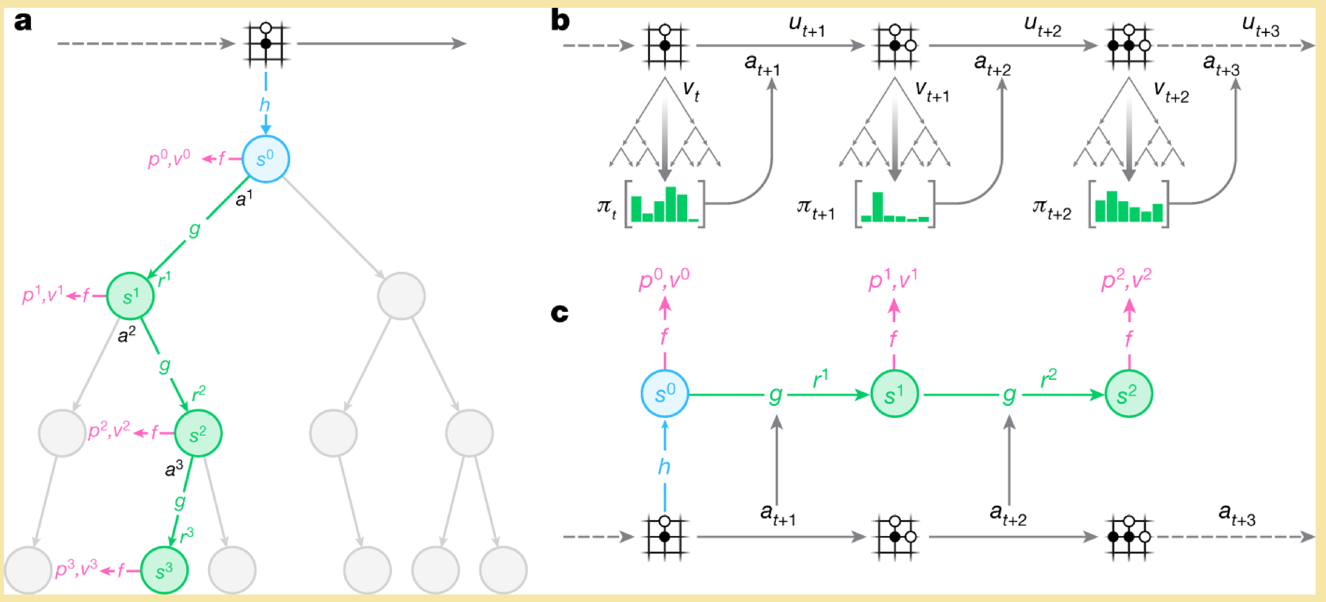
Methodology

Dataset / Environment

- **Game**: Video Pinball (via PyGame Learning Environment).
- **Preprocessing**: Grayscale frames, resize to 84x84 pixels, stack 4 frames.

Model Architecture

- **Representation Network**: CNN encodes the observation to a hidden state.
- **Dynamics Network**: Predicts the next hidden state and reward.
- **Prediction Network**: Outputs policy distribution and value estimate.



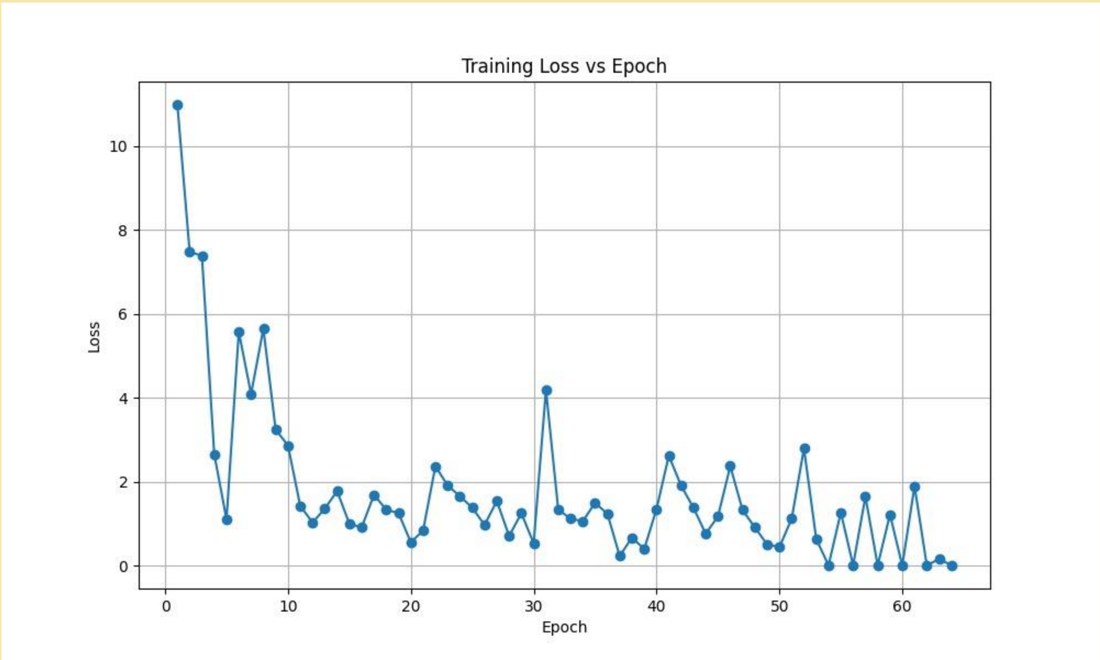
Training Pipeline

- **Self-Play**: Generate trajectories.
- **Replay Buffer**: Store experiences.
- **Training**: Sample mini-batches; compute reward, value and policy losses.

Results

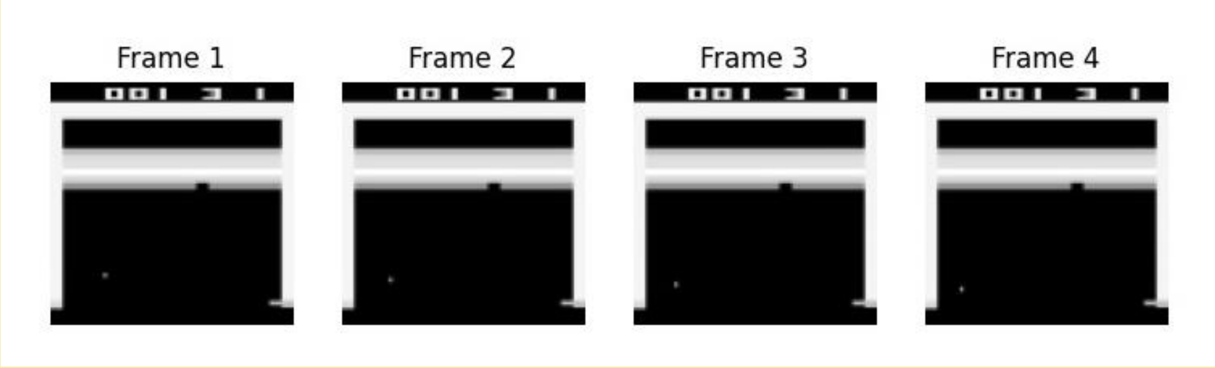
Quantitative

- Training loss is initially high, then decreases significantly by Epoch 10.
- Some noise and spikes remain, but an overall downward trend is clear.
- By Epoch 60, loss stabilizes below 1.0 in most cases, with some fluctuations.
- Indicates that the model is learning effectively from noisy self-play data.



Qualitative

- Replay Buffer and Trainer modules operational.
- Training runs successfully.
- Loss curve oscillates drastically (to be expected).

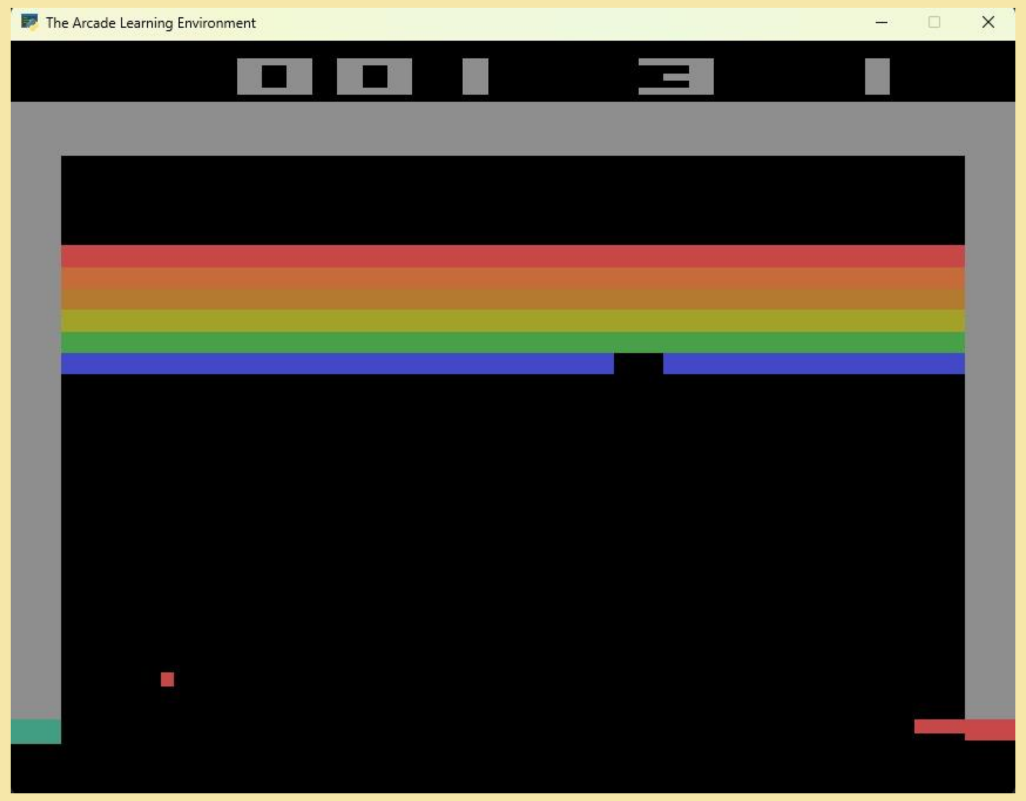


Next Steps

- Improve Monte Carlo Tree Search (MCTS) with Dirichlet exploration and temperature softmax sampling; finetune rollout steps and learning rate.
- Deploy Pinball environment training job on Oscar CCV.

Discussion

One of the main challenges we encountered is that training a MuZero agent takes a very long time, especially when training within a complex environment like pinball. To address this, we leveraged the modularity of our architecture and the abstraction provided by the PyGame Learning Environment to first validate our implementation on a simpler game such as Breakout. This allows us to ensure that our pipeline — including the replay buffer, training loop, and network components — works as expected without incurring long training times. Once validated, we can switch to the more complex Pinball environment and deploy a training job on the Oscar computing cluster to accelerate progress.



Future directions include tuning MuZero hyperparameters, developing deeper network architectures, and evaluating the agent's ability to generalize to different Pinball layouts.

Sources

1. Schrittwieser, J., et al. (2020). *Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model.* Nature. https://arxiv.org/abs/1911.08265
2. Duvaud, W., et al. (2020). *muzero-general: A Generalized Implementation of MuZero.* GitHub. https://github.com/werner-duvaud/muzero-general