Hooked on Phoenix:
Deep Q-Learning on the Classic Atari Game

Evan Darke, Jake Smola, Michael Mernagh

Phoenix

Phoenix is an Atari game in which the player controls a vehicle that moves left and right across the bottom of the screen, shooting up at swarms of alient birds while dodging their fire and dive bombs.

We developed an end-to-end reinforcement learning model that trains an RL agent to maximize its score in Phoenix' complex environment given raw pixels as input.

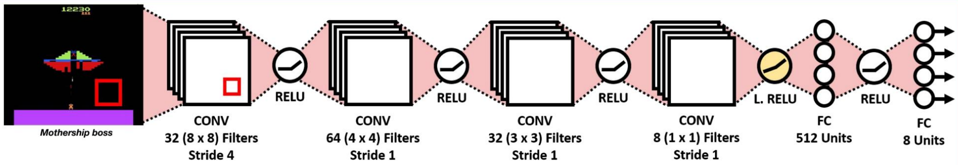
Challenges:
- Very large state space
- State space is not fully observable (agent does not know when shield is available).
- Very delayed returns when facing boss.

Proposal:
- Use autoencoder to reduce state representation and speed up learning.

Feature Extraction

- **Method 1 (Autoencoder)**:
  - Train a deep convolutional autoencoder to compress the state representation by a factor of 384.
  - Use output of autoencoder as input into feed forward network.
- **Method 2 (Downsampling)**:
  - Downsample image by 2 and convert to grayscale.
  - Use CNN to extract features and learn Q function.

Models



Deep Q-Learning, Convolutional Neural Network

Convolutional Neural Network

We leverage a deep convolutional neural network (CNN) to optimize Q-value targets as our agent plays Phoenix. The CNN passes raw pixel input through four convolution layers before flattening the subsequent output and passing it through two fully-connected layers. The precise architecture is shown in the figure above. To improve sampling efficiency, our model employs an experience replay buffer. Furthermore, to improve policy stability, we devise a distinct target network whose weights we update every 10,000 training steps.

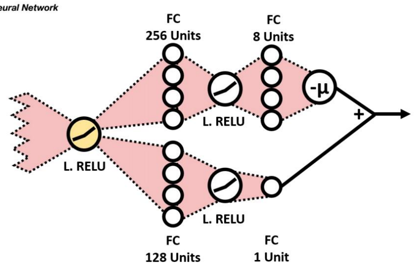
Dueling Architecture

To improve performance, we follow the footsteps of Wang et al. in utilizing a dueling network architecture. This adjustment comprises passing the output of the convolution layers through two distinct streams which respectively compute the state-value function and the advantage function. The state-value function concerns the recognition of preferred states while the advantage function supports action selection. Optimizing these two factors has been shown to improve overall performance in Atari games.

Double Q-Learning

To further improve performance, we also create a double q-learning model which accepts either of the aforementioned architectures at the graph-level. Van Hasselt et al. demonstrated that double q-learning can afford significantly better performance than traditional deep q-learning in the game of Phoenix. Taking a similar approach, we made as few changes as possible to the original deep q-learning logic to derive the double q-learning model. We leverage the already built target network and optimize the below objective.

$$Y_t^{\text{DoubleQ}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \underset{a}{\operatorname{argmax}}\, Q(S_{t+1}, a; \boldsymbol{\theta}_t); \boldsymbol{\theta}'_t)$$



Dueling Architecture

Results

Agent	*Best Avg Reward*
Random	460
Human	3875
DeepMind Architecture	2401
DQN	3250
Double DQN	3490
Dueling DQN	**4111**
Autoencoder (Dueling)	3143
Double Dueling DQN	3453
Final Model	**4423**

Agent-wise best average reward.

Loss Function	*Avg Reward*
Huber Loss	1580
Huber Loss w/ Clipping	2734
Squared Loss	**4102**
Squared Loss w/ Clipping	2725

Loss function reward comparison (DQN agent)

Model Performance

Key Takeaways

- The traditional deep-q learning model suffers from significant oscillation throughout training and only neared the performance of the other models towards the end of training.
  - This is likely due to the relatively lower stability of fixed-target DQN as opposed to the two-network model of double Q-learning.
- The double DQN, double dueling DQN, and dueling DQN models performed very similarly for the first half of training.
  - This can be due to the relative simplicity of earlier stages of Phoenix, the completion of which may not require the benefits of two-network and/or two-stream models.
- The dueling DQN model appeared to continue to improve well into the end of training while both double DQN models plateaued.
  - This finding is slightly surprising as the expectation is that the benefits of double q-learning and dueling networks can be co-exploited. Additional trials and/or longer training could alter this initial outcome.

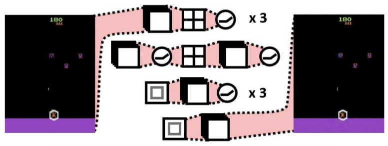
Autoencoder

Method

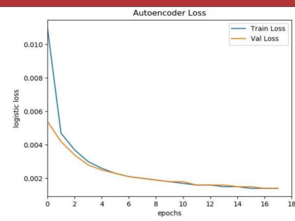
- Randomly sampled 200k frames with 1% probability from random play
- Created 80/10/10 train/dev/test split
- Trained network to reconstruct its input with a bottleneck layer

Architecture

- Gameplay Image → Convolution → Average Pool → Parameterized ReLU (3)
- Convolution → PReLU → Average Pool → Convolution → PReLU
- Up-sampling → Convolution → PReLU (3)
- Up-sampling → Convolution → Autoencoded Image



Autoencoder architecture, with actual input and output examples.



Advantages

- Outperforms baseline using 97% less RAM

Disadvantages

- Struggles to encode small objects like enemy fire
- Encodes irrelevant info like current score

Conclusion

Analysis

- The autoencoder requires a fraction of the memory needed to learn directly from pixels, but falls short of human performance. We suspect end-to-end RL does better because it learns to extract features specific to its task given an indefinite number of examples.
- After 10M iterations, the agent can reliably get to the boss, but doesn't learn to shoot through the middle of the mothership and instead stands off to the side.
- The agent doesn't alway dodge easily avoidable bullets, possibly because it tries to use its shield when it's not ready.

Future Work

- Transfer learning from other atari games may speed learning
- Longer exploration/training may help defeat the boss.
- Sufficient hyperparameter tuning can also lead to significant gains, as seen from Wang et al.
- Policy gradient networks, which directly learn a policy, tend to have more stable convergence.

References

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. CoRR, abs/1312.5602, 2013. http://arxiv.org/abs/1312.5602.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. CoRR, abs/1509.06461, 2015. http://arxiv.org/abs/1509.06461.

Ziyu Wang, Nando de Freitas, and Marc Lanctot. Dueling network architectures for deep reinforcement learning. CoRR, abs/1511.06581, 2015.
http://arxiv.org/abs/1511.06581.