Imitation Learning on MsPacman-v5

Homework 4 for COMPSCI 590-06: Elements of Deep Learning - by Haoran Zhang

Github Repository: https://github.com/Hollen0318/Haoran_Zhang_PacMan_v5

I. Improvement over Learning Algorithm

A. **What I did**

Imitation Learning (IL) is a field of machine learning where an agent learns to perform tasks by observing and mimicking the behavior of an expert. It's a powerful technique, effectively transforming complex task learning into supervised learning problems, making it more tractable.

I perform the IL by writing python scripts (in ***record.py***) to record the observation and my action as the training datasets, then using the deep neural network I constructed to learn from my recorded episodes.

B. Why I did it

In the context of MsPacman-v5, IL is an optimal choice for several reasons.

Firstly, this game has high-dimensional state spaces and nuanced game mechanics, which make traditional reinforcement learning algorithms struggle to find an optimal policy. Imitation learning can bypass this issue by directly mimicking expert gameplay, drastically reducing the learning curve.

Secondly, learning through trial-and-error methods, common in other learning paradigms, may be time-consuming and less efficient. With IL, the agent can leverage expert knowledge to quickly identify an effective strategy, resulting in faster and more stable learning.

Lastly, the deterministic nature of MsPacman-v5 is a perfect match for IL. Since every game state leads to a specific next state, learning the best action to take in each situation, as demonstrated by the expert, can lead to mastering the game. This makes IL a compelling approach for MsPacman-v5, offering efficiency, speed, and efficacy.

C. Result I got

On average, I can achieve 2000 points per episode.

II. Improvement over Model

A. Recurrent Model

In the forward () function inside the network, I weighted the current frame and previous frame by 90% and 10% (which is based on my own experiment studies).

B. Sophisticated Convolutional Model

Though both the hw4_agent.py and my model.py uses the convolutional neural network model, mine implmented the batch normalization and different activation functions.

C. get_action function

My implmentation will read the previous state and update it with the current state.

D. Performance

Baseline: 210 points per episode

Mine performance: 1652.35 points per episode * avg value on 200 episodes

```
Until Evaluation # 0 Average reward =  1640.0
Until Evaluation # 20 Average reward =  1670.96
Until Evaluation # 40 Average reward =  1637.32
Until Evaluation # 60 Average reward =  1607.22
Until Evaluation # 80 Average reward =  1606.08
Until Evaluation # 100 Average reward =  1628.46
Until Evaluation # 120 Average reward =  1635.73
Until Evaluation # 140 Average reward =  1620.01
Until Evaluation # 160 Average reward =  1638.27
Until Evaluation # 180 Average reward =  1639.02
Until Evaluation # 200 Average reward =  1652.35
```

III How to run the code

Prerequisite

1. Make sure you installed all the packages in the dependencies.yml

2. If you want to record your own expert training data, you need a windows console terminal window

User Cases

A. Generate your own data, train the model and evaluate

1. python record.py

This will open an ALE gym environment and let you play the game, when you are done recording your expert imitation, you can press P to exit the recording. The recorded data will be saved under the "saved_episode/Expert_2023-04-30T14-22-07.pt" with the datetime being present.

2. python train.py

This will train the ImitationCNNNet from the model.py to learn your generated expert data and save the model into the "saved_model/ best_model_2023-04-30.pth" with the date being present.

3. python eval.py

This will evaluate the model by the hw4_utils.py instructed by Professor Wiseman.

Or if you prefer to visualize it, you can use the visualize _eval.ipynb.

4*. Python visualize_play.py

This will create a real environment and let you see how the ImitationCNNNet model is doing and report the final reward when it ends.

B. Train the model with my expert data and evaluate

1. python train.py

This will train the ImitationCNNNet from the model.py to learn my generated expert data and save the model into the "saved_model/ best_model_2023-04-30.pth" with the date being present.

2. python eval.py

This will evaluate the model by the hw4_utils.py instructed by Professor Wiseman.

Or if you prefer to visualize it, you can use the visualize _eval.ipynb.

3*. Python visualize_play.py

This will create a real environment and let you see how the ImitationCNNNet model is doing and report the final reward when it ends.

C. Train the model with my expert data and don't evaluate

1. python train.py –eval False

This will train the ImitationCNNNet from the model.py to learn my generated expert data and save the model into the "saved_model/ best_model_2023-04-30.pth" with the date being present.

IV. References

Performances of DDQN Trained Agent with PER and Behavioral Cloning on Pacman Environment

https://www.youtube.com/watch?v=TFmyj-FKIcI