AI Player - 11942014

Samuel Perry

DESCRIPTION:

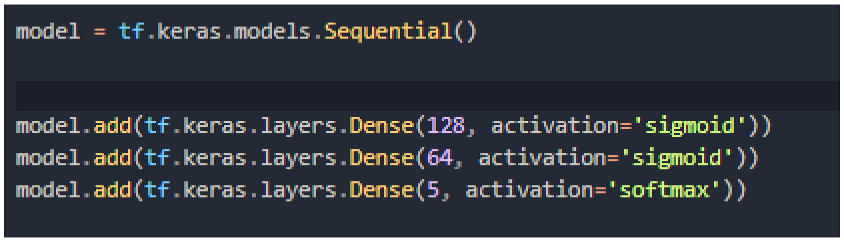
My AI player is a Sequential neural network that has been trained on a dataset of 15,000 hands of Pluribus - Facebook and Carnegie Mellon's world class poker AI - and championship poker winners. To run the AI, the player file imports the h5 model file and makes predictions based on information given.

To begin, I downloaded the data of thousands of hands of poker taken from multiple tournaments meant for researching pluribus vs poker championships. The data was free, but needed some processing to be able to use it for features. To get it into standard poker notation, I used http://kevinwang.us/lets-analyze-pluribuss-hands/ which was originally used to change games using Pluribus' notation into standard poker notation.

After getting a standard Notation for the data, I chose some features to focus on and to begin to train the Neural network. I decided to train the network on the strength of the Hand, pot size, current call needed, how much you've already bet, and # of players on every turn from Pluribus' point of view. I used these features because I needed simple features that could be implemented in the framework we were given and I believed that these would be good enough to get a solid base. With this, I got 100,000s of turns that Pluribus played and every action that the AI took given the features above - *actions.csv and model_train.py.*

After processing the data, I created a CSV of decisions (listed 0-4 - Fold, Raise, Check, call, bet) along with the features that pluribus made its decision. Then I moved to tensorflow and trained a model - *new_model.py*.

The structure I chose was a simple sequential neural network from the Keras api. I tried many structures of the network and the one I settled on was



```
model = tf.keras.models.Sequential()


model.add(tf.keras.layers.Dense(128, activation='sigmoid'))
model.add(tf.keras.layers.Dense(64, activation='sigmoid'))
model.add(tf.keras.layers.Dense(5, activation='softmax'))
```

This is a neural network with 3 layers, first with 128 nodes, 2nd with 64 nodes, and the 3rd output layer is 5 nodes representing the 5 decisions that could be made. The activation functions I also tested and settled on sigmoid for the hidden layers and softmax for the outer layer - *new_model.py*. I then trained using the RMS propagation (Root Mean Squared) which is another form of gradient descent that scales the learning rate. I then trained on the data for 1000 epochs, mainly due to time and the accuracy falling off towards the later epochs. The model was able to get about 70% of the Pluribus predictions correct and this was about the maximum I could get using the data and any structure I tested. With more time (I'll probably go back because it's really cool) I will try to improve that but it will be difficult because pluribus makes decisions based on much more information than the features I have.

After training and saving the model, I implemented it in the *11942014.py* file. This file has the baseline running and decision making from my model and I have run it using the code and moving it into the pokerstrat.py file and passing in the information as numpy arrays to make the prediction.

EXPLANATION:

The reason I decided to use Pluribus data is because the AI is one of the best poker playing AIs and players in the world. It also has some of the very few online poker databases for free.

The reason I chose a neural network is because I started by implementing CounterFactual Regret minimization (CFR) but then switched to a neural network when you mentioned different features that could be considered in a neural network. After that,I took the idea and ran with the neural network and researched some important features I could extract given the data I could download.

The structure I chose is just the result of the best structure after a lot of testing with the data I had.

PROBLEMS:

I included this part to be able to explain some decisions my AI makes as well as list problems I can fix in the future.

1. Pluribus folds **A LOT** because it is pretty optimal in a lot of situations and as such, a lot of the data is skewed towards folding. As such, my ai folds a lot and needs to be scaled down
2. My AI does much better with a lot of players in the mix, I need to go in and modify late game/1v1 poker
3. You need to import Tensorflow, keras, and numpy to be able to run the model and the model must take time to load at the beginning of the program.

NOTICE - I have a lot of files that I couldn't include that cover all my data processing, training, structure of Neural network, and other things that I can include in the zip file but it is all on my github https://github.com/shperry03 in the CS465 repository

References

http://kevinwang.us/lets-analyze-pluribuss-hands/

https://www.youtube.com/watch?v=yJm2l2Mh7nQ

https://keras.io/api

https://www.tensorflow.org/api_docs/python/tf

https://www.youtube.com/watch?v=NWS9v_r_IWk

https://int8.io/counterfactual-regret-minimization-for-poker-ai/

https://icml.cc/media/Slides/icml/2019/102(11-11-00)-11-12-15-4443-deep_counterfac.pdf

https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.295.2143&rep=rep1&type=pdf

https://papers.nips.cc/paper/2012/file/3df1d4b96d8976ff5986393e8767f5b2-Paper.pdf

http://modelai.gettysburg.edu/2013/cfr/cfr.pdf