September 30, 2017

Deep Convolutional Q Learning Applied to Tekken 7

Dillon Connolly
connollydillon@gmail.com

Domain Background

In fighting video games it is often a struggle to create challenging computer controlled combatants (bots). After some time playing the game, the bots are typically no match for the average player. The only way for a player to get better is to play against other human players in online matches. When online play isn't available or is limited, a player's personal skill can only increase so much. Professional matches at tournaments such as the Evolution Championship Series (Evo for short) shows just how large the skill gap is between casual players and professional players. As a casual player of various fighting games, I can't even imagine the skill gap between myself and professionals. By creating more advanced bots in fighting games for players to practice against, casual players will be able to drastically increase their personal skill level.

A perfect example of this is seen in the popular fighting game "Super Smash Bros. Melee" for the Nintendo GameCube. Players are quickly able to destroy the highest level bot that the game offers. However, pair an average player against a professional player and there is no competition. The professional player will completely wipe out the average player. A few students from MIT and NYU demonstrated that it is entirely possible to create superhuman level bots in fighting games as shown in their paper titled "Beating the World's Best at Super Smash Bros. Melee with Deep Reinforcement Learning." The students were able to utilize Deep Convolutional Q Learning to create a bot strong enough to fight equally against professional players. The best players of any sport only break through their skill ceilings when they are playing against more and more challenging opponents.

To avoid confusion, I will provide a brief overview of the strategy I will be applying for this problem. I will be running the game Tekken 7 on a Playstation 4 and interacting with the game from a Windows 10 machine. A combination of PS4 Remote play, REM4P and python libraries will be used to accomplish this. I will get into further detail of the plan in the Project Design section.

Problem Statement

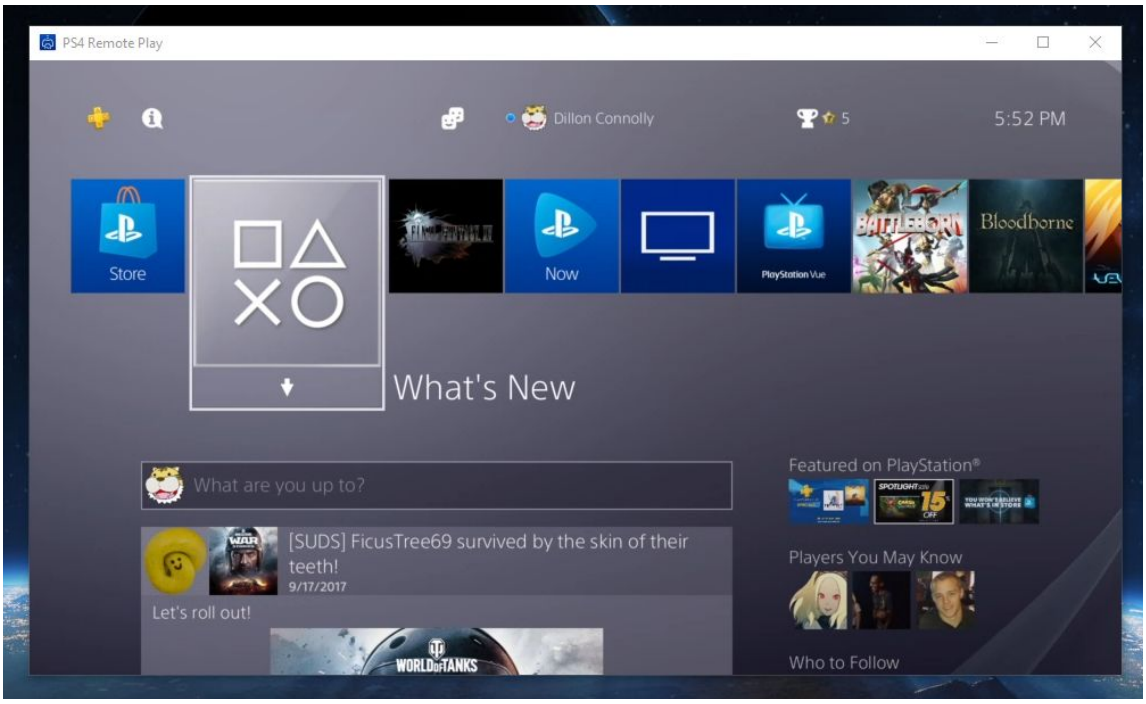
In fighting games there is currently no alternative to getting better than playing against people your skill level online. I want to introduce a new standard for fighting games in which computer controlled bots are taught how to play using deep learning. I believe facing more challenging bots will be the most optimal way for players to improve their skills. I will be

demonstrating this philosophy with the fighting game “Tekken 7.” Currently, at the time of writing this proposal, “Tekken 7” is the most popular competitive fighting game.

Datasets and Inputs

For this to work, I will have to train the neural network how to win. To start off I will be taking various screenshots in game to document what the health bar looks like in various states. I want the neural network to make decisions that are based off the reward it receives when it lowers its opponent’s health bar. Determining which health bar belongs to the bot and what it looks like as it is lowering is crucial to making this work. I will also be taking screenshots of the post-match screen that shows options for character select and playing again. Optimally, I want the bot to be able to always hit continue to engage in the next battle.

After these preliminary screenshots are learned, I will create an array that handles all of the button commands that the bot is able to press. Tekken 7 provides a combo list for each character that shows what button combinations to press to execute special moves on a particular character. I will only be allowing the bot to control one character to keep things simple for this project. The array will contain the basic combos provided by the game as well as the valid buttons by themselves. The neural network will have to decide on its own which is best to execute with a given input. Finally, for visual input, I will use the python library mss to capture raw pixel data and store it into a numpy array. The numpy array will then be fed to the neural network.



Example of PS4 Remote Play running on my Windows machine.

Solution Statement

Owing to the fact that I do not have 2 GPU's and an extremely strong computer at my disposal, I will be offloading the game rendering onto a PS4 to conserve resources. I will be using Sony's software "PS4 Remote Play" to stream the video content over a local area network to the Windows 10 machine. The PS4 Remote Play software allows the host computer to send commands to the remote PS4 as long as there is a controller plugged into the host computer via USB. Using a third party software called "REM4P" I am able to latch onto the PS4 Remote Play process and interpret keyboard inputs as controller inputs. This will allow me to control the PS4 with python. Using the libraries ctypes and mss, I am able to control and "see" what is happening on the PS4 in my python program. Ultimately, I want to utilize Deep Convolutional Q Learning to create a Deep Convolutional Q Learning Neural Network (DCQLNN) that can take raw pixel data from the game as input and then produce keyboard input as its output.

Benchmark Model

The best way to benchmark the success of this bot is to compare my personal win – loss ratio against it. By recording my win loss in an online environment and comparing it to my win - loss with the bot, I will be able to tell approximately what skill level the bot is currently playing at. I will record the resulting win - loss ratio and map it over time to see if there is a trend displayed. If the trend shows that the bot is winning more over time then that means the bot is getting better at the game.

To further push the DCQLNN bot, I will have the bot against itself for many hours. Most of the success in this field comes from the bot playing against itself for a long period of time. Ideally the bot will be able to handle any character and win most of the time.

Evaluation Metrics

The comparison of win – loss ratios between the bot and I will mostly determine the success of the bot. As a final test, I will match my bot against the online community on its own account. If the bot achieves a high win - loss ratio while online facing many different play styles and characters, it will show that on a whole the bot is highly successful.

Project Design

I will be designing a DCQLNN that is able to play the fighting game Tekken 7 on the Sony Playstation 4. Using Sony's free software "PS4 Remote Play" I will be able to connect a Windows machine over local area network or the internet. PS4 Remote Play remotely displays the visual output from the PS4 on the user's computer and allows the user to send controller inputs via a USB attached PS4 controller. In conjunction I will be using a third party software called "REM4P" to map the controller buttons to keypresses. By doing so I can control the PS4 using python. I will be utilizing the ctypes library to interface directly with the windows user32

api. This will allow me to control the window handler that contains the PS4 Remote Play process and then send keypresses to it to control the PS4.

The DCQLNN will use the python library mss to “see” the PS4 Remote Play window. Mss enables developers to screen capture and directly input the raw pixel data into a numpy array. The DCQLNN will be able to process the numpy array to decide what best action to take. To start with I will attempt to just get the bot controlling its selected character reliably. Once this is accomplished, I will set up 2 bots that run simultaneously and fight against each other. I will try different scenarios such as having the bots play the same characters for many hours versus having the secondary bot change its character every 50 matches. I think it will be beneficial to compare the performance of both approaches.

My environment is a Windows 10 machine with a GTX980M GPU, Intel i7-6700K CPU @ 4.00GHz, 16GB of DDR4 RAM and a 500GB SSD drive. My results will be limited to what my computer can produce in the given time frame. I will be using TensorFlow and Keras coupled with Nvidia’s CUDA and cuDNN to allow for running TensorFlow on my GPU to speed up the learning process. As mentioned earlier, I will be using ctypes and mss to capture input and control the PS4 over the local area network. To minimize resource usage, I will be running my python scripts from the command line and using Atom to write the scripts.

References

Ctypes: https://docs.python.org/3/library/ctypes.html

Mss: https://github.com/BoboTiG/python-mss

The Smash Brothers Documentary: http://www.eastpointpictures.com/documentary

Beating the World’s Best at Super Smash Bros. Melee with Deep Reinforcement Learning: https://arxiv.org/pdf/1702.06230.pdf

PS4 Remote Play: https://remoteplay.dl.playstation.net/remoteplay/lang/en/index.html

REM4P: https://tmacdev.com/