Muhammad Subhan 20I-0873  
Jawad Ahmed 20I-0945  
CS-D

AI Project-II Report

Contents

[Data Collection: 1](#_Toc134729964)

[Pre-Processing: 1](#_Toc134729965)

[Model Design: 1](#_Toc134729966)

[Saving the Model to File 2](#_Toc134729967)

[Loading the Model and Predicting Controls 3](#_Toc134729968)

[Random Forest Model (Not Used): 3](#_Toc134729969)

# Data Collection:

The data for the game was collected by playing the game against the given rule based bot in VS battle. Around 80 games were played using different player combination. This generated around 40,000 rows of data which were saved in a csv file. The Format of the csv file is as follows:

'timer','x\_diff','y\_diff','health\_diff','player1\_id','health1','x\_coord1','y\_coord1','is\_jumping1','is\_crouching1','is\_player\_in\_move1','Up1', 'Down1', 'Right1', 'Left1', 'Y1', 'B1', 'X1', 'A1', 'L1', 'R1', 'player\_id2', 'health2', 'x\_coord2', 'y\_coord2', 'is\_jumping2', 'is\_crouching2', 'is\_player\_in\_move2', 'move\_id1', 'Up2', 'Down2', 'Right2', 'Left2', 'Y2', 'B2', 'X2', 'A2','L2', 'R2','move\_id2'

# Pre-Processing:

In this step all of the data values were converted to integer format and normalized by dividing values with 100. The true/false values were left as it is and the controls were concatenated to binary strings which then converted to integer values. After all pre-processing and dropping out any NULL values, we dropped the unnecessary columns that didn’t contribute to the output of the model

'timer', 'x\_diff', 'y\_diff', 'health\_diff', 'player1\_id', 'health1', 'x\_coord1', 'y\_coord1', 'is\_jumping1', 'is\_crouching1', 'is\_player\_in\_move1', 'player\_id2', 'health2', 'x\_coord2', 'y\_coord2', 'is\_jumping2', 'is\_crouching2', 'is\_player\_in\_move2', 'Buttons1', 'Buttons2'

# Model Design:

We have tested neural network and a Random Forest regression model. Although the Random Forest Model was able to achieve a higher accuracy on the test set but did not perform well in the real time environment and lead to slow moves and less accurate moves. So, we proceeded with the Neural Network Model. Following are the design layers for the Neural Network Model.

Text

Description automatically generated

**5 Layers** were used with activation function **Sigmoid**. Other activation functions such as ReLU and Tanh were also used and resulted in low performance. Sigmoid performed quite well as compared to the others.

The data was split into test and train set to check the accuracy of the model before incorporating it into the game. **Mean Squared Error** was used to calculate the loss value and doing backward pass. Over **50,000** Epochs were done with a **learning rate of 0.001**. Overall, on the test set we were able to achieve a Mean Squared Error of **4330** which is quite high. But the model was able to capture the pattern of the player was playing the game quite well.

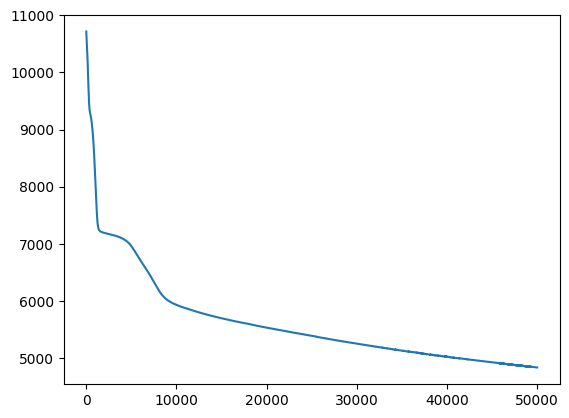


Figure 1: Learning Curve of the Neural Network

# Saving the Model to File

After all the training and testing we saved the model using “ torch.save(newt.state\_dict(), 'NN\_Model.pth')”. The model would then be saved to a file with **.pth** extension

# Loading the Model and Predicting Controls

In the game file bot.py, the current game state was received and then depending on the player’s ID we would create a new List and put all of the values in that list. Then list was converted to tensor and used to predict the Button number. The integer value was then converted to Binary String and then tokenized to get all of the control variables. The command object is then sent to the server and games continues.

# Random Forest Model (Not Used):

Since the problem that we had required the player controls to be simulated in real time, therefore we tried Random Forest Model to overfit the data such that the bot was able to replicate the exact human moves. Although the model showed high accuracy in remembering the data, but when used in the game, it took too long to respond and thus rendered it useless. Moreover, in real time the bot wasn’t able to take the right action with respect to the enemy moves. Following is the plot of the Random Forest Model.

