Assignment 1 - The Resistance

Sayed Maqbool Ahmed Inamdar

Games artificial intelligence can do the things that seems impossible by human at some extent. To elaborate more the technique used in the developing the games can make things more interesting, intelligent, and immersive. And the new technologies will make the system smarter than before. And capable of doing extraordinary things in the field of artificial intelligence.

Consider the game Resistance, by using the different techniques and different models in the development produces different percentages of wining ratio and efficiency. And by tuning it a little bit produces exceptional result. And things I did modified in this project will probably good example of which kind of results the game will produce. Moreover, the detailed modification and tuning are discussed in this report.

Introduction

Playing games are the common activity in human life, and since the 21st century the craze of computer games is increased rapidly. With the advancement of artificial and computational intelligence, several types of computer games whether they played online or offline have undergone tremendous transformations.

A lot of significant advancements have been made in the research field of game AI, Ever since the artificial intelligence was introduced in the games, we have seen a lot of enhancements and accuracy in our games, and it keep on increasing with time. Since various technologies and various methods provides different kinds of result, we needed to select the appropriate development model to get the best result possible.

However, games have long been utilised as a testing ground for AI algorithms and have gradually advanced to superhuman levels. Games are a good candidate for AI research since they are artificial settings that can be altered at will to include various traits and difficulties. For example, Tic-Tac-Toe is solved by searching forward through all possible final outcomes whereas the games like chess and The Resistance are harder to solve and require some sense of strategy.

In our game The Resistance, the random bots play the games and will get different win rate as the members of resistance or as spies. All this result is depending on the data present and how we are training those data to get the information needed to get the higher win ratio. So, after training those data using different training models and recording the results to select the suitable model to get the best outcomes.

Moreover, the methods, techniques and models used to enhance our bot is discussed in this report and kind of modifications done to the game is also mentioned which resulting a significant enhancement in the performance of our neural bot.

The Resistance:

The Resistance is a deception game played between group of players with their own secret identities. Some of them are called members of the resistance and rest are the spies. The resistance is the group who wanted to overthrow the malignant government. And the spies are the group who thwart the Resistance. Basically, there will be three members of resistance team and two members will spy.

The game is played in five rounds and in each round is considered as a mission and that round start with a new leader. And that leader selects the players to go on the mission. And at the same time other members need to do voting, whether to approve the mission or not. Play moves on to the mission phase if the majority players agreed, if majority of the players disagrees with the team players, then leader moves around the group clockwise, and the team-building phase is repeated. After the voting is done the elected players have option to succeed the mission or fail the mission, but they must vote in secret because they can't reveal their secret identity. if elected one is the member of spies, then that player can either make a mission succeed or fail, but if the player is the member of resistance team, then that player must need to do succeed mission if he elected in any round.

The interesting part of the game is only spies will know the secret identities of each other and of other players and they needed to hide their identity through the game otherwise if and if the resistance members have doubt on the any players as spy, then when it turns to leader that player will not pick that suspected player to go in the mission. In the result the chances of getting mission passed will increase. For the resistance to win the game, they must successfully complete three out of five mission. Otherwise, spies will win the game.

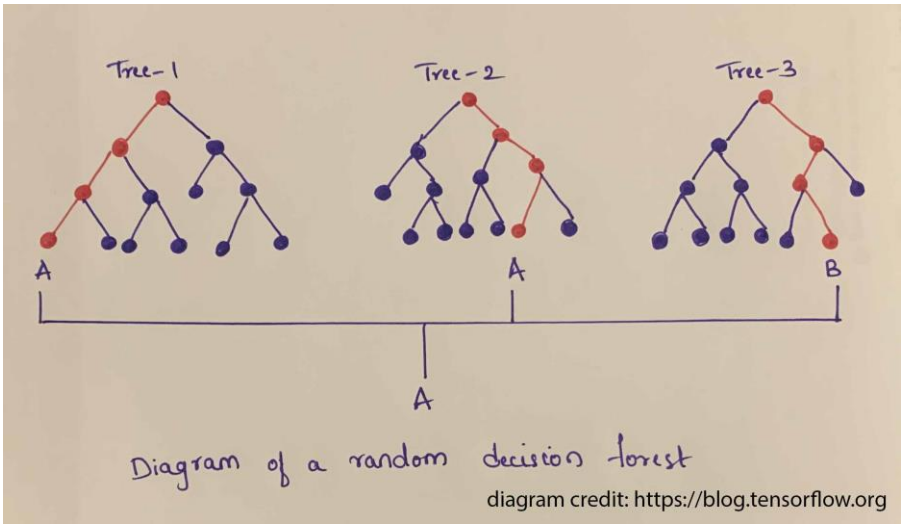
There is a game called "Chor Police", it is an outdoor role-playing games which is very popular in Indian subcontinent. This game played between five players. In which five different cards were shuffled and passed to all players. Each card hold the secret identity like king[1000], queen[500], police[100], cavalry[50] and thief[0]. Each holds different values, and whoever get the police card need to guess the thief. If the guess is correct, then he will get his points i.e.,100 otherwise thief will get polices points. And after certain round the total score will be calculated and whoever get the maximum score will win the game.

This game is almost like The Resistance, expect some rules. But the development model is almost similar. But it is quite easy to implement as compared to The Resistance, because there are less rules for example there is no voting thing in the game, and no body knows anyone's secret identity, just a pure guess based on the probabilities.

Background:

For this assignment, I replaced the Keras Sequential Model with the TensorFlow Decision Forest (TF-DF) to enhance its efficiency. And the more details of the TF-DF are described below which will help to understand better why was better option to enhance the structure.

TF-DF is an open-source algorithm for training, serving, and interpretation for decision forest models. Now that TensorFlow and Keras are more adaptable and composable, we may utilise these models for classification, regression, and ranking tasks.



Decision Forests are a family of machine learning algorithms that may compete in quality and performance with neural networks. When working with tabular data, they are simple to use and comprehend since they are constructed from several decision trees, and you can take advantage of a wide range of interpretability tools and approaches that are currently in use.

In Random Forest model if it is classification task then final or leaf node will be majority class or in other words majority votes as shown in above diagram or the average for the regression problems.

TensorFlow Decision Forest is based on Keras framework and it also compatible with Keras Tuner as well. And in this project, we used TF-DF Tuner because it automatically configures the objectives and if needed it automatically extracts the validation dataset. It supports shared dataset access between the trails.

In this project we used classification trees, and for each tree is grown by taking a random sample of rows from training data, and from that a subset of features will extracted for splitting on each tree and then those trees are grown to the largest extent specified by the parameters.

Techniques Implemented:

Firstly, I am using Random Forest model to enhance the accuracy, since Random Forest model not come by default with TensorFlow. This is installed TensorFlow_Decision_Forests library by using the command “!pip install tensorflow_desicion_forests”.

Note: In this project I modified only two files and those are classifer_loggerbot.ipynb and neuralbot.py from the bots directory. And all other remaining files remained untouched.

Inside classifer_loggerbot.ipynb I have used Random Forest model to get the better output, earlier it implemented using Sequential Model and it was producing the training accuracy of around 76% as shown below.

```
Epoch 118/120
1440/1440 [==============================] - 4s 3ms/step - loss: 0.4371 - accuracy: 0.7617 - val_loss: 0.4335 - val_accuracy: 0.7649
Epoch 119/120
1440/1440 [==============================] - 4s 3ms/step - loss: 0.4370 - accuracy: 0.7621 - val_loss: 0.4335 - val_accuracy: 0.7651
Epoch 120/120
1440/1440 [==============================] - 4s 3ms/step - loss: 0.4369 - accuracy: 0.7617 - val_loss: 0.4348 - val_accuracy: 0.7663
```

I used 50 trees and growing strategy “BEST_FIRST_GLABAL” and maximum depth of the tree is 50 and split axis is “SPARSE_OBLIQUE”

```
model2 = tfdf.keras.RandomForestModel(
    num_trees=50,
    growing_strategy="BEST_FIRST_GLOBAL",
    max_depth=50,
    split_axis="SPARSE_OBLIQUE",
    # inputs = tf.keras.layers.Dense(10,activation = 'relu')

)
```

Inside the model I used additional dense layers with 100 parameters with ReLu as the activation function.

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

model = keras.Sequential( name = 'my_neural_network')
layer1=layers.Dense(100, activation="relu", input_shape=(n
model.add(layer1)
layer2=layers.Dense(100, activation="relu")
model.add(layer2)
# layer4=layers.Dense(10, activation="tanh")
# model.add(layer4)
layer3=layers.Dense(num_outputs) #   No activation functic
model.add(layer3)

print(model(x_train[0:3]))
```

```
tf.Tensor(
[[ 0.26304847 -0.01731232]
 [-0.2796843  -0.29210907]
 [-1.0059555  -0.44771338]], shape=(3, 2), dtype=float32)
```

After this modification the system produces the accuracy more than 82% every time as shown below.

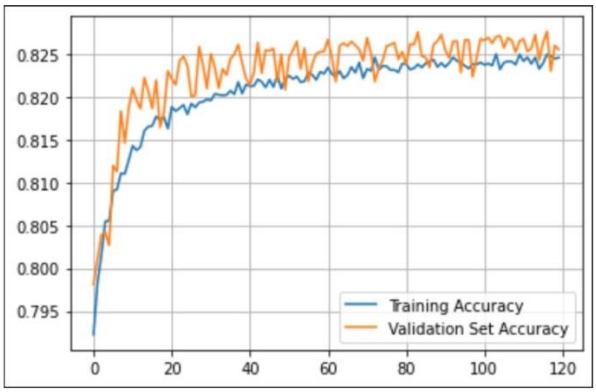


fig: final enhanced accuracy of the system using Random Forest Model

Experimental Study:

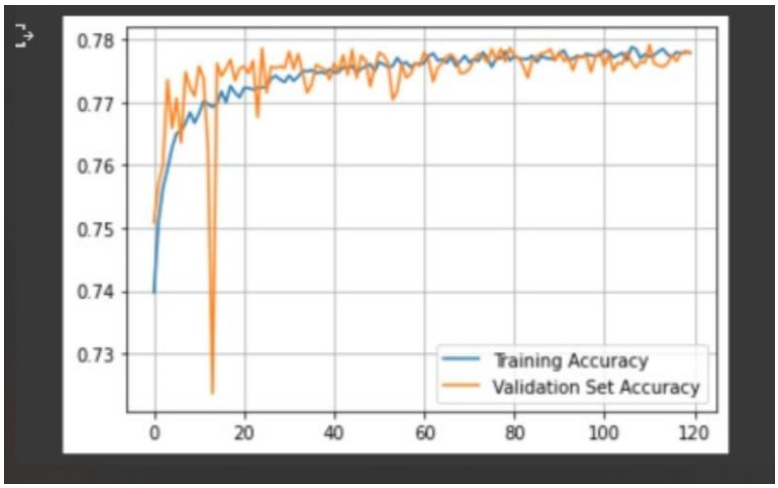
I have used Random Forest Model because it gives better result than default sequential model in TensorFlow.

Earlier I was getting accuracy around 78% after changing few params in sequential model but replacing that model with random forest model gave me accuracy around 82%.

I was experimenting with different params with different activation function like relu, softmax and optimizers like Adam, SDG and FTRL algorithm, dynamic output like sometimes accuracy decreased a lot and sometimes increased a little bit.

After experimenting with all this I figured it out SDG gives stable accuracy. And i.e. around 77.8%. as shown below.

```
1440/1440 [==============================] - 4s 3ms/step - loss: 0.3978 - accuracy: 0.7779 - val_loss: 0.4015 - val_accuracy: 0.7781
Epoch 120/120
1440/1440 [==============================] - 5s 3ms/step - loss: 0.3980 - accuracy: 0.7780 - val_loss: 0.4041 - val_accuracy: 0.7778
```

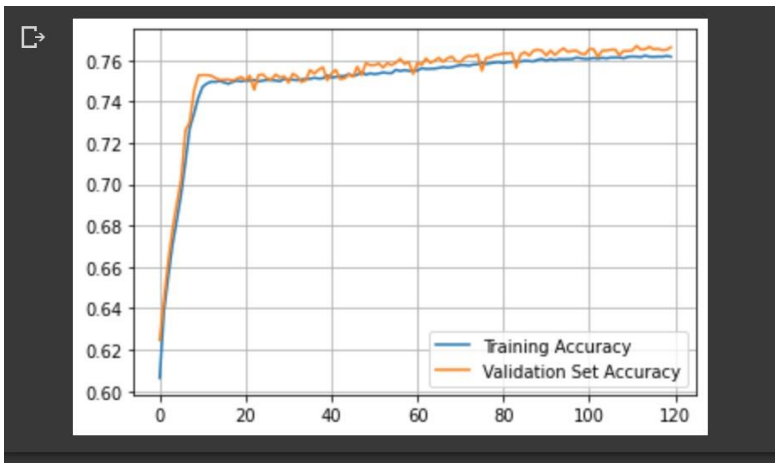


After experimenting with the different functions and different parameters and I did managed to get the Training accuracy of more than 82%.

Analysis:

Using the sequential Model without any kid of tuning to the original structure, the system was producing the Training Accuracy around just above 77% and validation Accuracy more than 77.5% with respect to 120 Epoch as shown in below graph.

```
Epoch 118/120
1440/1440 [==============================] - 4s 3ms/step - loss: 0.4371 - accuracy: 0.7617 - val_loss: 0.4335 - val_accuracy: 0.7649
Epoch 119/120
1440/1440 [==============================] - 4s 3ms/step - loss: 0.4370 - accuracy: 0.7621 - val_loss: 0.4335 - val_accuracy: 0.7651
Epoch 120/120
1440/1440 [==============================] - 4s 3ms/step - loss: 0.4369 - accuracy: 0.7617 - val_loss: 0.4348 - val_accuracy: 0.7663
```



I did surprise by the Random Forest Model while testing different kind of model because the increasement is comparable huge with the sequential model and always give more than 5% accuracy in every run.

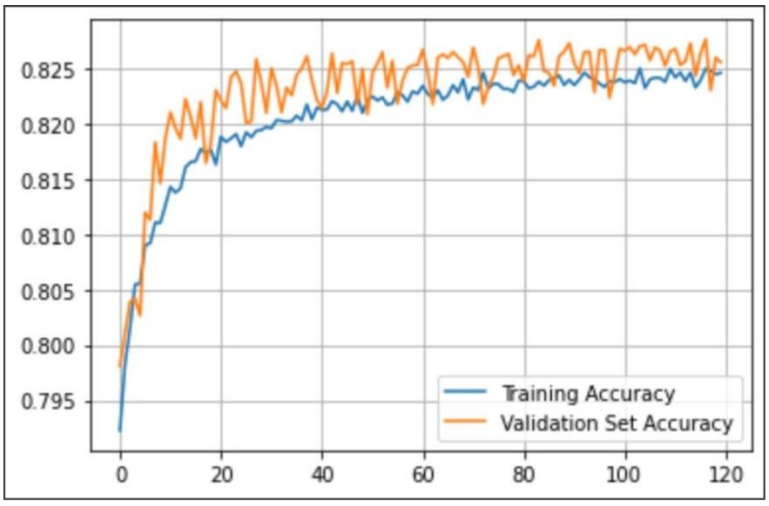


fig: final enhanced accuracy of the system using Random Forest Model

Overall conclusions:

So far, the concept of using different model concerning the game AI have been discussed. As a game AI research, current studies on TensorFlow Decision Forest tend to be limited to only a small number of games which are usually of the genre games. i.e., we still have a very long way ahead to understand the generalization of the game artificial intelligence.

In the meanwhile, with technique like decision tree, behaviour tree, neural network, TF-DF and hybrid intelligence gaining more attention, game artificial intelligence have promising future once these techniques are enhanced and different models implemented enough to be applied to game fields.

References

1. The Resistance Rules: https://www.ultraboardgames.com/the-resistance/game-rules.php
2. Random Forest: i) https://en.wikipedia.org/wiki/Random_forest

   ii) https://towardsdatascience.com/3-reasons-to-use-random-forest-over-a-neural-network
3. TensorFlow Decision Forest: https://www.tensorflow.org/decision_forests
4. The Sequential Model: https://keras.io/guides/sequential_model/
5. Module: tf.keras.optimizer https://www.tensorflow.org/api_docs/python/tf/keras/optimizers
6. Tf.keras.activations.reluhttps://www.tensorflow.org/tf/keras/activations/relu