CS583 Final Project – How close to touch down

Prepared By: Kun Wu

Introduction

Football is one of the most popular sports in the US. We all know how excited we are when we watched the player running to the touchdown zone with the ball. However, Football is a complex game. As a human, we never know who will win and more importantly, how they will win.

Therefore, on the Kaggle website, there is a competition about how many yards will an NFL player gain after receiving a handoff. In this project, I will develop a model to predict how many yards a team will gain on the given rushing plays as they happen.

Model

In this project, I need to predict how many yards the team can gain. It seems like a regression problem, however, the requirement of this competition indicates that for each play I must predict 199 values that represent its cumulative distribution from -99 to 99 yards gain. Therefore, this project becomes a classification problem.

I trained two models in this competition. The first one is a three-layers neural network. The structure of this model is shown in Figure1. The second is an RNN based model with multiple GRU layers.

As shown in Figure 2. The RNN based model takes three inputs, home sequence vector, away sequence vector, and an objective factor vector. Each input needs to go through a Dense layer as an encoder. Then, use objective factor as the initial hidden state of GRU layer, fed the team vector step by step. At last, we got two vectors, concatenate them. Finally, we used a Softmax output layer to make our prediction.

The basic idea of this RNN based model is I assume each play is related the previous play in one game. Then, I treated one game as one sample of data. Each play is a sequence of inputs.

Evaluation

This competition requires to use the Continuous Ranked Probability Score (CRPS) to show how the model performs. The CRPS is computed as follows:

$$C = \frac{1}{199N} \sum_{m=1}^{N} \sum_{n=-99}^{99} \left(P(y \le n) - H(n - Y_m) \right)^2$$

Where P is the predicted distribution, N is the number of plays in the test set, Y is the actual yardage and H(x) is the Heaviside step function. ($H(x) = 1$ for $x \geq 0$ and zero otherwise)

Result

The best CRPS score on validation set is 0.01351

Layer (type)	Output Shape	Param #
input_x (InputLayer)	(None, 516)	0
dense1 (Dense)	(None, 512)	264704
batch_normalization_1 (Batch	(None, 512)	2048
re_lu_1 (ReLU)	(None, 512)	0
gaussian_noise_1 (GaussianNo	(None, 512)	0
dropout_1 (Dropout)	(None, 512)	0
dense2 (Dense)	(None, 1024)	525312
batch_normalization_2 (Batch	(None, 1024)	4096
re_lu_2 (ReLU)	(None, 1024)	0
dropout_2 (Dropout)	(None, 1024)	0
gaussian_noise_2 (GaussianNo	(None, 1024)	0
dense3 (Dense)	(None, 256)	262400
batch_normalization_3 (Batch	(None, 256)	1024
re_lu_3 (ReLU)	(None, 256)	0
dropout_3 (Dropout)	(None, 256)	0
gaussian_noise_3 (GaussianNo	(None, 256)	0
output (Dense)	(None, 199)	51143

Total params: 1,110,727
Trainable params: 1,107,143
Non-trainable params: 3,584

Figure 1 NN Model

Layer (type)	Output Shape	Param #	Connected to
home_sequence_input (InputLayer	(None, 85, 253)	0	
hidden_input (InputLayer)	(None, 516)	0	
away_sequence_input (InputLayer	(None, 85, 253)	0	
home_sequence_encoder (Dense)	(None, 85, 200)	50800	home_sequence_input[0][0]
hidden_encoder (Dense)	(None, 200)	103400	hidden_input[0][0]
away_sequence_encoder (Dense)	(None, 85, 200)	50800	away_sequence_input[0][0]
dropout_5 (Dropout)	(None, 85, 200)	0	home_sequence_encoder[0][0]
dropout_4 (Dropout)	(None, 200)	0	hidden_encoder[0][0]
dropout_6 (Dropout)	(None, 85, 200)	0	away_sequence_encoder[0][0]
home_gru (GRU)	(None, 85, 200)	240600	dropout_5[0][0] dropout_4[0][0]
away_gru (GRU)	(None, 85, 200)	240600	dropout_6[0][0] dropout_4[0][0]
concatenate_1 (Concatenate)	(None, 85, 400)	0	home_gru[0][0] away_gru[0][0]
dropout_7 (Dropout)	(None, 85, 400)	0	concatenate_1[0][0]
output_layer (Dense)	(None, 85, 199)	79799	dropout_7[0][0]

Total params: 765,999
Trainable params: 765,999
Non-trainable params: 0

Figure 2 RNN based model

Further Work

There are two ways to improve the performance of my work. The first one is features engineering, there are still some features I didn't well use, for example, the column 'OffenseFormation'. The value in this column is like this, '1 RB, 1 TE, 3 WR'.

The second part is model, after reviewed many others work. The best models in this competition are Neural Network, Random Forest, and LightGBM. I think I can combine two of those model and get the better result.