Predicting Yardage Gained in Football Run Plays

Sahil Gupta#^1, Sooraj Kumar\*^2, Tiancheng Wu#^3

*#Dept. of Computer Science, University of Illinois at Urbana-Champaign  
Urbana, Illinois, United States of America*

1sjgupta2@illinois.edu

3twu54@illinois.edu

*\*Dept. of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign   
 Urbana, Illinois, United States of America*

2sskumar5@illinois.edu

***Abstract*— In this paper we showcase our approach and results on the ‘NFL Big Data Bowl’ Kaggle Competition [**[**1**](#bookmark=id.omttvp8oi6kf)**]. In football, many of the yards gained during a play come from “run” plays. The competition provides us with a dataset describing these plays with features such as player positions, game score, weather conditions, etc. and the objective is to be able to correctly predict the yardage gained in such plays. The evaluation metric is Continuous Ranked Probability Score (CRPS). The additional objective is to use the developed models to provide insights into what type of run plays are successful or rather what makes run plays successful. In our testing we were able to achieve a CRPS score of around 0.0131 using both Deep Neural Nets and an ensemble of Random Forest Classifiers. We also propose some promising future directions to improve upon this.**

***Keywords***— **Football, Deep Learning, Machine Learning, Bagging, Random Forest, Feature Engineering, Kaggle.**

1. Introduction

American Football is a complex sport. One of the main objectives is to move with the ball to “gain” yards (Figure 1). Many of these yardage gains come from “run plays”. Our objective is to predict yards gained on rushing plays given information about the game status at the start of the play, such as player positions, game time, player orientation, etc.

The dataset consists of player level attributes such as position, orientation, speed, etc. for each of the 22 players, play level features such as score, quarter, play rusher, etc. and game level features such as location, weather, team information, etc. In total about 360 features are provided for each play, consisting of both categorical and numeric features, along with the number of yards gained during the play. Information about over 23,100 such plays was provided in the dataset.

The evaluation metric used by the Kaggle competition is Continuous Ranked Probability Distribution (CRPS) which is defined as:

where P is the predicted distribution, N is the number of plays, Y is the actual yardage and H(x) is the Heaviside step function (i.e. H(x) = 1 for x 0, and 0 otherwise). So, the error is 0 when the yardage is predicted accurately (i.e. P(y = Ym) = 1) . The above metric requires us to predict a probability distribution over the possible yardage value space. In this paper, we treat each of the 199 possible yardage values (-99 to 99) as distinct classes and predict probability of belonging to each of these classes, and use this as our predicted distribution function.

1. Overview

For this task, we implemented three primary steps:

1. Transformation of the dataset from player-level feature set to a play-level feature set. This was required since the original dataset provided to us had features per player per play per game in each row. We had to identify the game-level, play-level, team-level and player-level attributes and transform the dataset such that each row represented all the information for each play.
2. We apply feature engineering to create ~50 meaningful features based on discussions in the football and ML community [[2](#bookmark=id.51m5s4m72wqa),[7](#bookmark=id.lafc3xxti30t),[8](#bookmark=id.qcpfsvn51sne)], some past research into this domain [[3](#bookmark=id.sgwdqeyg8ysi),[4](#bookmark=id.5bpfljygqrcm),[5](#bookmark=id.ahraob5epp6y),[6](#bookmark=id.gnlnozt5bkv5)] and some of our own intuition. We found that features related to the “rusher” and statistics about other players relative to the rusher made good features. We give a description of some of these features in section [III](#bookmark=id.3gzj4jbs0e4r).
3. We tried several Machine Learning classification and regression techniques to achieve good performance on the yardage prediction task, such as Decision Trees, Linear Regression, Naive Bayes, Deep Neural Nets and various Ensemble methods. We provide a description of the specific methods and their results in sections [IV](#bookmark=id.vixqcvtohezp) and [V](#bookmark=id.vps9pxvoai4w) respectively.

1. Feature Engineering

We utilized several unique features that can be organized into three categories: player-based features, team-based features and play-based features.

The most important player-based features are features based-off the rushing player or the “rusher” ([Figure 3](#bookmark=id.2w7pbifhbxil)). Intuitively, these features would be quite important as the rusher is the main driving force behind gaining yardage in a play. Features relating to attributes of the rusher and all the players relative to the rusher during the play would very likely influence the yardage finally gained on that play. We used features such as the distance of the rusher from the line of scrimmage, the mean, standard deviation, minimum and maximum distance from every player to the rusher, the rusher’s orientation, the rusher’s direction of movement, the rusher’s speed, and the projections of the direction of the rusher on the x- and y- axis. Aside from features relating to the rusher, we also included features such as the x and y coordinates of each player on the field (assuming that the bottom-left corner of the field was (0,0)), the speed of each player during the play, the acceleration of each player during the play, their height and age. We also included the projection of the velocity and direction of each player on the x- and y- axis.



For team-based features, we included features like formation ([Figure 2](#bookmark=id.vdmncms75rs5)), which team currently had possession, and if the team is in its own half of the field. Each of these features were one-hot encoded. We also include a unique feature called ‘DefendersInTheBox’ that calculates the ratio between the number of “defenders” in the box and the number of yards remaining to earn a first down. A very high value would correspond to many defenders lined up along the line of scrimmage since they have very few yards to lose before the opposing team obtains a first down. These features correspond to specific attributes about a team during a play and these attributes may underlie strategies that a team might be using to either gain yardage or to prevent the other team from gaining yardage.

****

Finally, in terms of game-based/play-based features, we included features such as the quarter, the down, the game clock in seconds, the direction of play, the score difference between the teams, and time-delta of the handoff to the snap. The quarter and the down could influence specific strategies that either team has decided to adopt. For example, if a team has possession and is in the fourth down, they might decide to punt the ball down the field. Another example is that if a team is leading and they are already in the fourth quarter, they might decide to play more defensively to preserve their advantage. The score delta is powerful in that it shows how much one team is leading or lagging. This again influences the specific strategies that a team may employ such as playing aggressively or desperately when it is lagging by a large amount.

1. Approach

During the scope of this project, we tried numerous classifiers to see how the results compare with each other. We first tried simple methods such as Gaussian Naive Bayes classifiers and Decision Tree Classifiers and later moved on to more complex models such as neural networks and ensemble methods. For each of the classifiers, we split the original dataset into two parts: 80% for training and 20% for testing/evaluating.

****

The first classifier that we tried is Gaussian Naive Bayes. Due to the big number of classes (199) and correlation between features, it performs poorly and only achieves a CRPS of only 0.162.

The next classifier we tried is Decision Tree, a simple yet effective classifier to approach this problem. The final CRPS we achieved with this classifier was 0.0259, which is a big improvement over our gaussian naive bayes approach.

****

After these attempts, we decided to build our own neural net architecture as described in [Figure 4](#bookmark=id.lmjz728o0pqb). The architecture consists of 4 modules with each module containing a linear layer, a ReLU activation, a DropOut layer, and a Batch normalization layer. The number of dimensions for each sample in the final output is 199, which corresponds to each of the discrete yards gain in range [-99, 99]. We apply softmax function to the final layer to produce our predicted probability distribution. On the evaluation set, we achieved a CRPS of 0.0131, which is a huge improvement over our previous methods.

For training our neural network, we used 10-fold cross-validation. That is, we split training data into 10 parts evenly, built a model based on 9 parts and evaluated the model on the left over part. This process was then repeated 9 more times with a unique block left out for evaluating the model each time. The optimizer we used is stochastic gradient descent (SGD). We also used a learning rate scheduler to dynamically change learning rate across epochs. We ran our training for roughly 100 epochs, where the loss converged near the end. Our loss function for neural network is categorical mean squared error.

Last but not least, we tried ensemble related methods to compare with our previous efforts. We tried Bagging and Adaboost ensembles of Random Forest Classifiers ([Figure 5](#bookmark=id.wzq5yn9za6y8)), Extra Tree Classifiers, Decision Tree Classifiers and K-nearest-neighbor Classifiers. We limited the maximum allowed depth for random forests to avoid overfitting. Of those, the one that performs best is the bagging ensemble of Random Forest Classifiers achieving a CRPS of it 0.0131, at par with our neural network performance..

1. Results

We have summarized our results in Figure 6, which showcases the performance of all the classifiers we tried.

Interestingly, the bagging ensemble of Random Forest Classifiers and our neural network model performed quite similarly. This could be due to several reasons:

1. Our neural network is made up of ReLU activations, and its nonlinearity is quite similar to decision trees since it allows splitting on some attributes (compositions of them) in parallel.
2. Due to the nature of the problem we are solving, the performance might be hard to improve upon beyond a threshold. This might be because you can’t really learn a lot of information just from the start position, and there is a lot of noise based on other information, for instance, if the rusher does a trick play by passing the ball behind and so on.
3. Conclusions

In this paper we have showcased our approach to get a competitive score in the Kaggle NFL Big Data Bowl Competition. We engineered ~50 features describing each run play, and then used various machine learning classification methods to predict a probability distribution over the possible yardage values. We found that both a neural net and a bagging ensemble of Random Forest Classifiers performed at par with one another, and provided a competitive score on the CRPS metric. The feature engineering provides the major contribution to the performance of the trained predictors. In particular, features of the rusher and the features of the other players relative to the rusher make for good attributes to train the yardage predictor.

1. Future Directions

We foresee the following potential avenues for further exploration in this problem space:

1. *Using an ensemble of neural nets*. As we mentioned, while Decision Trees and Random Forest Classifiers by themselves didn’t perform well, a bagging ensemble of them performed very well. We see potential in using an ensemble of shallow nets to get even better performance than our current methods.
2. *Predicting probability distribution parameters instead of the probabilities for each yardage class*. So, for instance, in the case of the gaussian distribution, we would predict the mean and standard deviation. The motivation for this comes from Variational AutoEncoders (VAEs) [[9](#bookmark=id.pq79h8jq148h)] and the structure of the CRPS metric. Such an approach would also preserve the continuous domain structure of the prediction, which is currently lost when using classification methods. We envision this being used with a custom differentiable loss function similar to the CRPS metric to train our neural networks to predict good probability distributions rather than probabilities for each class.
3. *Applying interpretability methods such as LIME [*[*10*](#bookmark=id.r80zhklj5df)*,*[*11*](#bookmark=id.nvbfsvby8wd5)*] to our predictors*. As we mentioned earlier, one of the objectives of this competition is to identify attributes of successful run plays, which can then be used to better inform coaches and their run-play strategies. Using model-agnostic interpretability methods would allow us to identify the set of features that correlate well with successful plays, to try and explain our model and help coaches better understand the factors leading to successful plays.

Acknowledgment

We would like to thank the course staff of CS 598PS Machine Learning for Signal Processing (Fall 2019), in particular Prof. Paris Smaragdis and the course TA Shrikant for their guidance and counsel throughout the semester.

References

1. NFL Big Data Bowl | Kaggle Competition

<https://www.kaggle.com/c/nfl-big-data-bowl-2020/overview>

1. Kaggle Discussion about Rusher Features

<https://www.kaggle.com/c/nfl-big-data-bowl-2020/discussion/113698#latest-675738>

1. Lee, Peter, Ryan Chen, and Vihan Lakshman. "[Predicting Offensive Play Types in the National Football League](https://tinyurl.com/y2gf33yd)."

1. Ota, Karson L. “[Football play type prediction and tendency analysis](https://dspace.mit.edu/handle/1721.1/113120)”. Diss. Massachusetts Institute of Technology, 2017.

1. Strange, Robert, and Lior Shamir. ["Prediction of American Football Plays Using Pattern Recognition."](http://search.ebscohost.com/login.aspx?direct=true&profile=ehost&scope=site&authtype=crawler&jrnl=16844769&AN=113234809&h=qkH7aZ8ab26rs03f17rkBJNfYzjRH7V988wf33TjI5BEWjBaatpOQ54KLbDN6%2BEFc1W1BR07yC3CYIBBJS7Xvg%3D%3D&crl=f) *International Journal of Computer Science in Sport (International Association of Computer Science in Sport)* 13.2 (2014).

1. Teich, Brendan, Roman Lutz, and Valentin Kassarnig. ["NFL Play Prediction."](https://arxiv.org/abs/1601.00574) *arXiv preprint arXiv:1601.00574* (2016).

1. Kaggle Shared Notebook implementing Rusher-based feature engineering

<https://www.kaggle.com/dandrocec/location-eda-with-rusher-features>

1. Kaggle Shared Notebook implementing some data cleaning and basic feature engineering

<https://www.kaggle.com/rutviklathiya/nfl-starter-mlp-feature-engg>

1. Kingma, D. P., and M. Welling. "[Auto-encoding variational bayes](https://arxiv.org/abs/1312.6114). arXiv 2013." *arXiv preprint arXiv:1312.6114* (2019).

1. Tulio Ribeiro, Marco, Sameer Singh, and Carlos Guestrin. "[" Why Should I Trust You?": Explaining the Predictions of Any Classifier.](https://arxiv.org/abs/1602.04938v3)" *arXiv preprint arXiv:1602.04938* (2016).

1. Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "[Model-agnostic interpretability of machine learning.](https://arxiv.org/abs/1606.05386)" *arXiv preprint arXiv:1606.05386* (2016)