A Deep Learning Approach to NFL Playoff Forecasting

Dylan Lasher
Computer Science Department
Idaho State University
Pocatello, ID. USA
lashdyla@isu.edu

Paul Grahek
Computer Science Department
Idaho State University
Pocatello, ID. USA
grahpaul@isu.edu

Justin Palmer
Computer Science Department
Idaho State University
Pocatello, ID. USA
palmjust@isu.edu

***Abstract*—Forecasting the results of American football playoff results is an intriguing application of machine learning due to the widespread popularity of the sport. However, predicting these results poses a difficult problem owning to the breadth of complex factors at play. These factors are too numerous and inter-related to model conventionally, so a machine learning approach appears a novel application. We propose a deep-learning neural network to compensate for the seemingly chaotic nature of the sport, in collaboration with a large dataset for training. Our team has gathered decades of relevant data and analyzed their cumulative role in anticipating football playoff results.**

***Index Terms*—Deep Learning, Machine Learning, NFL, Football, Sports, Sports Analytics, Data Analytics, Neural Network**

I. Introduction

This project aims to extend the predictive capabilities of sports forecasting. This has become possible thanks to the large troves of data that has been recorded in football matches.

The NFL Playoff games are some of the most watched program on television. Millions of people tune in every year to watch these games. With so much coverage on these events, a lot of time is put in by sports analyst, and fans to try and predict who would win these games. Our goal is to try and predict who would win a NFL playoff game, between any two teams, based on their regular season team statistics.

It is hard to try and predict winners of sporting events, because some time the better or stronger team does not win. When a team loses that should have won, or was predicted to win. We call that a upset. Upsets are pretty common in the NFL and especially the playoffs when each teams are playing at their best. This causes low accuracy when trying to predict games. Even the best professional sports analyzers, who put hundreds of hours in to researching teams can only marginally predict the outcome of games. Knowing this, we were not expecting to obtain an high accuracy, but were hopping to predict around sixty to seventy percent accuracy.

We have collected and sanitized more data than any existing Machine Learning forecasting study, allowing use to use our neural network to its fullest. In particular, we have implemented a Deep Learning Neural Network so that we may use our large input vector to its fullest. This should yield a predictive model that may predict NFL playoff winners with significant accuracy.

Note we are not trying to classify or rank the strength of individual teams instead. we are trying to predict the outcome of a game between any two teams in the NFL playoffs.

II. Background

A. Motivation

An important element of machine learning in sports is the ability to evaluate a team's chances of winning particular games. Outcomes of sports matches can be difficult to predict, which gives way to its competitive popularity. Football in particular is Traditional predictive analyses have simply used existing match results to evaluate overall team performance metrics and build statistical models to predict the results of future games. However, match results are an imperfect metric of a team's performance capabilities and therefore an incomplete metric for future forecasting.

The evolution of new Machine Learning techniques have allowed for better predictive performance in a wide range of analytical problems. With a combination of a deep learning approach with a previously unprecedented level of meta-data and game-variables analysis, we hope to provide a unique insight into the predictive forecasting of NFL playoff matches.

B. Related Works

In the field of sports analytics, computational analysis has been at the forefront for several decades. As we grew to discover, there are not many approaches that can rival that of Machine Learning, arguably the best solution to data-leveraged forecasting. For many sports, there have been several studies for helping to deconstruct the supposed randomness of the games.

[1] had proposed a data-mining predictor model of sports games. They put forward an approach that makes predictions based on a combination of several metrics derived from

historical game data. The main idea is that they analyze a set of teams that are the most similar to each of the competing teams, and use those game results to predict the outcome of the game between the original two teams. This proved effective when applied to college football matches. Their approach parses the teams and creates a map with every point representing a team. The distance between two points on the map is proportional to their similarity.

Potentially due to the American-centric nature of American football, understandably, the research into Machine Learning techniques for football predictive analytics is scarce. What we can also look at, however, is general Machine Learning approaches to sports in general. For instance, [2] had employed a Bayesian Network Model to predict the results of soccer matches. They divided the data set for the project into two: physiological factors (average player age, match frequency, injury statistics, etc.) and non-physiological factors (previous match history, weather, etc.). This team used NETICA software for model-building, and achieved a significant accuracy rating of 92%. While this lead us to believe that considering meta-data was a significant factor in forecasting, this study was only limited to one team's data.

Additionally, [3] proposed a Bayesian hierarchical model to predict soccer match results. The data they used was based on the attack and defense strength of each team. While they had a significant accuracy of 95%, their predictions seemed to only identify teams with a propensity to shoot more goals, which is much too simplistic in the face of all the extra data available. Again, highlighting a need for a greater consideration to variable breadth.

The issue is that these analytical approaches are either applied to the wrong subject matter or their approach to data is underwhelming. These are a deadly duo in the field of predictive analytics, especially for Machine Learning, where context and data quality is key to accurate forecasting. This is what lead us to propose a Deep Learning approach with a heavy breadth of consideration for every variable we had managed to scrape.

III. Data Gathering

Three options for collecting historical data about the NFL playoffs were considered. One option involved using services that connect to the NFL's API, but these would have cost money to use and did not provide data prior to the last 5 years. Another option was to scrape the NFL's website for data but the URLs were inconsistent and data did not go back the full 20 seasons needed for the model. The third option and the one used involved developing a web scraper to collect historical data from Pro Football Reference, a website dedicated to tracking a wide variety of statistics about the NFL far beyond the last 20 seasons of data, including season, game, coach, referee, and player statistics. The data gathered was formatted into an ARFF file which was then processed by the model into matrices to perform bulk mathematical operations during forward and backward propagation.

An initial list of features were selected based on common statistics relating to NFL teams over the regular season. This list can be found in Table I. Some early analysis of these features selected introduced one concern with how some statistics are determined. Team rankings in offense, defense, and special teams are relative to the teams in the season in question and are not necessarily an indicator of an individual team's actual strength. These rankings are also based on certain metrics. Offensive rank is based on the points scored by the team while playing on offense. As a result, these metrics are unreliable when training the model because the actual difference between two teams is better made as a comparison of the metrics themselves, not their relative positioning. (See Table 1 in Appendix)

Another feature that was removed was the playoff seed of the teams. Playoff seeds are a result of a team's record, strength of schedule, tie-breakers, and other attributes that are not explained well by a single attribute.

The quarterback rating (QBR) according to the NFL is a "fixed performance standard based on statistical achievements of all qualified pro passers since 1960". [4] Because the purpose of this forecasting is to find relationships between statistics aggregated by the QBR, this introduces a human bias in how the passing game of the team in question is evaluated. The QBR is not included as a feature for this reason.

The final attributes removed pertain to information about the stadiums that host the teams and the venue in which the game was played. There was not enough time to investigate whether these features have a direct impact on the outcome of a game but are possible avenue for exploration in the future.

The final feature set used for the model is seen in Table II. The continuous attributes are derived from season stats of the team in question, while the team type is the relationship of the team to the stadium in which the game is played. The game result is the team who won the game. The instances created for the feature set are the playoff games from the last 20 years of NFL play, with two instances per game where the teams are flipped between the t1 and t2 positions. (See Table 2 in Appendix)

IV. Model Design Choices

Due to the extreme complexity of the variables, and the widespread commercial use, we have elected to use a deep learning approach - A backpropagated Neural Network. This system is a collection of nodes (neurons) that algorithmically model the links between neurons in the human brain. Each neuron can receive a signal from other neurons and pass it on to the next layer of neurons. A neural network is composed of an input layer, with one neuron per input variable in the model data, an output layer, composed of a single neuron which gives the classification result, and a number of hidden layers between the two, containing a variable number of neurons in each layer.

A neuron which takes in an input x_j from a previous neuron then calculates its activation value through its activation function:

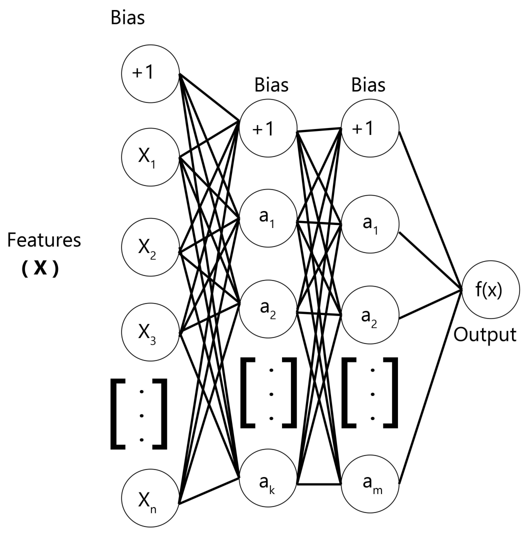

Fig. 1. Deep Learning Architecture

$$f : f(p_j) = a_j$$

The neuron's output o_j is then generated through its output function $_o$ such that $f_o(a_j) = o_j$.

This leads us to the propagation function which calculates the input x_j that a neuron j receives in the network:

$$p_j = \sum_i o_i w_{ij}$$

where w_{ij} is the weight between neurons i and j.

Training the Neural Network model involves setting the correct weight between each two neurons in the neural system. This is done through backpropagation. Here we input a training vector, calculate the gradient of the loss function, and update the weights from their output layer back through the network to the input layer. (The loss function is a function which quantifies the error between the network's prediction and the actual value.) This process is described by:

$$\delta w_{ij} = w_{ij}(t+1) = w_{ij}(t) + \eta \frac{\delta C}{\delta w_{ij}}$$

- η - The learning rate that determines the magnitude of the weight changes.
- C - The cost-loss function which depends on the learning type and neuron activation functions used.

The advantages to using a Neural Network over alternative models is that they usually achieve a higher predictive accuracy compared to alternative techniques. They do, however, require much larger sets of training data to optimize the model, but our approach of meta-data inclusion would address this.

In doing so, we have had to make several design choices. We began with a scaleable deep learning library, followed by the implementation of a couple refining features:

A. Gradient Descent

A gradient descent is the best method used to train deep learning models. A gradient descent is an optimization algorithm often used to find the weights/coefficients of machine learning algorithms. The model makes predictions on training data and uses the errors it makes to update the model in such a way as to reduce the error. It goal is to find the parameters that minimize the error of the model on the training dataset. It does this by making changes to the model that move it along a gradient of errors toward the minimum error value.

We have implemented a mini-batch gradient descent algorithm, which is a prime reason it is commonly used in deep learning models. This approach splits the training dataset into small batches that are used to calculate model error and update model coefficients. The frequency at which the model updates is higher than a regular batch gradient descent which allows for a better convergence, avoiding local minima.

This is in contrast to the alternatives: Stochastic Gradient Descent and Batch Gradient Descent. SGD is a computationally expensive approach that, because of its learning process, has a difficult time settling on an error minimum. A regular BGD, additionally, updates at the end of the training epoch, accumulating prediction errors across all training examples The mini-batch gradient descent is the algorithmically better choice in this case.

B. Regularization

Overfitting is the phenomenon where a neural network models the training data, and its noise, very well but fails when it is given non-training data from the same domain. L2 regularization is the most common form of regularization. This approach defines the regularization term as the sum of all the squares of the feature weights:

$$L2-Regularization-Term = \|w\|_2^2 = w_1^2 + w_2^2 + [...] + w_n^2$$

In this formula, weights close to zero have little effect on model, while large weights have a much larger impact. This Regularization Term and the loss function can then be minimized to further optimize the neural network.We are essentially preventing the model (weights) from fitting the training data too well, so we can keep the power of generalization.

C. Activation Functions

The purpose of neural activation functions is to convert an input signal into an output signal. Information is received by each neuron and these functions determine how the neurons individually respond.

For our hidden neurons, we used the ReLU (Rectified Linear units) function. Almost all deep learning models utilize the ReLU function because it rectifies the ”vanishing gradient” problem. When its derivative is back-propagated, there will be no degradation of the error signal (because 1 * 1 * [...] * 1 = 1). However, the ReLU activation function still maintains a non-linearity that can act as a ”switch” to turn on and activate the neuron.

Hence, we use the Sigmoid function for the output layer. This function outputs a binary value that represents the potential outcomes, which is ideal for our classification forecasting predictions.

V. Improving Results

An initial run of the final feature set was done with the parameters in Table III with the results of the test set consisting of last two seasons in Table IV. These parameters produced a test set accuracy of 0.59091, with a relative center of the output layer value around 0.505. This is a quite poor prediction accuracy at first glance, but the games in which the model predicted incorrectly warrant further discussion which will be expounded upon after discussing parameter tweaking.

A neural net architecture consisting of two layers of two times the size of the input features was selected to provide a balance between computational complexity and network complexity. More nodes did not impact accuracy, while more layers increased the epochs and time required for results to propagate, which was not acceptable for the time frame of the project.

Any change to ReLU activation for the hidden layer, Sigmoid activation for the output, and Binary Class Entropy for the loss function produced inferior results regardless of the other parameters. As such, these values were not changed for the following results.

Adjusting the mini-batch size generally produced poorer results if increased because too few batches create stronger local minima/maxima, while decreasing batch size to a stochastic level increased the computational cost of the training to a point that it is unviable for the time constraints of this project and produced gradients that did not accurately reflect the overall dataset.

Increasing the regularization constant as noted in Table V produced predictions that were slightly more consistent between permutations of a single game as seen in Table VI, but did not produce any accuracy increase. If the regularization constant is increased too much, the weights are not given enough space to grow and produces results that are more ”random” and the model may pick both teams to win between the two instances relating to the specific game. Lower values shifted the relative ”center” of the model further up, making it more likely to select the second team to win and decreasing accuracy.

The learning rate had little impact on this particular model except for reducing the number of epochs required for stopping the training. A learning rate of 0.1 was determined to be ideal as it allowed for relatively quick results while retaining the predictive accuracy of 0.59091. The results of this combination of parameters can be seen in Table VIII.

The combination of these investigations into each parameter resulted in the final values used for the model, located in Table VII. These parameters provide a strong balance between accuracy and training time for this investigation. Figure 2 shows the comparison between the three major parameter combinations tested.

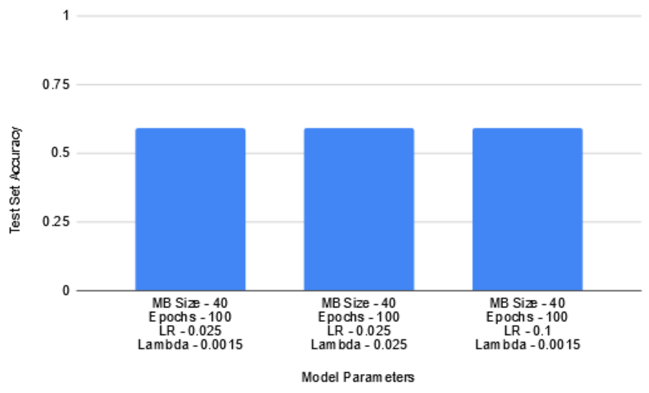

Fig. 2. Test Set Accuracy Versus Specific Parameters - Final Feature Set

VI. Discussion/Conclusion

A. Results

Looking at the model predictions of the last two seasonal playoffs, the first thing to note and a future improvement to be made is that the model struggles to determine a winner for the Super Bowl, where both teams were treated as ”away” teams when the data was gathered although one is designated as a home team at the time of the game for coin toss purposes. As a result the model is unable to predict an outcome for these situations because the ”middle” weight of the model is shifted towards the away team due to the extra data from the Super Bowls. The Sigmoid activation function does not perform well without a well-centered model.

Two notably incorrect predictions made by the model were the NFC and AFC title games from the 2018 season. Both of these games went to overtime and were determined by the single score. In the case of the NFC title game, a critical missed pass interference call against the Rams ended the Saint’s drive on the doorstep of a touchdown drive, resulting in a Rams field goal the following drive that ended the game. The Chiefs-Patriots game was another close game between two very good teams, with critical performances by Patriots receivers swinging the game in their favor. Another incorrect prediction was the Bears-Eagles game in the 2018 season where the Eagles led by Nick Foles, the previous year’s Super Bowl MVP and backup filling in for Carson Wentz again, led another stunning victory in the playoffs. This model does not account for the influence of injuries and individual player performances, meaning that it could not detect that the quarterback is a proven playoffs veteran. The final games of note in the 2018 season is the Colts at the Texans. This game was between two division rivals and the model does not consider how head-to-head records or division play during the season, which may make games that are seemingly not competitive more difficult to predict. Looking at the 2017 playoffs, a notable upset of the Jaguars over the Steelers tripped up the model as well as spectators at the time. The Jaguars were considered inferior to a tried and tested Steelers team, and their stunning 45-42 win was difficult for the model to predict.

B. Improvements

Multiple avenues of improvement to the model presented themselves during the execution of this project.

a) Home/Away distinction in the Super Bowl: Because the Super Bowl has normally been an "away" game for both teams, a third "middle" value could be investigated as an input for a team's type, where an away team has a 0.0 value, a home team has a 1.0 value, and Super Bowl teams have a 0.5 value. This would allow the weights normally associated with the home team to be partially applied to the game outcome and produce a more definitive result from the model.

b) Injury Metadata: Injuries have a potential to completely change the course of a team's season if a star player is removed from the picture. Including data about the number of inactive first-string players at the time of the game may help the model determine which team is more "complete" and in a better position to play at their full potential.

c) Stadium Metadata: NFL Teams play in a wide variety of venues, ranging from open grass fields in the frigid north to enclosed, air-conditioned domes with turf fields in the south. An analysis of how attributes such as altitude, field type, and stadium type in relation to the venue in which the game is played provide an opportunity to explore how teams may benefit or suffer from certain playing conditions.

d) Skill Position Player Metadata: Certain positions in football have a greater individual impact on the game than others. These positions include wide receivers, quarterbacks, running backs, kickers, and cornerbacks. Including how the starting player in these positions on every team performs may help the model correlate how a single skill player contributes to the overall performance of the team on their respective side of the ball.

e) All-Pro/Pro Bowl Players: Every season, a number of players in the league are given special distinction for their performance at their position. Tracking how many of these standout players are present on a team could help decide which team has the edge.

f) Model Performance: The model used has some issues with scaling node depth, node width, and input feature count. Optimizing the algorithms used and utilizing parallelization will allow this model to perform much better when testing the multitude of features and network layers that may provide better results.

C. Continuation of Work

There is a lot of potential for future projects to pivot off of this research.

a) Player-centrism: In mathematical chaos theory, a first principles approach would be best to break down complex group dynamics. A player-centric element to the model using player data could potentially better understand macro team dynamics. Instead of looking at overall team performance, a greater range of interacting agents may better perform for analyzing net behavior.

b) Betting Analysis: Another potential extension of this project could be to look at betting odds to see whether our model could recommend valuable bets to maximize long-term profit.

References

[1] Carson K. Leung, Kyle W. Joseph, Sports Data Mining: Predicting Results for the College Football Games, 18th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, Procedia Computer Science 35 (2014) 710 - 719

[2] Farzin Owramipur, Parinaz Eskandarian, and Faezeh Sadat Mozneb, Football Result Prediction with Bayesian Network in Spanish League-Barcelona Team, International Journal of Computer Theory and Engineering, Vol. 5, No. 5, (2013)

[3] Gianluca Baio and Marta Blangiardo, Bayesian Hierarchical Modelling for the Prediction of Football Results, Seminari del Dipartimento di Statistica, (2009)

[4] Quarterback Rating Formula. NFL-http://www.nfl.com/help/quarterbackratingformula

VII. Appendix

TABLE I
INITIAL FEATURE SET

Attribute Name (t1- attributes repeated for t2-)	attribute values
t1-team-type	home,away
t1-playoff-seed	1,2,3,4,5,6
t1-win-loss-ratio	continuous
t1-points-per-game	continuous
t1-points-allowed-per-game	continuous
t1-turnover-ratio	continuous
t1-offensive-rank	continuous
t1-defensive-rank	continuous
t1-kicking-and-punting-rank	continuous
t1-kicking-and-punting-return-rank	continuous
t1-relative-strength-of-schedule	continuous
t1-penalties-per-game	continuous
t1-penalty-yards-per-game	continuous
t1-rush-yards-per-game	continuous
t1-rush-yards-per-attempt	continuous
t1-rush-touchdowns-per-game	continuous
t1-pass-yards-per-game	continuous
t1-pass-yards-per-attempt	continuous
t1-receiving-touchdowns-per-game	continuous
t1-1st-downs-per-game	continuous
t1-3rd-down-conversions-pct	continuous
t1-4th-down-conversions-pct	continuous
t1-sacks-allowed-per-game	continuous
t1-fumbles-per-game	continuous
t1-fumbles-lost-per-game	continuous
t1-interceptions-thrown-per-game	continuous
t1-starting-qb-passer-rating	continuous
t1-rush-yards-allowed-per-game	continuous
t1-rush-yards-per-attempt-allowed	continuous
t1-rush-touchdowns-allowed-per-game	continuous
t1-pass-yards-allowed-per-game	continuous
t1-pass-yards-allowed-per-attempt-allowed	continuous
t1-receiving-touchdowns-allowed-per-game	continuous
t1-1st-downs-allowed-per-game	continuous
t1-3rd-down-conversions-allowed-pct	continuous
t1-4th-down-conversions-allowed-pct	continuous
t1-sacks-per-game	continuous
t1-forced-fumble-recoveries-per-game	continuous
t1-interceptions-per-game	continuous
t1-defensive-touchdowns-per-game	continuous
t1-kicker-field-goal-pct	continuous
t1-kicker-pat-pct	continuous
t1-avg-yardage-per-punt	continuous
t1-percent-blocked-punts	continuous
t1-opponent-avg-yardage-per-punt	continuous
t1-opponent-percent-blocked-punts	continuous
t1-kick-return-yards-per-return	continuous
t1-kick-return-touchdowns-per-return	continuous
t1-opponent-kick-return-yards-per-return	continuous
t1-opponent-kick-return-touchdowns-per-return	continuous
t1-punt-return-yards-per-return	continuous
t1-punt-return-touchdowns-per-return	continuous
t1-opponent-punt-return-yards-per-return	continuous
t1-opponent-punt-return-touchdowns-per-return	continuous
t1-stadium-altitude	continuous
t1-stadium-latitude	continuous
t1-stadium-longitude	continuous
t1-stadium-type	open,dome
t1-stadium-field-type	grass, turf
game-stadium-altitude	continuous
game-stadium-latitude	continuous
game-stadium-longitude	continuous
game-stadium-type	open,dome
game-stadium-field-type	grass,turf
game-result	t1,t2

TABLE II
FINAL FEATURE SET

Attribute Name (t1- attributes repeated for t2-)	attribute values
t1-team-type	home,away
t1-win-loss-ratio	continuous
t1-points-per-game	continuous
t1-points-allowed-per-game	continuous
t1-turnover-ratio	continuous
t1-relative-strength-of-schedule	continuous
t1-penalties-per-game	continuous
t1-penalty-yards-per-game	continuous
t1-rush-yards-per-game	continuous
t1-rush-yards-per-attempt	continuous
t1-rush-touchdowns-per-game	continuous
t1-pass-yards-per-game	continuous
t1-pass-yards-per-attempt	continuous
t1-receiving-touchdowns-per-game	continuous
t1-1st-downs-per-game	continuous
t1-3rd-down-conversions-pct	continuous
t1-4th-down-conversions-pct	continuous
t1-sacks-allowed-per-game	continuous
t1-fumbles-per-game	continuous
t1-fumbles-lost-per-game	continuous
t1-interceptions-thrown-per-game	continuous
t1-rush-yards-allowed-per-game	continuous
t1-rush-yards-per-attempt-allowed	continuous
t1-rush-touchdowns-allowed-per-game	continuous
t1-pass-yards-allowed-per-game	continuous
t1-pass-yards-allowed-per-attempt-allowed	continuous
t1-receiving-touchdowns-allowed-per-game	continuous
t1-1st-downs-allowed-per-game	continuous
t1-3rd-down-conversions-allowed-pct	continuous
t1-4th-down-conversions-allowed-pct	continuous
t1-sacks-per-game	continuous
t1-forced-fumble-recoveries-per-game	continuous
t1-interceptions-per-game	continuous
t1-defensive-touchdowns-per-game	continuous
t1-kicker-field-goal-pct	continuous
t1-kicker-pat-pct	continuous
t1-avg-yardage-per-punt	continuous
t1-percent-blocked-punts	continuous
t1-opponent-avg-yardage-per-punt	continuous
t1-opponent-percent-blocked-punts	continuous
t1-kick-return-yards-per-return	continuous
t1-kick-return-touchdowns-per-return	continuous
t1-opponent-kick-return-yards-per-return	continuous
t1-opponent-kick-return-touchdowns-per-return	continuous
t1-punt-return-yards-per-return	continuous
t1-punt-return-touchdowns-per-return	continuous
t1-opponent-punt-return-yards-per-return	continuous
t1-opponent-punt-return-touchdowns-per-return	continuous
game-result	t1,t2

TABLE III
MODEL PARAMETERS - ITERATION 1

Parameter	Value
Minibatch Size	40
Epochs to wait for improvement	100
Learning Rate	0.025
Regularization Constant	0.0015
Hidden Layer Activation Function	ReLU
Output Layer Activation Function	Sigmoid
Loss Function	Binary Cross Entropy

TABLE IV
MODEL RESULTS - ITERATION 1

Season	**Team 1**	**Team 2**	**Output Layer Value**	**Model Prediction**	**Actual Result**
2017	Falcons	Rams	0.51092	Rams	Falcons
2017	Rams	Falcons	0.49891	Rams	Falcons
2017	Titans	Chiefs	0.51106	Chiefs	Titans
2017	Chiefs	Titans	0.49931	Chiefs	Titans
2017	Jaguars	Bills	0.49865	Jaguars	Jaguars
2017	Bills	Jaguars	0.51143	Jaguars	Jaguars
2017	Saints	Panthers	0.49977	Saints	Saints
2017	Panthers	Saints	0.51026	Saints	Saints
2017	Patriots	Titans	0.49846	Patriots	Patriots
2017	Titans	Patriots	0.51156	Patriots	Patriots
2017	Eagles	Falcons	0.49833	Eagles	Eagles
2017	Falcons	Eagles	0.51150	Eagles	Eagles
2017	Vikings	Saints	0.49864	Vikings	Vikings
2017	Saints	Vikings	0.51115	Vikings	Vikings
2017	Jaguars	Steelers	0.51077	Steelers	Jaguars
2017	Steelers	Jaguars	0.49940	Steelers	Jaguars
2017	Patriots	Jaguars	0.49996	Patriots	Patriots
2017	Jaguars	Patriots	0.51024	Patriots	Patriots
2017	Eagles	Vikings	0.49921	Eagles	Eagles
2017	Vikings	Eagles	0.51000	Eagles	Eagles
2017	Eagles	Patriots	0.50442	Patriots	Eagles
2017	Patriots	Eagles	0.50607	Eagles	Eagles
2018	Colts	Texans	0.51085	Texans	Colts
2018	Texans	Colts	0.49942	Texans	Colts
2018	Cowboys	Seahawks	0.49973	Cowboys	Cowboys
2018	Seahawks	Cowboys	0.51064	Cowboys	Cowboys
2018	Eagles	Bears	0.51216	Bears	Eagles
2018	Bears	Eagles	0.49803	Bears	Eagles
2018	Chargers	Ravens	0.51080	Ravens	Chargers
2018	Ravens	Chargers	0.49943	Ravens	Chargers
2018	Chiefs	Colts	0.49992	Chiefs	Chiefs
2018	Colts	Chiefs	0.51012	Chiefs	Chiefs
2018	Rams	Cowboys	0.49981	Rams	Rams
2018	Cowboys	Rams	0.51038	Rams	Rams
2018	Saints	Eagles	0.49862	Saints	Saints
2018	Eagles	Saints	0.51135	Saints	Saints
2018	Patriots	Chargers	0.49924	Patriots	Patriots
2018	Chargers	Patriots	0.51049	Patriots	Patriots
2018	Patriots	Chiefs	0.51025	Chiefs	Patriots
2018	Chiefs	Patriots	0.49972	Chiefs	Patriots
2018	Rams	Saints	0.51107	Saints	Rams
2018	Saints	Rams	0.49921	Saints	Rams
2018	Patriots	Rams	0.50524	Rams	Patriots
2018	Rams	Patriots	0.50573	Patriots	Patriots

TABLE V
MODEL PARAMETERS - ITERATION 2

Parameter	**Value**
Minibatch Size	40
Epochs to wait for improvement	100
Learning Rate	0.025
Regularization Constant	0.025
Hidden Layer Activation Function	ReLU
Output Layer Activation Function	Sigmoid
Loss Function	Binary Cross Entropy

TABLE VI
MODEL RESULTS - ITERATION 2

Season	**Team 1**	**Team 2**	**Output Layer Value**	**Model Prediction**	**Actual Result**
2017	Falcons	Rams	0.51043	Rams	Falcons
2017	Rams	Falcons	0.49892	Rams	Falcons
2017	Titans	Chiefs	0.51055	Chiefs	Titans
2017	Chiefs	Titans	0.49929	Chiefs	Titans
2017	Jaguars	Bills	0.49864	Jaguars	Jaguars
2017	Bills	Jaguars	0.51093	Jaguars	Jaguars
2017	Saints	Panthers	0.49973	Saints	Saints
2017	Panthers	Saints	0.50980	Saints	Saints
2017	Patriots	Titans	0.49851	Patriots	Patriots
2017	Titans	Patriots	0.51105	Patriots	Patriots
2017	Eagles	Falcons	0.49838	Eagles	Eagles
2017	Falcons	Eagles	0.51099	Eagles	Eagles
2017	Vikings	Saints	0.49867	Vikings	Vikings
2017	Saints	Vikings	0.51065	Vikings	Vikings
2017	Jaguars	Steelers	0.51025	Steelers	Jaguars
2017	Steelers	Jaguars	0.49937	Steelers	Jaguars
2017	Patriots	Jaguars	0.49991	Patriots	Patriots
2017	Jaguars	Patriots	0.50977	Patriots	Patriots
2017	Eagles	Vikings	0.49922	Eagles	Eagles
2017	Vikings	Eagles	0.50956	Eagles	Eagles
2017	Eagles	Patriots	0.50419	Patriots	Eagles
2017	Patriots	Eagles	0.50577	Eagles	Eagles
2018	Colts	Texans	0.51034	Texans	Colts
2018	Texans	Colts	0.49940	Texans	Colts
2018	Cowboys	Seahawks	0.49968	Cowboys	Cowboys
2018	Seahawks	Cowboys	0.51013	Cowboys	Cowboys
2018	Eagles	Bears	0.51163	Bears	Eagles
2018	Bears	Eagles	0.49806	Bears	Eagles
2018	Chargers	Ravens	0.51029	Ravens	Chargers
2018	Ravens	Chargers	0.49939	Ravens	Chargers
2018	Chiefs	Colts	0.49986	Chiefs	Chiefs
2018	Colts	Chiefs	0.50964	Chiefs	Chiefs
2018	Rams	Cowboys	0.49975	Rams	Rams
2018	Cowboys	Rams	0.50991	Rams	Rams
2018	Saints	Eagles	0.49864	Saints	Saints
2018	Eagles	Saints	0.51086	Saints	Saints
2018	Patriots	Chargers	0.49923	Patriots	Patriots
2018	Chargers	Patriots	0.51000	Patriots	Patriots
2018	Patriots	Chiefs	0.50978	Chiefs	Patriots
2018	Chiefs	Patriots	0.49968	Chiefs	Patriots
2018	Rams	Saints	0.51055	Saints	Rams
2018	Saints	Rams	0.49920	Saints	Rams
2018	Patriots	Rams	0.50426	Rams	Patriots
2018	Rams	Patriots	0.50470	Patriots	Patriots

TABLE VII
MODEL PARAMETERS - ITERATION 3

Parameter	**Value**
Minibatch Size	40
Epochs to wait for improvement	100
Learning Rate	0.1
Regularization Constant	0.025
Hidden Layer Activation Function	ReLU
Output Layer Activation Function	Sigmoid
Loss Function	Binary Cross Entropy

TABLE VIII
MODEL RESULTS - ITERATION 3

Season	Team 1	Team 2	Output Layer Value	Model Prediction	Actual Result
2017	Falcons	Rams	0.51021	Rams	Falcons
2017	Rams	Falcons	0.49879	Rams	Falcons
2017	Titans	Chiefs	0.51034	Chiefs	Titans
2017	Chiefs	Titans	0.49918	Chiefs	Titans
2017	Jaguars	Bills	0.49852	Jaguars	Jaguars
2017	Bills	Jaguars	0.51076	Jaguars	Jaguars
2017	Saints	Panthers	0.49960	Saints	Saints
2017	Panthers	Saints	0.50958	Saints	Saints
2017	Patriots	Titans	0.49839	Patriots	Patriots
2017	Titans	Patriots	0.51087	Patriots	Patriots
2017	Eagles	Falcons	0.49826	Eagles	Eagles
2017	Falcons	Eagles	0.51079	Eagles	Eagles
2017	Vikings	Saints	0.49856	Vikings	Vikings
2017	Saints	Vikings	0.51042	Vikings	Vikings
2017	Jaguars	Steelers	0.51003	Steelers	Jaguars
2017	Steelers	Jaguars	0.49925	Steelers	Jaguars
2017	Patriots	Jaguars	0.49977	Patriots	Patriots
2017	Jaguars	Patriots	0.50954	Patriots	Patriots
2017	Eagles	Vikings	0.49911	Eagles	Eagles
2017	Vikings	Eagles	0.50935	Eagles	Eagles
2017	Eagles	Patriots	0.50400	Patriots	Eagles
2017	Patriots	Eagles	0.50556	Eagles	Eagles
2018	Colts	Texans	0.51010	Texans	Colts
2018	Texans	Colts	0.49925	Texans	Colts
2018	Cowboys	Seahawks	0.49953	Cowboys	Cowboys
2018	Seahawks	Cowboys	0.50988	Cowboys	Cowboys
2018	Eagles	Bears	0.51146	Bears	Eagles
2018	Bears	Eagles	0.49793	Bears	Eagles
2018	Chargers	Ravens	0.51005	Ravens	Chargers
2018	Ravens	Chargers	0.49924	Ravens	Chargers
2018	Chiefs	Colts	0.49967	Chiefs	Chiefs
2018	Colts	Chiefs	0.50936	Chiefs	Chiefs
2018	Rams	Cowboys	0.49959	Rams	Rams
2018	Cowboys	Rams	0.50965	Rams	Rams
2018	Saints	Eagles	0.49848	Saints	Saints
2018	Eagles	Saints	0.51062	Saints	Saints
2018	Patriots	Chargers	0.49907	Patriots	Patriots
2018	Chargers	Patriots	0.50976	Patriots	Patriots
2018	Patriots	Chiefs	0.50949	Chiefs	Patriots
2018	Chiefs	Patriots	0.49951	Chiefs	Patriots
2018	Rams	Saints	0.51028	Saints	Rams
2018	Saints	Rams	0.49903	Saints	Rams
2018	Patriots	Rams	0.50403	Rams	Patriots
2018	Rams	Patriots	0.50449	Patriots	Patriots