NRR Regularised Auto-Encoders for Player Embeddings in Cricket

Somesh Singh[1][0000−0003−4762−3785], Arshika Lalan[1][0000−0002−8176−2431], Nidhi Zare[1][0000−0002−4082−240X], and Nikhil Bisht[1][0000−0002−7557−1890]

Birla Institute of Technology and Science, Pilani, K.K. Birla Goa Campus, NH-17B, Zuarinagar, Goa
https://universe.bits-pilani.ac.in/Goa/

Abstract. Cricket betting is a multi-billion dollar industry. Therefore, there is a strong incentive for models that can predict games' outcomes and beat the odds provided by betters. Many computational methods predict a team's performance or player using essential features of a match like the toss, venue, and previous outcomes. However, these models consider outcomes solely based on a team's previous performance, instead of the constituent players. Therefore, they fail when rosters change significantly (like World XI). We first model these indicators and past performances to get an accuracy of 82% and introduce a regularised autoencoder-based approach to generate on the fly player/team embeddings, which considers players' previous performance to predict the outcomes of a match with a comparable accuracy (76%). We also predict the loss's margin in terms of Net Run Rate (NRR) instead of win/lose to get a more confident measure.

Keywords: Cricket · ODI · Machine Learning · Auto-Encoders · Net Run Rate · Sports Analytic.

1 Introduction

Cricket is arguably the most popular sport in India. Globally, more than 100 member states of the International Cricket Council (ICC) are a part of it. Millions of people tune into the live streams of tournaments, exhibiting a passion for their team and the sport. The game comprises 13 player teams, where two players are extras, and the remaining 11 are an assortment of batsmen, bowlers, and all-rounders. Typically, national-level playing teams have much more than 13 players playing for them and need to pick the right players to improve their winning chances carefully.

Various factors, such as the venue, home field, and toss, also come into play. Many internal factors, such as the past performances of players, also need to be considered. Given the game's global scale, teams and their analysts must consider all these variables while devising strategies to win the game. A robust win predictor, thus, gains utmost importance in the sport. The literature has mostly considered the history of outcomes between different teams, and these

match deciding factors. Mean Stats of players are used to model the strength of two teams and predict the outcome. In this paper, we discuss a regularised Auto-Encoder based approach to predict the outcome of the matches using the consistency and form of a player and a method to obtain the latent representation of a player or team in the form of embeddings.

Machine Learning techniques have recently gained popularity in sports. A number of approaches have been proposed in the literature for predicting the outcome of Cricket matches [3] [5] [6]. The literature has treated outcome prediction as a classification problem, only focusing on predicting the match-winner. However, it is equally important to check the winning team's dominance on the match and account for this; the paper follows a unique approach to predicting the scores and the net run rate for the team and using these values to predict the winner. Net run rate is a statistical method used in analyzing teamwork and performance in cricket. A team's net run rate is calculated by deducting from the average runs per over scored by that team throughout the competition. The average runs per over scored against that team throughout the competition. To the best of our knowledge, our contribution is novel, and we hope to guide research forward in this field.

2 Related Work

The literature on outcome prediction in the game of cricket is limited. Few researchers have studied the performance of cricket players. *Passi et al.* [6] predict the players' performance in One Day Cricket (ODI) by analyzing their characteristics using supervised learning techniques. *Kampakis et al.* [3] study the degree to which it is possible to predict the outcome of the cricket matches in the English T20 cricket world cup. *Dixit et al.* [2] compare three different Convolutional Neural Networks architectures on the task of classifying ball-by-ball outcomes for cricket videos, which was a novel approach in the domain of action recognition and obtained almost 80% accuracy on the validation dataset.

Some researchers have also used team-based analysis for predicting the match outcome. *Jhawar et al.* [5] uses the team combination as a factor in deciding the match outcomes. *Manivannan et al.* [4] uses the individual player performance of batsmen and bowlers for prediction. It includes taking into account the strike rate, runs scored, bowling average, and players' consistency. The features are encoded, and the output is predicted using CNN.

3 Dataset

3.1 Data Collection

We have crawled all the ODI Matches between 1971 and 2020 from [**?**] [1] using the Python requests and Beautiful-Soup [2] packages for collecting and parsing

[1] http://www.howstat.com/cricket
[2] https://pypi.org/project/beautifulsoup4/

them. Rain has been a spoilsport since ever in cricket; rain curtailed matches have used different methodology (Duckward Lewis is the latest method) to exclude all abandoned or curtailed matches. The matches aggregate to 3656; previous works that use individual player data have collected match by match data from Leagues such as IPL (Indian Premier League) or CWCs (International Cricket World Cups), where the data is limited. Associate teams like Scotland, Afghanistan, Ireland frequently miss out. Therefore we build our dataset for players and grounds using the performances beginning from the match on 5 January 1971 between Australia and England. Additionally, we collect all player attributes (Dexterity, Pace/Spin), which were available aggregating to 2132 players. We have only considered teams that have played the ODI-WC at least once after 2000 and played a total of 50 matches. The teams include Afghanistan (AFG), Australia (AUS), Bangladesh (BAN), England(ENG), India(IND), Ireland (IRE), Kenya (KEN), New Zeland (NZL), Pakistan (PAK), Scotland (SCO), South Africa (SAF), Sri Lanka (SRL), West Indies (WIN), and Zimbabwe (ZIM).

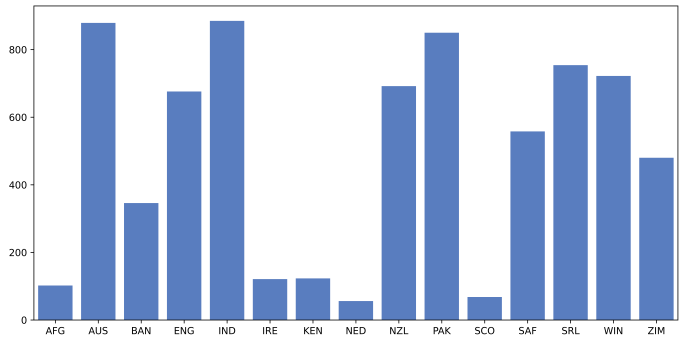

Fig. 1. Number of ODI matches played by current full time members

3.2 Data Pre-processing

The teams are label encoded from 0-14 (In ascending order of their names), Batsmen info per match scorecard includes the following: Dismissal, Runs scored, Balls Faced, 4s, 6s, Strike Rate (Runs per 100 balls), Contribution (Runs scored by batsmen / Runs Scored by the team). For Bowlers, Overs Bowled, Maidens, Wickets, Runs, Wicket Contribution, additionally we also scrape match details like Venue (Home/Away/Neutral), Ground, Batting Totals, Date.

Referring to previous work, we consider the following features to be important indicators for player performance, Batting: Batting Average (Average Runs per Innings), Strike Rate (Runs per 100 balls), Boundaries (Weighted total of 4s, 6s), Average Contribution. Bowling: Bowling Average (Average Runs conceded

per wicket), Economy (Average runs conceded per over), Bowling Strike Rate (Average Overs Taken for wickets).

All stats are min-max scaled, player Attributes are One-Hot Encoded, and debut players are replaced with zeros. For player performance prediction, we make the model predict the runs, strike rate, batting/bowling contributions, overs bowled, wickets taken, runs conceded, bowling contribution.

We replace the Win/Loss by Net Run Rate, which is the difference of run rates for both teams. For Indicator based models (Toss, Venue, Ground, Team) we have one hot encoded Toss and Teams, and label encoded Venue and Ground.

4 Experiments

4.1 Baseline

For this model, the provided input is more complicated and realistic as compared to the indicator based model. We take the following as the input:

- Team 1, Team 2
- Venue (more specifically - Home, Away or Neutral)
- Date of Match
- Team 1 and Team 2 Roster

Toss generation is random. We extract the matches played between the two teams in the past five years. We then use this data to extract match statistics including 4s, 6s, Strike Rate of each batsman and number of Maiden Overs bowled, number of Wickets taken, and Economy Rate of each bowler and use this data.

Once we have the dataset ready, we need to create the input based on the Team Rosters since we are using statistics like boundaries and averages. Consistency and form are different parameters used to predict a player and team's performance. The past few innings weigh the stats model's consistency at that point of time and form. We achieve this by using an exponentially weighted average (considered as the form) for each player. We take a player, extract all the player's matches in the last five years (irrespective of the opposing team), and use a weighted average to calculate each statistic mentioned above by giving the latest matches the highest weight. We get this for all players playing the prediction match and get an input vector for our model, which we use to predict the outcome.

Model Training and accuracy We input the vector containing batsmen and bowlers' performance to the model used to predict the match's outcome. Different models were used to check the outcome- decision tree regression, random forest regression, and linear regression. In the second approach, we used only the strike rate for batsmen and wickets and the economy rate for bowlers, which improved the results.

- Model 1 - Players Performance : The decision tree yield an accuracy of 50 %
- Model 2 - Players Performance : The decision tree yield an accuracy of 70 %

4.2 Decision Tree Regressor

The data from matches, venues and regions is combined to create the final dataset where the venues, regions and teams are ordinally labeled. Following this, team names were label encoded and toss and win features were one-hot encoded. The dataset was split into train and test data with a split ratio of 80:20

Analysis of features We did analysis of the following features:

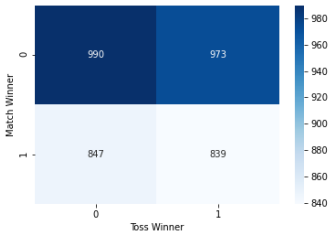

Fig. 2. Confusion matrix between match winner and toss winner

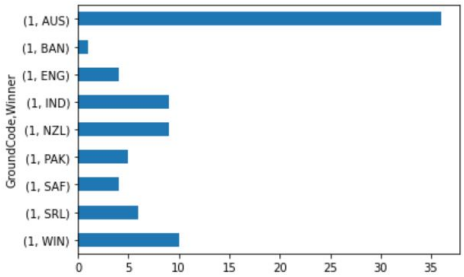

Fig. 3. There is a clear correlation between Australia playing a game in ground 1 and Australia winning. similarly Bangladesh is unable to perform well when playing in Ground 1

- TOSS: There is a small correlation between winning the toss and winning the match (The winning team had also won the toss approximately 53% of the time).
- GROUND: Some correlation was also found between the ground and winning team.
  We also used causal inference to confirm that the ground on which the match was played does in-fact influence the winning team. It was found that the Transfer Entropy value for the Winning Team given the ground was around 0.4. This indicates a small but definite casual relationship.
- VENUE: It was seen that the winning team played on the home turf approximately 30% of the time.

Thus, these features were taken to be relevant in analyzing the game. Several experiments were conducted after the data analysis.

Experiment 1 - Modelling the data on scores The data from both teams was first used to predict the scores for both teams. The scores were then compared, and the team with the more massive score was declared the winner. Thus, the first part of the modeling was a regression problem. Simultaneously, the accuracy was calculated based on the classification problem of the number of correct winners predicted by the model.

The models used for this experiment were Random Forest Regression, Linear Regression, Decision Tree Regression, and LassoCV. The Decision Tree Regressor was found to perform the best out of all the models.

- Model 1 - TOSS + Ground: The decision tree regressor yielded an accuracy of around 82%
- Model 2 - TOSS + Ground + Venue: The decision tree regressor yielded an accuracy of around 82%

Experiment 2 - Modelling the data on NRR It was found that while the model in Experiment 1 could identify the winner quite accurately, it suffered from high RMS values. One of the reasons for this is that the chasing team always wins by one run (or a similarly small amount), while the team that set the target would usually win by a much more resounding victory. This, in a sense, confused the model, and modeling on scores was found to be ineffective in this regard.

One solution to this can be modeled on the Net Run Rate (NRR hereafter) rather than the two teams' scores. The net run rate of a team is the run rate of that team minus the opposing team's run rate. Several conditions, such as whether the team went all out or not, have to be considered while calculating the net run rate. The net run rate was calculated for both the teams and added to the dataset.

The data from both teams was first used to predict the NRR for both teams. The team with the positive NRR value was declared the winner. Thus, the first part of the modeling was a regression problem. At the same time, the accuracy was calculated based on the classification problem of the number of correct winners predicted by the model.

The models used for this experiment were Random Forest Regression, Linear Regression, Decision Tree Regression, and LassoCV. The Decision Tree Regressor was found to perform the best out of all the models.

- Model 1 - TOSS + Ground: The decision tree regressor yielded an accuracy of around 79.9% . Improved RMS values were seen as compared to Experiment 1
- Model 2 - TOSS + Ground + Venue: The decision tree regressor yielded an accuracy of around 79.9% . Improved RMS values were seen as compared to Experiment 1

Experiment 3 - Reversing the dataset We swapped the order of teams, flipped the toss and winners and negate the NRR to make sure that the model was not learning from the order of teams.

The models used for this experiment were Random Forest Regression, Linear Regression, Decision Tree Regression and LassoCV. The Decision Tree Regressor was found to perform the best out of all the models.

1. Experiment 3.1 : Modelling on scores

- Model 1 - TOSS + Ground: The decision tree regressor yielded an accuracy of around 82%
- Model 2 - TOSS + Ground + Venue: The decision tree regressor yielded an accuracy of around 82% . Improved RMS values were seen as compared to Experiment 3.1

2. Experiment 3.2 : Modelling on NRR
   - Model 1 - TOSS + Ground: The decision tree regressor yielded an accuracy of around 79.5%
   - Model 2 - TOSS + Ground + Venue: The decision tree regressor yielded an accuracy of around 79.7% . Improved RMS values were seen as compared to Experiment 3.1

4.3 Regularized AutoEncoders

The motivation behind AutoEncoders is to obtain a latent unsupervised representation of the data [9, **?**]. For this experiment we have ignored players who have played less than 5 matches in their career and only considered matches after 1 January 1980 to get players to have some stat value before reconstruction. The data was split randomly 90-10, The MSE (Mean Squared Error) Loss for Player performance, Reconstruction, and NRR was monitored with early stopping. The Loss can be written as:

$$\lambda = \alpha_0 \lambda_{reconstruction} + \alpha_1 \lambda_{NRR} + \alpha_2 \lambda_{player} \tag{1}$$

$$\lambda_{reconstruction} = \frac{1}{N} \sum_{i=1}^{N} \|X_{input} - X_{reconstruct}\|^2 \tag{2}$$

$$\lambda_{player} = \frac{1}{N} \sum_{i=1}^{N} \|X_{performance} - X_{predicted}\|^2 \tag{3}$$

$$\lambda_{player} = \frac{1}{N} \sum_{i=1}^{N} \|X_{NRR} - X_{predicted}\|^2 \tag{4}$$

Player Embedding Generation An autoencoder is not a classifier or regressor, it is a nonlinear feature extraction technique. This is a dimensionality reduction technique, which is basically used before classification of high dimensional dataset to remove the redundant information from the data. Autoencoder architecture are also known as nonlinear generalization of Principal Component Analysis. Autoencoder is a neural network (also can be used as a tool to learn deep neural network) which is trained to replicate its input (i.e. original dataset) to its output, through encoding and decoding operation; and the encoded dataset are used as dimensionally reduced features.

Formally, An auto-encoder consists of two parts, the encoder and the decoder, which can be defined as transitions ϕ and ψ such that:

$$\phi : \chi \rightarrow F \tag{5}$$

$$\psi : F \rightarrow \chi \tag{6}$$

$$\phi, \psi = arg_{\phi,\psi} min \left\| X - (\psi \odot \phi) X) \right\|^2 \tag{7}$$

Inspired by [7] we also add small Gaussian noise to the inputs with a $mean = 0$, and $std = 0.1$ to obtain a robust representation.

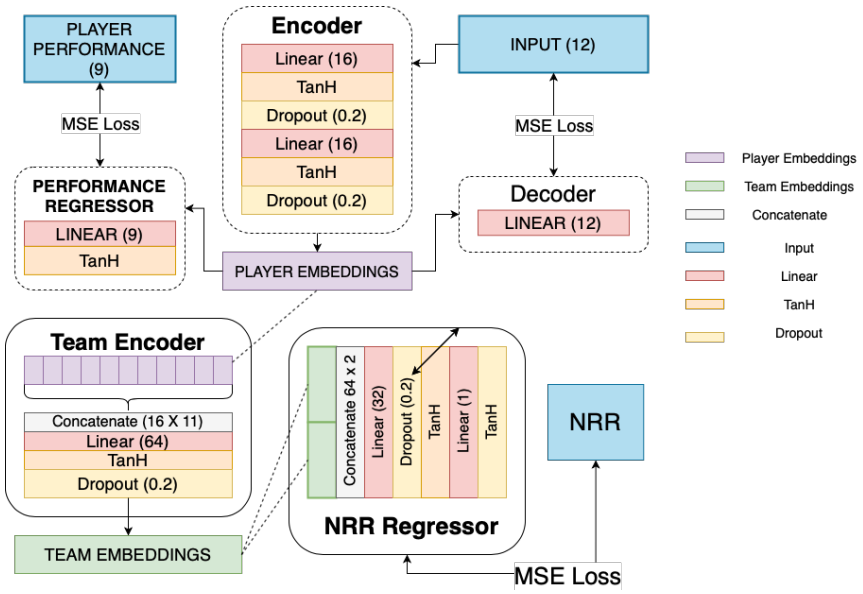

Fig. 4. Encoder-Decoder Set

Team Embeddings The Player Embeddings for all players of a team are concatenated and are fully connected two two Linear Layers, the output is used as a representation of the team and is used to predict the Net Run Rate.

All non final layers have a Dropout (0.2) added to them to prevent over fitting and TanH activation. We have used RMSProp with learning rate of 1e-3, α_0, α_1, α_2 have not been fine-tuned, to prevent ofverfitting each

5 Results and Discussion

It can be seen from all the experiments that modelling the data over NRR as opposed to scores leads to significantly better RMS value and comparable, if not better, overall accuracies. The classification reports for Experiments 1 to 3 are given in Table 1. and Table 2.

	Based on Scores:				**Based on NRR:**			
	precision	**recall**	**f1-score**	**support**	**precision**	**recall**	**f1-score**	**support**
Model 1*	**0.78**	**0.82**	**0.80**	**437**	**0.75**	**0.81**	**0.78**	**437**
Model 2	0.68	0.61	0.64	266	0.63	0.54	0.58	266
Accuracy			0.74	703			0.71	703
Macro avg	0.73	0.72	0.72	703	0.46	0.45	0.45	703
Weighted avg	0.74	0.74	0.74	703	0.70	0.71	0.70	703

Table 1. Classification report Australia vs All (Team vs All)

	Based on Scores:				**Based on NRR:**			
	precision	**recall**	**f1-score**	**support**	**precision**	**recall**	**f1-score**	**support**
Model 1*	0.70	**0.88**	**0.78**	**52**	0.69	**0.88**	**0.77**	**52**
Model 2	**0.83**	0.60	0.70	50	**0.83**	0.58	0.68	50
Accuracy			0.75	102			0.74	102
Macro avg	0.77	0.74	0.74	102	0.76	0.73	0.73	102
Weighted avg	0.76	0.75	0.74	102	0.76	0.74	0.73	102

Table 2. Classification report Australia vs West Indies (Team vs Team)

From the classification reports we can see that while both the models can act as decent win predictors, Model 1 offers great precision values, recall values, and f1-scores.

One of the significant reasons for the failure of our Auto-encoder based model is missing and sparse data. Usually, betters/critics consider previous performances in leagues/county matches and other one-day matches; since our approach is invariant to team-relationships and uses previous matches as the indicator for performance and, in turn, the total representation of the team, we can use metadata.

Amongst the cases where the difference between the ground truth and predicted NRR has an absolute difference of 75% (0.75 or above), 91% teams had substitutes or debut players. On the other hand, we see our model is 13% more accurate when handling matches where each player has played > 10 innings.

Auto Encoder Models	**Reconstruction**	**Player**	**NRR**	**Accuracy(%)**
Gaussian Noise	**0.009**	**0.026**	**0.008**	**76.09**
No Gaussian Noise	0.013	0.031	0.013	68.7
Without Player	0.001	-	0.016	62.1

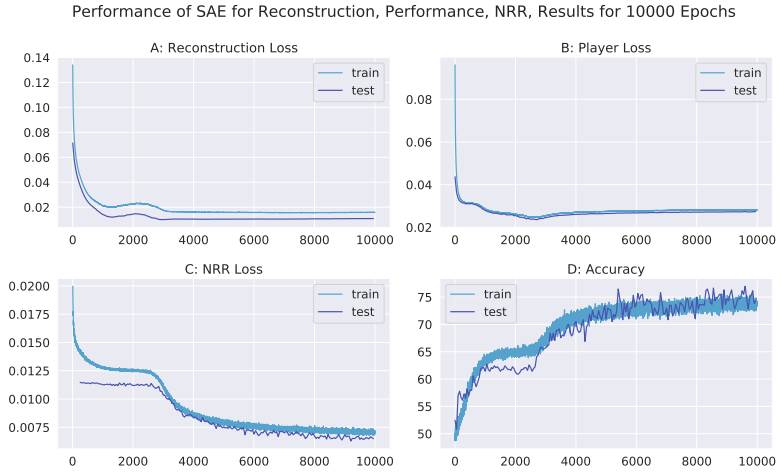

Fig. 5. Encoder-Decoder Set

We obtained best results at 8200 epochs

6 Conclusion

It is observed that by using indicators such as net run rate for modelling, an accuracy of approximately 82% can be achieved. We show that models can improve their performance by predicting the domination of wins/losses and getting a better understanding of the underlying distribution in previous outcomes. Meanwhile, we were able to use autoencoder based deep learning methods to obtain the representation of players and teams, which are enough to predict outcomes (76%) and can lead research to build better rosters and help coaches check other changes, including introducing experienced/young players, Left/Right coordination, Batting Orders, and focusing on form vs. consistency.

7 Future Work

Averaging is not the best indicator for an individual's performance since players perform according to their consistency and form. We tried a weighted average/moving window to represent the form. However, it leads to even more missing data points and worse results. Instead, the player encoder can use RNN based Networks, which have shown the state of the art results in many series prediction tasks.

Additional parameter that can be used is performance of fielders along with batsmen and bowlers. We used regex to extract fielders information from bats-

men dismissal. This information can further be modeled to account for performance of the fielders in the team as well.

We hope our work motivates research in machine learning and sport analytic to use domination based predictions and roster based models for better results with given confidence.

8 Acknowledgement

We'd like to humbly acknowledge the professors and teaching assistants of the BITS F464 course. This project would not have been possible without their constant guidance and support.
Teachers: Dr. Ashwin Srinivasan, Dr. Tirthiraj Dash
Mentors: Het Shah, Aditya Ahuja and Avishree Khare

References

1. Howstat Website, http://www.howstat.com/cricket. Last accessed 15 Nov 2020
2. K. Dixit et. al.: Deep Learning using CNNs for Ball-by-Ball Outcome Classification in Sports, University of Stanford (2016)
3. S. Kampakis et. al.: Using Machine Learning to Predict the Outcome of English County twenty over Cricket Matches, University College London (2016)
4. S. Manivannan and M. Kausik.: Convolutional Neural Network and Feature Encoding for Predicting the Outcome of Cricket Matches, In: 14th IEEE International Conference on Industrial and Information System (ICIIS) 2019
5. M. Jhawar and V. Pudi:Predicting the Outcome of ODI Cricket Matches: A Team Composition Based Approach, In: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD) 2016
6. K. Passi and N. Pandey.: INCREASED PREDICTION ACCURACY IN THE GAME OF CRICKET USING MACHINE LEARNING. In: International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.8, No.2, March 2018
7. Vaswani, A., Ganguly, R., Shah, H., SharanRanjit, S., Pandit, S., & Bothara, S. (2020). An Autoencoder Based Approach to Simulate Sports Games. ArXiv, abs/2007.10257.
8. Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., & Zhang, C. (2018). Adversarially regularized graph autoencoder for graph embedding. arXiv preprint arXiv:1802.04407.
9. Yu, W., Zeng, G., Luo, P., Zhuang, F., He, Q., & Shi, Z. (2013, September). Embedding with autoencoder regularization. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 208-223). Springer, Berlin, Heidelberg.