Discerning Football Scenarios using Sentence Embeddings

Ali Hariri
Mechanical Engineering Department
American University of Beirut
Beirut, Lebanon
aah71@mail.aub.edu

***Abstract*—Sports analytics has taken a new shape with the rise in Big Data Analytics. Technological advancements paved the way for a better data extraction from sports, allowing a better understanding of the patterns present by its competing athletes. While most papers shed light on raw numerical data in their analysis of a sport, the textual data remains under-explored despite its abundance and high information density. This paper proposes a method to label text commentaries as scenarios taking place in the game.**

Keywords—Natural Language Processing, Professional Footbal, Neural Networks, doc2vec.

I. INTRODUCTION

Sports is a lucrative sector that has seen a massive rise in investments in the 21st century, with estimated annual revenue of $91 billion, including $28.7 billion from the European football market alone [1]. That being said, Football is currently the most popular sport worldwide, with an estimated 4 billion fans. Long known of having "working-class roots", the financial side of the game was notably shaped in the past decades through the involvement of major corporations. Banks, broadcasting companies and many other Billion-dollar firms rush to make sponsorship deals in response to the rising popularity of the game, which increased with the rising exposure from social media platforms, spreading the sports even further around the globe.

Historically, sports betting have been part of football for several years. Despite its threat to the rules of the game, betting is now regularized and legal in several parts of the world, hence making the sports worth even more money. As a result, several companies have engaged in sports analytics and made it their core business to analyze games and predict their outcomes. In our project, we aim to automate the strategic aspect of the game using Natural Language Processing applied on textual game commentaries. Our dataset consists of both text commentaries and raw numerical data describing various aspects of the game in CSV format.

II. RELATED WORK

A. Analysis of the game using numerical raw data

Most of studies on sports mining focus on score prediction using advanced machine learning techniques. For instance, Tax et.al used 9 different classifiers in the aim of predicting scores of the Dutch League games. They have reached a maximum accuracy of 55% while considering numerous features that could affect a team both mentally and physically (Travel distance, previous encounters, matches with special importance) [2]. In addition, Razali et. Al have used Bayesian networks trying to predict English Premier League scores with their main features being listed in the table below. They succeeded in getting an accuracy of 59.21%, which is relatively well above average [3]. Nevertheless, in-depth features such as tactical analysis remain under-explored as they are more complex to study. In fact, metrics related to scoring alone cannot describe the style (i.e., the strategy) of a soccer team.

Several studies examined the passing style of teams throughout a given season. Yet, most of those made static analysis of the ball passing network (a set of passes between players of the same team aggregated into one graphical network). As a result, they disregarded the passing order and thus the way the play was built up. In contrast, other existing work had shed light on the way the ball was transitioned between each player [4] [5]. For instance, Gyarmati et. al built what was referred to as a "flow motifs profile", with one motif consisting of 3 consecutive passes, making a total of 5 motif types to study (ABAB, ABAC, ABCA, ABCB, ABCD) in order to monitor the passing sequence between players[1]. Eventually, 1000 random passing networks were generated, and the original motifs' prevalence was quantified by comparing them to passing networks of the same properties. As a result, each motif's z-score could be computed for each team, providing us with a characteristic of the latter's passing style (Fig.1). Quite similarly, the authors in [5] built a "ball flow network", which consists of interconnected nodes representing player positions. The connections are weighted according to the frequency of successful passes between two players. Two additional nodes were added to the network:

"shots to goal" and "shots wide". These were connected and weighted to other nodes according to the number of shots from each player. Eventually, the centrality of a player (how good of a playmaker he is) could be visualized by monitoring his node connection weights.

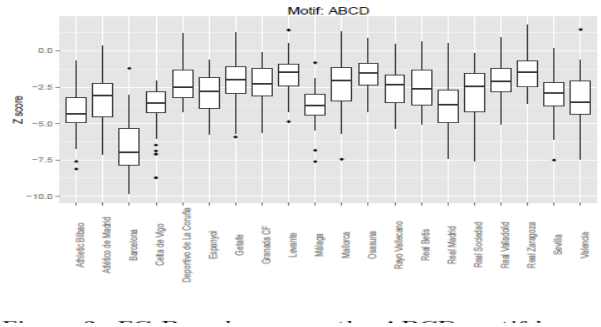


Figure 1: FC Barcelona motif profile

Similarly, in [6] and [7], passing was a main feature to discern a team's tactics. However, more events were studied (relative to [4]) such as tackling, fouls, etc. Also, the authors performed time-segmentation to keep track of the duration for which a team had a ball possession, in addition to the location of the passer and receiver of the ball at that specific time (See Table below).

Event ID	Timestamp	Passer ID	Receiver ID	Start position	End position
1574	02:27	[D]J.Alba	[D]J.Mathieu	(44.2,100.0)	(29.8,92.2)
1575	02:30	[D]J.Mathieu	[D]J.Mascherano	(25.5,81.7)	(17.8,42.9)
1576	02:33	[D]J.Mascherano	[D]D.Alves	(20.9,34.8)	(30.6,5.6)
⋮	⋮	⋮	⋮	⋮	⋮
1588	03:06	[M]I.Rakitic	[D]J.Mathieu	(52.4,14.4)	(45.2,56.7)
1589	03:10	[D]J.Mathieu	[M]A.Iniesta	(46.2,58.5)	(56.5,79.3)
1599	03:54	[G]C.Bravo	[M]S.Busquets	(4.8,40.6)	(21.9,49.8)
1600	04:01	[M]S.Busquets	[D]J.Mascherano	(33.6,49.6)	(36.7,20.7)
1601	04:06	[D]J.Mascherano	[M]I.Rakitic	(46.7,17.1)	(52.8,25.9)
1602	04:07	[M]I.Rakitic	[D]J.Mascherano	(52.8,25.9)	(45.1,17.8)
1611	04:43	[M]A.Iniesta	[D]J.Alba	(60.5,97.7)	(70.2,96.9)
1612	04:44	[D]J.Alba	[F]M.E.Haddadi	(70.2,96.9)	(80.4,82.8)
1613	04:45	[F]M.E.Haddadi	[F]L.Messi	(80.4,82.8)	(77.1,62.5)

Table 1

In [6], Wang et.al developed LDA and T³M models combining the spatial and temporal data over 90 minutes (per game) into heat maps providing details on the ball transition concentration (Bottom Figure-Left). As a result, the tactical patterns that result in the most goal rates could be deduced. In [7], Bialkowski et. Al used LDA alone to project occupancy maps of ball transition on the one hand (bottom figure-right), and another map projecting the heat map for every player role in a different color on the other hand (11 heat maps per figure).

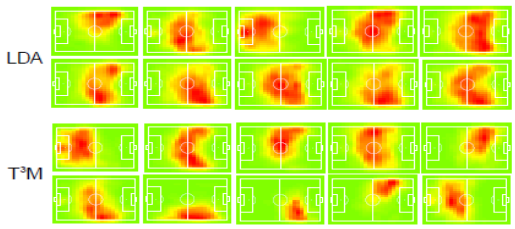


Figure 2: Heat maps for the 10 tactical patterns of Barcelona learned by LDA and T³M.

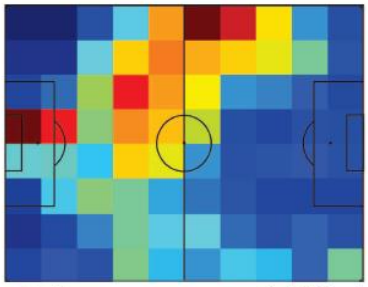


Fig. 4. Example ball occupancy map over a match half for a team attacking left to right. This example shows dominance of ball possession on the left side of the field which may be indicative of the team's playing style.

In a nutshell, the aforementioned studies have used football numerical data such as final scores and player statistics to monitor spatial and temporal movements, some of which were made easy to visualize through mapping in 2D heat maps across different timeframes and the set of projections resulting in the best performance (best scores) were determined. Nevertheless, the specific task of using Natural Language Processing (NLP) as a strategy for sports analysis remains under-explored. In fact, sports analysts and commentators provide a substantial amount of textual information, scrutinizing the details of each sport and providing professional post-match analysis.

B. Natural Language Processing

The amount of data generated every single day is estimated around 2.5 quintillion bytes [8]. As this growth expands over the years, it is compulsory for data governors to extract the useful information from it in the most computationally efficient manner. Yet, some data types remain relatively more difficult to classify, such as texts. Hence, the past decades have seen numerous language modeling attempts, using both supervised and unsupervised techniques. One example is the Bag of Words (BOW) method, which uses a binary vector representation of words in a given text, aggregates the occurrence of known words from a given vocabulary and scrutinizes their frequency in other documents. Consequently, this model is able to classify the semantics of each document. Despite being easy to implement, this approach results in a high sparcity representation of texts and is thus computationally expensive. Recently, a more artificial nature of semantics analysis has been introduced with the improvements on Deep Learning algorithms. The latter have outperformed conventional SVM and Naïve Bayes classifier on several occasions. Mariel et. al compared both methods by implementing them in sentiment analysis on online Tweets featuring prominent Indonesian institutions [9]. The deep

learning algorithm was significantly the better technique, outclassing SVM and Naïve Bayes algorithms with both precision and F1 scores being above 0.9 for balanced and imbalanced data. Moreover, NLP techniques in sports analytics have mainly used Latent Dirichlet Allocation (LDA). Miller et.al used this technique to describe possession sketches of NBA players by mapping each word in the BOW model to two three actions performed by three players simultaneously [10]. However, as mentioned earlier, the BOW model is computationally expensive and thus more innovative approaches in NLP for sports analytics are needed.

C. Sentence Embedding using Doc2vec

Recent NLP techniques have considered representing words and sentences in terms of numerical vectors in high-dimensional space, rather than merely a set of counters.

After training a large corpus of documents, a n-dimensional space is created, whereas words or sentences displaying close semantic similarities could be recognized through the cosine similarity value between the corresponding vectors given by the Euclidean dot product as follows: $similarity = \cos(\theta) = \frac{A.B}{||A||\,||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$. Bearing in mind that such vectors could be representative of either words or whole sentences, this mapping process is referred to as embedding. As a result, this technique is able to keep track entire sentences and the order of words within them, in contrast to the previously mentioned BOW method. Recent works have tested the efficiency of word2vec and do2vec embeddings. Moran et.al used a pretrained word2vec model for First Story Detection (FSD) on Twitter data, and compared it to conventional FSD methods such as tweet expansion using WordNet [11]. As a result, they got better Cmin (Topic Weighted Minimum Cost) values post-tuning, which decreases the probability of false alarms.

The input embedding vectors going into the network are obtained using doc2vec, which is an extension of word2vec. The latter works on sampling and classifying words having similar meanings, hence displaying vector embedding with high cosine similarity. This is done by negative sampling, which maximizes the dot product of words within the same context as follows:

$log\ \sigma(v'_{wo} T_{V_{wI}}) + \sum_{i=1}^{k} w_i \sim P_n(w)[log\ \sigma(-v'_{wi} T v_{wi})]$

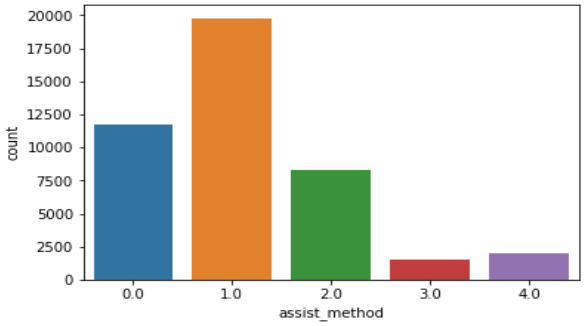
where k is the number of negative samples, v_w is the input word vector and v'_w is the vector of the word being sampled negatively [12].

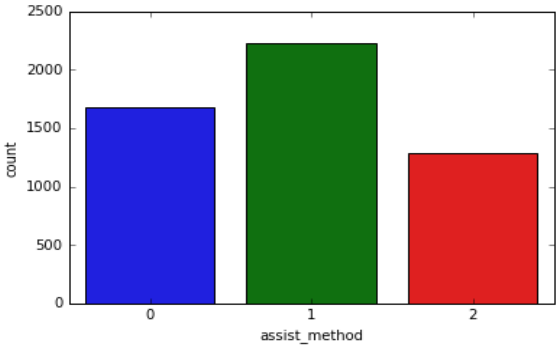
In a nutshell, our paper makes use of the aforementioned sections and combines them into a Deep Learning model that is able to recognize a game scenario given a textual input.

III. Methodology

A. Dataset

In this paper, we utilize a Kaggle dataset containing text commentaries from numerous games in different leagues across Europe [13]. The first phase of our work deals with data pre-processing. Out of 22 data features, our analysis uses only 3 which are believed to be the most representative of our approach to test our hypothesis. For instance, features such as team names, body parts of players and substitutions could be dropped for our purpose. In addition, the original data is highly unbalanced. For example, the feature *location* has over 90 000 instances with the attribute 15, as compared to 20 000 for the same feature. Similarly, the feature *assist method* has over 18 000 instances with attribute 1, which is significantly higher than other instances within the same category as shown in Figure 2. Therefore, a fair and representative re-sampling is done as shown below, preventing extensive noise in our data. Finally, team names between parentheses were removed from textual commentaries to prevent the model from learning them in an attempt to describe a football scenario rather than spotting a team name.





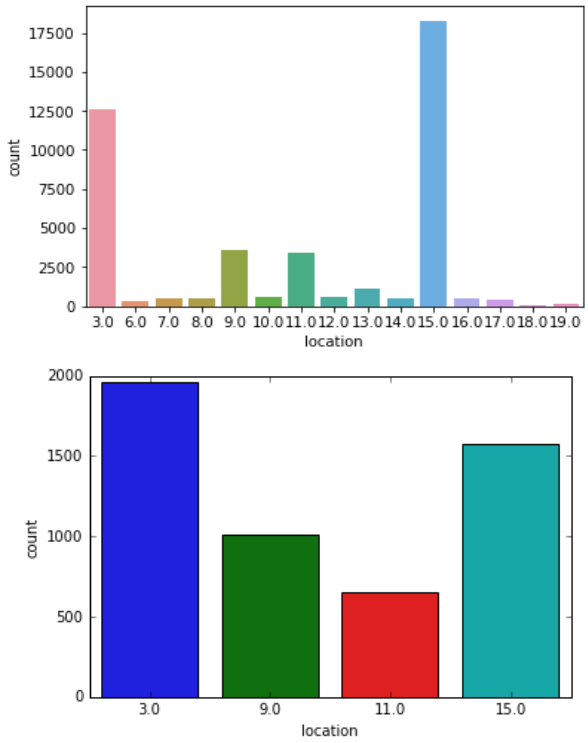


Figure 2:Pre and Post-Processing of features

Therefore, post-processing results in a normalized and fairly representative subset for training and testing whose shape is shown in Table 2. Figure 3 shows the final post-processes data.

	0	1	2	3
Assist Method	None	Pass	Cross	NaN
Shot outcome	NaN	On target	Off target	Blocked

Table 2

Index	text	assist_method	location	shot_outcome
0	Attempt missed. Mladen Petric left footed shot from the left side of the box is high and wide to the left. Assisted by Gokhan Tore.	1	9	2
1	Attempt missed. Shinji Kagawa right footed shot from outside the box is close, but misses the top right corner. Assisted by Mario Gotze.	1	15	2
2	Goal! Borussia Dortmund 1, Hamburg 0. Kevin Grosskreutz left footed shot from the left side of the box to the bottom right corner. Assis…	1	9	1
3	Attempt blocked. Mats Hummels right footed shot from outside the box is blocked.	0	15	3
4	Attempt blocked. Tomas Rincon right footed shot from outside the box is blocked.	0	15	3
6	Attempt blocked. Ilkay Gundogan left footed shot from outside the box is blocked.	0	15	3
7	Attempt saved. Mats Hummels header from the centre of the box is saved in the centre of the goal. Assisted by Chris Lowe with a cross.	2	3	1

Figure 3: Text and corresponding labels

B. Inferring vectors to embeddings using gensim

As doc2vec is designed for large data analysis, it should be trained on large corpora in order to get effective results. Therefore, we used a doc2vec model that was pre-trained on a collection of Wikipedia articles that it got from the English Wikipedia database dump using the WikiExtractor code [14].
That being said, our code was designed to take as input an array of size (4,300) which represents the doc2vec embeddings for a text commentary input and its corresponding 3 labels. 300 corresponds to the size of the inferred embedding vector for each instance.
To proceed, a function is defined to infer each column instance belonging to the same row, while appending them in a list *emat* which would be converted to an array later on. Next, a label of 1 is assigned to each array in a new list *emat_label*, informing the network that such a sequence description is correct. In a similar manner, a "false classification" is introduced into a list *emat_wrong* by randomly shuffling the labels, and assigning 0 to them in *emat_wrong_label*. Consequently, the network learns that the labels generated do not accurately describe the original text scenario. Eventually, a vertical stacking is performed on *emat* and *emat_wrong* followed by *emat_label* and *emat_wrong_label.* Eventually, the latter 2 columns correspond to our training sets X_train and Y_train. Our hypothesis is that our CNN model to which doc2vec embeddings are fed is able to learn a labeling process describing the game scenarios associated with each commentary.

C. Building and Tuning Parameters of Nerual Networks

Our model architecture is presented in Figure 5. The input and hidden layers will consist of rectified linear units acting as activation functions. Cui et.al performed text classification using this function based on SAE algorithm, and compared it to the sigmoid function [15]. It was shown that Relu activation functions, despite being prone to sparsity, can prevent overfitting and improve the accuracy of text classification as compared to the sigmoid function, will be used for our binary output (Table 3). Knowing that our text instances differ in length, we added the option of padding. Finally, our input shape is (4,300) for which we use a kernel size of 2 for the feature detectors. The convolutional layers are followed by a Dropout regularizer of with a rate of 0.5, further reducing the possibility of overfitting. The final layout of our model is shown in Figure 5 below.

Activation function	Training error	Test error
SAE (Sigmoid)	8.569%	46.625%
SAE (Tanh)	3.46%	24.063%
SAE (ReLU)	24.465%	10.938%

Table 3

```
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_7 (Conv1D)            (None, 4, 256)            153856
_________________________________________________________________
dropout_7 (Dropout)          (None, 4, 256)            0
_________________________________________________________________
conv1d_8 (Conv1D)            (None, 4, 128)            32896
_________________________________________________________________
dropout_8 (Dropout)          (None, 4, 128)            0
_________________________________________________________________
conv1d_9 (Conv1D)            (None, 4, 64)             8256
_________________________________________________________________
dropout_9 (Dropout)          (None, 4, 64)             0
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 2, 64)             0
_________________________________________________________________
flatten_3 (Flatten)          (None, 128)               0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 129
=================================================================
Total params: 195,137
Trainable params: 195,137
Non-trainable params: 0
```

Figure 4: CNN model

Afterwards, our unseen training dataset is loaded into the model and inferred as new embedding vectors. Finally, we use conventional hyperparameters that are known to have good results on large corpora, with $\alpha = 0.1$ and $\alpha_{min} = 0.0001$ [12]. Finally, we use the binary crossentropy loss function, and the RMSprop optimizer.

On the other hand, we try a Recurrent Neural Network to compare accuracies and performance with CNN. The RNN has the architecture given below, with LSTM being added at the input.

Dataset	Total row size	Features	Training size utilized	Validation size utilized	Testing size utilized	Feature size utilized
Events	941009	22	6256	1564	1380	3

Table 4

D. New commentary testing

In the final stage, our code transforms the new data from Flashscore.com into embedding vectors. Next, a for loop is initiated to run over all permutations of possible labels that are combined with the input text vector and predicted. An array of size 36 (total number of permutations) will be created as a result, containing values between 0 and 1. The maximum probability (>0.5) at a given row containing a set of labels means this labels are to be assigned to the text. The pseudo-code for this process is given next:

```
alist=list(itertools.product([0, 1, 2],[3,9,11,15],[1,2,3]))
maxnum=0
locate=[0,0,0]
track=[]
for i in range(len(liverform)):
    forma=[]
    maxnum=0
    for j in range(len(alist)):
            forma.append([envec(liverform.iloc[i,0]),envec(alist[j][0]),envec(alist[j][1]),envec(alist[j][2])])
            monitor=model.predict(np.array(forma))
            maxnum=0
    for k in range(len(monitor)):
            if(maxnum<monitor[k]):
                maxnum=max(maxnum,monitor[k])
                locate=[alist[k][0],alist[k][1],alist[k][2]]
    track.append(locate)
```

Figure 5 Pseudo code used to find compatible labels

Our methodology is summarized in the flowchart below:

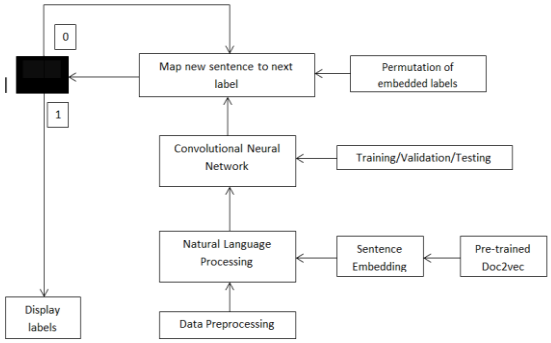


Figure 6: Flow chart summarizing our methodology

IV. RESULTS

Accuracies for training, validation and testing are reported in Figures 8 and 9, with the corresponding loss plotted in Fig 10. That being said, our model is able to label scenario events from a given textual input with about 70% accuracy.

```
Epoch 86/90
6256/6256 [==============================] - 2s 336us/step - loss: 0.5500 - acc: 0.7048 - val_loss: 0.5294 - val_acc: 0.7244
Epoch 87/90
6256/6256 [==============================] - 2s 318us/step - loss: 0.5423 - acc: 0.7110 - val_loss: 0.5273 - val_acc: 0.7295
Epoch 88/90
6256/6256 [==============================] - 2s 324us/step - loss: 0.5432 - acc: 0.7136 - val_loss: 0.5352 - val_acc: 0.7295
Epoch 89/90
6256/6256 [==============================] - 2s 328us/step - loss: 0.5499 - acc: 0.7075 - val_loss: 0.5251 - val_acc: 0.7410
Epoch 90/90
6256/6256 [==============================] - 2s 323us/step - loss: 0.5451 - acc: 0.7059 - val_loss: 0.5349 - val_acc: 0.7212
```

Figure 7: Training accuracy after 50 epochs for CNN

```
1380/1380 [==============================] - 0s 90us/step
Test set
  Loss: 0.550
  Accuracy: 0.728
```

Figure 8: Testing accuracy after 50 epochs for CNN

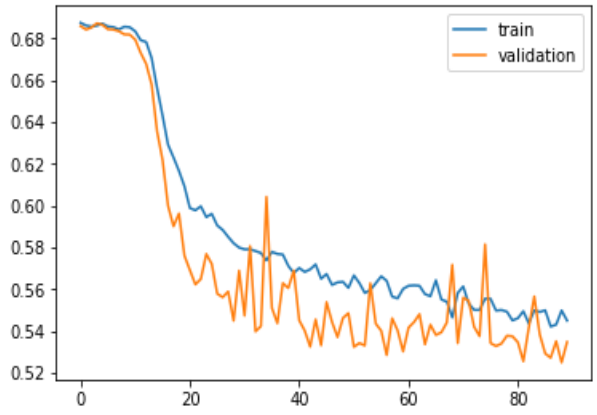


Figure 9: Plot of training and validation loss for CNN

The following confusion matrix was obtained as described in Table X, hence yielding a precision of 66%.

	Predicted no	Predicted yes
Actual no	228	369
Actual yes	69	714

On ther other hand, the RNN gave the resuls below :

```
Epoch 47/50
6256/6256 [==============================] - 2s 345us/step - loss: 0.4467 - acc: 0.7858 - val_loss: 0.4404 - val_acc: 0.7999
Epoch 48/50
6256/6256 [==============================] - 2s 334us/step - loss: 0.4366 - acc: 0.7956 - val_loss: 0.4164 - val_acc: 0.8069
Epoch 49/50
6256/6256 [==============================] - 2s 346us/step - loss: 0.4413 - acc: 0.7898 - val_loss: 0.4758 - val_acc: 0.7583
Epoch 50/50
6256/6256 [==============================] - 2s 358us/step - loss: 0.4401 - acc: 0.7951 - val_loss: 0.4261 - val_acc: 0.8031
```

Figure 10: Training accuracy after 50 epochs for RNN

```
1380/1380 [==============================] - 0s 94us/step
Test set
  Loss: 0.480
  Accuracy: 0.778
```

Figure 11: Testing accuracy after 50 epochs for RNN

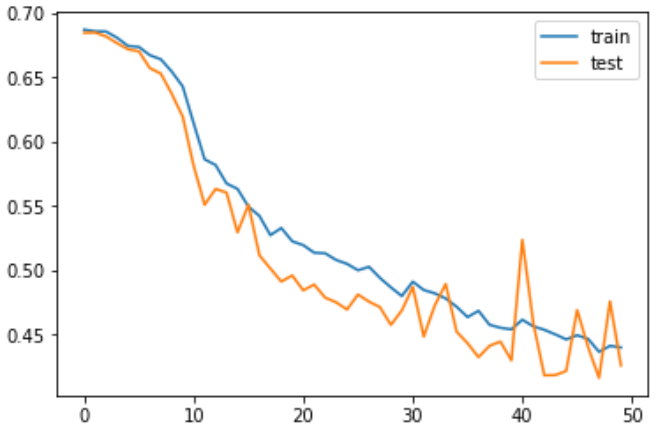


Figure 12: Training and validation loss for RNN

```
from sklearn.metrics import confusion_matrix
cm2 = confusion_matrix(Y_test, hello2)
cm2
```

```
array([[362, 228],
       [ 83, 707]])
```

Figure 13: Confusion matrix for RNN

As observed, the RNN came with a higher accuracy than CNN, which could be due to the usage of LSTM. Generally, CNNs can handle long-term dependencies better than RNNs, by convolving numerous kernel filters over an input vector with the kernel size being proportional to the vector size. However, the usage of LSTM solves the issue of long-term dependencies for the RNN, hence providing it with the advantage it had over CNN.

At this stage, our model is ready to be tested on completely unseen data. We introduce the game commentaries of the Liverpool vs Arsenal game in March 2017 obtained from Flashscore.com, containing sentences describing Liverpool actions.

Index	Text
0	Xherdan Shaqiri (Liverpool) takes the free kick and immediately restarts play with a short pass.
1	Xherdan Shaqiri (Liverpool) misses a good chance to score. A perfect cross into the box finds Xherdan Shaqiri (Liverpool) who rises for a header, but he sends the ball well over the bar.
2	Trent Alexander-Arnold produces a cross from the resulting corner and finds Divock Origi (Liverpool) inside the box. He pulls the trigger and scores, sending the ball into the top left corner. What a brilliant finish. 4:0.
3	James Milner (Liverpool) sends a teasing cross into the area, but Marc-Andre ter Stegen intercepts the ball.
4	It's a goal! Georginio Wijnaldum (Liverpool) makes it 3:0. He jumped highest to connect with a perfect cross from Xherdan Shaqiri and planted his close-range header into the left side of the goal. Marc-Andre ter Stegen was helpless.
5	Goal! Trent Alexander-Arnold plays a pass to the feet of Georginio Wijnaldum (Liverpool), and he shoots into the back of the net from inside the box. It's 2:0.
6	A cross following the corner kick finds its way to Virgil Van Dijk (Liverpool) inside the box and he manages to steer it to the middle of the target. He is about to start celebrating a goal, but Marc-Andre ter Stegen makes a save in the nick of time and maintains the current score.

Index	Type	Size	
0	list	3	[0, 9, 2]
1	list	3	[2, 3, 2]
2	list	3	[2, 3, 2]
3	list	3	[0, 3, 2]
4	list	3	[2, 3, 2]

As a result, the model seems to be able to spot the scenario in many cases. For instance, the commentary *"Xherdan Shaqiri (Liverpool) misses a good chance to score. A perfect cross into the box finds Xherdan Shaqiri (Liverpool) who rises for a header, but he sends the ball well over the bar."* was labeled [2,3,2] which refers to the assist method being a cross labeled 2, the location being inside the box labeled 3, and the shot outcome being off target labeled 2. That being said, the scenario is indeed that of a cross being played inside the box and shot off target!

V. Conclusion

Label learning has been successfully done using both CNN and RNN. The latter came with a higher accuracy of 79% while having LSTM at the input, preventing problems associated with arbitrary long sequence inputs. Our model could be further developed into an unsupervised learning approach in sports analytics. In the future, we aim to generalize this model to cover entire games and more detailed features.

VI. References

[1] Deloitte , "Global sports market-total revenue from 2005 to 2017 (in billiob u.s dollars)," In Statista-The Statistics Portal , 2017.

[2] N. Tax and Y. Joustra, "Predicting The Dutch Football Competition Using Public Data: A Machine Learning Approach," in *TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*.

[3] N. Razali, A. Mustapha, F. Yatim and R. Ab Aziz, "Predicting Football Matches Results using Bayesian Networks for English Premier League (EPL)," in *IOP Conference Series: Materils Science and Engineering*.

[4] L. Gyarmati, H. Kwak and P. Rodriguez, "Searching for a

Unique Style in Soccer," *https://arxiv.org/abs/1409.0308*, 2014.

[5] J. Duch, J. Waitzman and L. Amaral, "Quantifying the Performance of Individual Players in a Team Activity," *https://doi.org/10.1371/journal.pone.0010937*, 2010.

[6] Q. Wang, H. Zhu, Z. Shen and Y. Yao, "Discerning Tactical Patterns for Professional Soccer Teams: An Enhanced Topic Model with Applications," in *KDD*, 2015.

[7] A. Bialkowski, P. Lucey, P. Carr, Y. Yue, S. Sridharan and I. Matthews, "Identifying Team Style in Soccer using Formations Learned from Spatiotemporal Tracking Data," *IEEE International Conference on Data Mining Workshop*, 2014.

[8] B. Marr, "Forbes. How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read," 21 May 2018. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#60b7020460ba.

[9] W. C. F. Mariel, S. Mariyah and S. Pramana, "Sentiment analysis: a comparison of deep learning neural network algorithm with SVM and naive Bayes for Indonesian text," in *International Conference on Data and Information Science* , 2018.

[10] A. C. Miller and L. Bornn, "Possession Sketches: Mapping NBA Strategies," in *MIT SLOAN SPORTS ANALYTICS CONFERENCE* , 2017.

[11] S. Moran, R. McCreadie, C. Macdonald and I. Ounis, "Enhancing First Story Detection using Word Embeddings," in *SIGIR '16 Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*.

[12] J. H. Lau and T. Baldwin, "An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation," *Workshop on Representation Learning for NLP, Berlin, Germany*, 2016.

[13] A. Secareanu. [Online]. Available: https://www.kaggle.com/secareanualin/football-events.

[14] G. Attardi, "A tool for extracting plain text from Wikipedia dumps," [Online]. Available: https://github.com/attardi/wikiextractor.

[15] J.-L. Cui, S. Qiu, M.-y. Jiang, Z.-l. Pei and Y.-n. Lu, "Text Classification Based on ReLU Activation Function of SAE Algorithm," in *International Symposium on Neural Networks* , 2017.

[16] V. Sharma, R. Kulshreshtha, P. Singh and N. Agrawal, "Analyzing Newspaper Crime Reports for Identification of Safe Transit Paths," *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 17-24, 2015.

[17] D. Blei, A. Ng and M. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning*, 2003.

[18] M. Cordeiro, "Twitter event detection: combining wavelet analysis and topic inference summarization," in *DSIE'12 the Doctoral Symposium on Informatics Engineering*, 2012.

[19] [Online]. Available: https://www.flashscore.com/match/U5PUe8fH/#live-commentary;0.

.