# Authorship Attribution of NBA Tweets

**Pedro Sousa** **Holly Foster**

## 1 Introduction

Authorship attribution (AA) is a multi-class classification problem that involves identifying the author of a given text based on linguistic patterns and features extracted from the text. As each author uses different writing styles, vocabularies, and structures, it is challenging to identify them accurately. Machine learning models and algorithms can be used to solve this problem by analysing and extracting text features that indicate the author's writing style (stylometric features) (Yin et al., 2016).

From forensic analysis (Ainsworth and Juola, 2018) to plagiarism detection (Stamatatos and Koppel, 2011), authorship attribution has many applications. An exciting recent application of AA is in the ever-growing industry of social media, notably for Twitter (Layton et al., 2010) (Bhargava et al., 2013). Twitter is a popular platform that allows users to share short messages known as tweets. Its vast user base and broad use cases make it a unique and dynamic platform that has become integral to online communication and social media marketing. In particular, the National Basketball Association (NBA) has received praise for using the platform to drive fan engagement with the sport. With top NBA players having a large influence over their fans, key tweets can boost ticket sales and team support. Here, we aim to conduct AA on tweets posted by the most popular players in the league.

In this report, we propose an exploration of different sequence classification approaches and techniques used to solve this NLP problem. We aim to evaluate and compare the performance of such models on our collected dataset of 9 top NBA players' tweets. Namely, our baseline models will be a Support Vector Machine (SVM) and a Naive Bayes classifier, with a Bi-Directonal LSTM and BERT model as the two main comparison models. The desired output for each model will be a correct classification of an author to an unseen tweet. Additionally, we will discuss the challenges and limitations of this authorship attribution task and potential directions for future follow-on work.

## 2 Methodology

### 2.1 Dataset

Our dataset was built using the Twitter API (Twitter, 2023). The dataset was generated manually because there is not a pre-collected dataset of tweets for our required niche group of users. Using the selected NBA player's Twitter handles as our `AuthorIDs`, the API queries consisted of checking for any tweets, excluding retweets, posted by those accounts. The time span for these queries was the start of the 2010 season (28-10-2010) up until the All-Star Weekend break (20-01-2023). The resulting found tweets were then appended to a `CSV` data file. Upon reviewing ethical and data protection principles, the `AuthorIDs` were stored as numerical values as opposed to the author's account name. The challenge encountered during the dataset creation was the API's request rate limit (Riggins, 2023). The Twitter API imposes a limit on the number of API calls that can be made in a given time frame, meaning we were not able to create a dataset as large as we would have liked. However, we used the `Tweepy.paginator` library to retrieve more results than is usually returned in a single response in order to help overcome this issue. This resulted in a dataset of 44892 tweets.

It is worth noting here that this dataset focuses only on tweets that use the English language.

### 2.2 Data Preprocessing

After reading the `CSV` file of 'raw tweets', the data was separated by `AuthorID`. Any authors with fewer than 10 tweets were removed as there wouldn't be enough example data to gather writing style patterns with. As emojis are heavily used in social media, these were removed from the tweets, along with any punctuation, special characters, links, and user mentions. Word contractions were also removed as they contribute to text standardisation and are useful when working with this type of data, as the words play an important role in sentiment analysis. Full hashtags appearing at the end of a tweet were removed, but just the symbol itself was removed for those in the middle of a sentence. This meant the sentence structure could be preserved. In addition to tokenising each word in the tweet, the `PorterStemmer` was used to stem the words. Initially, we lemmetised the words too. However, the results were better with the stemmer alone. Each author's data was then combined into a dataframe of 'cleaned tweets', ready to be used by the models.

The resulting clean dataset consisted of 9 unique authors corresponding to 9 top NBA league players. The data was split into separate training, validation, and testing datasets, using a 70%, 10%, and 20% split, respectively. As the classes were unbalanced for each author, the training set was oversampled in order to enforce the same count of tweets for all 9 authors. This value was taken from the author with the highest number of tweets (6532 tweets). The `CountVectorizer` method was used to create a bag of words and consequently transform the train and test sets. The `TfidfTransformer` was then used to apply the TF-IDF value to attribute weights to the word vectors based on their frequency.

## 2.3 Naive Bayes

As one of our baseline models for the classification task, we implemented a multinomial naive Bayes classifier. Assuming conditional independence between features, the likelihood of observing feature vector **x** given author $A_k$ is,

$$\log p(\mathbf{x}|A_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i p_{ki}^{x_i} \tag{1}$$

where $p_{ki}$ is the probability of feature $i$ occurring in a tweet by author $k$. Multinomial naive Bayes becomes a linear classifier when expressed in log-space,

$$\log p(A_k|\mathbf{x}) \propto \log(p(A_k) \prod_i p_{ki}^{x_i}) \tag{2}$$

$$= \log p(A_k) + \sum_i x_i \log(p_{ki}) \tag{3}$$

$$= b_k + \mathbf{w}_k^T \mathbf{x} \tag{4}$$

The parameters $p_{ki}$ and $p(A_k)$ are calculated using the distribution of training data features with maximum likelihood estimation. To classify the author of a given tweet, Equation 4 is used to calculate the log-likelihood of each author and then the author with the largest value is assigned (Chen et al., 2017) (Rennie et al., 2003).

This model was trained using scikit-learn's `MultinomialNB` class on the input and output training values, with the predict method used on the test input data to obtain model predictions (SKLearn, 2023b).

## 2.4 Support Vector Machine

An SVM was our second baseline model. The algorithm tries to find a hyperplane that can separate the data points of all the classes in the best possible way. The objective of SVM is to maximise the margin between the hyperplane and the closest data points of each class. In our one-versus-one approach, the SVM trains a binary classifier for each pair of classes. When classifying a new data point, the SVM predicts the class that wins the most pairwise comparisons (Daengduang and Vateekul, 2017).

In a similar fashion to the naive Bayes classifier, this model was trained using scikit-learn's `svm.SVC` class (SKLearn, 2023a).

## 2.5 Bi-directional LSTM

The LSTM is a variant of a recurrent Neural Network. The input sequence created by the embedding layer is fed into two separate LSTM single layers, one in the forward direction and one in the backward direction. During the training process, the weights of both the forward and backward LSTMs are updated based on the negative log-likelihood loss (NLLL) function computed from the concatenated output. The outputs of both layers are concatenated and used as the final output. Log softmax normalisation is then applied to the output in order to obtain a probability prediction. These model predictions are then compared to the author of the text, and the model is trained to minimise the NLLL between the two. We opted for an embedding layer of 200 and a hidden dimension of 100, with 5 epochs. To create the embeddings, we used Word2Vec (TensorFlow, 2023). Additionally, the AdamW optimiser was used to adjust the learning rate of each weight and bias in the model for faster convergence and better performance.

This model was trained using PyTorch's neural network module (PyTorch, 2023).

### 2.6 BERT

In BERT (Bidirectional Encoder Representations from Transformers), the last layer of the transformer encoder is used to generate the final representations of the input text. Specifically, the last layer output of BERT is a sequence of hidden states, one for each token in the input sequence. These hidden states contain information about the context of each token and can be used as embeddings. Like with the LSTM, the AdamW optimizer is used to enhance the model. In our case, the model minimises the cross entropy loss by adjusting the weights and biases of the layers through back-propagation.

Again, this model was built using PyTorch (PyTorch, 2023) and the `Transformers` library for BERT from Hugging Face (HuggingFace, 2023).

Due to current state-of-the-art performances being achieved via BERT models, we expect this model to give us the most accurate results.

## 3 Evaluation

In order to compare the results of the different models, the same main experimental setup was used for each of our models. Each model used the same proportion splits for the training, validation, and testing datasets.

By building BERT and Bi-directional LSTM models, we expected to achieve the correct authorship attribution of a given tweet with higher accuracy than the baseline SVM and Naive Bayes models. We predicted that the BERT model would achieve the best results due to its higher complexity and the evidence shown from state-of-the-art models.

Here, our results focus on the overall accuracy achieved by the model. Accuracy measures the proportion of correct predictions among all predictions made by the model. Whilst our classification reports provided other evaluation metrics such as precision, recall, and F1 scores, we felt accuracy was the most appropriate metric to use, particularly as our classes were balanced.

The final accuracy results for our models are shown in Table 1.

Here we can see that the BERT model performed the best, with the highest accuracy of 58.61%. This is a 2.6% increase from the second best, our SVM

| Model | Accuracy |
|---|---|
| Niave Bayes | 53% |
| SVM | 56% |
| Bi-LSTM | 55.63% |
| BERT | 58.61% |

Table 1: Testing accuracy for NBA tweets author classification.

baseline model. However, the Bi-LSTM was only 0.37% less accurate than the SVM model, meaning it still performed highly in comparison. The second baseline model, the naive Bayes, was the poorest model at classifying the correct author to the given tweet. We also manually tested our BERT model by inputting the latest tweets from one of the used players, i.e. posts since the end of the All-Star Weekend, and seeing if the author was classified correctly.

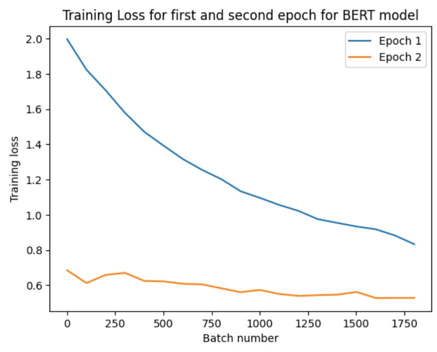
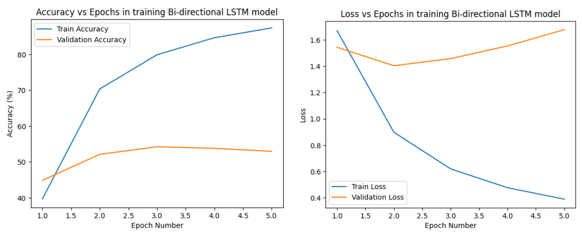


Figure 1: Training Loss for First and Second Epoch for BERT Model.

Figure 1 shows how the training loss for the BERT model declined significantly with the second epoch. As shown in the graph, the line stabilises as the batch number continues to increase. This suggests the loss functions converge, and the model improves greatly over the second epoch.



(a) Accuracy vs Epochs. (b) Loss vs Epochs.

Figure 2: Training and Validation Comparisons for Bi-directional LSTM model.

As shown in Figure 2, the number of epochs used in the model causes different results. The training accuracy of the Bi-LSTM increases as the number of epochs increases. However, the validation accuracy peaks at 3 epochs and then plateaus. The training loss consistently decreases with more epochs. This is not the case for the validation loss; it decreases until it reaches 2 epochs, and then increases linearly. Both of these graphs suggest that the model is overfitting the training data. This means that the model fits too closely to the training data, including the noise, and is not generalising well to new, unseen data. As a result, the validation accuracy, which measures the model's performance on new data, decreases as the model becomes too specialised to the training set.

## 4 Discussion

The results we have achieved slightly contradict our hypothesis, as the table shows our accuracy did not significantly improve with the more complex models. We expected the Bi-LSTM model to perform significantly better than both of the baseline models. However, this was not the case. On the other hand, our BERT model did outperform all other models, as we predicted. We were still surprised to see only a few percentage differences between the models' performances.

All of our results were unfortunately below the 60% mark. This could be down to certain hyperparameter tuning of the models or due to an issue with the dataset. As each accuracy was within the same region, we believe the results were on the lower end of the scale because of the restrictions on our dataset size. With the API restrictions and niche tweet criteria, we would ideally have used either a bigger dataset or potentially fewer authors, for example, 5 instead of 9. Due to time limitations, we could also not spend as much time experimenting with the hyperparameter tuning as we would have liked, for example, BERT's learning rate.

However, the results still show evidence suggesting that the models successfully learned a relationship between the written texts and their authors.

When compared to related existing work in AA, our values are below the typical standard. For example, in (Fabien et al., 2020), their BERT models achieved majority accuracies of 90%+, for datasets such as IMDB and Enron (emails). Although, their results for their Blog dataset were also in the 50s. This highlights that the results are highly dependent on the quality of the dataset.

Our baseline models were also below average when compared to previous work. For example, in (Chen et al., 2017), their Naive Bayes model reaches an accuracy of 84.4%. This is significantly higher than our result of 53%. Their SVM performed better (87.6%), which is in line with the pattern we have also found. We do have to bare in mind that models can vary significantly depending on their features (e.g. if Bag-Of-Words, Word2Vec, or POS tagging etc. was used), so direct comparisons to others' work aren't completely fair.

## 5 Conclusion

An authorship attribution model evaluated on a set of Tweets achieves good classification accuracy using Naive Bayes, SVM, Bi-directional LSTM, and BERT approaches. Classifying tweets from 9 separate authors (NBA players), all models exceeded the baseline Naive Bayes model's performance of 53%/. The highest performing model was achieved using a BERT, which utilises a multi-layer bidirectional transformer encoder, giving an accuracy of 58.61%. Although this shows additional features and more powerful models do contribute to higher accuracy, they don't significantly outperform.

There are multiple directions we could take this work in in the future. First, we'd like to investigate and reduce any bias within the models, especially in the Word2Vec features, as there were some low training accuracies. Regarding the Bi-LSTM, we could try adding more layers to increase the complexity of the model. We would also like to add some regularisation techniques, e.g., weight decay, to help with any overfitting issues, as mentioned previously. For our BERT model, we would love to further pre-train it and spend longer on the hyperparameter tuning in order to improve the performance further. In general, we'd like to assess how the models differ in performance as more authors are added and explore which factors caused the misclassifications.

As a next step to AA, we could extend our approach to Authorship Verification, which is a subfield of AA. It involves the task of determining whether two or more texts have been written by the same author or not.

## References

Janet Ainsworth and Patrick Juola. 2018. Who wrote this: Modern forensic authorship analysis as a model for valid forensic science. *Wash. UL Rev.*, 96:1159.

Mudit Bhargava, Pulkit Mehndiratta, and Krishna Asawa. 2013. Stylometric analysis for authorship attribution on twitter. In *Big Data Analytics: Second International Conference, BDA 2013, Mysore, India, December 16-18, 2013, Proceedings 2*, pages 37–47. Springer.

Luke Chen, Eric Gonzalez, and Coline Nantermoz. 2017. Authorship attribution with limited text on twitter. *CS Stanford. edu*.

Suthipong Daengduang and Peerapon Vateekul. 2017. Applying one-versus-one svms to classify multi-label data with large labels using spark. In *2017 9th International Conference on Knowledge and Smart Technology (KST)*, pages 72–77. IEEE.

Maël Fabien, Esaú Villatoro-Tello, Petr Motlicek, and Shantipriya Parida. 2020. Bertaa: Bert fine-tuning for authorship attribution. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 127–137.

HuggingFace. 2023. Hugging face bert documentation. https://huggingface.co/transformers/v3.0.2/model_doc/bert.html.

Robert Layton, Paul Watters, and Richard Dazeley. 2010. Authorship attribution for twitter in 140 characters or less. In *2010 Second Cybercrime and Trustworthy Computing Workshop*, pages 1–8. IEEE.

PyTorch. 2023. Pytorch neural network documentation. https://pytorch.org/docs/stable/nn.html.

Jason D Rennie, Lawrence Shih, Jaime Teevan, and David R Karger. 2003. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 616–623.

Jennifer Riggins. 2023. What we can learn from twitter's outages. https://thenewstack.io/what-we-can-learn-from-twitters-outages/.

SKLearn. 2023a. C-support vector classification. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

SKLearn. 2023b. Naive bayes classifier for multinomial models. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html.

Efstathios Stamatatos and Moshe Koppel. 2011. Plagiarism and authorship analysis: introduction to the special issue. *Language Resources and Evaluation*, 45:1–4.

TensorFlow. 2023. Word2vec. https://www.tensorflow.org/tutorials/text/word2vec.

Twitter. 2023. Twitter api documentation. https://developer.twitter.com/en/docs/twitter-api.

Xiangyu Wang and Mizuho Iwaihara. 2021. Integrating roberta fine-tuning and user writing styles for authorship attribution of short texts. In *Web and Big Data: 5th International Joint Conference, APWeb-WAIM 2021, Guangzhou, China, August 23–25, 2021, Proceedings, Part I 5*, pages 413–421. Springer.

Feiyang Yin, Zhilin Yao, and Jia Liu. 2016. *Authorship Attribution Using Stylometry and Machine Learning Techniques*, pages 113–125.