# Authorship Attribution for Neural Text Generation

**Arjun Ramesh Kaushik**
University at Buffalo
kaushik3@buffalo.edu

**Dhiraj Gunasheela**
University at Buffalo
dgunashe@buffalo.edu

**Sahithi Pendry**
University at Buffalo
sahithip@buffalo.edu

**Smarana S. Pankanti**
University at Buffalo
smaranas@buffalo.edu

## Abstract

Authorship attribution is the process of identifying the author of a given text. It has essential applications in various fields, such as cyber-forensics, plagiarism detection, and political socialization. This report explores various Machine Learning models and has compared their performances for each task of authorship attribution.

## 1 Introduction

With the advent of powerful Natural Language Generators, synthetic text generation is close to the realistic text. This also sprouts challenges like detecting plagiarism, detecting bots on social media, and also detecting the legitimate sources of a given text. The problem is triple-faceted:

1. **Problem 1** To determine if two given texts $T_1$ and $T_2$ are produced by the same generator (NLG/human)

2. **Problem 2** To determine if a given text $T_1$ is synthetic, i.e., is it written by a human or generated by an NLG

3. **Problem 3** To determine the generator of a given text $T_1$ among multiple NLGs and human

Our dataset consists of text, generated by 10 natural language generators (NLGs) in response to 1066 prompts. These NLGs include CTRL, GPT, GPT2, GROVER, XLM, XLNET, PPLM, FAIR, as well as GPT3 and INSTRUCT GPT whose responses are generated using OpenAI API and also HUMAN-generated responses. We are utilizing this NLG-generated data to perform tasks 1, 2, and 3. For the Reddit task, we are specifically using HUMAN, GPT3, and INSTRUCT GPT NLGs to accomplish all the above tasks.

In our work, we have shown how each model performs. Based on the task, we can use the appropriate model with requisite embeddings to complete the task

## 2 Related Work

The report explores various classification algorithms that are used to perform authorship attribution. It discusses the use of features such as n–grams, POS-tags, topic modelling, POS-Noise and LIWC features and also writing style. Apart from these features, we also look into style, content and hybrid features [reference].

- Style : function words, digits and punctuations

- Content : bags of n-grams

- Hybrid : character n-grams

The classification algorithms used for authorship attribution task range from simple machine learning models such as Naive Bayes, SVM, Conditional Trees, Random Forest, and KNN to deep learning methods such as CNNs and RNNs. The report also uses large language models like RoBERTa and GROVER-DETECT. Skip-gram word embeddings by fastText tend to perform better than Word2Vec or GLoVe embeddings in Authorship Attribution[1]

Deep learning frameworks have outperformed most state-of-the art approaches in a multitude of language processing tasks such as machine translation, sentiment analysis, and speech recognition. Recent works like those by Fereshteh Jafariakinabad [6] include a combination of different deep learning models. They propose using CNNs and BiLSTM sequentially where CNNs behave as Parts-of-Speech tags encoder for sentence classification. Further, Shriya TP Gupta [5] examines the working of BiLSTMs with a Max Pooling layer and concludes that there is a 4% increment in the model performance. RNNs have been explored with pre-trained embeddings by Chen Qian[7] to achieve considerable accuracies. However pre-trained embeddings limit the performance of the model, owing to their high dimensionality. Further, they do not consider the possibilities where the word usage matters as well. Because these word vectors are meant to capture similarities between words, certain words such as 'therefore' and 'furthermore' will have very similar word vectors. We, believe this could be one of the reasons for the decreased performance of our models while using POS tags.

# 3 Model Architecture

The models implemented in solving these tasks range from simple machine learning algorithms in conjunction with different architectures that encode the given text into representation vectors, to simple neural models such as RNN and BiLSTMs to large language models such as RoBERTa.

## 3.1 Machine Learning Algorithms:

1. *Vector Representations:* We used multiple methods to extract vector representations from the data:
   (a) TF-IDF vectors
   (b) Open AI Embeddings: text-embedding-ada-002[8]
   (c) Sentence Transformers: all.mpnet-base-v2[9]
2. *Classification Algorithms:* We used the vector embeddings to train the following models
   (a) Logistic Regression
   (b) SVM Classifier
   (c) Naive Bayes Classifier
   (d) Random Forest Classifier
   (e) XGBoost Classifier

## 3.2 RNN and BiLSTM

In our baseline model, we had two RNN/BiLSTM layers with dropout regularization and Dense layers. Dropout is being used to combat overfitting. We improve our model as shown in figure 1, by adding CNN layers, Max Pooling layer(size = 5 for task P1,P2 and size = 3 for P3) and a Global Max Pooling layer. Sparse Categorical cross-entropy is used as the loss function. Specifically, the final dense layer uses 'tanh' activation function instead of 'relu'.

## 3.3 Large Language Models

Large Language models like RoBERTa, GPT-2 and BERT were optimized to obtain the highest accuracies for the tasks.

# 4 Results

The performance of different machine learning models for two different projects (P1, P2 and P3) are compared. These models are evaluated based on various metrics like Accuracy, Precision, Recall, and F1 Score, which are key metrics in evaluating the performance of a classification model.

In P1, the XGBoost Classifier performed best among the baseline models for both the balanced data with TF-IDF vectors and binary random with TF-IDF vectors. Among the improvements, the POS (Part-of-Speech) with CNN+BiLSTM+MP (Convolutional Neural Network + Bidirectional Long Short-Term Memory + Max Pooling) achieved the highest results.

In P2, the baseline results show that the model "roberta-base" with the RoBERTa architecture performed the best. For the improved models, BERT-based models with additional features such as Stylistic Features and Character n-gram showed exceptional performance. Logistic Regression and Naive Bayes Classifier with BERT+Style+Character n-gram achieved the highest scores.

In P3, the highest baseline accuracy is achieved by the RoBERTa model with roberta-base embedding (84.4%). Maximum accuracy among the improved models is achieved by the XGBoost Classifier with Stemmming + TF-IDF vectors (90.01%).

# 5 Discussion and Error Analysis

The tables provided detail the performance of various models on three tasks, labeled as P1, P2, and P3. The evaluation metrics used are accuracy, precision, recall, and F1 score. The models and configurations vary across tasks, but the baseline models are generally the same for P1 and P2, including Logistic Regression, Random Forest Classifier, XGBoost Classifier, Naive Bayes Classifier, and SVM Classifier.

The most striking results are as follows:

1. In P1, XGBoost Classifier achieved the highest accuracy (63.4%) among the baseline models when using balanced data with TF-IDF vectors. However, the best model in terms

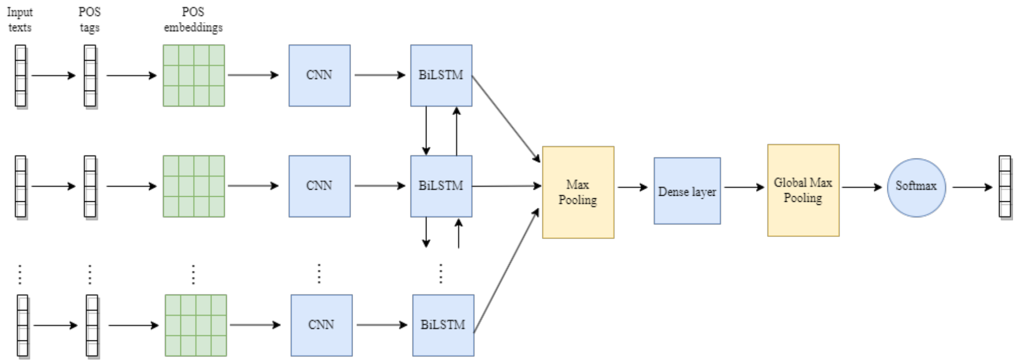


Figure 1: BiLSTM model architecture

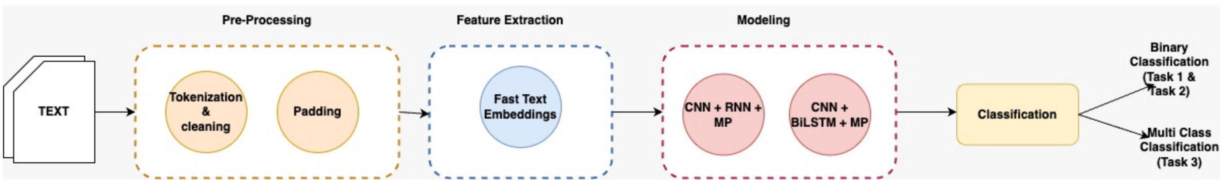


Figure 2: Deep Learning Pipeline

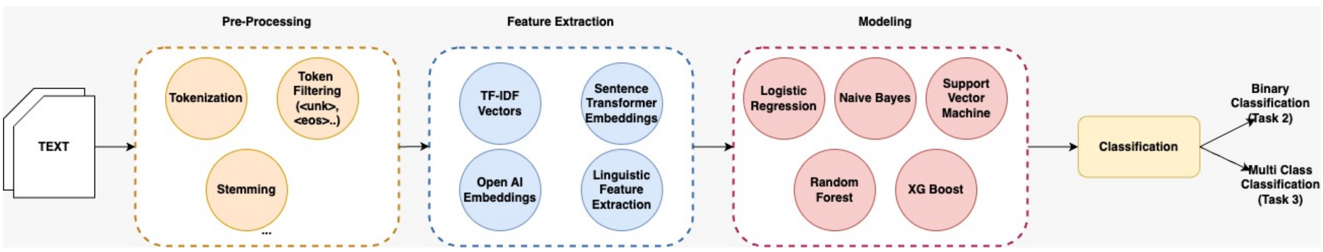


Figure 3: Machine Learning Pipeline

of F1 score in the improvements section was a combination of Convolutional Neural Network (CNN), BiLSTM, and Max Pooling (MP) model with part-of-speech (POS) features, which reached an F1 score of 75.01%.

2. In P2, RoBERTa with roberta-base embedding performed exceptionally well, with accuracy reaching 92.3%. In the improvements section, Logistic Regression with BERT Embeddings + Stylistic Features or BERT+Style +Character n-gram feature configuration also achieved a high accuracy of 99.16%.

3. For P3, RoBERTa with roberta-base encoding achieved the highest accuracy (84.4%) among all the models. In the improvements section, XGBoost Classifier with Stemming + TF-IDF vectors achieved the highest accuracy of 90.01%.

It's worth noting that some models perform significantly better with certain data configurations or embeddings, highlighting the importance of feature selection and model choice depending on the specific task.

## 6 Conclusion

Our investigation into authorship attribution for neural-based language models has yielded significant insights. The implemented models, particularly when incorporating stylometric features alongside BERT, exhibited impressive performance. Notably, models such as GPT2, GPT3, and InstructGPT generated higher-quality texts that could frequently deceive machine classifiers.

The disparity in text generated by AI and humans is quite high, which makes authorship attribution a relatively easier task. We can see that there is a lot of scope to improve text generated by text generators and make them more human-like. Until then, future direction would be to improve upon the produced models using other linguistic features and generalizing it by adding more data.

## 7 Contribution

I have worked on the BiLSTM and RNN models. Built the baseline models and improved them for all tasks P1, P2, P3.

## 8 References


1. https://www.researchgate.net/publication/325564535_Long_short-term_memory_for_machine_remaining_life_prediction
2. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8663814
3. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8631977
4. https://digital.csic.es/bitstream/10261/268091/1/EICC_2022_paper_7792%283%29.pdf
5. https://shriya999.github.io/pdfs/ICISDM_19.pdf
6. https://arxiv.org/pdf/1902.09723.pdf
7. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/reports/2760185.pdf
8. https://aclanthology.org/2020.icon-main.16.pdf
9. https://aclanthology.org/C18-1029.pdf
10. https://platform.openai.com/docs/guides/embeddings/what-are-embeddings
11. https://www.sbert.net/docs/pretrained_models.html

| P1 | | | | | |
|---|---|---|---|---|---|
| Baseline Results | | | | | |
| data config | Model | Accuracy | Precision | Recall | F1 Score |
| balanced data with TF-IDF vectors | Logistic Regression | 44.4 | 44.3 | 44.4 | 44.4 |
| | Random Forest Classifier | 52.5 | 52.5 | 52.5 | 52.4 |
| | XGBoost Classifier | 63.4 | 63.4 | 63.4 | 63.3 |
| | Naive Bayes Classifier | 34.5 | 34.0 | 34.5 | 33.9 |
| | SVM Classifier | 49.3 | 48.9 | 49.3 | 44.8 |
| binary random with TF-IDF vectors | Logistic Regression | 90.8 | 53.7 | 50.0 | 47.7 |
| | Random Forest Classifier | 91.3 | 95.6 | 52.1 | 51.9 |
| | XGBoost Classifier | 92.5 | 84.1 | 62.9 | 67.8 |
| | Naive Bayes Classifier | 58.7 | 51.2 | 53.7 | 44.9 |
| | SVM Classifier | 75.3 | 49.8 | 49.6 | 49.3 |
| tokens | RNN | 48.83 | 48.67 | 48.01 | 48.33 |
| tokens | BiLSTM | 51.23 | 50.98 | 50.91 | 50.94 |
| Improvements | | | | | |
| data config | Model | Accuracy | Precision | Recall | F1 Score |
| tokens | CNN+RNN+MP | 50.41 | - | - | 66.72 |
| fastText tokens | CNN+RNN+MP | 49.79 | - | - | 65.61 |
| POS | CNN+RNN+MP | 51.82 | 48.73 | 76.7 | 59.6 |
| tokens | CNN+BiLSTM+MP | 64.95 | - | - | 70.12 |
| fastText tokens | CNN+BiLSTM+MP | 61.2 | - | - | 67.76 |
| POS | CNN+BiLSTM+MP | 70.34 | 64.58 | 89.65 | 75.01 |

Table 1: P1 Results

| Embedding | Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|---|
| P2 | | | | | |
| Baseline Results | | | | | |
| text-embedding-ada-002 | Logistic Regression | 85.4 | 85.6 | 85.2 | 85.3 |
| | Random Forest Classifier | 81.0 | 81.3 | 80.8 | 80.8 |
| | XGBoost Classifier | 81.7 | 85.6 | 85.2 | 85.3 |
| | Naive Bayes Classifier | 78.5 | 78.5 | 78.6 | 78.5 |
| | SVM Classifier | 74.9 | 76.1 | 74.5 | 74.3 |
| all-mpnet-base-v2 | Logistic Regression | 84.1 | 84.2 | 84.3 | 84.1 |
| | Random Forest Classifier | 89 | 89.3 | 88.8 | 88.9 |
| | XGBoost Classifier | 91.2 | 91.4 | 91.1 | 91.2 |
| | SVM Classifier | 76.3 | 76.4 | 76.4 | 76.3 |
| | Naive Bayes Classifier | 72.9 | 77.5 | 72.2 | 71.3 |
| TF-IDF vectors | Logistic Regression | 84.1 | 84.2 | 84.3 | 84.1 |
| | Random Forest Classifier | 89 | 89.3 | 88.8 | 88.9 |
| | XGBoost Classifier | 91.2 | 91.4 | 91.1 | 91.2 |
| | SVM Classifier | 76.3 | 76.4 | 76.4 | 76.3 |
| | Naive Bayes Classifier | 72.9 | 77.5 | 72.2 | 71.3 |
| tokens | RNN | 45.45 | 44.98 | 45.23 | 45.10 |
| tokens | BiLSTM | 48.03 | 48.01 | 47.93 | 47.96 |
| roberta-base | RoBERTa | **92.3** | 92.4 | 92.0 | 92.2 |
| Improvements | | | | | |
| Stemmming + TF-IDF vectors | Logistic Regression | 90.00 | 90.15 | 90.04 | 90.00 |
| | Random Forest Classifier | 95.50 | 95.50 | 95.50 | 95.50 |
| | XGBoost Classifier | 95.83 | 95.94 | 95.80 | 95.83 |
| | SVM Classifier | 91.50 | 91.56 | 91.53 | 91.50 |
| | Naive Bayes Classifier | 74.5 | 78.94 | 74.24 | 73.34 |
| bert-uncased | BERT | 99.00 | 98.35 | 99.66 | 98.99 |
| Stylistic Features | Logistic Regression | 62.16 | 62.24 | 62.21 | 62.15 |
| | Random Forest Classifier | 88.50 | 88.51 | 88.49 | 88.50 |
| | XGBoost Classifier | 89.16 | 89.22 | 89.19 | 89.17 |
| | Naive Bayes Classifier | 59.50 | 59.49 | 59.49 | 59.49 |
| BERT Embeddings +Stylistic Features | Logistic Regression | 99.16 | 99.17 | 99.18 | 99.17 |
| | Random Forest Classifier | 88.50 | 88.56 | 88.47 | 88.49 |
| | XGBoost Classifier | 89.16 | 89.22 | 89.19 | 89.17 |
| | Naive Bayes Classifier | 99.00 | 99.00 | 99.01 | 99.00 |
| Character n-gram | Logistic Regression | 66.33 | 67.39 | 66.16 | 65.66 |
| | Random Forest Classifier | 91.50 | 91.53 | 91.48 | 91.49 |
| | XGBoost Classifier | 91.33 | 91.39 | 91.31 | 91.33 |
| | Naive Bayes Classifier | 61.16 | 61.27 | 61.22 | 61.14 |
| BERT+Style +Character n-gram | Logistic Regression | 99.16 | 99.17 | 99.18 | 99.17 |
| | Random Forest Classifier | 92.66 | 92.66 | 92.67 | 92.67 |
| | XGBoost Classifier | 89.16 | 89.22 | 89.19 | 89.17 |
| | Naive Bayes Classifier | 99.00 | 99.00 | 99.01 | 99.0 |
| tokens | GPT2 | 83.00 | 83.00 | 84.00 | 83.00 |
| tokens | CNN+RNN+MP | 51.32 | - | - | 67.45 |
| fastText tokens | CNN+RNN+MP | 47.58 | - | - | 48.23 |
| POS | CNN+RNN+MP | 53.08 | 52.17 | 96.55 | 67.75 |
| tokens | CNN+BiLSTM+MP | 83.28 | - | - | 84.29 |
| fastText tokens | CNN+BiLSTM+MP | 68.33 | - | - | 69.65 |
| POS | CNN+BiLSTM+MP | 81.82 | 83.33 | 80.46 | 81.87 |

Table 2: P2 Results

| P3 | | | | | |
|---|---|---|---|---|---|
| Embedding | Model | Accuracy | Precision | Recall | F1 Score |
| text-embedding-ada-002 | Logistic Regression | 77.4 | 77.7 | 77.5 | 77.5 |
| | Random Forest Classifier | 55.1 | 54.5 | 54.3 | 53.7 |
| | XGBoost Classifier | 66.8 | 66.5 | 66.7 | 66.5 |
| | Naive Bayes Classifier | 56.8 | 56.9 | 56.8 | 56.7 |
| | SVM Classifier | 69.0 | 69.4 | 69.1 | 68.8 |
| all-mpnet-base-v2 | Logistic Regression | 65.9 | 65.4 | 65.8 | 65.6 |
| | Random Forest Classifier | 50.0 | 47.3 | 49.5 | 47.25 |
| | XGBoost Classifier | 59.1 | 58.2 | 58.8 | 58.4 |
| | SVM Classifier | 59.2 | 58.9 | 59.2 | 58.8 |
| | Naive Bayes Classifier | 46.7 | 45.2 | 46.4 | 45.1 |
| TF-IDF vectors | Logistic Regression | 72.8 | 73.8 | 72.9 | 73.3 |
| | Random Forest Classifier | 60.1 | 71.7 | 60.1 | 57.3 |
| | XGBoost Classifier | 80.8 | 81.4 | 80.8 | 81.0 |
| | SVM Classifier | 40.1 | 58.1 | 40.7 | 37.5 |
| | Naive Bayes Classifier | 72.1 | 76.3 | 72.1 | 72.1 |
| tokens | RNN | 41.15 | 41.19 | 41.01 | 41.1 |
| tokens | BiLSTM | 43.32 | 43.12 | 43.01 | 43.06 |
| roberta-base | RoBERTa | **84.4** | 85.1 | 84.7 | 84.6 |
| Improvements | | | | | |
| Stemmming + TF-IDF vectors | Logistic Regression | 81.41 | 82.59 | 81.54 | 81.84 |
| | Random Forest Classifier | 87.44 | 88.83 | 87.40 | 87.75 |
| | XGBoost Classifier | 90.01 | 90.34 | 90.02 | 90.13 |
| | SVM Classifier | 85.38 | 87.28 | 85.47 | 85.93 |
| | Naive Bayes Classifier | 56.27 | 65.22 | 57.05 | 52.29 |
| bert-uncased | BERT | 88.57 | 88.67 | 88.62 | 88.80 |
| Stylistic Features | Logistic Regression | 43.63 | 43.63 | 43.75 | 42.50 |
| | Random Forest Classifier | 82.73 | 83.14 | 82.80 | 82.86 |
| | XGBoost Classifier | 83.41 | 83.70 | 83.52 | 83.57 |
| | Naive Bayes Classifier | 44.28 | 50.60 | 43.66 | 39.52 |
| BERT Embed-dings +Stylistic Features | Logistic Regression | 88.48 | 88.74 | 88.53 | 88.58 |
| | Random Forest Classifier | 89.05 | 89.39 | 89.10 | 89.19 |
| | XGBoost Classifier | 87.38 | 88.92 | 87.38 | 87.84 |
| Character n-gram | Logistic Regression | 60.21 | 60.10 | 60.10 | 59.04 |
| | Random Forest Classifier | 85.26 | 85.98 | 85.31 | 85.37 |
| | XGBoost Classifier | 86.25 | 86.66 | 86.37 | 86.44 |
| | Naive Bayes Classifier | 44.28 | 50.60 | 43.66 | 39.52 |
| BERT+Style +Charac-ter n-gram | Logistic Regression | 88.45 | 88.71 | 88.50 | 88.55 |
| | Random Forest Classifier | 89.05 | 89.37 | 89.12 | 89.20 |
| | XGBoost Classifier | 87.38 | 88.92 | 87.38 | 87.84 |
| tokens | CNN+RNN+MP | 66.54 | - | - | 66.72 |
| fastText tokens | CNN+RNN+MP | 47.48 | - | - | 47.50 |
| tokens | CNN+BiLSTM+MP | 71.34 | 72.23 | 71.46 | 71.30 |
| fastText tokens | CNN+BiLSTM+MP | 65.47 | - | - | 66.87 |
| POS | CNN+BiLSTM+MP | 64.80 | 66.75 | 64.89 | 64.94 |

| Task | Model | Accuracy | Precision | Recall | F1 Score |
| --- | --- | --- | --- | --- | --- |
| P1 balanced data | XGBoost | 76.8 | 76.9 | 76.8 | 76.8 |
| P1 binary random | XGBoost | 66.6 | 62.2 | 61.8 | 62.0 |
| P2 | Logistic Regression | 91.9 | 91.1 | 92.0 | 91.9 |
| | XGBoost Classifier | 94.9 | 95.0 | 95.0 | 94.9 |
| | Random Forest Classifier | **95.4** | **95.9** | 95.4 | 95.4 |
| | SVM Classifier | 90.9 | 90.9 | 90.9 | 90.9 |
| | Naive Bayes Classifier | 91.4 | 92.7 | 91.3 | 91.3 |
| P3 | Logistic Regression | 80.2 | 80.4 | 80.3 | 80.4 |
| | XGBoost Classifier | **83.2** | 83.6 | 83.3 | 83.3 |
| | Random Forest Classifier | 81.9 | **83.9** | 81.9 | 82.2 |
| | SVM Classifier | 39.1 | 41.2 | 38.8 | 30.7 |
| | Naive Bayes Classifier | 79.5 | 81.3 | 79.6 | 80.0 |
| Improvements | | | | | |
| P1 balanced data | BiLSTM | 81.58 | 87.10 | 72.97 | 79.41 |
| P1 binary random | BiLSTM | 88.07 | 86.78 | 80.25 | 83.39 |
| P2 | BiLSTM | 94.06 | 94.06 | 94.06 | 94.06 |
| P3 | BiLSTM | 72.25 | 75.57 | 74.90 | 74.10 |

Table 3: Reddit Results with TF-IDF Vectorization