Authorship Identification using Enron Email Dataset

Chandrahas Reddy Mandapati
cmandap@ncsu.edu
North Carolina State University
Raleigh, NC, USA

Pathey Vipulbhai Shah
pshah7@ncsu.edu
North Carolina State University
Raleigh, NC, USA

Pratyush Ajay Vaidya
pavaidya@ncsu.edu
North Carolina State University
Raleigh, NC, USA

KEYWORDS

datasets, neural networks, machine learning, text classification

1 PROBLEM STATEMENT

With the internet becoming increasingly accessible, the number of emails sent and received globally has increased each year. Despite the host of benefits offered by email, there are certain disadvantages, that are to be dealt with a great deal of attention such as privacy and data breaches. The rise of email usage has created an anonymous environment for cyber-criminals to take advantage of nascent individuals. One of the most common practices has been to send suspicious emails by gaining access to a user with the interest of stealing data. In this project, we make an effort to identify the author of emails by making use of the prominent Enron Email dataset which plays an important role in differentiating original emails from fraudulent ones.

Authorship identification implies identifying the person who has written the text. The task can be achieved by learning a person's writing styles, use of special characters, use of phrases, and use of special words sentiments. The goal of this project is to identify the original sender of emails to ensure that users' privacy was not compromised.

2 RELATED WORK

Enron dataset is a large corpus dataset used in model training of various email-based classification NLP applications. The application of stylometric analysis techniques[1] on content generates lexical based, content-based, syntactic based and semantic-based features from a given piece of text to identify an author based on writing style. Another set of features can be obtained by assigning weights to each term in the email body by using the tf-idf transformation scheme[3]. For classification of the mails, classifiers such as random forest classifier, decision tree classifier, AdaBoost and gradient boosting classifier[4] are suggested. Anirudh Harisinghaney[2] performed text and image-based spam email classification using KNN, Naive Bayes and reverse DBSCAN algorithm on Enron Corpus. We used these approaches to extract stylometric based features and tf-idf values which we are using to train our classification models.

3 APPROACH

Based on Literature survey and previous work done for the authorship identification task we are using two types of approaches for our task. In our first approach we are extracting stylometric features along with TF-IDF and count vectors and training Random Forest Classifier and Multinomial Naive Bayes Classifier, and Support Vector Classifier using those features. While in second approach we are first generating tokens from dataset using Keras tokenizer then by using glove we are generating word-embeddings, And we are training Recurrent Neural network based models using this embedding matrix. "Here are the details of all the models that we have been using. "

Random Forest Classifier. Random Forest Classifier is an ensemble machine learning method for classification tasks that operate by constructing multiple decision trees at the time of training. A random forest classifier combines the output of multiple decision tree classifiers to generate the final output this technique makes Random Forest Classifier resilient to overfitting the training data. Since we have a large number of features compared to the amount of training data available overfitting is a serious issue for our model. The Random Forest classifiers are effective in dealing with the high dimensional noisy data in text classification. We are applying gridsearchCV on sets of hyperparameters to find out the best set of parameters for the Random Forest Classifier.

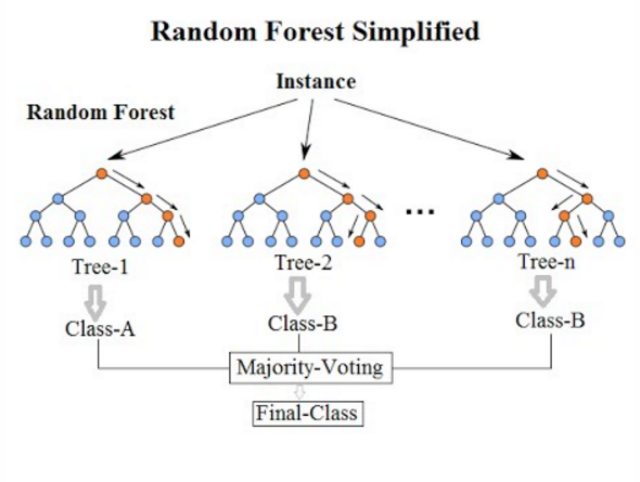

Figure 1: Random Forest Classifier architecture

Support Vector Machine. Support Vector Machines are supervised learning models where each data point is plotted in n-dimensional space with feature value as its coordinate value and performs classification by finding the right hyperplane which splits the data points into classes. SVM is effective in cases where the number of dimensions is greater than the number of samples. SVMs naturally support powerful kernel functions, which implicitly project data through a non-linear transformation to high dimensional spaces very efficiently. One remarkable property of SVMs is that their ability to learn can be independent of the dimensionality of the feature space. This property plays a vital role, especially in text classification where the feature space can be greater than the number of samples.

Multinomial Naive Bayes. Multinomial Naive Bayes is considered a probabilistic approach to text classification in the context of acknowledging the term frequencies. This classifier achieves well on discrete types as the word frequencies in a document. The key advantage of the Multinomial Naive Bayes classifier is it can be easily scaled with large sets of features. As the Multinomial Naive Bayes classifier uses Bayesian probabilities to perform classification tasks it is a well-suited model for the text classification task. Another key advantage of using the Multinomial Naive Bayes classifier is it is a simple model and can be trained quickly.

Long short-term memory. LSTMs stand for Long Short Term Memory which is a special kind of Recurrent Neural Network which enable a model to remember long time dependencies in a sequence(text sequence in this case). LSTMs can help with the classification as they can keep track of the past events while processing a text and is, therefore, better at dealing with contexts. LSTM models are effective in Natural language processing tasks because of their recurrent neural network architecture along with forget and output gate it can very well control the flow of information throughout the network. Below is the architecture of LSTM model.

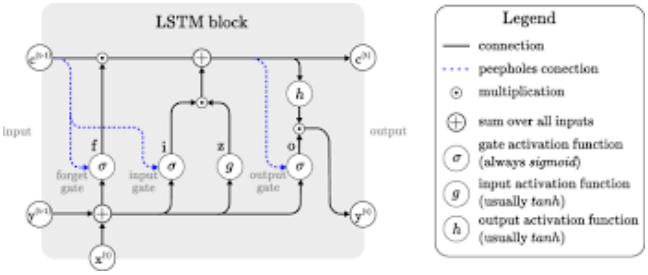

Figure 2: LSTM architecture

Bi-directional LSTM. The Bi-directional LSTM Model consists of two LSTM models one taking input in the forward direction while the other in the backward direction. Bi-directional LSTM has higher information flow than the vanilla LSTM model as it propagates information in both forward backward directions. It can understand language context better than the vanilla LSTM model because it can understand the predecessor and successor of a word. As the accuracy of the authorship identification task is dependent upon how well a model can understand the writing style of a person Bi-directional LSTM model is a better fit for the task compared to the vanilla LSTM model. We are using a Bi-directional LSTM model with a fully connected layer of 128 nodes and dropout layer with a 0.3 dropout value and an output layer of 10 nodes. The model summary of the Bi-directional LSTM model is presented in the figure ahead. The following are the sequence of layers built as part of the Bi-directional LSTM model.

- Embedding layer: The embedding layer enables us to convert each word into a fixed-length vector of defined size.
- Bidirectional LSTM: There are two layers included in it, where one taking the input in a forward direction, and the other in a backwards direction. Both try to identify sentence embeddings from word embeddings and vice versa.
- Dense Layer: A dense layer is a fully-connected layer, i.e. every neuron of the layer N are connected to every neuron of the layer N+1

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 1500)]	0
embedding (Embedding)	(None, 1500, 100)	2719200
bidirectional_1 (Bidirectional)	(None, 200)	160800
dense_2 (Dense)	(None, 128)	25728
dropout_1 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 10)	1290

Total params: 2,907,018
Trainable params: 2,907,018
Non-trainable params: 0

Figure 3: Bi-directional LSTM model summary

4 RATIONALE

Our task is to determine the author from the set of authors using the email body so it is a classification task. Our first task is to pre-process the data to extract features that we can use to train the models mentioned in the approach. As part of pre-processing, we extracted first the email content and author of the email from the dataset. Then we selected the users who have a higher number of emails than the threshold values. On those email content using Python libraries, we extracted different stylometric features along with TF-IDF vectors which are used to train the classifier model. These stylometric features are particularly useful for authorship identification tasks because they reflect the unique writing style of a person. For training the recurrent neural network-based models we tokenized the email content and generated word embeddings using the glove. As to train the Recurrent neural network-based model we need to convert our text into vectors, the key reason for using glove instead of word2vec is word2vec only relies on local content local information statistics while glove also takes global statistics into account as well. Due to this word embeddings generated using glove have a better vector representation of the text.

We are using Random Forest Classifier, SVM Multinomial Naive Bayes as our baseline model to measure the performance of RNN-based models Vanilla LSTM, Bidirectional LSTM hierarchical LSTM mdoel. As our baseline classification model, we are using SVM, Random Forest Naive Bayes to get baseline results. For the final classification task, we have trained RNN models like LSTM and bidirectional LSTM. We extracted a total of 27 stylometric features along with the TF-IDF vector from the initial dataset which we are using to train our baseline model.

Support Vector Machine. The problem that we are currently working on involves dealing with high dimension features be it Stylometric features, TFIDF features or Count vectorizer features. The feature dimension scales with the amount of text we are working on. Stated in this way, Support Vector Machines are good classifiers when we are working with higher dimensional datasets. As support

vector machines use hyperplane in n-dimensional vector space to classify the data it can be trained with lower training examples a high number of features. Another key advantage of using SVM is it uses support vectors to compute hyperplanes because of this approach it has fewer memory requirements compared to other models. The only major drawback of using the SVM model is for a large corpus with a high number of features it will take a larger training time and it is not feasible to add new features to the model. We can play around with the SVM kernel trick to improve training speed.

Random Forest Classifier. A random forest classifier is a popular classification algorithm that is quite straightforward to apply. The biggest advantage of using it is that it uses the bagging technique internally to combine the output from multiple individual decision trees decision stumps, making it resilient to over-fitting the dataset. Since we have a large number of features compared to available training data overfitting is a serious concern for our model. Due to ensemble methods used by Random Forest Classifier it reduces the probability of model over-fitting on training data. As accuracy of Random Forest classifier highly dependent upon selection of hyperparameters we applied GridSearchCV to find best set of hyper parameters for our classifier.

Multinomial Naive Bayes Classifier. Multinomial Naive Bayes is a powerful algorithm used for text classification problem. It is a good classifier for multi class classification problem. Since we are using 10 classes in our dataset Multinomial Naive Bayes classifier is a good fit our multi class classification problem. The key advantage of using a Multinomial Naive Bayes classifier is it uses a Bayesian probability model for predicting the probability of output tasks and which makes it more robust towards overfitting, which is a serious concern for our task. Another key advantage of using Multinomial Naive Bayes Classifier is it is a simple model and easy to train compared to SVM classifier.

LSTM. The classifier models which we used earlier (Random Forest, SVM, Multinomial Naive Bayes) uses extracted features from text to perform classification task, so the accuracy of classification is highly dependent upon the feature extraction task. Because of that, these classification models have limitations on the performance they can achieve. While the LSTM model uses word embeddings to represent text in vector space because of its recurrent neural network architecture LSTM based models performs well in natural language processing task compared to the other classification model.

Bi-directional LSTM. In the vanilla LSTM model flow of information is in the forward direction only because it can not understand the context of successor words which impacts the classification accuracy. Bi-direction LSTM model uses two-layer LSTM models one for forward propagation of information other for backward propagation of information. Due to this bi-directional LSTM layer, it can better understand the context of language compared to the vanilla LSTM model. In our authorship identification task, the accuracy of identification is highly dependent upon how well a model can understand the language context and writing style of a person. This makes bi-directional LSTM model the best choice for our task.

5 DATASET

The Enron Email Dataset is a collection of approximately half a million emails from about 150 users, mostly senior management of Enron, organized into folders owned by the Federal Energy Regulatory Commission after the downfall of Enron. In this project, we used a dataset composed by William W. Cohen.

The dataset of 1.77 GB consists of several folders where each folder represents a user and consists of several files depicting respective formatted email messages. Each message consists of a unique message-id, sender email address, list of recipients, date, email subject, folder to which this email belonged, name of the person who rolled out the email, name of the person to whom this email was intended, and the email content. The email content also includes the original message and forward message constructs if it's a forwarded message. As we try to identify the email author, the most relevant attribute of the above-mentioned is email content. Label Encoding is performed on the author name attribute to transform the categorical values into numerical values. The process involved in extraction is elaborated on in the subsequent section. The total number of emails in the raw dataset was approximately a hundred thousand and with the interest of data balance, we have extracted the top 10 authors with more data records. Finally, after cleaning and filtering the data, we ended up with 41254 emails from 10 authors and their distribution is shown in the figure ahead. The dataset split for the training and testing purposes is discussed in the subsequent section.

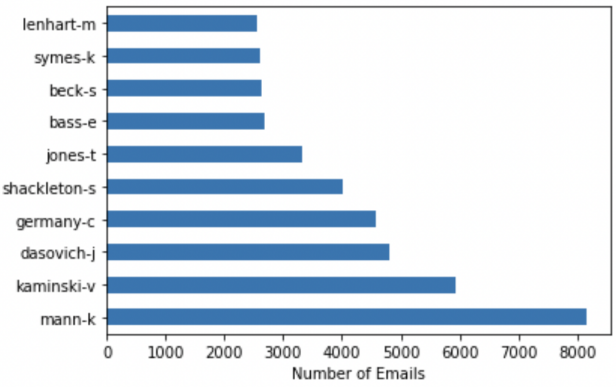

Figure 4: Number of emails per author

6 HYPOTHESIS

The primary hypothesis we try to define and investigate in this project is: Can we predict the author of an email provided with email content and a list of available authors. In other words, can we identify whether the originated email is fraud? The importance of the problem has been mentioned in the problem statement section.

7 EXPERIMENTAL DESIGN

The following are the steps involved in achieving the project's idea.

7.0.1 ***Data Extraction.*** Each sub-folder in the dataset represents a category of emails such as inbox, sent_emails, and deleted_items.

We considered files under sent, sent_items, and __sent__mail subfolders as they contain all the emails sent by that particular author. The main objective of this process is to create a dataset with directory names as labels and extract the content of all files under every directory. The output of this step is a CSV file containing the file content and author name acting as a label.

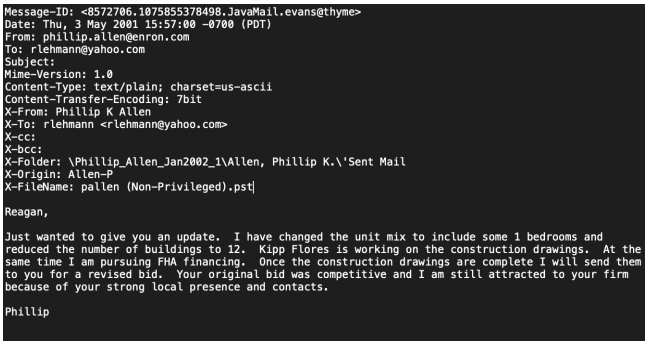

Figure 5: Sample Data Structure

7.0.2 ***Data Cleaning.*** In this process incomplete, inaccurate, incorrect records are identified as part of this process and any further discrepancies in the data are resolved. Email content consists of several unwanted email chains and forwarded messages apart from the actual email content. In addition to that, there are emails with no email content. We used python's email library to parse the file content and extract the sender's email address and email body. On top of the parsed content, we used regex to eliminate the unwanted chain conversations and forwarded messages. The records with no email content are removed, ensuring every record contains actual email content.

7.0.3 ***Feature Extraction.*** Writing styles play a vital role in identifying authors. In order to capture the writing styles, there has to be at least a threshold amount of data that is guaranteed in considerable documents. The email content is usually different from the document's vocabulary and structure. Therefore, email classification has to be dealt with differently. We try to obtain several stylometric features such as lexical, syntactic and semantic features. In addition to these features, we try to obtain several readability indexes. The following are the feature categories and features extracted as part of that category.

- **Sentence-Based Features** Sentence based features such as average sentence length, and the average number of sentences per paragraph are useful in capturing unique writing styles and differentiating authors. The number of sentences and the average length of sentences is extracted as sentence based features.
- **Word-based features** Different authors have different preferences of words to convey a particular thing. The choice of words used can also provide a hint of differentiation between the authors. The average number of words per sentence, number of words, frequency of most commonly used words, number of stop words, number of functional words, Type Token Ratio(TTR), hapax Token Ratio(HTR) are word specific features extracted.
- **Character-based features** Special characters and symbols used can also help identify the writing styles. Features extracted under this category are the number of characters, special characters, and the average number of characters per word.
- **Punctuation-based features** The number of commas, dots, question marks, exclamatory marks, colons, and semicolons is extracted as punctuation based features.
- **Syntactic-based features** Syntactic features are concerned mainly with the parts of speech tags and the accordance of sentence structure with the standard one. The average number of verbs and POS tags are the features extracted under this category.
- **Semantic-based features** Authors use different tones in conveying the same information, identifying the number of positive words and the number of negative words is useful in interpreting the tone of an author. Polarity score, number of positive words and negative words are extracted from the email content.
- **Readability indexes** Readability indexes represent what level of education will need to be able to interpret the text easily. Readability indexes are useful in differentiating authors based on their content complexity. smog, dale-chall, and Flesch–Kincaid readability indexes are calculated under this category.

7.0.4 ***Feature Transformation.*** Once all the necessary features have been obtained, we performed several feature transformation steps such as Standardization to scale the data points of different attributes to a relative comparable range, Label Encoding to convert non_numerical or categorical attributes to numerical attributes, Count Vectorization to obtain vectors from text indicating occurrences of every word, and TFIDF Vectorization to extract TF-IDF features.

7.0.5 ***Model Training.***

- **Baseline Models** We executed two different procedures for splitting the dataset into training and testing, they are the Stratified Cross-Validation and HoldOut approaches using the stratified sampling method. We selected approximately 42000 records from the raw dataset through data extraction and data cleaning with a minimum of 2500 records per author. The models were then trained using the stratified cross-validation in the first phase and using the 80:20 train test split using stratified sampling in the second phase along with the extracted features. In the interest of handling the class imbalance problem, stratified sampling method was employed in all the training phases.
- **LSTM Networks** As mentioned already, we had 42k emails from 10 different authors. For network training, we split the data into an 80:20 ratio for the training and testing set. Furthermore, the training dataset is split again into an 85:15 ratio for training and holdout sets. We used holdout set to tune hyperparameters of our LSTM model. To train our model we first tokenized email content using Keras

tokenizer. Then we used glove to generate the word embeddings for each word token. Next we created embedding matrix for input data created embedding layer for LSTM Bi-directional LSTM model.

8 RESULTS

We have performed several phases of model training, experimenting with the count vectorization, TFIDF vectorization, and several extracted stylometric features.

8.0.1 ***TFIDF and Count Vectorization.*** We initiated the model training using TFIDF and Count vectorizers derived from the email content column. The TFIDF vectorization seems to perform well as compared to the Count vectorization and is expected because it not only considers the frequency of words in the document but also takes the importance of words relative to the corpus into consideration. On the other hand, Count vectorization represents the frequency of words and does not provide much about the significance. The results of the model evaluation are depicted in the figure below. We could see that Random Forest performed well on both the vectorizers when compared to the Multinomial Naive Bayes. Although Random Forest outperforms the Multinomial Naive Bayes model, the training time taken by Multinomial Naive Bayes is far less compared to the Random Forest.

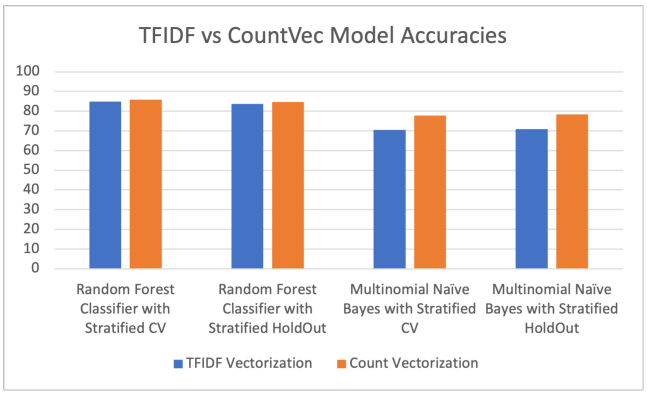

Figure 6: Baseline Models Accuracy with TFIDF, and Count vectorizers

8.0.2 ***TFIDF and Count Vectorization with Stylometric Features.*** In this phase of training, we included the extracted stylometric features in addition to the TFIDF and Count vectorizers. Adding stylometric features, accounted for the improvement in the accuracy of models. The accuracy achieved by Random Forest when evaluated on top of TFIDF and Count vectorizers is 83.7% and 83.5%, whereas Random Forest with stylometric features along with the TFIDF and Count vectorizers resulted in 84.82% and 83.78% accuracies respectively. An interesting observation is that the Multinomial Naive Bayes did not improve its performance rather, the accuracy decreased by 2%. This observation is valid since adding stylometric features introduced continuous attributes into the dataset and Multinomial Naive Bayes performs is well suited for categorical data attributes. The model accuracies when including stylometric features along with the TFIDF and Count vectorizers are presented in the below figure.

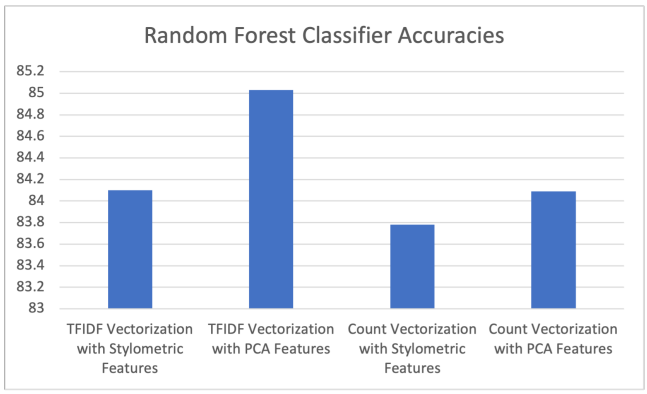

Figure 7: Random Forest Model Accuracy with Stylometric features before and after PCA

8.0.3 ***Principal Component Analysis on Stylometric Features.*** We extracted 27 stylometric features including syntactic, semantic, lexical and readability features which we thought would best represent the text. As the training model with all features did not provide much improvement in the model accuracy, we decided to perform Principal Component Analysis to examine the variance and spread of the data. We observed that only 15 among all the extracted features are explaining almost 96.78% of the variance therefore, we decided to iterate the model over the principal components. The extracted features corresponding to the principal components are listed in the table.

8.0.4 ***LSTM Model Training.*** We trained the Bi-directional LSTM model using 80 percent of total data as training data, To perform hyperparameter tunning we further divided our training data and kept 85 percent of total training data as actual training data and 15 percent of training data as validation data. Using validation accuracy during training we tuned hyperparameters for the LSTM model like the number of hidden layers at the end, Dropout ratio, and the number of neurons in each layer. We trained our Bi-directional LSTM model with 8 epochs, 128 batches and each epoch takes approximately 35-40 minutes to complete.

Random Forest performed well than the other three other algorithms. Although it is in accordance with the literature study that we performed, we achieved higher accuracy compared to the models we examined during the literature study. Random Forest acts as a good benchmark model in any classification task due to its ease of handling a different variety of features and is very quick to train. We have performed hyperparameter tuning to find the best estimator count and Although the accuracy gets increased with the increase in the number of estimators, the delta is not much worth noting.

To understand the role of the size of training data, We increased the training data to 43000 records from 18000 records of 10 authors and observe that there is an increase in the model's accuracy with the increase in the size of training data. The Support Vector

	0	1
0	PC0	num_words
1	PC1	smog_idx
2	PC2	num_pos_email
3	PC3	avg_num_chr_word
4	PC4	polarity_score
5	PC5	avg_verb_email
6	PC6	num_neg_word
7	PC7	smog_idx
8	PC8	num_neg_word
9	PC9	num_pos_email
10	PC10	num_pos_email
11	PC11	num_named_entity
12	PC12	avg_num_word_sen
13	PC13	highest_frequency
14	PC14	num_long_words

Figure 8: Features corresponding to PCA components

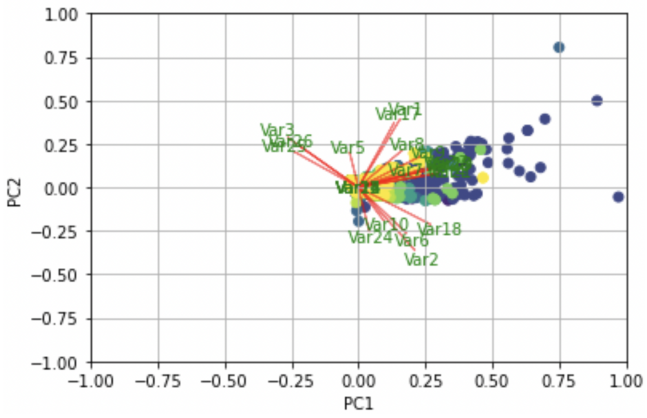

Figure 9: PCA Biplot overlaying a score plot and a loadings plot

Classifiers performed decently well on the training data but, takes huge time to train for large datasets. Multinomial Naive Bayes on the other hand, performed decently well irrespective of the size of training data and extracted stylometric features.

Finally, The results of Bi-directional LSTM have been presented. The classification report of bi-directional LSTM shows that the model achieved a precision, recall, and accuracy of 89%, 90%, and 90% respectively. The Bi-directional performed very well as compared to the baseline model evaluations.

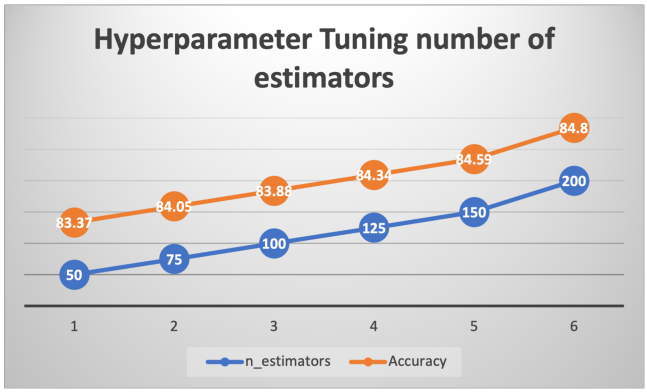

Figure 10: Random Forest Hyperparameter Tuning - Number of Estimators

```
Epoch 1/8
220/220 [==============================] - 2145s 10s/step - loss: 1.2707 - acc: 0.5686 - val_loss: 1.0286 - val_acc: 0.6441
Epoch 2/8
220/220 [==============================] - 1953s 9s/step - loss: 0.8664 - acc: 0.7029 - val_loss: 0.9690 - val_acc: 0.6663
Epoch 3/8
220/220 [==============================] - 1898s 9s/step - loss: 0.7051 - acc: 0.7586 - val_loss: 0.9121 - val_acc: 0.7063
Epoch 4/8
220/220 [==============================] - 1906s 9s/step - loss: 0.5974 - acc: 0.7986 - val_loss: 0.6837 - val_acc: 0.7720
Epoch 5/8
220/220 [==============================] - 1961s 9s/step - loss: 0.5215 - acc: 0.8246 - val_loss: 0.5992 - val_acc: 0.7948
Epoch 6/8
220/220 [==============================] - 2125s 10s/step - loss: 0.4559 - acc: 0.8472 - val_loss: 0.5824 - val_acc: 0.8039
Epoch 7/8
220/220 [==============================] - 2039s 9s/step - loss: 0.4015 - acc: 0.8672 - val_loss: 0.5629 - val_acc: 0.8158
Epoch 8/8
220/220 [==============================] - 2057s 9s/step - loss: 0.3563 - acc: 0.8820 - val_loss: 0.5469 - val_acc: 0.8227
```

Figure 11: Training Bi-directional LSTM model

	precision	recall	f1-score	support
0	0.81	0.83	0.82	545
1	0.96	0.97	0.97	517
2	0.91	0.85	0.88	888
3	0.92	0.85	0.88	912
4	0.77	0.89	0.82	670
5	0.94	0.97	0.96	1213
6	0.77	0.85	0.81	538
7	0.91	0.89	0.90	1591
8	0.96	0.89	0.92	811
9	0.96	0.97	0.96	566
accuracy			0.90	8251
macro avg	0.89	0.90	0.89	8251
weighted avg	0.90	0.90	0.90	8251

Figure 12: Classification report Bi-directional LSTM model

Model	Accuracy	Precision	Recall
Random Forest	85.09	85	85
Bi-directional LSTM	90	89	90

Table 1: Metrics of best baseline model and Bi-directional LSTM.

9 DISCUSSION

One interesting difference between email and document classification is the amount of content available. Working with short content will provide a low number of features and doesn't provide enough word co-occurrence. Therefore, preprocessing plays an important role to extract the best out of short content. Our baseline models

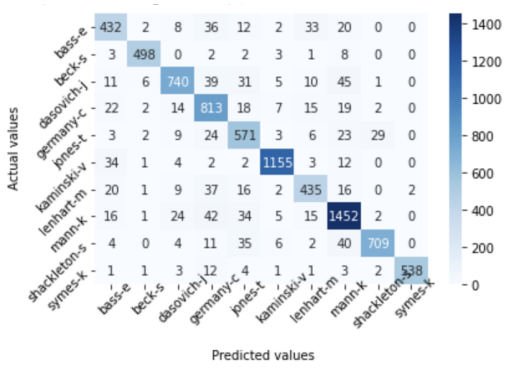

Figure 13: Confusion matrix for Bi-directional LSTM model

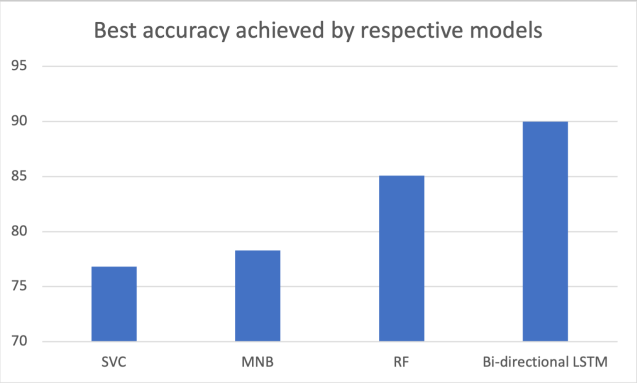

Figure 14: Best accuracy achieved by respective models

performed well overall and the Random Forest Classifier achieved the highest accuracy of 85.09% when trained and evaluated on top of the TFIDF vector along with stylometric features. Prior studies in the field of text classification or author attribution especially, include the models that were either trained on TFIDF and Count vectorizers or trained on stylometric features independently. In our proposed approach, we combined stylometric features with TFIDF features and achieved higher accuracy compared to the models implemented previously in this field.

The Random Forest classifier outperformed every other baseline model in different categories of model training. We noticed that the accuracy of Random Forest was increasing with the increase in the number of estimators and achieved 84.8% accuracy with 150 estimators. One interesting observation to be noted is that the increase in accuracy with the increase in the number of estimators is very less.

The accuracy of Bi-directional LSTM is significant as compared to the baseline models. Also, based on the confusion matrix we can see that authors Germany-eg, Mann-k, and jones-t have more general writing styles as they have more misclassifications among all authors. In addition to that authors Symes-k,beck-s have the most unique writing styles as they have got the highest recall and precision. Bi-directional LSTM achieved an accuracy of 90% which is exceptional considering the limited computing resources and the number of epochs for which the model is trained. We know that neural networks are more effective in the context of text classification and can work well with imbalanced data. Although stated in this way, we won't be able to make the most out of the model if undertrained.

Precision and recall are the most relevant metrics for any model evaluation and are important in determining authenticity. Precision and recall for Random Forest Classifier are 85% and 84% respectively, which is good for any machine learning model in the context of text classification. On the other hand, the precision and recall for Bi-directional LSTM are 89% and 90% respectively. Finally, we believe that the Bi-directional model performed really well given the amount of training and can evolve into a great model when trained for more additional epochs, therefore, confirming our hypothesis that was mentioned is supported by our model. Due to high recall and precision, our model will be able to correctly identify fraudulent emails and ensure that users' privacy is not compromised.

10 CONCLUSION

In this project, we suggested a novel approach to extracting stylometric features along with TF-IDF Count Vectorization to classify the author of an email. We tried several classification models like SVM, Random Forest Classifier, and Multinomial Naive Bayes classifier on these extracted features. The results we achieved using these classification algorithms indicated that the features were good identifiers to predict the author of an email. One novel aspect which is worth noting is making use of readability indexes as they covey the complexity of sentence structures and can well differentiate different writing styles. The important thing we noticed while working on the classification models was that the accuracy of the results was highly dependent on the data preprocessing and feature extraction task. We learned how to use several regex patterns to make sure that the forward chains and any additional signatures are eliminated and the extracted content only belongs to the author.

As our task was an NLP task we used recurrent neural network-based architectures like LSTM, Bi-directional LSTM by generating word embeddings from the email task. We learned the Bi-directional architecture and how to tweak the same to add more layers on top of an existing model. We used a novel approach of using two LSTM layers (forward backwards) so the model can better understand the context of language. Training LSTM models was a challenging task as it requires high computation power, we used the Google collaboratory tool to train our LSTM model for 8 epochs. The LSTM model gave promising results with good accuracy, F1-Score and we can train the LSTM model for the higher number of epochs to obtain even better results.

Because of the limitation in computational power and amount of datasets available we did 10 authors authorship identification tasks and for future work we can include a higher number of classes and use state-of-the-art NLP models like BERT to achieve better results. From this project, we can say that authorship identification from a small corpus of text like email is a difficult task but if we can extract appropriate features with a bi-directional LSTM model and given sufficient training examples we can achieve good results in classifying authors of an email.

11 ONLINE COLLABORATION

The following are the dates and times when our team has met and everyone has attended all the meetings.

- March 26th,2022 4:00pm - 6:30pm
- March 31st,2022 4:00pm - 6:30pm
- April 7th,2022 11:00am - 2:30pm
- April 9th,2022 6:30pm - 8:30pm
- April 15th,2022 10:00am - 1:30pm
- April 16th,2022 4:00pm - 6:30pm
- April 22nd,2022 4:00pm - 6:30pm
- April 23rd,2022 5:00pm - 6:30pm

12 CODE LINK

https://github.ncsu.edu/pshah7/CSC522_S22_P12

REFERENCES

[1] Ahmed Abbasi and Hsinchun Chen. 2008. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)* 26, 2 (2008), 1–29. https://doi.org/10.1145/1344411.1344413

[2] Anirudh Harisinghaney, Aman Dixit, Saurabh Gupta, and Anuja Arora. 2014. Text and image based spam email classification using KNN, Naïve Bayes and Reverse DBSCAN algorithm. In *2014 International Conference on Reliability Optimization and Information Technology (ICROIT)*. 153–155. https://doi.org/10.1109/ICROIT.2014.6798302

[3] Sarwat Nizamani and Nasrullah Memon. 2013. CEAI: CCM-based email authorship identification model. *Egyptian Informatics Journal* 14, 3 (2013), 239–249. https://doi.org/10.1016/j.eij.2013.10.001

[4] LZ Wang. 2017. News authorship identification with deep learning. https://cs224d.stanford.edu/reports/ZhouWang.pdf