Machine Learning – Project Report
Authorship Attribution Using Machine Learning Techniques

Ashwin Srinath Sureshkumar

(N16317183)

1.Definition

1.1 Problem Statement

The Problem that I am trying to tackle in this project is of Authorship Attribution. Authorship attribution means given a set of documents from a set of authors, create a system which, given a new unseen document, can tell the original author of that document [1]. These systems have become extremely popular now-a-days. The primary task at hand is to define an appropriate characterization of the text files so that it captures the writing styles of the authors efficiently. This is a problem that can readily be framed as a text classification task, "where author represents a class (label) of a given text [7]," One important scenario where such systems are being used is the identification of disputed documents. The problem arises when 2 or more people claim the authorship for a document. Another scenario could be to attribute the old historical pieces of writings to different eras, or perhaps the original author as well. Hence there is a need to have strong authorship attribution systems.

1.2 Overview

A couple of decades ago, author attribution analysis was typically applied to longer texts, such as books and letters with a minimum length of 250 words that was required for the stylometric characteristics to be apparent. However, recent research has demonstrated that with the

implementation of various new machine and deep learning techniques the word length can be reduced to about 25 words to successfully determine the author of a tweet/text/passage/poem etc [9]. To tackle this problem, I plan to use 2 different techniques, one is CNN based approach that uses CNN with character n-grams as input and the second method is an SVM based approach that makes use of a feature set comprising the words in the text excerpts. In doing this project, I have made use of a dataset containing 68,000 sentence-long excerpts from 8 authors along with the associated labels which identifies the author of every excerpt. This dataset was made with the help of books which were downloaded from Project Gutenburg (https://www.gutenberg.org/).

The dataset is initially preprocessed so that the chapter headings have been removed from the files prior to the actual analysis. Text excerpts were taken from multiple works, to build a balanced dataset, for authors whose novels tended to be shorter in length. The novels used to create the dataset are given in Figure 1:

Author	Novel	Genre	Year
Louisa May Alcott	Little Women	Coming of Age	1869
Jane Austen	Emma	Romance	1815
Jane Austen	Pride and Prejudice	Romance	1813
Charlotte Bronte	Jane Eyre	Gothic Romance	1847
Wilkie Collins	The Woman in White	Mystery	1859
Arthur Conan Doyle	The Hound of the Baskervilles	Mystery	1902
Arthur Conan Doyle	A Study in Scarlet	Mystery	1887

Arthur Conan Doyle	The Sign of the Four	Mystery	1890
L.M. Montgomery	Anne of Avonlea	Coming of Age	1909
L.M. Montgomery	Anne of Green Gables	Coming of Age	1908
Bram Stoker	Dracula	Horror	1897
Mark Twain	The Adventures of Huckleberry Finn	Adventure	1984
Mark Twain	The Adventures of Tom Sawyer	Adventure	1976

Figure 1: Novels used to create the author attribution dataset

1.3 Project Design:

1) Collect, explore and preprocess the author attribution data;

2) Randomly split the dataset into training and test subsets using an 80%/20% split.

3) Create the n-gram character sequences and bag-of-words feature sets.

4)Fit and tune a CNN to the (training) n-gram character sequences.

5) Fit and tune an SVM to the bag-of-words (training) feature set.

6) Using the test dataset, evaluate and compare the fitted models.

Yet making the whole model of authorship attribution, effective and user- friendly so that it can be used from small text documents to even large-scale novels or books.

1.4 Metrics

Performance of the models is evaluated based on the following metrics:

• Accuracy: Proportion of all test cases correctly classified by the model;

• Weighted average precision: The weighted average of the (test) precision values for all classes, where precision is the probability an example is actually labelled x given it is predicted to be x;

• Weighted average F1 score: The weighted average of the (test) F1 score values for all classes. F1 score is a metric that combines precision and recall (the probability an example is predicted to be x given it is actually labelled x) into a single value.

• Training time: Time required to train the model against the entire training dataset.

• Prediction time: Time required to predict the class of all datapoints in the test dataset.

•Confusion matrix: It provides a visual comparison of the actual versus predicted labels of examples at a class level.

Since the dataset is first preprocessed before the actual analysis is conducted the accuracy metric is enough to evaluate the validity of the machine learning model. However, I have considered other metrics also in order to further evaluate the validity and performance. The confusion matrix has also been calculated to determine if there are any systematic patterns displayed in the misclassification of the text snippets.

2.Methodology

2.1 Data Preprocessing

The dataset used here, that is in my case the books of those 8 authors are used for the processing, this consisted of 68,000 sentence-long snippets along with their respective labels identifying the individual authors. This was then normalized to 8,500 text excerpts per author so that no one author has higher probability of being selected.

I made sure that there were no corrupt labels. There were also no typos or spelling mistakes in the final dataset. In the preprocessing part all the characters were converted to lowercase, all the chapter/section headings were removed along with invalid characters and white spaces. Every text snippet was split into individual words, hence the word count, character count and average word length, was found. It was seen that the average excerpt is 17.88 words long and most of the snippets were smaller than 100 words in length. There was no threat from any outliners. This data was then split (80%/20%) into training and test set. Finally, the following modifications were undertaken, in order to make the data suitable for fitting the models.

• Bag-of-words vectors was implemented using the TfidfVectorizer which is available in sklearn. The words had to be first stemmed, using the NLTK Porter Stemmer before the model was filt so that the stop words are removed. Only words which reoccur 5 times or more while training is used as a feature.

• N-gram sequences was created from the text snippet. This was done by first converting the sentences into strings of characters having n-grams (n =1,2,3), and then using the Keras one-hot function. The one-hot function converted the strings variables into sequences of numbers. This was done to fit the CNN ML model properly. It was implemented using the sklearn's Label Binarizer function.

2.2 Implementation

The two main techniques used in my project was support vector machines (SVMs) and convolutional neural networks (CNNs) [4][1]. Neural networks (NNs) model uses a mathematical model, which works similar to the operation of the human brain for the classification of data. They

are composed of multiple layers of nodes called neurons, which receive input data. This data is weighted, combined and then transformed, by an activation function, in order to produce an output value. The output values of one layer become the input values of the next in the case of neural networks with multiple layers. CNNs are a special case of NNs which "share their parameters across spaces". Here I have applied to 1D array of sequences of character n-grams. For this project, the rectified linear unit (ReLU) activation which is one of the most common deep neural networks is used. I have done CNN implementation using Keras.

In the case of the CNN, we a 3channel network, with each channel comprising the following:

a) A convolutional layer with 500 filters, kernels of sizes 3, 4 and 5 for each of the three channels respectively.

b) A max pooling layer with pool size 2.

c) An embedding layer of dimension 26, in the 1-gram case, 300 in the 2-gram case and 600 in the 3-gram case.

d) A 25% drop-out layer.

The 3 channels were merged at the output layer, which has eight output units, one for each author. In training this model, an Adam optimizer with learning rate of 0.0001 and a batch size of 32 was used. The model was trained for 5 epochs as this was the point at which validation accuracy was found to plateau. The Adam optimizer is known for its adaptive learning rate optimization algorithm used in training deep NN models. The optimizer computes the first and second moments of the gradients in order to adapt the learning rate for each weight in the neural network model.

SVMs are mainly used for text categorization tasks because of their ability to process thousands of different inputs. In applying SVMs to author attribution, Diederich et al. (2003) considered a different SVM kernels, like linear, quadratic, cubic and RBF. He concluded that the choice of

SVM kernel function had little to no effect on the model performance in terms of loss. I have considered a linear kernel for this project, I also tried using the RBF kernel, but it seemed to be sensitive to the choice of the γ parameter in some cases.

We know that both SVMs and CNNs accept only numeric data as inputs. Since we are working with text data, I had to first pre-process the dataset such that information contained in the text was converted to numeric values. I have deployed 2 techniques to do this as described below:

• Tfidf Vectorization: This algorithm involves converting each text excerpt to a vector of word counts where the length of the vector is equal to the size of the vocabulary represented by the entire text excerpts. It is then divided by the frequency of the individual word count. So Tfidf says how frequent a word appears in a document but not across documents.

• Word/Character Embedding: It is a deep learning technique which involves efficiently mapping one-hot encoded words to feature vectors. Here the words that are used in a similar way have similar feature representations. This technique was applied to character n-grams.

Combining these techniques with the algorithms previously described gives us the two models that we will consider as part of this analysis (in addition to a random model): Therefore to recap the first model was a CNN with character n-grams as inputs, which is processed using a (character) embedding layer and the second model is a SVM with a bag-of-words feature set created using tfidf vectorization.

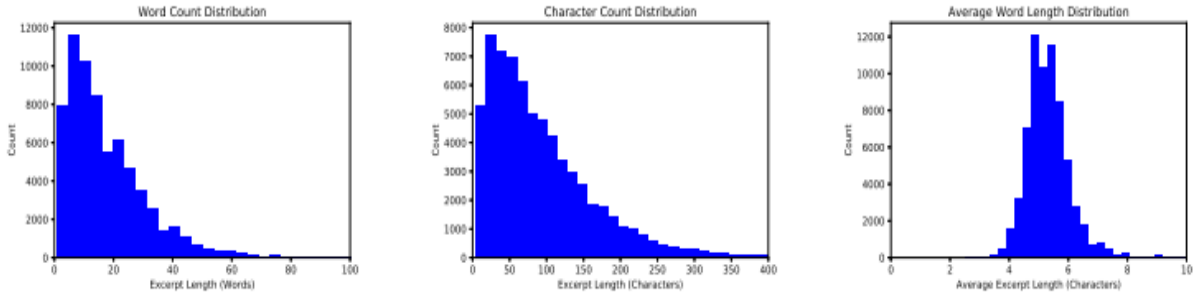
3. Project Results

I have successfully generated an excel file which is the new dataset moving forward. The excel sheet consists of preprocessed data that was collected from all the books mentioned in section 1.2. A collection of lists of these sentences were generated for each author. From these lists 8500 were

then chosen from every author to bring consistency in the ML decision making process. I have then randomly shuffled the data in order to avoid any issues arising from the bunching together of sentences by a single author. This excel sheet containing 6800 sentence-long text excerpts from the works of famous authors, along with respective labels identifying the authors of the excerpts. A screenshot of a small dataset of that excel sheet is shown below.

72	'They always likes a bone or two to clean their teeth on about tea-time, which you 'as a bagful.'	Stoker
73	When he went out in the morning he was bewildered by small commissions for the captive mamma, if he came gaily in at night, eager to embrace his family, he was quenched by a "Hush!	Alcott
74	You shall lie on one, and I on the other, and our sympathy will be comfort to each other, even though we do not speak, and even if we sleep".	Stoker
75	On the Saturday Mr. Hartright had left before I got down to breakfast.	Collins
76	They should call it let me see the White Way of Delight.	Montgomery
77	She does seem a charming young woman, just what he deserves.	Austen
78	I consider him to be eminently ill-tempered and disagreeable, and totally wanting in kindness and good feeling.	Collins
79	The translation of a few pages of German occupied an hour; then I got my palette and pencils, and fell to the more soothing, because easier occupation, of completing Rosamond Oliver's r	Bronte

Distribution of word count (left), character count (middle) and average word length (right) for all authors combined is also found. A screenshot of the graph is shown below:

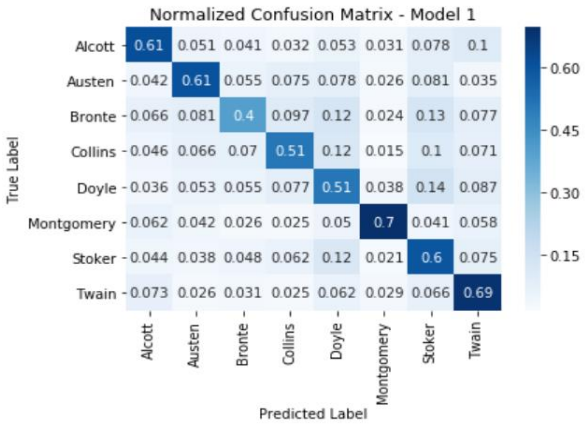


I have further removed punctuations, white spaces, invalid characters and made the individual characters to lower case in order to normalize all the excerpts. This processed dataset is then split into training and testing subsets for using them in the CNN and SVM based ML models. The SVM model that I have implemented here has a linear kernel, with C values in the set as {1,10, 100}. The SVM was implemented using the sklearn's SVC function. The optimal value of C was

determined using sklearn's GridSearchCV function with 3-fold cross-validation and accuracy is used as the scoring metric.

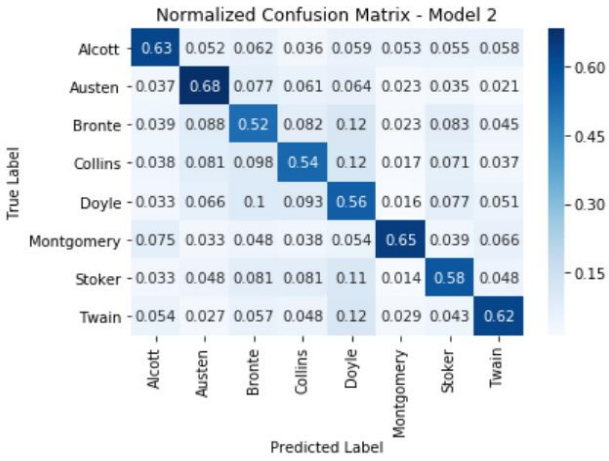
The following is a screenshot of the output of n-gram CNN based approach (Model 1):

```
Accuracy: 0.5779411764705882
Ave. Precision: 0.5834731226487566
Ave. Recall: 0.5779411764705883
Ave. F1 Score: 0.5774801929044945
Training Time: 1520.5548295974731 seconds
Prediction Time: 24.316569805145264 seconds
```



The following is a screenshot of the output of Bag of Words SVM based approach (Model 2):

```
Accuracy: 0.5981617647058823
Ave. Precision: 0.6057913357704325
Ave. Recall: 0.5981617647058823
Ave. F1 Score: 0.6005337108664128
Training Time: 218.93059182167053 seconds
Prediction Time: 32.63461709022522 seconds
```



We can see that, Model 2 performing slightly better than Model 1 with regards to accuracy, precision and F1 score. But, the training time for Model 1 is almost 7 times that of Model 2. Model 1 has lower prediction time compared to Model 2. Hence Model 2 is the preferred model with respect to both speed and accuracy.

The confusion matrices show a higher variation in the level of accuracy for Model 1(Best – 70% and Worst – 40%) than for Model 2 (Best – 68% and Worst – 52%). Hence Model 2 is again chosen because it performs similarly well across all classes.

5.Conclusion

In this analysis, I was able to correctly identify the authors of the text excerpts that were 25 words long, on average, with a reasonably good test accuracy level of approximate 60% which is over 4.5 times what we would expect based on random chance. Hence we proved that it was possible to apply ML algorithms to determine the authorship of short, tweet-length excerpts of longer works, which proves that the conclusion made by, Forsyth and Holmes in 1996 that a text needed to be a minimum of 250 words in length for the authorship attribution to be apparent was wrong.

I had considered 2 different models, one a CNN fitted to 3-gram character sequences having 3 channels and second an SVM fitted to a bag-of-words feature set. We saw that the SVM model gave a slightly better result. I believe that the accuracies can be further improved for example in the CNN model we could try using n gram word sequence instead of n gram character sequence or we could try using RNN architecture. In the case of the SVM model we can try changing the kernel model or try changing the Tfidf Vectorizer parameters. Overall,

both of the discussed models fit my expectations for a solution to the authorship attribution problem.

6. References:

[1] Green, R. and J. Sheppard (2013). Comparing frequency- and style-based features for Twitter author identification. Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference, 64–69.

[2] Patrick Juola, "JGAAP: A Modular Software Framework for Evaluation, Testing, and Cross-Fertilization of Authorship Attribution Techniques".

[3] Mohsen, A., N. El-Makky, and N. Ghanem (2016). Author identification using deep learning. Proceedings of the 15th IEEE International Conference on Machine Learning and Applications, 898–903.

[4] Shrestha, P., S. Sierra, F. Gonz´alez, P. Rosso, M. Montes-y G´omez, and T. Solorio (2017). Convolutional neural networks for authorship attribution of short texts. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. Valencia, Spain, 669–674.

[5] Schwartz, R., O. Tsur, A. Rappoport, and M. Koppel (2013). Authorship attribution of micro-messages. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Seattle, Washington, USA, 1880–1891.

[6] Zheng, R., J. Li, H. Chen, and Z. Huang (2006). A framework for authorship identification of online messages: Writing-style features and classification techniques. Journal of the American Society for Information Science and Technology 57(3), 378–393.

[7] Diederich, J., J. Kindermann, E. Leopold, and G. Paass (2003). Authorship attribution with support vector machines. Applied Intelligence 19, 109–123

[8] Brownlee, J. (2017). What are Word Embeddings for Text? https://machinelearningmastery.com/what-are-word-embeddings/.

[9] Forsyth, R. and D. Holmes (1996). Feature finding for text classification. Literary and Linguistic Computing 11(4), 163–174