B.Sc. in Computer Science and Engineering Thesis

# Authorship Attribution in English Literature Using Machine Learning

Submitted by

SK SAIF AL ISLAM
16151012

JAHIDUL KABIR
16151037

SUNGIDA MUNIR
16151054

MARUF ALAM
17151097

Supervised by

SUPRATIP GHOSE (PHD)

**Department of Computer Science and Engineering**
**Bangladesh University of Engineering and Technology**
Dhaka, Bangladesh

JANUARY 2020

# CANDIDATES’ DECLARATION

This is to certify that the work presented in this thesis, titled, “Authorship Attribution in English Literature Using Machine Learning”, is the outcome of the investigation and research carried out by us under the supervision of SUPRATIP GHOSE (PHD).

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

SK SAIF AL ISLAM
16151012

JAHIDUL KABIR
16151037

SUNGIDA MUNIR
16151054

MARUF ALAM
17151097

# CERTIFICATION

This thesis titled, **“Authorship Attribution in English Literature Using Machine Learning”**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in JANUARY 2020.

**Group Members:**

**SK SAIF AL ISLAM**

**JAHIDUL KABIR**

**SUNGIDA MUNIR**

**MARUF ALAM**

**Supervisor:**

SUPRATIP GHOSE (PHD)
NULL
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

# ACKNOWLEDGEMENT

To the begin with i would like to thanks Almighty Allah without his blessing none of us can do anything. Then i would like thanks our parents specially my Mother for her unending love and support so that i have been able to complete my undergrad education in Computer Science. We are indebted to to our Honourable Teacher's without their proper guidance it was totally impossible for us to complete our Graduation.

Last but not least we would like to remember all of our friends for their wonderful co-operation which has played an important role to purse our B.Sc Degree.

Dhaka
JANUARY 2020

SK SAIF AL ISLAM

JAHIDUL KABIR

SUNGIDA MUNIR

MARUF ALAM

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# ABSTRACT

Authorship Attribution is the process of finding an author from anonymous text. This is a problem that can readily be framed as a text classification task, “where author represents a class (label) of a given text,” and as a result, recent research into author attribution analysis has focused almost exclusively on the use of machine learning techniques. The aim of this analysis is to develop a machine learning-based model capable of successfully identifying the authorship of several stories from English Literature. In doing this, we consider two approaches: a CNN-based approach, similar to that proposed by Shrestha et al. (2017) and a bag-of-words SVM-based approach, similar to that used by Green and Sheppard (2013). And finally We have gained 58 Percent accuracy because of we have worked on longer text.

# Chapter 1

# Introduction

Authorship Attribution is the process of finding an author from anonymous text. This is a problem that can readily be framed as a text classification task, “where author represents a class (label) of a given text,” and as a result, recent research into author attribution analysis has focused almost exclusively on the use of machine learning techniques.

## 1.1 Overview

Author attribution “is the task of identifying the author of a given text from a (given) set of suspects” This is a problem that can readily be framed as a text classification task, “where author represents a class (label) of a given text2,” and as a result, recent research into author attribution analysis has focussed almost exclusively on the use of machine learning techniques.

Prior to the advent of social media, author attribution analysis was typically applied to longer texts, such as books and letters. In fact, Forsyth and Holmes (1996) concluded that a text had to be a minimum of 250 words in length for the stylometric characteristics to be apparent. However, recent research (for example, Green and Sheppard (2013) and Schwartz et al. (2013)) has demonstrated the successful application of author attribution techniques to Twitter messages (“tweets”), which “average less than 25 words” in length and are “often less than 10” words long3.

Tweets are currently limited to 280 characters, and prior to November 2017, were limited to 140 characters. As a result, “tweets are relatively self-contained and have smaller sentence length variance compared to excerpts from longer text4.” It is possible that these characteristics are the reason why author attribution techniques, that have previously fallen apart when applied to shorter texts, have succeeded when applied to tweets. It is also possible that, had Forsyth and

Holmes (1996) considered more modern machine learning algorithms in their analysis, such as support vector machines (SVMs) and neural networks, which were not in common use in 1996, that they would have drawn different conclusions about the minimum text length required to successfully identify the author of a text.

In this analysis we explore these hypotheses by applying techniques that have been demonstrated to succeed in determining the authorship of ten stories from ten different authors. In performing this analysis, we make use of of a dataset compris-ing 40,000 sentence-long excerpts from the (fiction) works of ten classic authors (Charles Dickens, L.M.Montgomery, Bram Stoker, Jane Austen, Charlotte Bronte, Wilkie Collins, Jonathan Swift, Margaret Thatcher, Mark Twain, Louisa May Alcott, Arthur Conan Doyle, and Upton Sinclair), along with labels identifying the author of each excerpt.

This dataset was created using novel texts sourced from Project Gutenburg5, with chapter/section headings manually removed from the files prior to processing. To allow for the creation of a balanced dataset, for authors whose novels tended to be shorter in length, text excerpts were taken from multiple works.

The novels used to create the dataset are given in Figure 1:

| Author Name | Novel |
|---|---|
| Charles Dickens | A Tale of Two Cities |
| Lucy Maud Montgomery | Anne of Avonlea |
| Bram Stoker | Dracula |
| Jane Austin | Emma |
| Jonathan Swift | Gulliver's Travels |
| Margaret Thatcher | Hard Times |
| Mark Twain | Huckleberry Finn |
| Louisa May Alcott | Little Women |
| Arthur Conan Doyle | The Adventures of Sherlock Holmes |
| Upton Sinclair | The Jungle |

Table 1.1: Novels used to create the author attribution dataset

## 1.2 Problem Statement

The aim of this analysis is to develop a machine learning-based model capable of success- fully identifying the authorship of ten different stories. In doing this, we consider two approaches: a CNN-based approach, similar to that proposed by Shrestha et al. (2017) and a bag-of-words

SVM-based approach, similar to that used by Green and Sheppard (2013). The CNN-based approach uses a CNN with character n-grams (that is, sequences of n characters) as inputs, while the SVM-based approach makes use of a feature set comprising the words in the text excerpts themselves.

For benchmarking purposes, the performance of these two models is compared to that of a random model. That is, a model where the author of a given text extract is predicted by selecting one of the eight authors uniformly at random.

All analysis was undertaken in Python 3, predominantly using the Scikit-Learn (sklearn), Keras and Natural Language Toolkit (NLTK) packages. At a high level, the steps involved in performing the analysis are as follows:

- Collect, explore and preprocess the author attribution data;
- Randomly split the data set into training and test subsets using an 80%/20% split;
- Create the n-gram character sequences and bag-of-words feature sets;
- Fit and tune a CNN to the (training) n-gram character sequences;
- Fit and tune an SVM to the bag-of-words (training) feature set;
- Using the test data set, evaluate and compare the fitted models.

Given that our data set comprises 8 authors, if an author attribution model is successful in identifying the authors of text inputs, we would expect it to achieve accuracy levels of greater than 12.5% (i.e. better than selecting the author of a given text uniformly at random).

In their analysis, Shrestha et al. (2017) achieved accuracy levels of over 76% in deter- mining the author of tweets (from a pool of 50 possible authors) using a character n-gram CNN, while Green and Sheppard (2013) achieved accuracy levels of approximately 60% in determining the author of tweets (from a pool of 3 possible authors) using a bag-of-words SVM. In general, we would expect accuracy levels to decrease as the number of possible authors increases.

## 1.3 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 data set; and
- Prediction time: Time required to predict the class of all data points in the test data set.

We also examine the confusion matrix for each of the models. The confusion matrix provides a visual comparison of the actual versus predicted labels of examples at a class level.

Given we are using a balanced data set, accuracy should be a sufficient metric to evaluate the performance of our model. However, we have considered multiple metrics to evaluate the performance of the models, anyway, to effectively serve as a ”second opinion” with regard to performance and allow us to view model performance from multiple angles. In particular, the confusion matrix has been included as one of the evaluation metrics to allow us to determine if there are any systematic patterns in the mis classification of the text excerpts.

Weighted average recall is not included in the above list, as it can be shown to be mathematically identical to accuracy.

# Chapter 2

# Analysis

## 2.1 Data Exploration and Exploratory Visualization

As mentioned previously, the data set used in this analysis (the author data set) comprises 40,000 sentence-long excerpts from the works of Charles Dickens, Lucy Maud Montgomery, Bram Stoker, Jane Austin, Jonathan Swift, Margaret Thatcher, Mark Twain, Louisa May Alcott, Arthur Conan Doyle, Upton Sinclair along with labels identifying the author of each excerpt. The data set is balanced, with 4000 text excerpts per author. There do not appear to be any missing or corrupt labels.

The texts excepts are presented in plain text, with punctuation included and no obvious types or spelling errors. Four sample text excerpts are presented below:

**Excerpt 10,753** :

It seemed altogether past belief that anyone could make such a will, or that they would pay such a sum for doing anything so simple as copying out the Encyclopædia Britannica.

**Excerpt 27,910** :

I observed, with much pleasure, that these two persons were in good intelligence with each other; and Cæsar freely confessed to me, “that the greatest actions of his own life were not equal, by many degrees, to the glory of taking it away”.

**Excerpt 35,244** :
The governor, at my request, gave the sign for Cæsar and Brutus to advance towards us.

**Excerpt 27067** :
It is pleasant to me to observe, Watson, that you have so far grasped this truth that in these little records of our cases which you have been good enough to draw up, and, I am bound to say, occasionally to embellish, you have given prominence not so much to the many causes célèbres and sensational trials in which I have figured but rather to those incidents which may have been trivial in themselves, but which have given room for those faculties of deduction and of logical synthesis which I have made my special province".

The characters contained in the text excerpts were examined and 35 instances of an invalid character were identified, as well as 42 excerpts containing accented characters and 2606 excerpts containing large blocks of white space. An examination of the excerpts containing accented characters indicated that the accented letters were being correctly used (e.g. in foreign names) and were not an indication of corrupt data. However, the invalid characters and blocks of white space were flagged for removal during the preprocessing stage.

Each of the text excerpts was split into individual words and the word count, character count and average word length distributions were examined. Figure 2 presents summary statistics for these distributions, for all authors combined. These distributions are also plotted in Figure 2:

| | Word Count | Character Count | Ave. Word Length |
|---|---|---|---|
| Minimum | 1 | 5 | 3 |
| Maximum | 277 | 1570 | 18 |
| Mean | 20.80 | 11.22 | 5.36 |
| Median | 16.00 | 86.0 | 5.29 |
| 1st Percentile | 1.00 | 7.00 | 3.91 |
| 99th Percentile | 85.00 | 467.00 | 8.00 |
| 99.9th Percentile | 138.00 | 781.00 | 12.00 |

Table 2.1: Summary statistics for the word count, character count and average word length distributions

graphicx

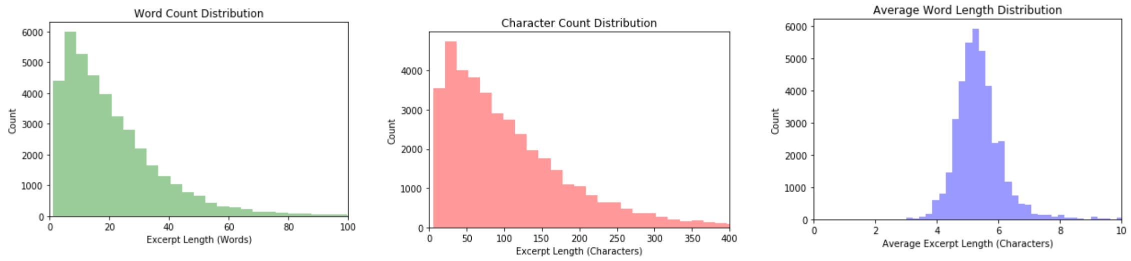

Figure 2.1: Distribution of word count (left), character count (middle) and average word length (right) for all authors combined

The word count and character count distributions are heavily skewed to the right. Although the average excerpt is only 17.88 words long (or 94.97 characters long) and the vast majority of excerpts are less than 100 words in length, the tail of the word count Machine Learning for Author Attribution Genevieve Hayes distribution is quite long, with the longest excerpt reaching 266 words in length. By contrast, the average word length distribution is almost perfectly bell-shaped, although, again, this distribution does contain some outliers, with one excerpt achieving an average word length of 22 characters.

The extrema of each of these distributions (at both the low and high extremes) were examined to check for any data issues. However, none were identified.

## 2.2 Algorithms and Techniques

The two main algorithms used in this analysis are convolutional neural networks (CNNs) and support vector machines (SVMs). These algorithms were chosen due to the success previous researchers (specifically, Shrestha et al. (2017) and Green and Sheppard (2013) respectively) have had in applying them to the tweet author identification problem in the past.

Neural networks (NNs) use a mathematical model, inspired by the operation of the human brain, to classify data. Although first developed in the 1940's, NNs have recently come to prominence due to their successful application to traditionally difficult to model problems, such as speech and image recognition. NNs are composed of multiple layers of nodes or neurons, which receive input data. The input data is weighted and combined, then transformed, via an activation function, to produce an output value. For NNs with multiple layers, the output values of one layer become the input values of the next.

CNNs are a special case of NNs which "share their parameters across space6." They are commonly applied to image data, in which they "group together adjacent pixels in an image and treat them as a collective7." However, they can also be applied to one- dimensional arrays of values, such as sequences of character n-grams, which was how Shrestha et al. (2017) applied them to author attribution analysis.

In fitting a NN, it is necessary to select values for the number of (hidden) layers in the network, as well as the number of neurons in each layer and an appropriate activation function. For this analysis, the rectified linear unit (ReLU) activation function will be used, as it is the activation function most commonly used in deep neural networks.

SVMs are often used for text categorization tasks due to their “ability to process many thousand different inputs. This opens the opportunity to use all words in a text directly as features8.” They involve constructing a decision boundary to separate the data into classes, which may be non-linear if the “kernel trick” is used to transform the data into a higher dimensional space. As such, the key decision that must be made when fitting an SVM classifier is the choice of kernel (in addition to tuning any hyperparameters specific to that kernel).

In applying SVMs to author attribution, Schwartz et al. (2013) and Green and Shep- pard (2013) both used a linear kernel, while Zheng et al. (2006) used a polynomial kernel. Diederich et al. (2003) considered a range of different SVMs kernels, including linear, quadratic, cubic and radial basis function (RBF) and concluded that “the choice of SVM kernel function has little or no effect on (model) performance in terms of loss),” . . . al- though “the RBF kernel seems to be sensitive to the choice of the parameter in some cases,” and so is “considered to be less adequate.” For this analysis, we shall, therefore, only consider the linear (polynomial with degree = 1), kernel for our model.

Both SVMs and CNNs accept only numeric data as inputs. Given that we are working with text data, it is, therefore, necessary to pre-process our data in such a way that the information contained in the text is converted to numeric values. In this analysis, we will employ two techniques in order to achieve this:

- **Word/Character Embedding:** Word embedding is a deep learning technique that involves efficiently mapping one-hot encoded words to feature vectors such that words that are used in similar ways have similar feature representations9. This technique is more efficient than count vectorization for large vocabularies and can also be applied to character n-grams; and
- **Tfidf Vectorization:** This technique involves converting each text excerpt to a vector of word counts (which will be zero for words not contained in the excerpt), where the length of the vector is equal to the size of the vocabulary represented by the entire corpus of text excerpts, and then dividing these counts by the frequency of each word in the overall corpus. The length of the vectors may be reduced by word stemming (i.e. reducing words to their base or root form, such that words with the same stem are considered to be the same) and by excluding from the vocabulary stop words and words that appear less than a pre-specified number of times in the entire corpus of text excerpts.

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):

- **Model 1:** CNN with character n-grams as inputs, which are processed using a (character) embedding layer; and
- **Model 2:** SVM with a bag-of-words feature set created using tfidf vectorization.

# Chapter 3

# Methodology

## 3.1 Data Preprocessing

The first step involved in preprocessing the data was to remove the invalid characters and large blocks of white space, identified in Section 2.1. Once this was done, the data was normalized, by removing all punctuation and converting all excerpts to lowercase, and the data was randomly split into training and test sets, using an 80%/20% split. Finally, the following modifications were undertaken, in order to make the data suitable for fitting the models described in Section 2.2:

- **Creation of n-gram Sequences:** n-gram sequences were created from the text excerpts (for n = 1, 2 and 3) by first converting the excerpts into strings of character n-grams, and then using the Keras one-hot function to convert these strings into sequences of numbers. As it is necessary for all of the inputs to a CNN to be the same size, these sequences were then resized to a length of 350 by cutting down any overly long sequences and padding any sequences of length less than 350. A length of 350 was selected because it is long enough that over 95% of all excerpts will not need to be cut down, but short enough to avoid having to massively pad out the majority of sequences in order to accommodate the small number of very long excerpts. Values of n in the range 1 to 3 were selected for creating the n-gram sequences, broadly in keeping with Shrestha et al. (2017), which found that 1-gram and 2-gram letter sequences produced the best results;
- **Creation of Bag-of-Words Feature Vectors:** Bag-of-words vectors were created using the sklearn TfidfVectorizer. Prior to fitting the TfidfVectorizer, the words in the text excerpts were stemmed, using the NLTK PorterStemmer, and stop words were removed. In keeping with Green and Sheppard (2013), only words that occur more than 5 times across all training text excerpts were used as features;

- **One-Hot Encoding of Labels:** In order to fit the CNN, it is necessary to one-hot encode the data labels. This was done using sklearn's LabelBinarizer function.

## 3.2 Implementation

The CNN was implemented using Keras, while the SVM was implemented using sklearn's SVC function.

In the case of the CNN, we used the same model architecture as was used by Shrestha et al. (2017) wherever possible. That is, a 3 channel network, with each channel compris- ing the following:

- An embedding layer of dimension 26, in the 1-gram case; 300, in the 2-gram case; and 600, in the 3-gram case;
- A 25% drop-out layer;
- A convolutional layer with 500 filters, kernels of size 3, 4 and 5 for each of the three channels respectively, and a ReLU activation function;
- A max pooling layer with pool size 2.

The three channels were merged at the dense softmax output layer, with 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, and the model was trained for 5 epochs (since this was the point at which validation accuracy was found to plateau).

In the case of the SVM, we used a linear kernel and considered C values in the set 1, 10, 100. The optimal value of C was determined using sklearn's GridSearchCV function with (default) 3-fold cross-validation and accuracy used as the scoring metric.

No complications occurred during the implementation phase, as both sklearn and Keras are very well documented, and easy to use, packages.

## 3.3 Refinement

Using sklearn's GridSearchCV function, the optimal C value for the SVM was determined to be C = 1. No further refinements were undertaken.

In the case of the CNN, the model was fit to 1-gram, 2-gram and 3-gram letter sequences. Although 1-gram and 2-gram sequences were found to produce the best results for Shrestha et al. (2017), in our analysis, the best results (with regard to validation accuracy) were achieved using 3-gram sequences, so these were ultimately favored over the 1-gram and 2-gram sequences.

Some experimentation was undertaken to determine if better results could be achieved by adding an extra channel (with kernel size = 6) to the model. However, this change resulted in a slight decrease in validation accuracy, so was ultimately abandoned.

Based on this analysis, the following hyper-parameters, etc, were selected for the CNN and SVM:

- **Model 1(n-gram CNN):** 3 channel CNN fit to 3-gram letter sequences; and
- **Model 2(Bag-of-Words SVM):** Linear SVM with C = 1.

These are the final models that are evaluated in the next section.

# Chapter 4

# Results

## 4.1 Model Evaluation and Validation

The final models described in the previous section were each fit to 100% of the training data and then evaluated against the test data. The evaluation metrics described in Section 1.3 were calculated for each model, as well as for the random (benchmark) model, and are presented in Figure 4. In the case of the random model, all reported metrics are averages calculated over 10 runs of the model. The standard deviations of the metrics for these 10 runs are shown in brackets next to the averages. The evaluation metrics for the random model are compared with those for the two final models in Section 4.2, so are not discussed any further in this section.

| Metric | Model-1 | Model-2 | Random Model |
|---|---|---|---|
| Accuracy | 0.62 | 0.21 | 0.10255 |
| Ave. Precision | 0.27 | 0.27 | 0.10238 |
| Ave. F1 Score | 0.618 | 0.26 | 0.10224 |
| Training Time(s) | 11655.49 | 101.70 | N/A |
| Prediction Time(s) | 336.26 | 19.51 | N/A |

Table 4.1: (Test) evaluation metrics for the final models

From Figure 4, we can see that, with regard to accuracy, precision and F1 score, Models 1 and 2 achieve similar levels of performance, with Model 2 performing slightly better than Model 1. However, the training time for Model 1 is almost 7 times that of Model 2. The only metric for which Model 1 outperforms Model 2 is prediction time. Nevertheless, the prediction times for both models are small relative to the training times, and unless a very large number of predictions are to be made from the model, any benefits with regard to faster predictions are likely to be outweighed by the costs in terms of reduced accuracy and increased training time. Consequently, with regard to both speed and accuracy, Model 2 is the preferred model.

Figure 5 presents the (normalized) confusion matrices for Models 1 and 2.

The confusion matrices show much greater variation in the level of accuracy by author for Model 1 than for Model 2. Model 1 achieves accuracy levels of around 84% for the best performing authors (Jonathan Swift), but only 40% for the worst performing author (Charles Dickens) - a variation of 44% between best and worst. By comparison, the accuracy levels for Model 2 are more consistent from author to author, with accuracy levels ranging from 84% for Jonathan Swift to 57% for Barm Stocker - a range of just 27%. It is preferable to have a model that performs similarly well across all classes than one that performs very well on a few classes but poorly on others. Therefore, on this front, Model 2 is, again, preferred.

graphicx

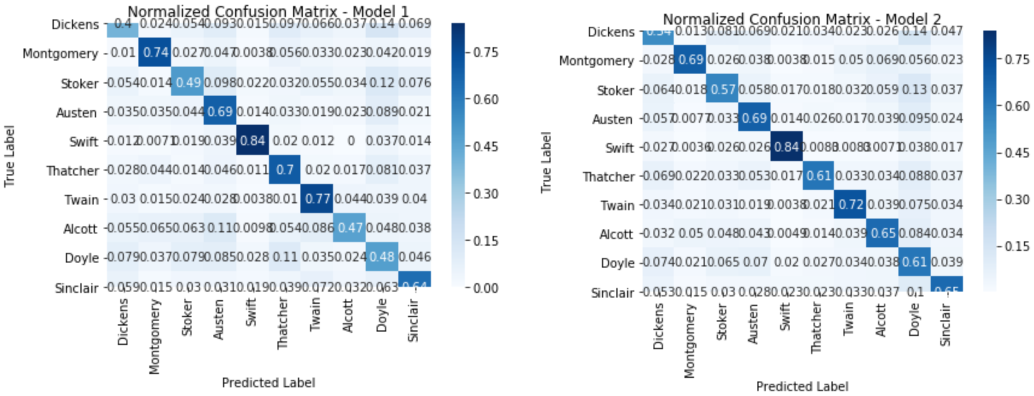

Figure 4.1: Confusion matrices for Model 1 (left) and Model 2 (right).

To explore why the models may be classifying some excerpts correctly and others incorrectly, a sample of 100 text excerpts were examined, along with their true and predicted labels under each of the two models. Based on this sample, it appeared that both models find it easier to correctly identify the authors of longer excerpts than shorter excerpts, as would be expected, although if the sample contains a word specific to the vocabulary of a particular author then both models can successfully identify the author regardless of how short the excerpt is. For example, both models were able to correctly identify the author of the single word excerpt: “Shucks,” as Mark Twain.

To test whether excerpt length has a significant impact on whether or not the models can correctly identify the author of a text, the average word count and average character count were compared for correctly vs. incorrectly classified excerpts, for each model.

Welch’s t-test was used to compare the average word count and average character count for correctly and incorrectly labelled examples, for each model. In each case, the null hypothesis (that the averages are equal) was rejected at the 5% significance level. Therefore, even though both models are capable of correctly classifying very short text excerpts, the longer the excerpt is, the easier it is for the models to classify.

To test the robustness of the models, with regard to small changes in the training data, each model was refit three times using a different 67% subset of the training data each time (with the 67% subsets randomly selected using the sklearn KFold function) and the three non-time test evaluation metrics recalculated for each of the three runs. For each of the models, and each of accuracy, precision and F1 score, the resulting metrics for the three runs all fell within less than 0.01 of each other, validating the robustness of both models under consideration and suggesting that the results from either of these models can be trusted.

## 4.2 Justification

As demonstrated by Figure 4, both models under consideration perform significantly bet- ter than the random (benchmark) model, with both models achieving accuracy, precision and F1 scores over 4.5 times those achieved by the random model. As a result, we can conclude that it is possible to use machine learning techniques to successfully identify the authorship of short, tweet-length excerpts of longer works.

However, comparing our results to those achieved by Shrestha et al. (2017) and Green and Sheppard (2013) using models similar to our Models 1 and 2 respectively, applied to actual tweets, Shrestha et al. were able to achieve an accuracy score of 76% for their n- gram CNN when applied to the much more difficult task of identifying the correct author of tweets from a pool of 50 authors; while Green and Sheppard achieved an accuracy score of 60%, which is similar to the accuracy score achieved by our Model 2, but for the simpler task of identifying the correct author of tweets from a pool of only 3 authors. This suggests that our Model 1 underperformed relative to the Shrestha et al. (2017) n-gram CNN, while our Model 2 outperformed the Green and Sheppard (2013) bag-of-words SVM.

It is impossible to say conclusively why our models have underperformed or overper- formed relative to their counterparts. However, one possibility is that it may be due to our models being fit to a dataset comprising tweet-length excerpts of novels, rather than actual tweet data, as was the case for Green and Sheppard (2013) and Shrestha et al. (2017).

Tweets have certain characteristics, such as the use of symbols or abbreviations in place of words, that are not present in novels. These characteristics may make them more suitable for author identification via n-gram CNNs. By contrast, unlike tweets, novels typically feature a relatively small number of characters and locations, the names of which are repeated numerous times throughout the work - a fact that a bag-of-words SVM can exploit to its benefit, which it may not be able to do in the case of tweets.

Regardless of the cause of these differences, overall, our results provide sufficient justification to be able to declare the author attribution problem posed by this analysis to be adequately solved.

# Chapter 5

# Conclusion

## 5.1 Reflection

In this analysis, we set out to determine whether it was possible to use machine learning techniques to determine the authorship of short, tweet-length excerpts of longer works. Previously, Forsyth and Holmes (1996) had concluded that a text needed to be a mini- mum of 250 words in length for the stylometric characteristics to be apparent. However, through applying techniques that had previously proven to be successful in identifying the authorship of tweets to the problem, we were able to prove Forsyth and Holmes wrong, and were able to identify the author of excerpts that were 18 words long, on average, with test accuracy levels of around 60% - over 4.5 times what we would expect based on random chance.

Two different models were considered in this analysis: a CNN fitted to 3-gram charac- ter sequences; and an SVM fitted to a bag-of-words feature set. Of these models, the best results were produced by the SVM. This was an interesting result, since, based on the results of previous studies, we would have expected the CNN to be the superior model. One possible explanation for this was that previous researchers had fit their models to tweet data, whereas our models were fit to tweet-length excerpts of novels. However, without further analysis, it is impossible to conclude whether or not this is the case.

That said, in light of Forsyth and Holmes' conclusions about the minimum text length required for successful author identification, the fact that either model produced above- benchmark results meant that the results of this analysis exceeded my expectations.

The results of this analysis were interesting to me, with regard to the light they shed on the similarities and differences between the writing styles of the different authors. By observing which authors were most likely to be mistaken for each other, in the confusion matrices presented in Figure 5, we can see that writers who write in the same or similar genres are more likely to be mistaken for one another than those who write in more distinct genres. For example, a text excerpt from the works of mystery writer Wilkie Collins is more likely to be incorrectly classified as being written by fellow mystery writer Arthur Conan Doyle than by L.M. Montgomery, author of numerous coming-of-age novels. From this, we can infer than authors who write in the same or similar genres tend to adopt similar vocabularies and stylometric quirks to each other.

Overall, both models considered in this analysis fit my expectations for a solution to this problem and I believe both could be used in a more general setting to solve similar problems to the author attribution problem presented in this analysis.

## 5.2 Improvement

In fitting the character n-gram CNN presented in this analysis, I remained largely faithful to the model architecture presented by Shrestha et al. (2017). Some experimentation was undertaken, with regard to the model architecture and hyperparameters. For example, I considered adding an extra channel to the model and fitting the model to 3-grams, instead of 1-grams or 2-grams. However, due to time constraints, it was not possible to experiment with variations in as many of the model parameters as I would have liked to. Consequently, I believe improvements could potentially be made to the above results by experimenting with different model hyperparameters and architectures for this model.

Some changes that could be made to the CNN, which may potentially result in im- provements include:

- Using n-gram sequences, for n values greater than 3, as inputs;
- Using n-gram word sequences, instead of character sequences;
- Incorporating additional convolutional layers into each of the CNN model channels;
- Changing the number of filters used in the convolutional layers of the model; and
- Incorporating some aspects of Recurrent Neural Network (RNN) architecture into the model, for example, long short-term memory (LSTM) units, which have previ-ously been successfully applied to natural language problems.

These changes would all result in a much more complex version of Model 1 than the final model produced by this analysis, and such a model would undoubtedly take longer to train. However, this additional complexity would be justifiable if an improvement in model accuracy could be achieved as a result.

In the case of the SVM, the model hyperparameters used in this analysis were largely based on those used by Green and Sheppard (2013) in their bag-of-words SVM. Again, due to time constraints, it was not possible to experiment with as many variations on this model as might have been desirable. However, some changes that could be made to the SVM, which may potentially result in improvements include:

- Using a different kernel function. For example, a polynomial kernel of degree greater than 1;
- Adjusting the parameters of the TfidfVectorizer;
- Using a different feature set. For example, stylometric features (such as charactercounts) instead of the bag-of-words feature set; and
- Not excluding stop words prior to creating the feature set.

Nevertheless, the implementation of these potential improvements, to both the CNN and the SVM, are beyond the scope of this analysis.

# References

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

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

Forsyth, R. and D. Holmes (1996). Feature finding for text classification. *andLinguistic Computing 11 (4), 163–174.*

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.*

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.*

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.*

Shrestha, P., S. Sierra, F. Gonzalez, P. Rosso, M. Montes-y Gomez, 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.*

Udacity (2017). Deep Learning Nanodegree [Online Course].*https://www.udacity. com/course/deep-*

*learning-nanodegree–nd101.*

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.*

# Appendix A

# Algorithms

## A.1 Convolutional neural network

In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics.

graphicx

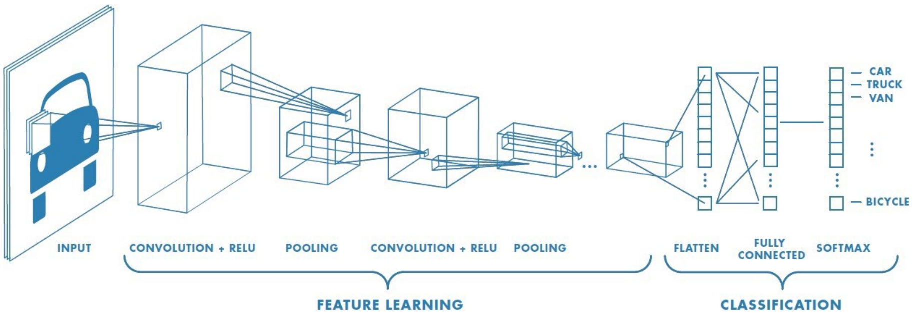

Figure A.1: Workflow of CNN).

Basically CNN is work for image classification apart from that CNN also works for text classification. Below is the figure how CNN works for text Classification:

graphicx

sectionSupport Vector Machine

SVM is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs.

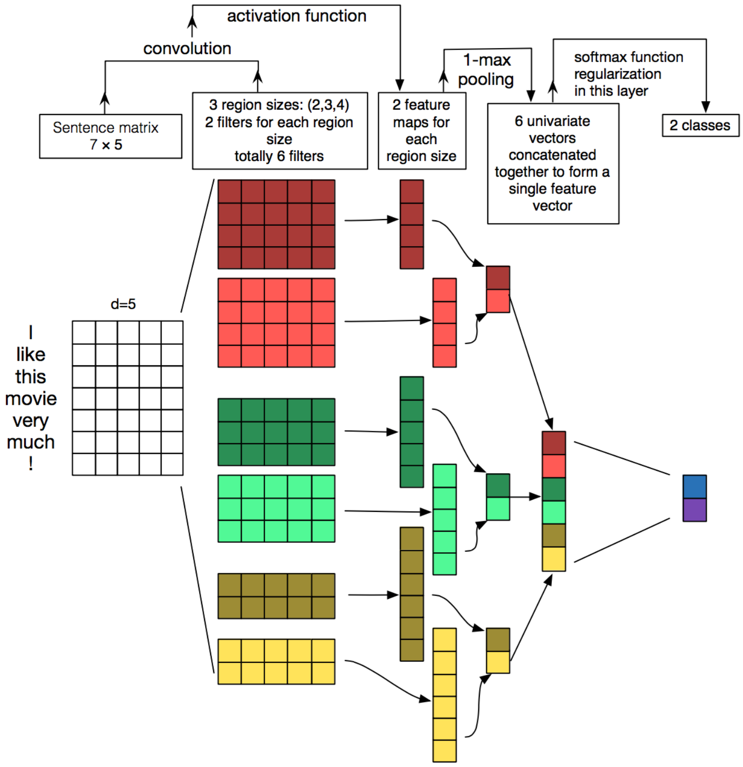
activation function
convolution
Sentence matrix
7 x 5
3 region sizes: (2,3,4)
2 filters for each region size
totally 6 filters
2 feature maps for each region size
1-max pooling
6 univariate vectors concatenated together to form a single feature vector
softmax function regularization in this layer
2 classes
d=5
I
like
this
movie
very
much
!

Figure A.2: CNN for Text Classification).

Generated using Undegraduate Thesis LaTeX Template, Version 1.4. Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh.

This thesis was generated on Wednesday 8th January, 2020 at 5:14pm.