Capstone Project Phase A
Project 21-2-R-13

Department of Software Engineering
Braude College of Engineering, Israel

Authorship Analysis Using Impostors Method

In partial fulfillment of the requirements for

Capstone Project in Software Engineering (61998)

Karmiel - June 2021

Raz Itzhak Afriat	208220418	razafriat101@gmail.com
Konstantin Maltcev	336492640	k.maltcev@outlook.com

Supervisors:

Prof. Vladimir (Zeev) Volkovich

Dr. Renata Avros

Contents

Abstract:

The plagiarism event is occurring since the early days and is very hard to prove. Lately, thanks to advanced technological systems, obtaining evaluations regarding the true author of literature is possible. For example, the novelist Mikhail Aleksandrovich Sholokhov was accused of plagiarism countless times for his work ‘And Quiet Flows the Don’. All the attempts to determine if he is the real author from a mathematical standpoint did not state plagiarism eventually. For many interested parties, it is important to make sure that these sorts of crimes will be revealed.

This paper proposes a solution for the Authorship Analysis problem and plagiarism detection based on the “Impostors method” algorithm alongside the use of a complex CNN-LSTM model. The solution involves taking the “Tweets Approach” and dividing the input text into small chunks to recognize short patterns in it. With the aim to improve predecessors’ systems accuracy, we will build a system alongside a GUI to operate it based on these models. The system's initial parameters are set and can change via the GUI to find the best combination of parameters and improve accuracy. The program is evaluated using Sholokhov literature and other authors works during the test phase. While these tests are being performed, the system’s behavior is compared with expected behaviors. The expected behaviors are predefined in test cases from a designed test plan.

Keywords: cnn; lstm; authorship analysis; impostors method; natural language process;

1. Introduction

Plagiarism is defined as the practice of taking someone else’s work or ideas and passing them off as one’s own. In early times it was extremely hard if not impossible to detect such a crime, but recently, with the technological advancement and the usage of sophisticated systems, it is possible to produce a favorable evaluation. Without using modern technology, one cannot, in most cases, prove plagiarism. That is because overlapping words are not enough proof by themselves. Often, it is impossible to find sufficient evidence for the claim of authorship. As a form of theft, steps to stop and reveal plagiarism are important and should be taken.

A variety of parties are interested in the development of an accurate system of Authorship Analysis. Victims are eager to prove their claims, lawyers, and all the judgment systems involved with plagiarism. Readers want to know the actual author of their beloved literature to know whom they should follow, admire, etc. Publishers and investors want to know the true artists so they can make suitable investments.

Mikhail Aleksandrovich Sholokhov was a Russian novelist who had won a Nobel Prize in Literature in 1965. ‘And Quiet Flows the Don’ is an epic novel by Sholokhov, considered among his most significant works. For this creation, Sholokhov was accused of plagiarism repeatedly under many different claims. Moscow State University’s research in 2020 claimed that ‘And Quiet Flows the Don’ indeed belongs to Sholokhov. Their claims are backed up with graphical results of a text distance measure algorithm called Burrow’s Delta. Other approaches and attempts to solve this matter were taken and supported Sholokhov.

Simply put, Tweets are messages sent on Twitter. In reference, the term Tweet is used to describe any short text, usually limited to a fixed size and straightforward, most often in the field of social networks. Tweet analysis is now a big trend in Natural Language Processing due to the need to detect malicious bots' emails and analyze the huge amount of data from Facebook, Twitter, etc. It is dividing a document into tweets that yields more significant results in the field of sentiment analysis since it enables the analysis of a widely changing document. For instance, in the case of having many external quotations of text that do not belong to the author, the small chunks of the author's writing make all the difference.

The impostors method is a widely used Machine Learning (ML) supervised classifier that excels in authorship verification tasks. The system is learning to distinguish between a collection of one author's work, and a collection of another author's literature, using a two-class classifier. After that, a test is applied with the questionable literature alongside genuine works of the accused author, and the behavior of it is being observed under the assumption that if the piece is indeed the author's work, it should be classified the same as his other works. The project's agenda is to improve an existing model implementation of the algorithm based on Convolutional Neural Network (CNN) with the Long Short-Term Memory (LSTM) model.

2. Background and Related Work

2.1. Background

2.1.1. Natural Language Processing NLP

NLP is a subfield of artificial intelligence focused on studying the interaction between human languages and computer systems. Due to the persuasion of the development of systems capable of "understanding" human documents, many algorithms are now widespread in NLP, for example, Keyword Extraction algorithms, sentiment analysis algorithms, etc.

2.1.2. Neural Networks

Neural Networks are types of Machine Learning (ML) algorithms developed with inspiration from the human brain's neural network. A neural network's model can learn and analyze huge amounts of data, making it possible to detect patterns in it and accomplish other different tasks as well. These days neural networks are extremely popular and are applied in most ML problems' solutions.

2.1.3. Convolutional Neural Network - CNN

Convolutional Neural Networks are a commonly used type of neural network. Their model originated from a study of the human brain's cortex in the 1980s. They use the convolution function and have three main types of layers. The convolutional layer requires the input data, a filter, a feature map and is responsible for most of the computational work. The pulling layer, or Downsampling, conducts dimensionality reduction and reduces the input's number of parameters. Lastly, the Fully Connected layer is responsible for the classification tasks.

2.1.4. Recurrent Neural Network - RNN

Recurrent Neural Network is another type of neural network in which the output of a stage is processed as input for the step after it. However, sometimes it is required to output a "forecast" based on input, for instance, when one wants to predict the next word based on previous words in sentences. RNN is solving this task, using a unique layer named "Hidden" and thus, the heart of RNN is the ability to have the hidden state, which remembers information of the previous states.

2.1.5. Long Short-Term Memory - LSTM

Long Short-Term Memory Networks is a kind of Recurrent Neural Network only that LSTM is capable of memorizing states for the long term. For example, it is common to predict current output; the information processed and stored a long time ago is required. Though, RNN is incapable of maintaining and reaching these long-term dependencies and thus fails these sorts of tasks. When using LSTM, it is not required to store a finite number of beforehand states. Architecturally, LSTM is extending the RNN hidden layer to four different layers interacting with each other, producing as output the layer's output alongside its state, making it possible to maintain such a memory.

2.1.6. Word Embedding

Word Embedding is a method of bridging between humans' representation of languages and computers. In an N-dimensional space, the method holds representations for texts in a way that one representation should be close to another if the words' meanings are similar. It is based on mathematics and the ability to produce good numerical representations, which computers can comprehend with. These vectors of word representations are crucial for solving almost any of the natural language processing tasks.

2.1.7. ELMo embedding framework

ELMo is an NLP framework developed by AllenNLP. They are implementing a novel way of embedding words and representing them in vectors. It handles complex characteristics of words and, as well, the way these words' uses are varied. Elmo was trained with a lot of data, no less than a corpus of 5.5 billion words. ELMo is achieving excellent results in several NLP tasks and is now widely used by researchers and the industry.

2.1.8. ReLU activation function

The Rectified Linear Unit is the most used activation function in deep learning models. It returns the value of its input, or zero if the input was negative. Even though it is simple and only composed of two linear pieces, the ReLU function works great in most applications and is widely used. Among the ReLU advantages is its computational cheapness, the absence of the vanishing gradient problem, which many other activation functions are suffering from. ReLU is also natural and intuitive because it is sparsely activated, mimicking the biological neural network better.

2.1.9. Natural Language Toolkit (NLTK)

Natural Language Toolkit is a platform used for building any kind of product in statistical natural language processing (NLP). It contains text processing libraries for tokenization, parsing, classification, stemming, tagging, and semantic reasoning. It also includes graphical demonstrations and sample data sets, and a cookbook that explains the principles behind the underlying language processing tasks.

2.2. Related Work

Plagiarism detection is considered a Natural Language Processing task, and therefore, most of the proposed solutions for syntactic or lexical plagiarism are of “concept extraction” using a corpus. NLP and its subtasks are hot trends in researches and the industries as well. Authorship Analysis is closely linked to other fields of NLP, such as authorship attribution and sentiment analysis. This section is a review of a few of the attempts made to solve these tasks.

As mentioned above, to achieve greater results when solving NLP tasks, it is best to choose and use the fittest embedding method. It is not a simple task, especially where resources of the specific task are few. For instance, [3] proposed a model for sentiment analysis of Arabic tweets even though, relatively, there are not enough available resources like tools or research on the specific challenges of the language in the field of NLP. [3] used the AraVec word embedding and an ensemble model of CNN and LSTM, to predict the sentiment of the tweets. [3]s outstanding approach is applied. For the activation function, ReLU was chosen at the fully connected layer, just before a dropout and SoftMax output layers.

A paper by Z. Volkovich (2020) [2] presents a novel approach to solve the task of the General Authorship Attribution problem and investigates medieval Arabic documents. [2] produced evaluations regarding two manuscripts, questionably attributed to the famous Islamic jurist, theologian, and mystical thinker Abu Hamid Al Ghazali. Taking the “Tweets Approach”, he divided the documents into tweets and analyzed them. The division into tweets is important when “The devil is in the details", meaning that the general characteristic of the document may not be close to the author’s writing style, but there are few points in the texts that add the author’s “touch”. Medieval writings require this approach the most since it is full of external quotations and texts that do not belong to the writer.

[7] Used and tested the Impostors Method to classify blog pairs and assign them either to “Belongs to author” or “Does not belong to the author”. Tests are conducted to evaluate performance with different algorithms and different ways of choosing input impostors. Among the other insightful findings, [7] shows that best results are yielded when using the impostors' method and especially when picking impostors with relatively similar writings.

3. Expected achievements

3.1. Outcomes

The main goal is to develop an improved system capable of detecting plagiarism with an accuracy rate higher than its predecessors. In addition, the system should include a comfortable and simple-to-use Graphical User Interface (GUI) so that it would be usable by any professional and recreational users.

Firstly, the improved system should be trained with the creations of pairs of authors and should distinguish properly between them. A CNN and LSTM model fed with tweets from the authors' works in supervised training should classify the literature. Then, a test text applied to the system should be classified into one of the classes. The chosen class represents a writing style that is more like the tested author than the other one. The algorithm is expected to pick the suitable class, from a mathematical standpoint with very high accuracy since it is the most important building block. Even if a man cannot notice any semantic or so ever differences between the authors' writings and the tested piece, the piece should still be classified as mathematically closer to a certain style. Eventually, if all of the same person's test documents were written, they should all be classified the same. So, to validate plagiarism or to object to the claim, a comparison is made to make sure that the questionable literature was classified the same as the author's other works. Since there is a chance that the system will state plagiarism when it is not the case or the opposite, the algorithm should run iteratively with many different pairs of authors, assuming that extreme cases like copycat writers or genuine writers, deciding to write distinctively different may occur.

3.2. Graphical User Interface (GUI)

While the algorithm might be complex for non-professionals, using it should be intuitive and straightforward. End users only need to point the system at the literature. A brief look at the GUI's screen is expected to be enough to receive all the information needed, such as the main statement (i.e., whether the author is genuine) and the accuracy measure of the produced statement, alongside a clear, pleasant, illustrated graph.

Another great and unconventional feature would be the display of a meter, representing the quality of the given input. Mathematically it would be better to have two relatively close classes (or writing styles) as input. For an extreme instance, taking medieval documents alongside modern ones would not yield good results since most of the medieval, for example, will be closer to a medieval text even if their semantics are distinct.

4. Research process

4.1. Process

After examining the components alongside our supervisors, the proposed model of [3] seems to be the most robust and will be followed during the project. It is to be remembered that with the intention to analyze documents of Russian authors several adjustments need to be made.

For preprocessing, The Natural Language Toolkit (NLTK) is the fittest, known as one of the best tools with comprehensive documentation and support of many languages. Usage of NLTK was proven to yield better results, and it is also easy to use.

The ELMo Embedding is the fittest technique of word representation. It is because ELMo looks at the entire sentence before assigning each word an embedding. Thus, it takes into consideration the context meaning and can improve the system's accuracy. However, ELMo does not provide Russian language support. Therefore, in this project, pre-trained embeddings by [8] is being used.

These constraints appear since the research process in the field of authorship analysis aims to embed emotions to improve accuracy and writing styles that are not mathematically defined well.

- In the Russian language, like in other Slavic languages and especially in literature, complex sentences containing two or more clauses are usual. Also standard is the usage of complex words with two or more roots and diverse meanings. Therefore, it might pose a challenge with embedding semantics.
- The embedding technique’s corpora are trained with modern writing from Wikipedia (also, scientific writing mostly), and thus embedding style characteristics, emotions, or hidden meaning of non-modern writings could be challenging.

The heart of the system is its algorithm and the ability to analyze authorship accurately. Therefore, the first and most major principle that is followed when building the GUI is “simplicity”. Complete model and research work need to be transferred into a usable system available for anyone, regardless of knowledge with computer science theory and background with IT systems.

CNNs excel in classification problems, capable of extracting essential and relevant features for the classification task. The Bi-LSTM model is great because it handles long-term dependencies. Text divided into tokenized chunks keeps the semantics and context of the next and previous chunks. For instance, if an external writing style from a quotation has been recognized, it counts in the following chunks, and the model recognize it.

4.2. Product

This part describes the main configuration of the system and its web-based dashboard prototype that is developed to allow experimenting and endpoint using the model. First, datasets of literature are being preprocessed to be ready for training. Then, chosen LSTM and CNN models will be trained for distinguishing between two different authors(“Impostors”) labeled as A and B. At the beginning of the test phase, the investigated author texts will be loaded for a test. All his works are expected to be labeled as only one of A or B. Specifically, the work-under-test should also be classified the same if it is not plagiarism. Then the whole process is being repeated several times with the next Impostor’s pair C and D etc. At the end of the experiment, it is expected that the investigated author will behave the same in every test iteration. Below is the flow diagram, visualizing the system flow of actions involving users’ actions and machines.

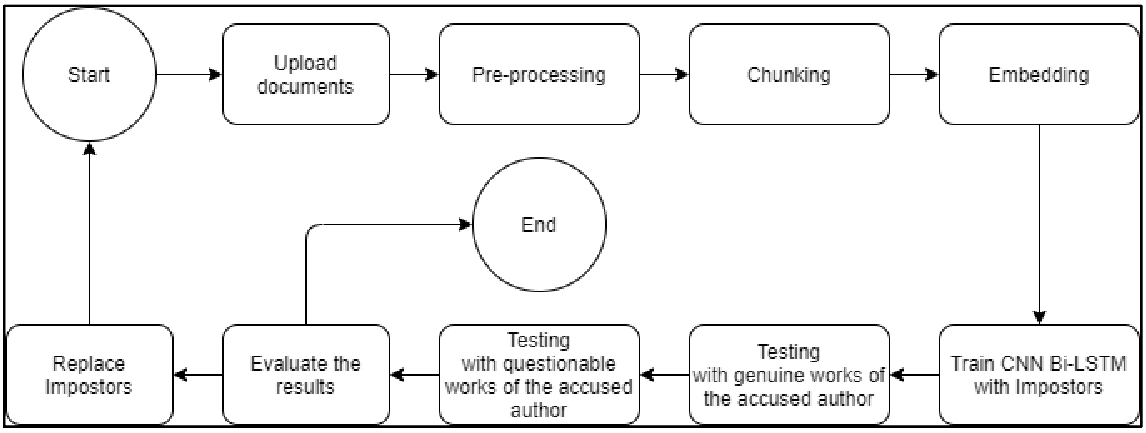

Figure 1. Flow diagram of the training process

4.2.1. Data preprocessing and embedding

In this step, the input documents are being cleaned from unwanted noises such as numbers, dates, and special characters. NLTK is capable of preprocessing Russian words and texts and therefore, it is a good tool to have in the system.

With the aim to analyze short patterns of text, the system prepares tweets or chunks from the document. The tweets are basically a variable-length set of words from the text. To embed the refined text, the algorithm is using the ELMo embedding library, where each chunk is represented as a 2D vector of dimension $n\ x\ d$ where n is the number of words in chunk and d is the dimension of the embedding space.

4.2.2. CNN

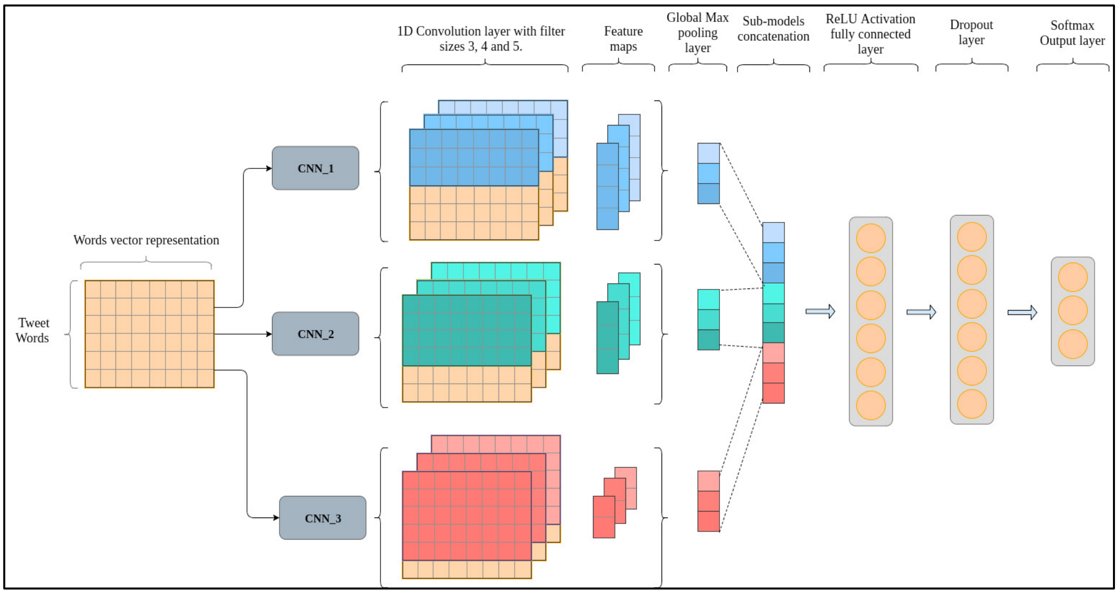

Figure 2. CNN Architecture model

The CNN model is built to distinguish between two authors' writing styles. The model of [3] is followed. The input of the network is a matrix of "tweets", which are tokenized into words. This model contains three parallel CNN sub-models where each of them has a certain filter size s and m filters. Feature maps received from each filter will be of size $n - s + 1$. Each sub-model has the same number of filters. Each output yields a vector of size m.

Several convolution operations are applied to the matrix of chunks. The convolution involves a filtering matrix $w \in R^{h \times d}$, where h is the size of the convolution. The convolution operation is defined as:

$$c_i = \mathrm{f}\left(\sum_{j,k} w_{j,k}\left(X_{[i:i+h-1]}\right)_{j,k} + b\right)$$

- $b \in R$ – Bias value
- $f(x)$ – Nonlinear function (during this research, ReLU function)

Later then, a max-pooling is used and producing $c_{max} = max(c)$. This gets the most important feature for each c_i. Also, it is applied to allow us to combine all the c_{max} of each of the m filters into a single vector. The vector is then passed through a small FC ReLU activation layer and then in turn through a soft-max layer to produce final probabilities of the classifications. After the max-pooling and the FC layers, an added dropout layer is intended to prevent overfitting.

4.2.3. Bi-LSTM

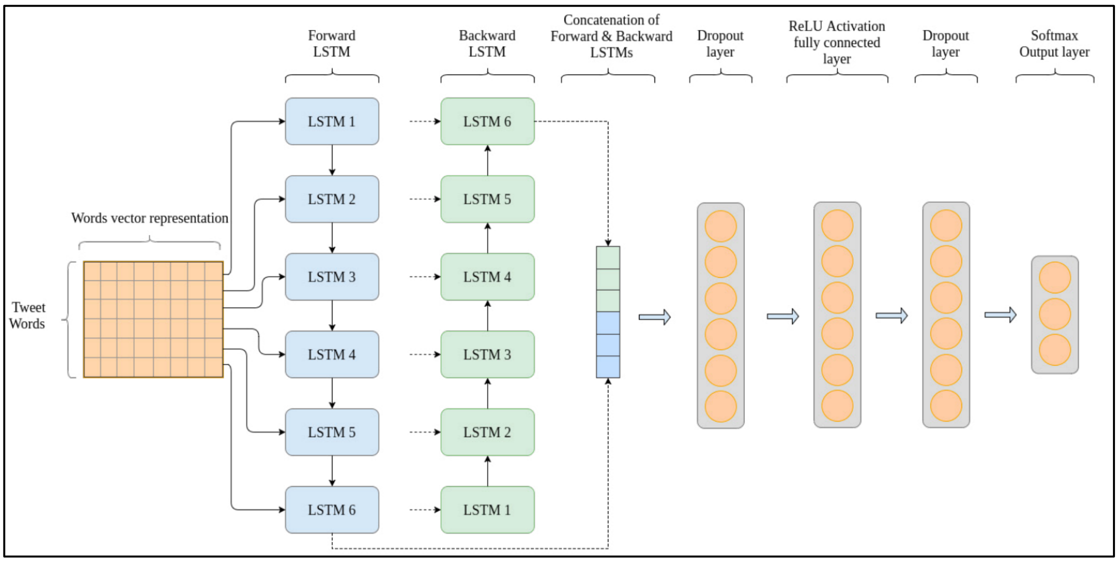

Figure 3. LSTM Architecture model

As an extended kind of RNN, LSTM is capable of learning long-term dependencies. The RNN hidden state value at a certain time is calculated as:

$$h_t = \mathrm{f}(W_h \cdot x_t + U_h \cdot h_{t-1} + b_h)$$

Where x_t is the embedded word at the time t, W_h and U_h are weight matrices, b_h is a bias value and f is a non-linear function, usually $tanh$. Due to the Vanishing Gradient problem, this simple RNN does not fit our task where predictions based on previous states are needed. LSTMs solve this problem and calculates its hidden state by:

$$f_t = \sigma\left(W_f \cdot x_t + U_f \cdot h_{t-1} + b_f\right)$$

$$i_t = \sigma(W_i \cdot x_t + U_i \cdot h_{t-1} + b_i)$$
$$o_t = \sigma(W_o \cdot x_t + U_o \cdot h_{t-1} + b_o)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ tanh(W_c \cdot x_t + U_c \cdot h_{t-1} + b_c)$$
$$h_t = o_t \circ tanh(c_t)$$

The problem with LSTM is that it reads only in one direction. Thus, it does not consider post-word information well enough. Bi-LSTM, or two LSTMs with their outputs stacked together, is solving this matter excellently. Since there are two LSTMs instead of one, they can read forward and backward now. The chunks are passed to each layer of the LSTM, each LSTM is of size h. The final output of each LSTM is concatenated to generate a vector of length $2h$. The concatenation output is fed to a FC ReLU activation layer, afterwards it is going through a soft-max layer. Lastly, an additional dropout layer is used in a similar way to the way it was in the CNN architecture. The dropout layers are places before the FC ReLU activation function and before the Soft-Max layer.

4.3. Network Parameters

In these tables, the initial suggested parameters values of the system are shown. Using the developed GUI, these parameters may be changed in future use. Finding the best combination of parameters for the system is a complicated task but crucial to achieving optimal results. We are going to prove it via a massive numerical experimental study.

4.3.1. CNN Component default parameters:

Hyper-parameter	Value
Filter sizes	[3, 4, 5]
Number of filters	200
Number of units in fully connected layer	30
Dropout rate	0.5
Learning rate	0.001
Number of epochs	10
Batch size	50

Table 1. CNN default parameters

4.3.2. Bi-LSTM Component default parameters

Hyper-parameter	Value
LSTM hidden state dimension	200
Number of units in fully connected layer	30
Dropout rate	0.5
Learning rate	0.001
Number of epochs	10
Batch size	50

Table 2. Bi-LSTM default parameters

4.4. Graphical User Interface

The prototype for a web-based dashboard built with the Streamlit library and Python. At the beginning of each experiment, the GUI allows uploading of all the required datasets and adjusting the parameters of the preprocessing phase and the neural network/LSTM parameters as well. Whenever Impostors datasets are uploaded, it shows a computed distance in the 'Impostors proximity meter'. After the user presses the button "Start training", it begins the training process and shows up all required information and plots to analyze the author's authorship.

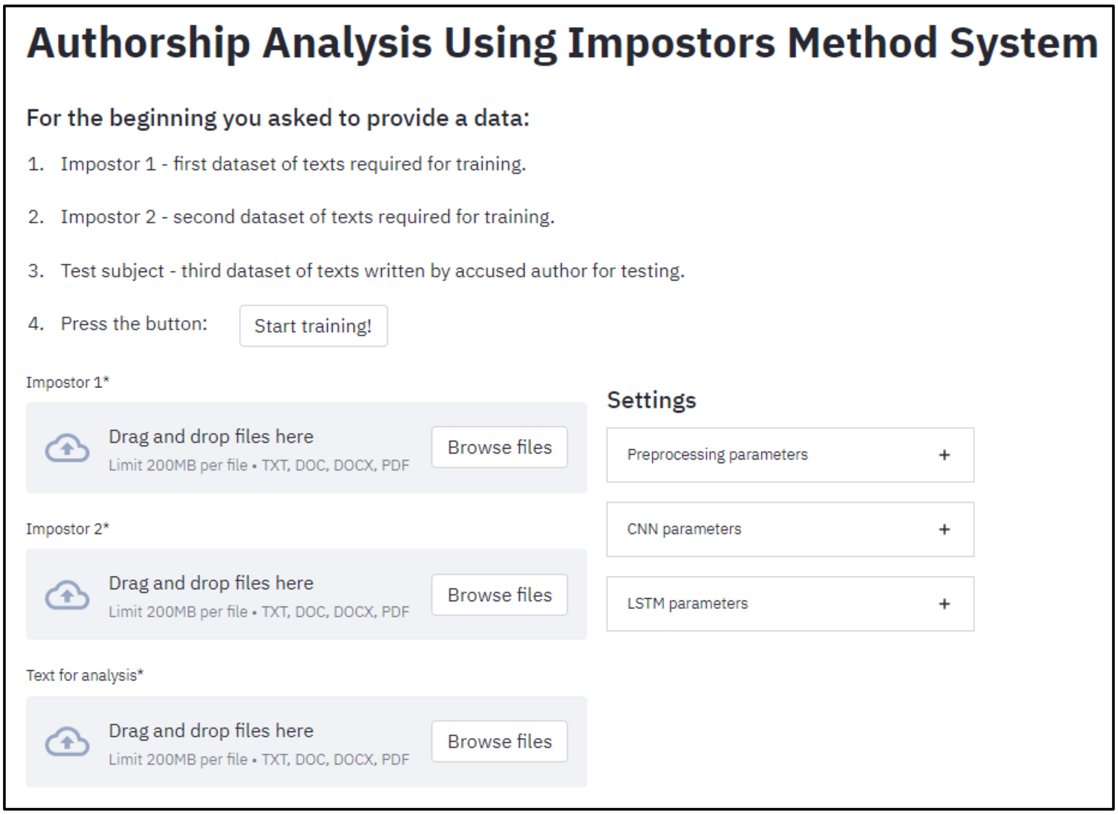

Figure 4. First screen with settings

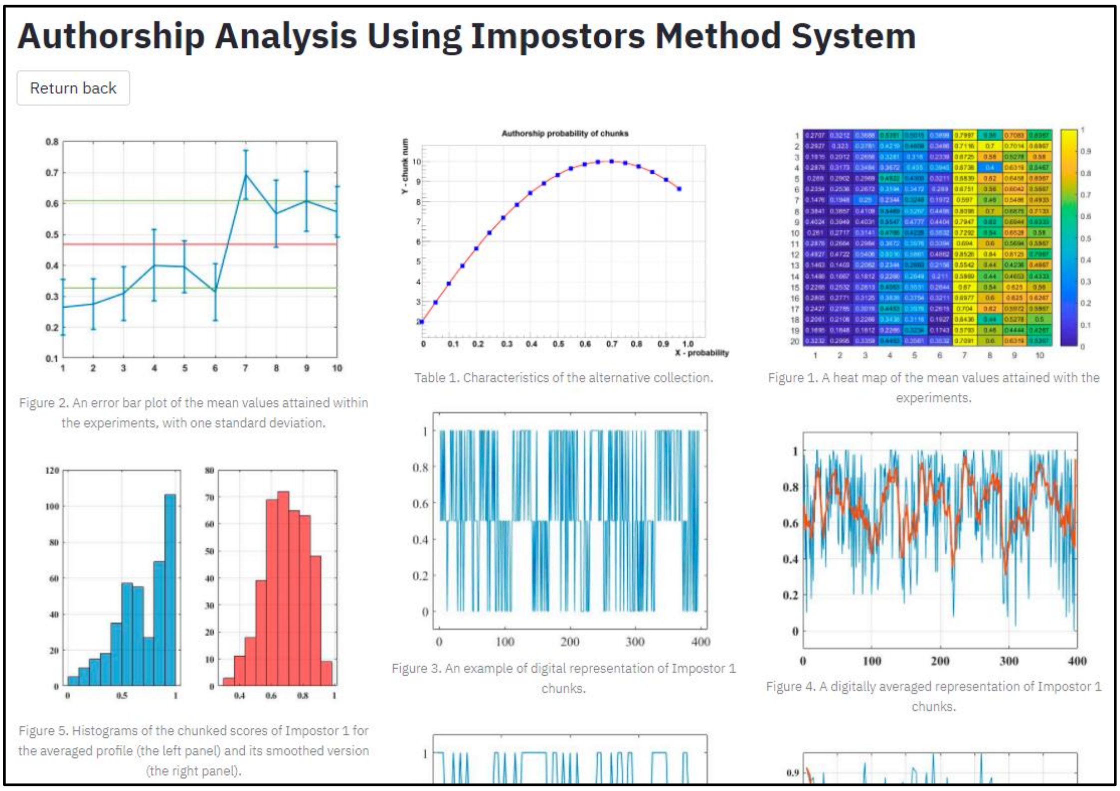

Figure 5. Second screen with plots and data

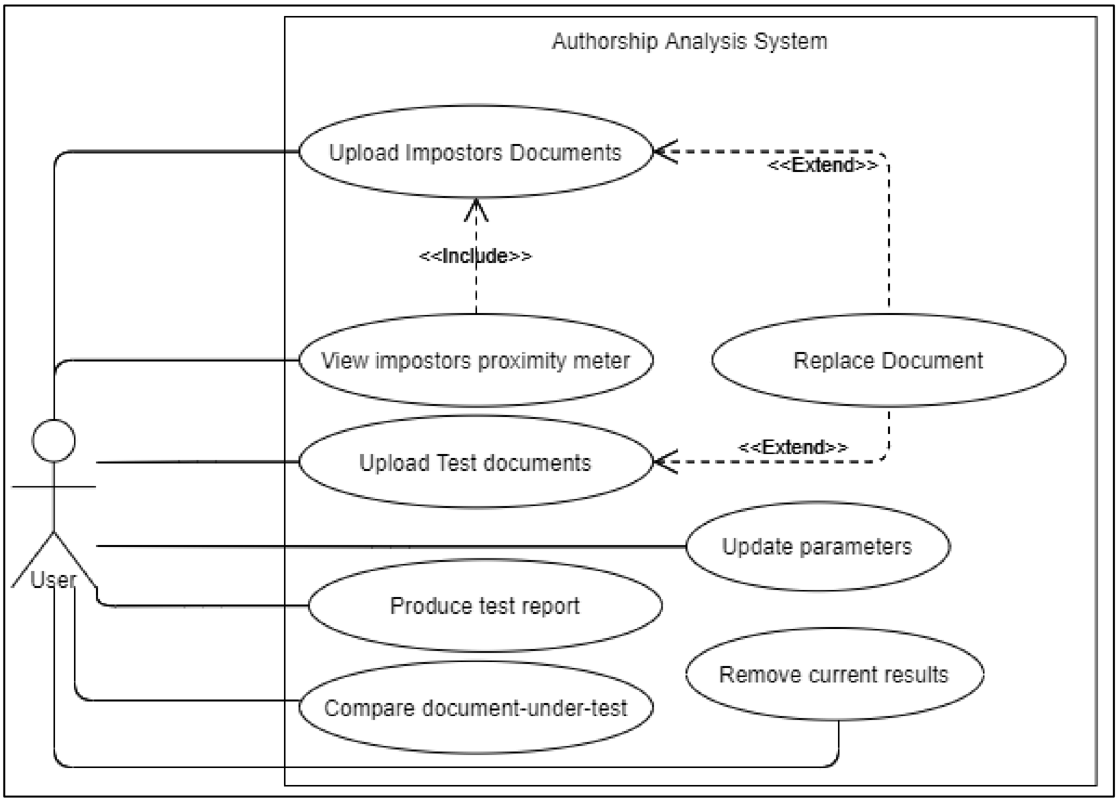

Figure 6. Use case diagram of the GUI's use case

5. Evaluation and Verification Plan

The best course of action for evaluating the system behavior would be the widely used "Black-Box Testing" approach alongside monitoring the algorithms' loss and Accuracy values. Cover testing is the most acceptable form of development of test cases, practiced by industries and researchers worldwide. Thus, the "Cover testing" principles are being applied and used in this research as the main form of the system evaluation. Previously tested and accepted tools are used to build the system, and they will not be tested again, for example the preprocessing or embedding frameworks, assuming they are functioning correctly. ELMo embeds, and CNN classifies, and there is no need to make sure of it. The need is to make sure that the classification and other predefined tasks are being solved accurately. Thanks to the "Impostors proximity meter," avoiding underfitting is possible; this aspect should not be checked.

In table 3, test cases for possible combinations of the impostors' datasets and the system expected behavior regarding the "Impostors' proximity meter" functionality. Table 4 presents Black-Box tests of possible classifications of the questionable literature under the assumption that the system is previously fed with an appropriate pair of the impostors and the assumption and tested with genuine works. Finally, GUI testing is presented in table 5 and is validating the proper functioning of most of the GUI operations by presenting operations available for the user and the required outcome of them, done by the system.

Test Case	**Input**	**Expected Result**
Case A	Two impostors with similar writing style	Impostors' proximity meter displays good feedback.
Case B	Two impostors of different times/different genre	Impostors' proximity meter displays bad feedback.
Case C	The two impostors set both consist of the same author different works	Impostors' proximity meter displays a great feedback
Case D	Two impostors writing of different languages	Impostors' similarity meter displays an erroneous feedback: "Impostors are too distinct"
Case E	The two impostors' sets are the same	Impostors' similarity meter displays an erroneous feedback: "impostors are the same"

Table 3. Expected distance computations of impostors

Test Case	Input	Expected Result
Case A	A genuine work of the accused author	Classified the same as most of the other works with high accuracy rate
Case B	Plagiarism – A work that has been written by an author different then the accused	Classified as either of the classes regardless of the genuine work's classifications
Case C	A partially plagiarized text – A work that was written by the accused author and by another as well	Classified the same as most of the other works with low probability

Table 4. Expected predictions for model inputs

Test Case	Event	Expected Result
Case A	No datasets had been uploaded or only one and the user clicked "start"	GUI displays Error: "Please enter the three required datasets"
Case B	Two datasets had been uploaded and the user clicked "start"	GUI displays the "Impostors proximity meter" and prompt "There is no text for analysis"
Case C	Three datasets had been uploaded and the user clicked "start"	GUI displays the "Impostors proximity meter", starts training and append results to dashboard
Case D	Invalid files uploaded and the user click "start"	GUI displays the "One or more files are invalid"

Table 5. Uploading datasets

An excellent text suited for the evaluation of the system would be "And Quiet Flows the Don" as the literature under-test alongside unquestionable kinds of literature of Sholokhov. It is to be remembered that the vast majority of researchers around this novel supported Sholokhov from a mathematical perspective. Therefore, another document is gathered to execute the test cases in an unbiased manner properly.

References

1. el Mostafa, Hambi & Benabbou, Faouzia. (2020). A deep learning based technique for plagiarism detection: a comparative study. IAES International Journal of Artificial Intelligence (IJ-AI). 9. 81. 10.11591/ijai.v9.i1.pp81-90.

2. Volkovich, Zeev. (2020). A Short-Patterning of the Texts Attributed to Al Ghazali: A “Twitter Look” at the Problem. Mathematics. 8. 10.3390/math8111937.

3. Heikal, Maha & Torki, Marwan & El-Makky, Nagwa. (2018). Sentiment Analysis of Arabic Tweets using Deep Learning. Procedia Computer Science. 142. 114-122. 10.1016/j.procs.2018.10.466.

4. Kim, Yoon. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. 10.3115/v1/D14-1181.

5. Cliche, Mathieu. (2017). BB_twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs. 573-580. 10.18653/v1/S17-2094

6. Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. Neural Comput. 9, 8 (November 15, 1997), 1735–1780. DOI:https://doi.org/10.1162/neco.1997.9.8.1735.

7. Koppel, Moshe & Winter, Yaron. (2014). Determining If Two Documents Are Written by the Same Author. Journal of the Association for Information Science and Technology. 65. 10.1002/asi.22954.

8. D. R. Baymurzina, D. P. Kuznetsov, and M. S. Burtsev, ‘‘Language model embeddings improve sentiment analysis in Russian,’’ in Proc. Int. Conf. Dialogue Comput. Linguistics Intell. Technol., vol. 18, 2019, pp. 53–63.