Doppelgänger Detection Algorithm

Study Project Report

Alvin Fauzi Murod
Mina Lee

Date of Submission: 24th of March, 2021

This report is submitted to the

Chair of IT Security

Brandenburg University of Technology, Germany

Examiners:
Prof. Dr.-Ing. Andriy Panchenko
Asya Mitseva, M.Sc.

Contents

1 Introduction

In today's digital era, the development of the Internet is very fast and provides many advantages to people. The Internet provides a capability so powerful and general that it can be used for almost any purpose that depends on information, and it is accessible by every individual who connects to one of its constituent networks. The Internet supports human communication via social media, electronic mail (e-mail), chat rooms, newsgroups, and audio and video transmission and allows people to work collaboratively at many different locations. In the absence of the Internet, it is likely that we would not be able to communicate remotely to people in different regions and time zones.

People mostly tend to spend their time by surfing on the Internet such as viewing the social media, reading article news and perhaps doing online shopping. Social media forums are a useful way for users to share opinions, obtain information about products and services, as well as to discuss political issues. They are also an essential channel for word-of-mouth marketing. While doing online shopping, people prefer to purchase the product which has good reviews even though they do not know whether the review comes from a real user or a fake user. Besides the advantages, the usage of the Internet can also be misused by some people. The spread of fake news, hate speech, opinion manipulation, and cyberbullying are some of the prominent examples that are happening on the Internet and it can not be controlled. These abuses can extend to defamation, propaganda, even worse, e.g., chaos. The practice of opinion manipulation has been a reality since the rise of online community forums. It has been shown that user opinions about products, companies and politicians can be influenced by other users' opinions [24].

Statista [35] which is a statistic website, reports that there are over three billion users of social media in the world, many of whom maintain accounts on multiple platforms. The total number of social media accounts may be more than three billion, making the task of sorting out people (and their braver/sassier/sexier online alter-egos) from commercial, political, and general trolling accounts (an account that is used to spreading the hate or make a troll) a huge technical challenge. According to Facebook [14], almost all fake profiles were discovered before they became active users of the social network. So they would not be added to the number of users that the company publishes on a regular basis. Facebook estimates that at least 412 million Facebook accounts are fake or duplicate which is approximately 5% of Facebook's worldwide monthly active users [14]. In the second and third quarters of 2020, Facebook says it disabled more than 2.8 billion fake profiles [9].

Furthermore, fake information or fake news were also affecting the election in United States of America. In the final stage of the American election campaign, most U.S. citizens did not gain an understanding of how to differentiate between fake news and real news through their social networks and this can yield a huge problem and affect the election. People will

not believe every single false message. But if the distortions and forgeries accumulate, the boundaries of what can be said and conceived are shifted [8].

Facebook states in a transparency report [14] and announced the progress in the fight against harmful contents/posts such as terror, suicide, child abuse and drugs, bullying, hate speech and other categories which are categorized in the transparency report [14]. Facebook had removed around 18.5 million harmful contents within six months. In the previous six months, it was 13 million. The increase of the number of removed harmful contents is due to improved detection technology that is implemented by Facebook. However, the technologies are not perfect and need to be improved. Therefore, Facebook continues to invest in better recognizing posts that violate its own principles [29].

The research on how to mitigate the risks or impacts that come from fake accounts or doppelgänger accounts, i.e., duplicated accounts has been done by some researchers and it is still going on. According to Wikipedia [36], "a doppelgänger is a biologically unrelated look-alike, or a double, of a living person". Thus, a doppelgänger account is more likely a duplicated account that is being used to do something online without revealing its identity, i.e., pseudonymization. Jin et al [19], proposed an approach to detect the fake accounts in online social networks. They used a technique that analyzes and characterizes the behaviours of a fake account based on attribute similarity. On the other hand, Afroz et al [2], proposed another technique to detect a doppelgänger by using stylometric features in a comment news' section.

Controlling the spread of fake news, fake information, hate speech, and opinion manipulation is hard to do without the help of technologies or algorithms. However, there is still a way to detect the doppelgängers on the Internet and prevent the spread of misleading information caused by doppelgängers. In our project, we will implement the doppelgänger detection algorithm on the news website which is *ZEIT ONLINE*. Due to anonymization techniques, e.g., using VPN (Virtual Private Networks) or Tor (The onion routing), it is quite challenging to detect and find the relationship, e.g., find the similarity behaviour between two or more accounts that correspond to the same user [12]. Thus, we proposed to use machine learning algorithms in terms of detecting the doppelgänger by observing the writing styles with the help of stylometric features. Stylometric is a technique of analyzing texts for evidence of authenticity and author identity. Some examples of stylometric features are word-length frequency distribution, sentence length, word n-grams, character n-grams, PoS (parts of speech) tags, function words and content words.

The aforementioned reasons come to our research question which is "How effective is the doppelgänger algorithm in order to detect doppelgänger in *ZEIT ONLINE* News' comment by using stylometric features?". With regard to implement the algorithm, the project is formulated as follows :

1. Collect the article title, comments, published date, and the author's name from *ZEIT ONLINE* news website. We only collected the information of comments from unique users which have written more than 100 comments on the *ZEIT ONLINE* comments' section and stored it in the database as a dataset.

2. Implement the doppelgänger detection algorithm using the dataset that is collected from *ZEIT ONLINE*'s comment section.
3. Evaluate the implementation of doppelgänger detection for news comments which based on stylometric features.

The remaining sections are structured as follows. Section 2 is a review of background. Section 3 introduces our proposed approach. Section 4 presents the experiments conducted to evaluate our doppelgänger detection algorithm. Section 5 makes concluding remarks and discusses future work.

2 Background

In this chapter, we would like to elaborate the terms that are related to our project. This chapter consists of the explanation of doppelgänger and machine learning algorithms which will be used further in our project.

2.1 Doppelgänger

According to Wikipedia [36], "a doppelgänger is a biologically unrelated look-alike, or a double, of a living person". The terms of doppelgänger sometimes used in a fiction and folklore which refers to a specter or wraith of an alive person. In online forums, doppelgänger can be referred to as someone who has duplicated account. The purpose of people having a doppelgänger account mostly because they do not want to reveal his/her identity in online forums. Nevertheless, the advantage of having doppelgänger account sometimes can be misused by some people and the motives are vary. Nowadays, some people use doppelgänger account to do such bad things such as spreading hate speech, providing fake information, cyberbullying, faking the online review of a product, and even propaganda.

2.2 Machine Learning

According to the Machine Learning for Dummies book [28] Machine Learning is a branch study of Computer Science and Artificial Intelligence (AI) that focused on building applications that learn from data and improve their accuracy over time by itself without explicitly programmed. There are two types of machine learning algorithm such as supervised and unsupervised. The difference between supervised and unsupervised machine learning lies on the existence of labels in the training dataset [5]. A machine learning algorithm will iteratively learn from data to improve its accuracy, describe the data and predict the outcomes from the data. To do so, the process of training the data is needed. A machine learning algorithm will then produce a model based on the data which will be used to generate the output from the data. Machine learning intersects mathematics and statistics fields.

2.2.1 Supervised Machine Learning

Supervised machine learning is one of the machine learning algorithms based on training a data sample from data source with correct classification already assigned [28]. Supervised machine learning is used when the dataset contains a label that is associated with the data then the algorithm learns to predict the output from the input data. For instance, supervised

machine learning can predict the email spam, the house prices and many more. In the dataset, the labels are already provided and they can be a category, e.g., spam or not spam, or a number, e.g., 0 or 1. When training the data, the machine learning algorithm will be given a data and it will create a model. The training phase will incrementally improve the model's ability to predict the result.

Support Vector Machines

Support Vector Machines (SVM) is a supervised machine learning algorithm that based on vector representations which can be used either for classification or regression problem. SVM does separation between data of two classes using separating line, i.e., hyperplane. In classification problem, SVM uses margin to determine the optimal hyperplane. Generally, the margin can be defined as a distance between the hyperplane and the closest points which will be used for generalization [26]. SVM can scale the high dimensional spaces data effectively but it is not suitable for large dataset and can take a lot of resources for processing.

K-Nearest Neighbors

K-Nearest Neighbor (k-NN) is a machine learning algorithm based on a distance method. Basically, k-NN will classify the data based on the nearest neighbors. For instance, if there is a new data given to the machine learning then k-NN will calculate which class is suitable for the new data [26]. Suppose we are given a new data (new point) that needs to be classified, choosing a number of neighbors or K will be the first step and it will affect the classification. Calculation of each distance's point can be done by using euclidean distance. Once the distances are calculated, to choose either the new point is classified as class A or B, k-NN uses a majority voting to do classification process. However, the number of Kshould be odd number and bigger than 2 to avoid confusion between two classes of data. If $K = 1$, the data point obviously will be classified easily by looking at the nearest point but the classification will be biased. If $K = 2$, classifying process will be quite challenging since the k-NN uses a majority voting to classify the data when the number of class is equal, e.g., one data for class A and one data for class B. Hence, if we use $K = 3$, the classification process will be done easily by looking at the majority points associated with the classes, e.g., two data for class A and one data for class B then the data will be classified as a class A [26]. Calculation of each distance's point can be done by using euclidean distance. Once the distances are calculated, to choose either the new point is classified as class A or B, k-NN uses a majority voting to do the classification process. K-NN is considered as a lazy learner since it does not have a training phase thus, it does not save the model for predicting but redoing a calculation of distance for each prediction instead.

Random Forests

Random Forests (RF) is a machine learning algorithm based on decision trees. RF will construct multiple decision trees during the training phase as a model then will take a decision to classify the data. The more number of trees (n) in the random forests, the more robust the

prediction as well as the accuracy. The number of trees (n) is established before taking the majority voting of predictions. RF is also very useful with regard to have a good prediction result due to its versatility and can handle large dataset with high dimension and dealing with missing values [6]. However, one of the disadvantages of RF algorithm is when using a large number of trees (n), it can slow down the performance for real-time predictions.

2.2.2 Unsupervised Machine Learning

Contrary to a supervised machine learning algorithm, an unsupervised machine learning algorithm does not contain a label in the data. The unsupervised can be defined as clustering the data into the class without a prior knowledge of the classification. Clustering is the organizing process of unstructured data into groups which have similarity [5]. In other words, unsupervised machine learning classifies the data and makes a new class based on its similarity.

Principle Component Analysis

There are several techniques to implement unsupervised machine learning such as Principal Component Analysis (PCA). According to Shlens et al [34], the main idea of PCA is to reduce the dimension or the number of features which have a high variance within the dataset. One of the advantages of PCA is reducing the processing time. In order to do calculation of PCA, there are some steps namely :

1. Standardize the data
2. Calculate the Covariance Matrix
3. Calculate the Eigenvectors & Eigenvalues
4. Compute the Feature Vectors

Standardize The Data

The main goal of standardizing the data is to standardize the range of the attributes thus, each one of them lie within similar boundaries. This process involves the removal of mean from the variable values and scaling the data with respect to the standard deviation [13].

$$Standardized\ value\ of\ x_i = \frac{x_i\ - \overline{x}}{standard\ deviation\ of\ x_i} \tag{2.1}$$

The formula 2.1 [18] shows how to standardize by calculating the variable values (x_i) and subtracting it with the mean of variables ($\overline{x}$) then dividing it with the standard deviation of (x_i).

Covariance Matrix

Covariance matrix is used to express the correlation between any two or more attributes in a multidimensional dataset. The covariance matrix has the entries as the variance and covariance of the attribute values [13]. The calculation of the covariance matrix can be seen in the formula 2.2 [2]. X and Y are two random variables, whereas (x_i) and (y_i) are the variable values and the $(\overline{x})$ and $(\overline{y})$ are the mean of X and Y. N is the number of the data, in our project, N is the total number of the comments.

$$cov(X,Y) = \sum_{i=1}^{N} \frac{(x_i - \overline{x})(y_i - \overline{y})}{N} \tag{2.2}$$

Eigenvectors and Eigenvalues

Eigenvectors and eigenvalues are the mathematical values that are extracted from the covariance table [13]. They are responsible for the generation of new set of variables from old set of variables which further lead to the construction of principal components. Eigenvectors is a vector that will not change during the transformation, whereas eigenvalues are the scalars of the eigenvectors. The formula 2.3 shows how to calculate eigenvectors and eigenvalues. The matrix is denoted by A where v is the eigenvectors and λ is the eigenvalues [18].

$$Av = \lambda v \tag{2.3}$$

Feature Vectors

Feature vector is a matrix that consist of eigenvectors of the components that we decide to keep as the columns. The less significant prinicipal components will be discarded. The formula 2.4 shows how to calculate the PCA [2] to reduce the number of features or dimension. The selected dimension is denoted by K, whereas N is the original dimension and λ is the eigenvalues.

$$PCA = \frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > 0.999 \tag{2.4}$$

3 Our Approach

Our project consists of three steps: (1) collection of a sufficient number of news comments posted by numerous users in the news website *ZEIT ONLINE*, (2) generation of literary features from the collected user comments, (3) implementation of a doppelgänger detection algorithm for news comments.

3.1 Dataset

To proceed the steps of our project, collecting a sufficient number of user comments posted in response to articles in a news website *ZEIT ONLINE* is needed first. We collected the comments from 50 unique users with at least 100 different comments posted by each of users and only when each comment contains at least 50 words because the comment which is not long enough is not suitable for getting proper writing style from the user.

To collect the comments dataset from the web pages, we downloaded the web pages which contain required information on a news website *ZEIT ONLINE* using *requests*[1] library and examine the HTML structure closely to identify the particular HTML element from which to extract data. To do this, right click on the web page in the browser and select *inspect* option to view the HTML element structure. Then, *BeautifulSoup*[2] library is used to find the particular element from the HTML response and extract data which is needed to get by defining the name of tags in the HTML markup. Since HTML is made up of tags and it stores data in them, we can get data which is needed when finding the tags which contain data we need through inspecting the HTML structure.

We created two tables which are "articles", "comments" in the local database. In the table "articles", name of article title, URL of article, author name are stored and information about name of article title, username, content of comment, published date of comment are stored in the table "comments". The data which is the URL of article from "articles" table is used to obtain information about comments. We crawled each article which is stored in "articles" table and collected detailed information about the comments posted in response to this article as well as the users posted the corresponding comments. When storing data into the database, only data which is not already present in the database is stored and that means the database does not contain any duplicate data. As a result, we obtained 68,702 comments posted in response to 26,712 articles from 1,856 users. When manually checking the data, some comments have exactly same contents but they were written by different users.

[1] `https://requests.readthedocs.io/`
[2] `https://www.crummy.com/software/BeautifulSoup/`

3.2 Pre-processing

For generating literary features from the comments collected from web crawling, the pre-processing is needed first to standardize the texts. Pre-processing is a technique that is used to convert the raw data into a clean data set or an understandable format. Here, we conduct pre-processing by removing the stop words and lemmatizing all words. Removing stop words can potentially help improve the performance as there are fewer and only meaningful tokens left. Thus, it could increase classification accuracy. Similar to the reason why we remove stop words, lemmatization is also implemented to get more classification accuracy since machine could identify the meaning of root words more than inflected words. The result of pre-processing is saved in a CSV format. The CSV format, which stands for comma-separated values, is a file format used by many external machine learning tools.

The stop words are words which do not add much meaning to a sentence. Therefore, they can safely be ignored. For example, in English, the words like "the", "a", "an", "in", "I", "we", etc. are stop words and in German, the words which are "der", "die", "das", "mit", "und", "oder", etc. are stop words. The Natural Language Toolkit, *NLTK*[3] was used for removing stop words.

The lemmatization is the process of converting a word to its base form. For example, in English, the lemmatized word "play" is from "playing", "plays", or "played" and for German, the word "spielen" is lemmatized from "spielt", "spielte", or gespielt". *spaCy*[4] was used for lemmatizing. *spaCy* is a library for advanced Natural Language Processing.

3.3 Feature Generation

Since our doppelgänger detection algorithm is based on the literary styles of users, we generate a set of stylometric features from our collected comments. Data obtained from pre-processing and stored in a CSV file is used to generate features.

Features are separated by word-level, vocabulary richness, sentence-level, frequency of used punctuation and the repeated occurrence of white space, content-based, idiosyncratic, and additional features. The total number of features per comment is 35 in our case.

Word-level Features. We counted the average number of characters, lowercase/uppercase letters, and digits per word from each comment. Moreover, we calculated the total number of words per user comment and the frequency of large words. We defined the word which is composed of at least 10 characters as a large word.

Vocabulary richness. The number of syllables per word, type-token(i.e., word types in a comment) ratio, the entropy of different vocabulary items in a comment, *Simpson's D measure*, *Sichel's S measure* were computed.

[3] `https://www.nltk.org/`
[4] `https://spacy.io/`

We used Shannon's Entropy to obtain entropy of different vocabulary items in a comment. Shannon's Entropy is an estimation of the average amount of information stored in a random variable. The sub-package *scipy.stats*[5] from *SciPy*[6] was used to calculate Shannon's entropy.

Simpson's D measure is a vocabulary richness measure defined by Simpson. It computes the probability that two words which are randomly selected from a comment will be the same words. Equation 3.1 shows how Simpson's D measure is measured [7].

$$D = 1 - \left(\frac{\sum n(n-1)}{N(N-1)} \right) \tag{3.1}$$

n: the total number of a particular word
N: the total number of all words

Sichel's S measure is a vocabulary richness measure defined by Sichel and it describes an individual's lexical diversity. Further information about how to calculate Sichel's S measure can be found in [4].

Sentence-level Features. The number of short sentences, the number of long sentences, and the average sentence length in characters per comment are counted. We defined a short sentence when it consists of equal to or less than 20 words and a long sentence is a sentence which contains at least 50 words. The sentence The sentence which contains more than 20 words up to 49 words is considered as a medium sentence and it is neither counted as a short sentence nor a long sentence. In addition, we extracted features that calculate the *Flesch-Kincaid grade* level of the sentences and the average number of words per sentence.

The *Flesch Kincaid Grade* level is a widely used readability formula which assesses the approximate reading grade level of a text. It is equivalent to the US grade level of education. For example, if a text has a Flesch Kincaid level of 8, this means the reader needs a grade 8 level of reading or above to understand it. The grade level is calculated with the following formula [21]:

$$0.39 \left(\frac{total\ words}{total\ sentences} \right) + 11.8 \left(\frac{total\ syllables}{total\ words} \right) - 15.59 \tag{3.2}$$

To obtain this feature, we made use of a function called *flesch_kincaid_grade* from *textstat*[7] which calculates statistical features from text.

Frequency of punctuation and repeated whitespaces. The frequency of the used punctuation and the repeated occurrence of white spaces per comment and per sentence were also measured. Punctuation is the system of symbols that is used to separate written sentences and parts of sentences, and to make their meaning clear. Some common punctuation marks are the period, comma, question mark, exclamation point, apostrophe, quotation mark and hyphen. The property *string.punctuation*[8] was used to obtain the all sets of punctuation.

[5] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.entropy.html`
[6] `https://www.scipy.org/`
[7] `https://github.com/shivam5992/textstat`
[8] `https://docs.python.org/3/library/string.html`

Content-based Features. We measured the average positivity and sensitivity per word and per sentence. Through this sentiment analysis, we can detect that how positively or negatively a user tends to write comments. For detecting the sentiment of German texts, a package, called *germansentiment*[9], was utilized. Since this package uses the Googles Bert architecture trained on 1,834 million samples, we can get the result of whether the text is positive or negative when putting texts. Further information about how this package works can be found in [17].

Idiosyncratic Features. The number of grammar mistakes within a sentence and a comment and the uppercase word usage per sentence and comment were counted. We used *LanguageTool*[10] to detect grammar errors and spelling mistakes. This library provides grammar checking for the text. Therefore, we are be able to count the number of grammar mistakes within a sentence and a comment.

Additional Features. Other than features that are mentioned above, Noun phrase, Named Entity Recognition, language detection, top 3 words used in a comment, ease-reading, and gunning fog were measured additionally.

Noun phrase is a phrase that has a noun as its head. It could also include other kinds of words, such as adjectives, ordinals, determiners. It is useful for understanding the context of each comment. The property *noun_phrases* from *TextBlob*[11] was used to extract noun phrases.

NER, Named Entity Recognition, is getting the entity and the label in order to identify and categorize the key information in texts. The library *spaCy*[12] was used to pick out the entities discussed in a text and classifying them into pre-defined categories such as "person", "organization", "location" and so on.

Language detection is literally finding out which language is used in a comment. We used *langdetect*[13] library and it is ported from Google's language-detection.

We found the most frequent 3 words from each comment that are being used by a user with the method *most_common* from *collections.Counter*[14]. This method returns a list of the n most common elements and their counts from the most common to the least. These top 3 words can be helpful for determining the writing behavior from a user.

Ease-reading is scoring how easily readable the comment is. The Flesch reading-ease gives a text a score between 1 and 100, with 100 being the highest readability score. In the Flesch reading-ease test, higher scores indicate material that is easier to read and lower numbers mark passages that are more difficult to read. The formula for the Flesch reading-ease score test is [31]:

$$206.835 - 1.015\left(\frac{total\ words}{total\ sentences}\right) + 84.6\left(\frac{total\ syllables}{total\ words}\right) \tag{3.3}$$

[9] https://github.com/oliverguhr/german-sentiment-lib
[10] https://github.com/jxmorris12/language_tool_python
[11] https://github.com/sloria/TextBlob
[12] https://spacy.io/
[13] https://github.com/Mimino666/langdetect
[14] https://docs.python.org/3/library/collections.html

The function *flesch_reading_ease* from *textstat*[15] was utilized to obtain values of the Flesch reading ease score.

Although two tests of the Flesch Kincaid Grade level and the Flesch reading-ease use the same core measures which are word length and sentence length, they have different weighting factors. The results of the Flesch Kincaid Grade level and the Flesch reading-ease correlate approximately inversely: a text with a comparatively high score on the Flesch reading-ease test should have a lower score on the Flesch Kincaid Grade level test.

Gunning fog index is a readability test estimating the years of formal education a person needs to understand the text on the first reading. It generates a grade level between 0 and 20. For instance, a fog index of 8 requires the reading level of eighth-graders. The higher fog index shows the more difficulty to read. The Gunning fog index is calculated with the following algorithm [15]:

1. Select a passage (such as one or more full paragraphs) of around 100 words and do not omit any sentences.
2. Determine the average sentence length by dividing the number of words by the number of sentences.
3. Count the "complex" words consisting of three or more syllables. Do not include proper nouns, familiar jargon, or compound words. Do not include common suffixes (such as -es, -ed, or -ing) as a syllable.
4. Add the average sentence length and the percentage of complex words.
5. Multiply the result by 0.4.

The complete formula is:

$$0.4\left[\left(\frac{words}{sentences}\right)+100\left(\frac{complex\ words}{words}\right)\right] \tag{3.4}$$

We used the function *gunning_fog* from *textstat* to acquire the fog index of the given text.

3.4 Doppelgänger Finder

3.4.1 Reduction of the number of features

To implement doppelgänger detection algorithm, we reduced the number of features by calculating PCA, Principal Component Analysis. PCA is a statistical procedure which extracts the most important features of the dataset and its goal is to reduce the number of features whilst keeping most of the original information. The main goal of using PCA is decreasing the training time while preserving as much information as possible. To obtain PCA value from feature values, the calculation of the covariance matrix, eighenvectors and eigenvalues are needed. We converted the feature values into matrix form and calculated the

[15] `https://github.com/shivam5992/textstat`

covariance matrix. The covariance matrix measures how much the features vary from the mean with respect to each other. The eigenvector with the highest eigenvalue is the most dominant principal component of the dataset.

Some of the features are not in the same shape which can yield an error while calculating PCA. For instance, some features have one value per comment or per word and other features have multiple values per comment. Moreover, some of features have only one value of all comments. Thus, we only selected the features having one value per comment for calculating PCA and they were 11 features (total words per comment, frequency of large words per comment, Simpson's D measure, Sichel's S measure, average sentence length per comment, frequency of used punctuation per comment, frequency of repeated occurrence of white space per comment, number of grammar mistakes per comment, uppercase word usage per comment, ease-reading per comment, gunning fog per comment). As a result of calculating covariance matrix, eigenvalue, eigenvector and PCA, the number of features is reduced from 11 to 7 or 8. The total reduced number of features is different depending on the number of comments since the number of comments affects the value of covariance matrix which is needed to calculate PCA.

3.4.2 Implementation of Doppelgänger Finder

There exist labels of users and the feature vectors which are computed by PCA as dataset. We separated the feature vector as a dependent variable and a label of users as an independent variable. Then, the variables of dependent and independent are split into the training set and the testing set by using train_test_split from scikit-learn[16]. Scikit-learn is a well-known library for machine learning in Python providing a lot of efficient tools for machine learning. After obtaining the sets of training and testing, we trained the classifier with the training set. Then, for each pair of users A and B, we calculated the probability of A's comment being attributed to B and the probability of B's comment being attributed to A with the testing set (we can call these probabilities as pairwise probabilities). To compute the pairwise probabilities, the classifier model of each user obtained from training the classifier was compared. And we combined pairwise probabilities into a single probability per pair of users.

We implemented three ways of combining the pairwise probabilities into a single value to provide a variety of options getting a combined probability and one of them can be chosen from the command line. These are average, multiplication, and squared average. The average probability is the mean of two probabilities, the multiplication probability is multiplying the pairwise probabilities, and the squared average probability is squaring each probability and getting the average value of squared values.

After obtaining the combined probability per pair of users, we obtained the threshold value from the command line and the threshold was compared to the combined probability. Two users, A and B are considered as doppelgängers if their combined probability is greater than the threshold. We labeled a set of doppelgängers as 1 and a pair of non-doppelgängers as 0.

[16] `https://scikit-learn.org/`

Figure 3.1 and Figure 3.2 show the results implemented by doppelgänger finder.

```
Please enter the name of author A from 'Valid Author List': too late
Please enter the name of author B from 'Valid Author List': no-panic
Please enter way to combine probabilities
('multiple' for multiplication, 'average' for average 'squared' for squared average): average
Pr(A->B):  0.010201298929745884 & Pr(B->A):  0.011051865174013742
Please enter the threshold (range: 0-1): 0.002
Combined probability is  0.010626582051879812
>>  too late  and  no-panic  are Doppelgängers
```

Figure 3.1: Result of a pair of doppelgängers.

```
Please enter the name of author A from 'Valid Author List': knowwhereman
Please enter the name of author B from 'Valid Author List': margherita
Please enter way to combine probabilities
('multiple' for multiplication, 'average' for average 'squared' for squared average): squared
Pr(A->B):  0.00977487418050556 & Pr(B->A):  0.008078271004759338
Please enter the threshold (range: 0-1): 0.02
Combined probability is  8.040331383552484e-05
>>  knowwhereman  and  margherita  are not Doppelgängers
```

Figure 3.2: Result of a pair of non-doppelgängers.

4 Evaluation

To test our doppelgänger detection algorithm, two experiments which are known number of doppelgängers and unknown number of doppelgängers were implemented. For known number of doppelgängers, we artificially split users into two sets of pseudonyms to construct known pairs of doppelgängers. Furthermore, for a more realistic scenario, we evaluated our doppelgänger detection algorithm when the number of doppelgängers is unknown.

4.1 Automated Threshold and Statistical Measures

Since manually selected threshold is not so precise, we automatically determined the optimal threshold by computing confusion matrix. A confusion matrix for binary classification shows four different combinations of predicted and actual values (True Negative, False Positive, False Negative, True Positive). Because f_1-score is used to determine the threshold, confusion matrix is calculated. To obtain f_1-score, values of precision and recall are needed and confusion matrix is used to compute precision and recall.

With the values from confusion matrix, the value of accuracy, precision, recall, and F_1-score are computed. Accuracy is a ratio of correctly predicted observations to the total observations. It measures how often the algorithm classifies a data point correctly. Because it is one single measure used to summarize performance of the classifier, accuracy is widely used for evaluating the algorithm. The result of precision indicates a value between 0 for no accuracy and 1 for the perfect accuracy. Precision is implied as the measure of the correctly identified positive cases from all the predicted positive cases. It is a measure of the reproducibility of a set of measurements. The result of precision indicates a value between 0 for no precision and 1 for the perfect precision. Recall is the ratio of the data in which the predicted and the actual value matched positive among the the actual values which are positive. The result of recall shows a value between 0 for no recall and 1 for the perfect recall. F_1-score is the harmonic mean of precision and recall. The range of F_1-score is from 0 to 1 and it tells how accurate the classifier is. F_1-score is considered perfect when it becomes 1.

The formulas for computing accuracy, precision, recall, and F_1-score are [32]:

$$\begin{aligned} Accuracy &= \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative} \\ Precision &= \frac{True\ Positive}{True\ Positive + False\ Positive} \\ Recall &= \frac{True\ Positive}{True\ Positive + False\ Negative} \\ F_1 - score &= \frac{2 \times Precision \times Recall}{Precision + Recall} \end{aligned} \tag{4.1}$$

To obtain the optimal threshold, all possible thresholds between 0 and 1 with a step size of 0.001 were tested. We applied each possible threshold of all possible thresholds to a combined probability from the pairwise probabilities per pair of users and mapped values to 1 if the probability is equal to or greater than the threshold and to 0 if the probability is less than the threshold. Then, we made an array which contains F_1-scores with the mapped values and the actual labels of whether a pair of users is doppelgänger or not (1 if they are doppelgängers, 0 if they are non-doppelgängers). A value from the array whose index has the largest F_1-score is an optimal threshold.

4.2 Known Number of Doppelgängers

The initial evaluation of our doppelgänger detection algorithm was implemented when the number of doppelgängers is known. For establishing known pairs of doppelgängers, we artificially split a user into two pseudonyms. Comments written by a user were divided randomly into two sets of pseudonyms. We can assume two pseudonyms are from one user and they are doppelgängers.

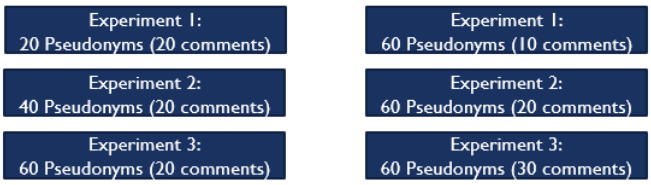


Figure 4.1: An Overview of executed experiments.

We implemented six experiments in total to evaluate our doppelgänger detection algorithm. As you can see in Figure 4.1, three experiments were implemented according to the number of pseudonyms and other experiments were conducted with varying numbers of comments per pseudonym. For number of pseudonyms, we used three different numbers of pseudonyms

with the same number of comments per pseudonym such as 20, 40, and 60 pseudonyms with 20 comments. And for the number of comments per pseudonym, the same number of pseudonyms with different numbers of comments such as 60 pseudonyms with 10, 20, and 30 comments were conducted. Only comments consisting of at least 250 characters were used for experiments.

Figure 4.2 shows the process overview of assessing the doppelgänger detection algorithm with known number of doppelgängers.

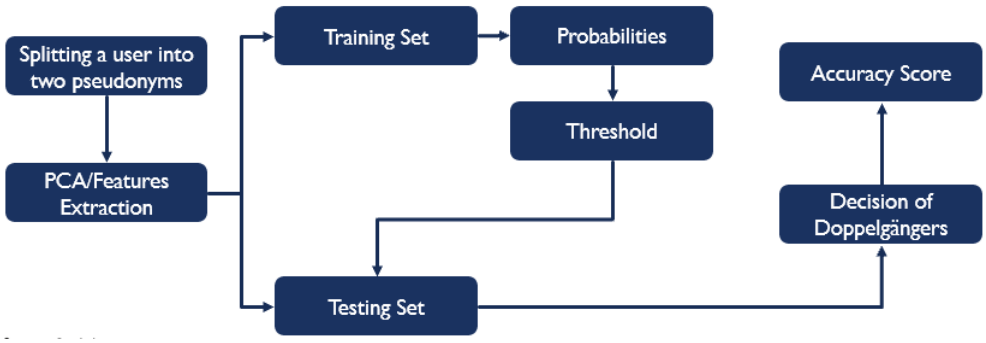


Figure 4.2: Process of evaluation with known number of doppelgängers.

First, we split each user into two pseudonyms with comments written by the user. The names of users were encoded and we used the encoded values as labels in a range of 0 and (*number of users - 1*) and we artificially added 100 for pseudonym 1 and 200 for pseudonym 2 to the labels of users since we experimented less than 100 users. Therefore, we could distinguish which pseudonyms are from a same user by looking at the labels of pseudonyms (e.g., labels of pseudonyms, 101 and 201 are from a same user so that they are doppelgängers). Then, we computed PCA values from feature values that were generated from the comments. Dataset which are feature vectors and labels of pseudonyms is divided into training set and testing set using cross-validation with *cross_val_score*[1] from *sklearn.model_selection*[2] module.

Cross-validation is used to access the effectiveness of the classifier model. Three folds are conducted in our case. K-fold cross-validation is where a given dataset is split into K number of folds. We used *cross_val_score* from *sklearn.model_selection* module for splitting dataset into three folds. As shown in Figure 4.3, in the first iteration, the first fold is used to test the model and the rest folds are used to train the classifier. In the second iteration, the second fold is used as the testing set while the rest folds serve as the training set. The same process is also conducted in the third iteration. In each fold, $\frac{1}{3}$ fold is used for testing the model and $\frac{2}{3}$ folds are used to train the classifier.

We computed the pairwise probabilities of each pair of pseudonyms with the training set and combine them into a single probability per pair of pseudonyms. Then, we applied the

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html`
[2] `https://scikit-learn.org/stable/modules/classes.html#module-sklearn.model_selection`

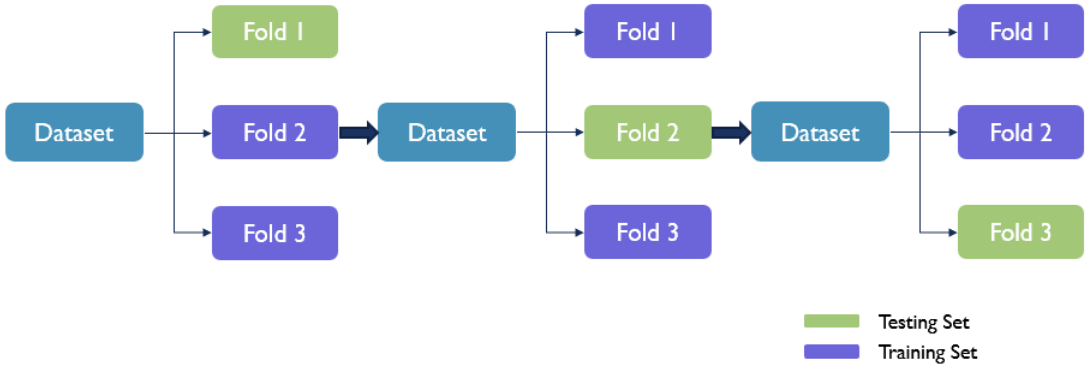


Figure 4.3: K-Fold Cross-Validation.

threshold which is automatically defined (cf. Section 4.1) to the testing set. The decision whether each two pseudonyms are doppelgängers or not can be done by comparing the threshold and the combined probabilities. With the predicted value of doppelgänger detection which is obtained from the previous process and the actual value, we can acquire the accuracy score of the doppelgänger detection algorithm.

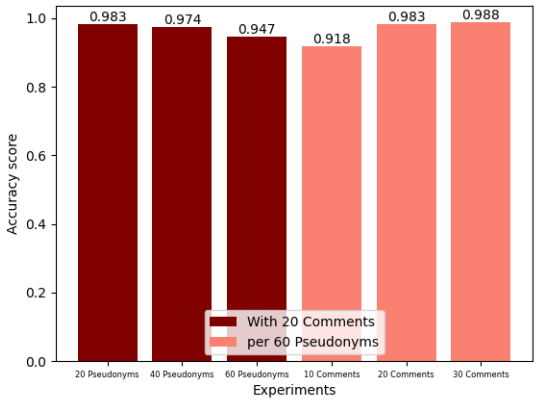


Figure 4.4: Accuracy score of experiments.

Figure 4.4 shows the accuracy score of each experiment. When conducting the experiments of 20, 40, 60 pseudonyms with 20 comments per pseudonym, we acquired accuracy scores, 0.983, 0.974, 0.974, respectively. The accuracy score declined when using more number of pseudonyms. When experimenting 10, 20, 30 comments per pseudonym from 60 pseudonyms, the accuracy scores were 0.918, 0.983, 0.988, respectively. The accuracy score rose when the number of comments increase. The highest accuracy score was when experimenting 30 comments per pseudonym from 60 pseudonyms which is the largest amount of data among the experiments executed.

To sum up, we could get to know an increasing number of pseudonyms decreases the accuracy

score while the number of comments and the accuracy score is inversely proportional. It means that the classification can be more accurate when classifying less number of independent variables.

4.2.1 Comparison with Baseline

To compare the performance of our doppelgänger detection algorithm with a distance-based method which is supervised, we used the euclidean distance method. Euclidean distance is used to find the nearest distance between two points on a real line. Since we computed the euclidean distance between literary features of any two users, we utilized the formula of calculating the euclidean distance between two one-dimension arrays. The distance between them can be computed by [11]:

$$\begin{gathered} ||u - v||_2 \\ \left(\sum (w_i |\, (u_i - v_i)\, |^2)\right)^{\frac{1}{2}} \end{gathered} \tag{4.2}$$

n, v : input arrays
w: the weights for each value in u and v

scipy.spatial.distance.euclidean[3] was used to compute the euclidean distance between feature values from two users. To implement the supervised distance-based approach, the dataset which is literary features per user is split into the training and the testing sets by using *train_test_split*[4] function from *sklearn.model_selection*[5]. We trained the classifier using the euclidean distance between two users each in the training set and the approach was tested using the euclidean distance between two users in the testing set. We acquired a threshold from the command line and compare the threshold with the euclidean distance. Two users are considered as a pair doppelgängers if the distance between them is less than a threshold.

For experiments, we used the number of users as 20, 40, 60 with 20 comments per user and the number of comments as 10, 20, 30 per user from 60 users. Figure 4.5 shows the comparison of the classification results obatined by our doppelgänger detection algorithm and a distance-based method which is euclidean distance. We set an average value of the minimum distance and the maximum distance of each experiment as a threshold.

The accuracy score obtained by doppelgänger detection algorithm decreased when experimenting more number of users. On the other hand, not all the accuracy scores obtained by a distance-based method reduced when conducting experiments with an increasing number of users. With a distance-based method, the accuracy score when experimenting 60 users with 20 comments per user was higher than when experimenting 40 users with 20 comments per

[3] https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.euclidean.html
[4] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
[5] https://scikit-learn.org/stable/modules/classes.html#module-sklearn.model_selection

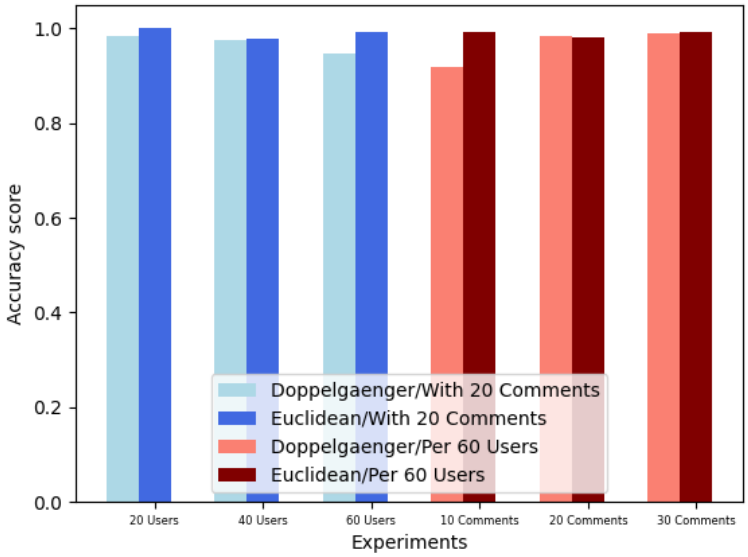


Figure 4.5: Comparison of accuracy scores obtained by our doppelgänger detection algorithm and a distance-based method (Euclidean distance).

user. When experiments with different numbers of comments were implemented, the accuracy score acquired by doppelgänger detection algorithm increased as the number of comments increased. However, the same occurrence did not happen to the accuracy score obtained by a distance-based method. When a distance-based method was used, the accuracy score for the experiment applying 10 comments per user from 60 users was slightly lower than the accuracy score of experimenting 20 comments per user from 60 users.

Both the number of users and the number of comments affected the classification results obtained by our doppelgänger detection algorithm and also a distance-based method which is euclidean distance. However, the results obtained by doppelgänger detection algorithm were gradually influenced by the number of users and comments while the results acquired by a distance-based method were not.

4.3 Unknown Number of Doppelgängers

To evaluate our doppelgänger detection algorithm in a more realistic scenario, we set the number of doppelgängers as unknown. For all experiments, three-fold cross-validation was applied.

As Figure 4.6 indicates, four different settings that contain different quotas of doppelgängers are considered to generate a situation which we do not have any knowledge about the number of doppelgängers: *(i) None*: the dataset contains no simulated doppelgänger, *(ii) Single*:

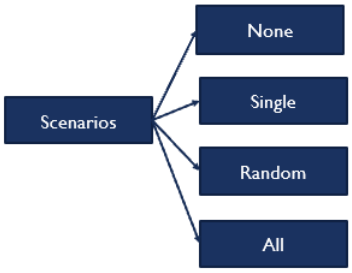


Figure 4.6: Scenarios.

the dataset contains exactly one doppelgänger pair, *(iii) Random*: the dataset contains between 25% and 75% doppelgängers, and *(iv) All*: each user in the dataset has exactly one doppelgänger. For each scenario, two experiments were executed in which the number of pseudonyms is 50 and 100 with 20 comments from each pseudonym. We made a use of comments whose minimum text length per comment is 750 characters.

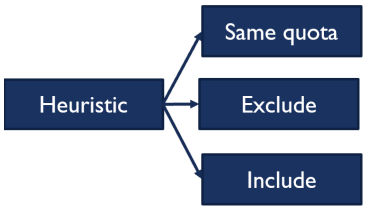


Figure 4.7: Heuristics.

Moreover, we implemented three different heuristics as shown in Figure 4.7 to identify the threshold: *(i) Quota Same*: determining the threshold applied to testing fold, whereas the quota of doppelgängers in the training and the testing sets is the same, *(ii) Quota Exclude*: determining the threshold applied to testing fold, whereas for training folds we consider scenarios with varying number of doppelgängers and exclude the quota of doppelgängers of the testing fold, *(iii) Quota Include*: determining the threshold applied to testing fold, whereas for training folds we consider scenarios with varying number of doppelgängers and include the quota of doppelgängers of the testing fold.

The process of assessing the doppelgänger detection algorithm with unknown of doppelgängers as in Figure 4.8 is similar to the way of evaluating the doppelgänger detection algorithm with known number of doppelgängers. First, the dataset is split into training and testing set using three-fold cross validation (cf. Section 4.2). We obtained pairwise probabilities by comparing the classifier model of each pseudonym obtained from training the classifier with the training set. Once obtaining probabilities per pair of pseudonyms, the probabilities are combined into a single probability which is an average value of pairwise probabilities. Then, F_1-score is computed with a combined probability and the actual label of doppelgänger detection which is 1 (a pair of doppelgängers) or 0 (a pair of non-doppelgängers) to obtain the optimal threshold. The computed threshold is then applied to the testing set.

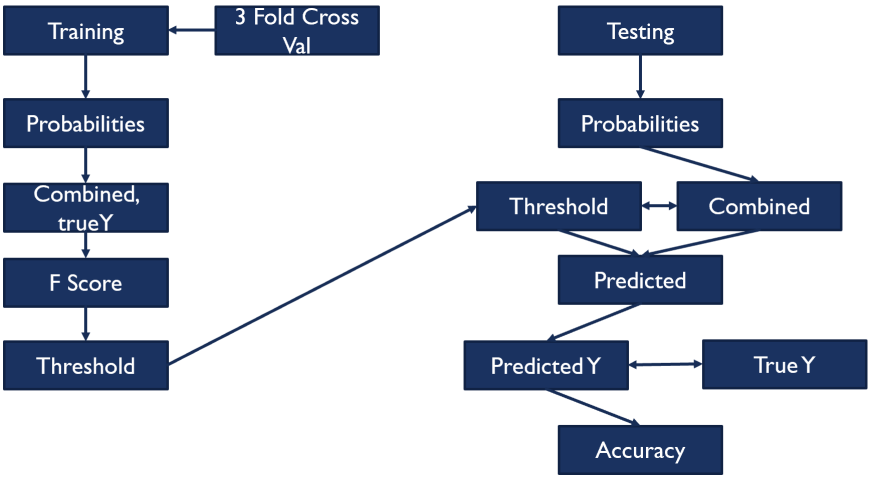


Figure 4.8: Process of evaluation with unknown number of doppelgängers.

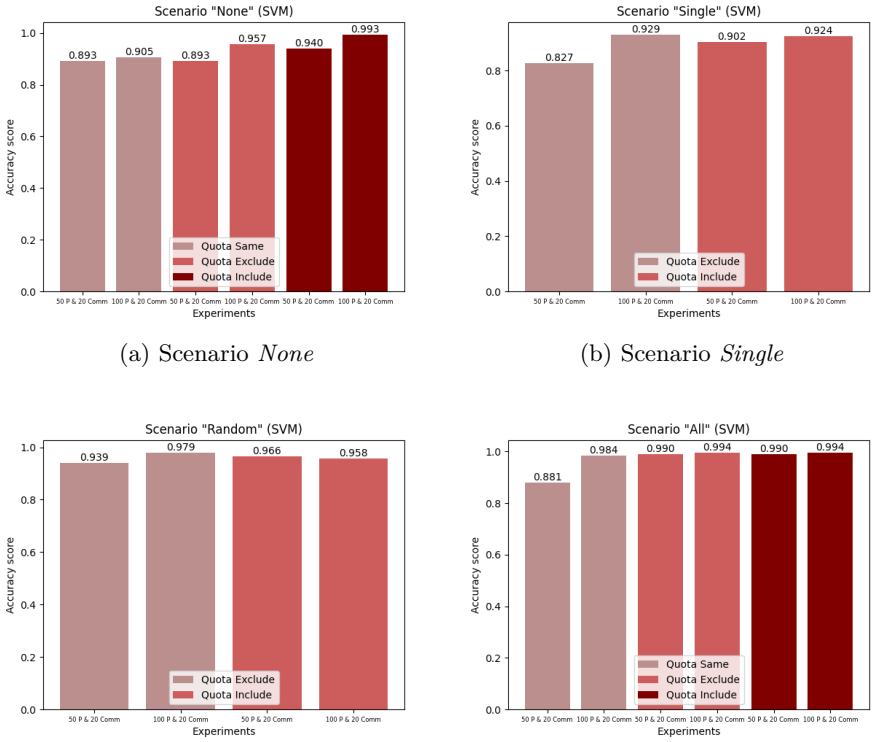
It is compared with the combined probability from testing dataset and the predicted label of doppelgänger detection is obtained. With the actual label of doppelgänger detection and the predicted label, we can get the accuracy score of classifier which is being used in the doppelgänger detection algorithm.

We used three different classifiers, k-NN (k-Nearest Neighbors), Random forests and SVM (Support Vector Machines) for evaluating our approach and Figure 4.9 shows the accuracy score of SVM in each scenario.

In scenario *None*, the accuracy score of each experiment in all heuristics increased when experimenting more number of pseudonyms. The highest accuracy score (0.993) was when the heuristic *Quota Include* was conducted in experiments with 20 comments per pseudonym from 100 pseudonyms. 0.893 was the lowest accuracy score when experimenting 50 pseudonyms with 20 comments per pseudonym in the heuristics *Quota Same* and *Quota Exclude*. The accuracy score was higher when the heuristic *Quota Exclude* was implemented in experiments than the heuristic *Quota Same* applied to experiments.

In scenario *Single*, the heuristics *Quota Exclude* and *Quota Include* were implemented without the heuristic *Quota Same* because we have one pair of doppelgängers in scenario *Single* and the quota of doppelgängers can not be set as same in the training and the testing sets. Similar to the result of scenario *None*, the accuracy score of each experiment in all heuristics increased when implementing experiments with more number of pseudonyms. The highest accuracy score was 0.929 and it was obtained when the heuristic *Quota Exclude* was experimented with 100 pseudonyms with 20 comments per pseudonym and the lowest accuracy score was 0.827 when implementing the heuristic *Quota Exclude* with 50 pseudonyms with 20 comments per pseudonym. The gap between two experiments with 20 comments per pseudonym from 50 pseudonyms and 100 pseudonyms in the heuristic *Quota Exclude* was wider than in the heuristic *Quota Include*.

In scenario *Random*, similar to the scenario *Single*, the heuristic *Quota Same* was not



(a) Scenario *None*

(b) Scenario *Single*

(c) Scenario *Random*

(d) Scenario *All*

Figure 4.9: Accuracy score of SVM in each scenario.

implemented since we can not set the quota of doppelgängers in the training set and the testing sets as same when the quota of doppelgängers is random. Differently from the result of scenario *None* and *Single*, all accuracy scores of the experiments did not increase gradually when experimenting more number of pseudonyms. In the heuristic *Quota Include*, the accuracy score which is 0.966 of a experiment with 20 comments per pseudonym from 50 pseudonyms was lower than the accuracy score of the experiment with 20 comments per pseudonym from 100 pseudonyms. The highest and lowest accuracy scores were 0.979, 0.939, respectively and they were happened when experimenting the heuristic *Quota Exclude*.

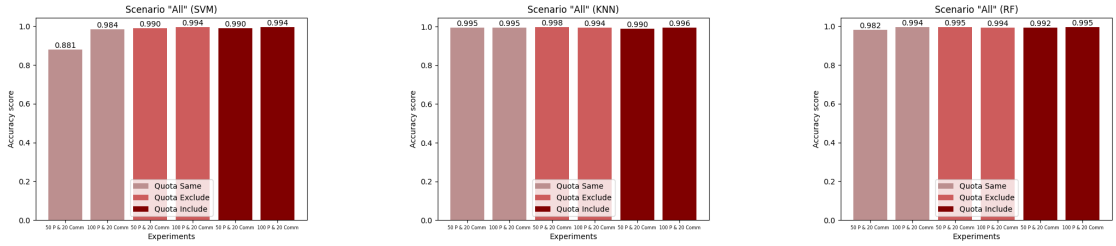
In scenario *All*, the accuracy score of each experiment in all heuristics rose when implementing the experiments with more number of pseudonyms. The lowest accuracy scores which are 0.881 was when the heuristic *Quota Same* was implemented. The accuracy scores which are 0.99, 0.994 when experimenting in which the number of pseudonyms is 50, 100, respectively with 20 comments per pseudonym were same when the heuristics *Quota Exclude* and *Quota Include* were implemented in the experiments.

Overall, the number of pseudonyms influenced the accuracy score in all heuristics. The highest accuracy score was 0.994 in the scenario *All* when the heuristics *Quota Exclude* and *Quota Include* were implemented in experiments with 20 comments per pseudonym from 100 pseudonyms. The lowest accuracy score was 0.827 in the scenario *Single* when the

experiment applied to the heuristic *Quota Exclude* with 20 comments per pseudonym from 50 pseudonyms.

4.3.1 Doppelgänger Finder with Multiple Machine Learning Techniques

To extend our doppelgänger detection algorithm, we conducted experiments using three different machine learning techniques which are k-NN (k-Nearest Neighbors), Random forests and SVM (Support Vector Machines).



(a) SVM (b) k-NN (c) Random forests

Figure 4.10: Accuracy score of classifiers in scenario *All*.

For each classifier, all scenarios were experimented with each heuristic applied and we implemented two experiments of each heuristic in which the number of pseudonyms is 50 and 100 with 20 comments per pseudonym. The minimum text length per comment is 750 characters. Figure 4.10 represents the accuracy score of three different classifiers which are SVM, k-NN, Random forests used in the scenario *All*, where each user in the dataset has exactly one set of doppelgängers.

In the result of SVM, the accuracy score was higher when the number of pseudonyms increased. The accuracy scores when the heuristics *Quota Exclude* and *Quota Include* were applied to the experiments of 50, 100 pseudonyms with 20 comments per pseudonym were the same, 0.99, 0.994, respectively. The lowest accuracy score was 0.881 when the heuristic *Quota Same* was implemented in the experiment of 50 pseudonyms with 20 comments per pseudonym. The highest accuracy score was 0.994 when the heuristics *Quota Exclude* and *Quota Include* were implemented in the experiment of 100 pseudonyms with 20 comments per pseudonym.

In the result of k-NN, the accuracy scores of both experiments when the heuristic *Quota Same* was implemented were same at 0.995. When the heuristic *Quota Exclude* was applied to the experiments, the accuracy score declined from 0.998 to 0.994 as the number of pseudonyms rose. Conversely, the accuracy score increased from 0.99 to 0.996 when more number of pseudonyms were used. The lowest accuracy score was 0.99 when the heuristic *Quota Include* was implemented in the experiment of 50 pseudonyms with 20 comments per pseudonym. The highest accuracy score was 0.998 when the heuristic *Quota Exclude* was experimented

with 20 comments per pseudonym from 50 pseudonyms. All accuracy scores were more than 0.99.

In the result of Random forests, the accuracy score was higher as the number of pseudonyms increased when the heuristics *Quota Same* and *Quota Include* were implemented. When the heuristic *Quota Exclude* was implemented, the accuracy score slightly declined from 0.995 to 0.994 when more number of pseudonyms was used. 0.982 was the lowest accuracy score when the heuristic *Quota Same* was implemented in the experiment of 50 pseudonyms with 20 comments per pseudonym. The highest accuracy score was 0.995 when the heuristic *Quota Exclude* was applied to the experiment of 50 pseudonyms with 20 comments per pseudonym and when the heuristic *Quota Include* was implemented in the experiment of 100 pseudonyms with 20 comments per pseudonym.

Overall, the lowest accuracy score was 0.881 when SVM and the heuristic *Quota Same* was applied to the experiment of 50 pseudonyms with 20 comments per pseudonym. The highest accuracy score which is 0.998 was happened when k-NN was used as a classifier and the heuristic *Quota Exclude* was implemented in the experiment of 50 pseudonyms with 20 comments per pseudonym. When experimenting the scenario *All*, among three classifiers, k-NN provided the highest accuracy score on average which is 0.994 and SVM got the lowest accuracy score on average which is 0.972 containing the lowest accuracy score (0.881). The average accuracy score when Random forests was applied to the experiments was 0.992 and it was close to the the average accuracy score of k-NN. The accuracy score of classifiers was various depending on the scenarios.

4.3.2 Complexity Analysis

The next evaluation part is to compute the complexity analysis of doppelgänger detection algorithm such as the running time and the memory usage of the algorithm for feature extraction as well as the execution of doppelgänger algorithm for each classifiers (SVM, k-NN and Random Forest). This evaluation part is important in order to obtain the results of which classifier is having the lowest running time and the memory usage. In this evaluation part, there are two scenarios for computing the complexity analysis they are :

1. The experiment which consist of 20, 50, 70, and 100 pseudonyms and 20 comments for each experiment.
2. The experiment which consist of 5, 10, 15, 20, and 25 comments and 100 pseudonyms for each experiment.

As we can see in figure 4.11 the running time for feature extractions of each experiment were increased by putting more number of pseudonyms and the running time reached its peak in approx. 6000 seconds (1.5 hour) for the first scenario and 7000 seconds (below 2 hours) for the second scenario. The running time of SVM gradually increased by putting the more number of comments and pseudonyms for each heuristic. However, the result for k-NN and Random forest seem satisfactory since the running time were below 100 seconds for each experiment. Due to an iteration process to obtain the feature vectors, the doppelgänger algorithm took more time than other experiment.

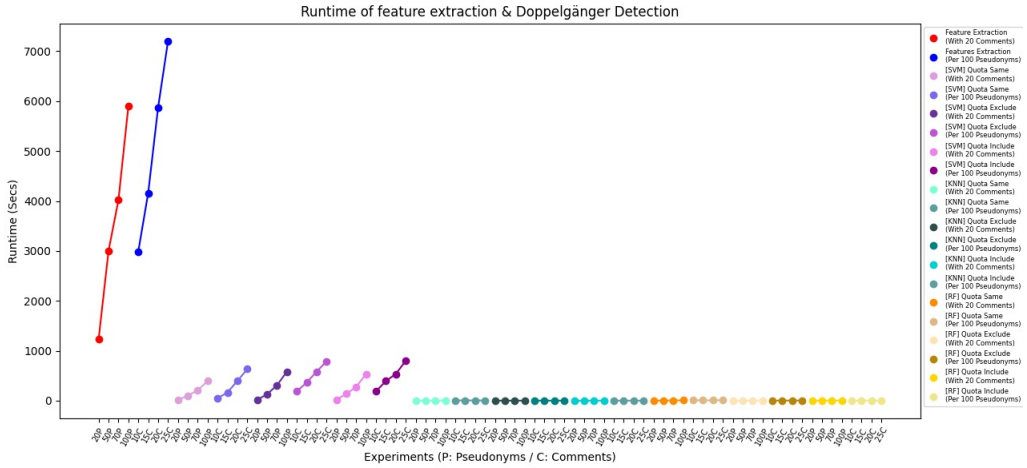


Figure 4.11: Runtime of each process in doppelgänger detection algorithm

The figure 4.12 shows the memory usage of doppelgänger algorithm detection for each experiments. The number of memory that were utilized for feature extraction rapidly increased from around 350 MB to 700 MB. SVM utilized the highest memory than k-NN and Random Forest. SVM reached it peak in approximately 700 MB while k-NN and Random Forest were remain steady in below 100 MB. Due to the disadvantage and incapability of SVM to handle large datasets, SVM utilized more memory than other classifiers.

In this experiment we obtained the results of complexity analysis and can conclude that SVM used longer running time and higher memory usage than other classifiers such as k-NN and Random Forest. It can be seen that the number of pseudonyms, comments and heuristic have influenced the running time and memory usage.

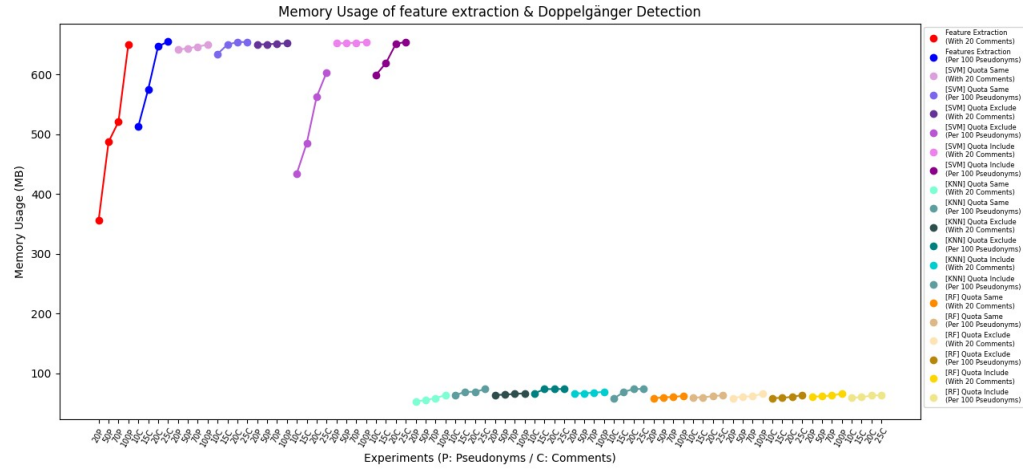


Figure 4.12: Memory usage of each process in doppelgänger detection algorithm.

5 Conclusion

Social media forums are used by most people to obtain and exchange information. However, social media can also be misused by some people to do bad things such as spreading the fake information and opinion manipulations. Most of the misleading information was carried out by fake account or duplicated account (doppelgänger account). Some of the researchers have done research to thwart the growth of fake account and doppelgänger account by using different approaches. Therefore, in this study project, we also tried to detect the doppelgänger which are widely spread in the online world. We have implemented an algorithm to detect doppelgänger on online news website, which is *ZEIT ONLINE*. The doppelgänger algorithm works by observing and analyzing the writing styles in the comments that are written by the *ZEIT ONLINE*'s users. The doppelgänger algorithm used features that can help to analyze the writing styles such as *Sichel's S measure*, vocabulary richness, word-level features, idiosyncratic features, and other features that can help to detect doppelgänger. In order to obtain the best result and accuracy to detect the doppelgänger, we have done various experiments such as using different numbers of pseudonyms , i.e., 20, 40, 60, 100 pseudonyms and using different numbers of comments, i.e., 5, 10, 20, 25 comments. During the experiments, the minimum number of characters per comment was increased from 250 to 750 characters per comment. From the experiments, we can conclude that the number of comments have influenced the accuracy score. Experiments were also done by applying automated threshold for each experiment. The machine learning algorithms that we used in doppelgänger detection algorithm are SVM, k-NN and Random Forest. The results from the experiments showed that Random Forests obtained the higher accuracy score and took lesser memory usage and runtime process than SVM and k-NN. This project still needs to be improved, thus, for future work, we suggest to do more experiments to obtain the best threshold and increase the accuracy score to detect the Doppelgänger using stylometric features.

Abbreviations and Acronyms

HTML	Hypertext Markup Language
URL	Uniform Resource Locator
CSV	Comma-Separated Values
NLTK	Natural Language Toolkit
NLP	Natural Language Processing
NER	Named Entity Recognition
PCA	Principal Component Analysis
SVM	Support Vector Machines
k-NN	k-Nearest Neighbors
VPN	Virtual Private Network
PoS	Parts of Speech

Bibliography

[1] S. Afroz, M. Brennan and R. Greenstadt. *Detecting hoaxes, frauds, and deception in writing style online*. In: *2012 IEEE Symposium on Security and Privacy*. IEEE. 2012, pp. 461–475.

[2] S. Afroz, A. C. Islam et al. *Doppelgänger finder: Taking stylometry to the underground*. In: *2014 IEEE Symposium on Security and Privacy*. IEEE. 2014, pp. 212–226.

[3] H. Ahmed, I. Traore and S. Saad. *Detection of online fake news using n-gram analysis and machine learning techniques*. In: *International conference on intelligent, secure, and dependable systems in distributed and cloud environments*. Springer. 2017, pp. 127–138.

[4] H. Baayen. *Statistical models for word frequency distributions: A linguistic evaluation*. In: *Computers and the Humanities* 26.5 (1992), pp. 347–363.

[5] M. W. Berry, A. Mohamed and B. W. Yap. *Supervised and unsupervised learning for data science*. Springer, 2019.

[6] G. Biau. *Analysis of a random forests model*. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 1063–1095.

[7] F Bibi, Z Ali et al. *Measurement of diversity indices of avian communities at Taunsa Barrage Wildlife Sanctuary, Pakistan*. In: *The Journal of Animal & Plant Sciences* 23.2 (2013), pp. 469–474.

[8] K. Biermann and A. Ginzel. *War without blood*. `https://www.zeit.de/digital/internet/2017-02/bundestag-elections-fake-news-manipulation-russia-hacker-cyberwar/seite-5?utm_referrer=https%3A%2F%2Fwww.google.com%2F`. 2017.

[9] J. Burns. *How Many Social Media Users Are Real People?* `https://gizmodo.com/how-many-social-media-users-are-real-people-1826447042`. 2018.

[10] Y. Chen, Y. Zhou et al. *Detecting offensive language in social media to protect adolescent online safety*. In: *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*. IEEE. 2012, pp. 71–80.

[11] T. S. community. *scipy.spatial.distance.euclidean - SciPy v1.6.1 Reference Guide*. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.euclidean.html`. 2021.

[12] M. Conti, R. Poovendran and M. Secchiero. *Fakebook: Detecting fake profiles in on-line social networks*. In: *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE. 2012, pp. 1071–1078.

[13] U. Demšar, P. Harris et al. *Principal component analysis on spatial data: an overview.* In: *Annals of the Association of American Geographers* 103.1 (2013), pp. 106–128.

[14] Facebook. *Community Standards Enforcement Report.* `https://transparency.facebook.com/community-standards-enforcement`. 2020.

[15] R. Formulas. *The Gunning's Fog Index (or FOG) readability formula.* In: ().

[16] S. Girgis, E. Amer and M. Gadallah. *Deep learning algorithms for detecting fake news in online text.* In: *2018 13th International Conference on Computer Engineering and Systems (ICCES).* IEEE. 2018, pp. 93–97.

[17] O. Guhr, A.-K. Schumann et al. *Training a Broad-Coverage German Sentiment Classification Model for Dialog Systems.* In: *Proceedings of The 12th Language Resources and Evaluation Conference.* Marseille, France: European Language Resources Association, 2020, pp. 1620–1625.

[18] J. Heiny and T. Mikosch. *Eigenvalues and eigenvectors of heavy-tailed sample covariance matrices with general growth rates: The iid case.* In: *Stochastic Processes and their Applications* 127.7 (2017), pp. 2179–2207.

[19] L. Jin, H. Takabi and J. B. Joshi. *Towards active detection of identity clone attacks on online social networks.* In: *Proceedings of the first ACM conference on Data and application security and privacy.* 2011, pp. 27–38.

[20] D. Kagan, Y. Elovichi and M. Fire. *Generic anomalous vertices detection utilizing a link prediction algorithm.* In: *Social Network Analysis and Mining* 8.1 (2018), pp. 1–13.

[21] J. P. Kincaid, R. P. Fishburne Jr et al. *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.* Tech. rep. Naval Technical Training Command Millington TN Research Branch, 1975.

[22] H. Li, Z. Chen et al. *Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns.* In: *Proceedings of the International AAAI Conference on Web and Social Media.* Vol. 9. 1. 2015.

[23] T. Mihaylov, G. Georgiev and P. Nakov. *Finding opinion manipulation trolls in news community forums.* In: *Proceedings of the nineteenth conference on computational natural language learning.* 2015, pp. 310–314.

[24] T. Mihaylov, T. Mihaylova et al. *The dark side of news community forums: Opinion manipulation trolls.* In: *Internet Research* (2018).

[25] T. Mihaylov and P. Nakov. *Hunting for troll comments in news community forums.* In: *arXiv preprint arXiv:1911.08113* (2019).

[26] A. Mohamed. *Comparative Study of Four Supervised Machine Learning Techniques for Classification.* 2017.

[27] M. Mohammadrezaei, M. E. Shiri and A. M. Rahmani. *Identifying fake accounts on social networks based on graph analysis and classification algorithms.* In: *Security and Communication Networks* 2018 (2018).

[28] J. P. Mueller and L. Massaron. *Machine learning for dummies*. John Wiley & Sons, 2021.

[29] Z. Online. *Facebook sperrte seit Jahresbeginn Milliarden Fake-Profile*. `https://www.zeit.de/digital/2019-11/gefaelschte-accounts-facebook-fake-profile-anstieg-sperrung`.

[30] H. Ramnial, S. Panchoo and S. Pudaruth. *Authorship attribution using stylometry and machine learning techniques*. In: *Intelligent Systems Technologies and Applications*. Springer, 2016, pp. 113–125.

[31] F. Rudolf. *How to write plain english*. In: *University of Canterbury* (2016).

[32] T. Saito and M. Rehmsmeier. *The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets*. In: *PloS one* 10.3 (2015), e0118432.

[33] Z. Shan, H. Cao et al. *Enhancing and identifying cloning attacks in online social networks*. In: *Proceedings of the 7th international conference on ubiquitous information management and communication*. 2013, pp. 1–6.

[34] J. Shlens. *A tutorial on principal component analysis*. In: *arXiv preprint arXiv:1404.1100* (2014).

[35] H. Tankovska. *Most popular social networks worldwide as of January 2021, ranked by number of active users*. `https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/`. 2018.

[36] Wikipedia. *Doppelgänger*. `https://en.wikipedia.org/wiki/Doppelg%C3%A4nger#:~:text=listen)%2C%20literally%20%22double%2D,a%20harbinger%20of%20bad%20luck.`.