# Towards enhancing cyber-hate speech detection: An experimental investigation on text vectorization techniques

**Mullah, Nanlir Sallau & Dashe, Miapmuk Obadiah**

Federal College of Education Pankshin, PMB1027 Pankshin, Plateau State, Nigeria.
Email: mullakns@gmail.com, Mobile Number: 08037607681
Mobile Number: 08039453140.

## Abstract

Text analysis is one of the important application domains for machine learning algorithms (MLAs). Notwithstanding, text data cannot be fed directly to the MLAs because algorithms use numerical feature vectors having fixed size lengths. To address this problem, advances in natural language processing have developed many techniques for converting text data to a machine-understandable format. The text-to-number conversion is called the text feature extraction (TFE).TFE has played a significant role in improving text classification tasks such as hate speech detection applications. However, despite the overwhelming importance of feature extraction, identifying the optimal TFE technique for hate speech detection on social media (SM) data remain a challenge to researchers. Therefore, this study conducted experiments on SM data to determine the most suitable TFE technique(s) to be used for cyber-hate detection. The researchers identified four widely used TFE techniques (Count Vectorization, TF-IDF, Word n-gram and Character n-gram) and five classifiers for hate speech detection. The researchers used Kaggle's publicly available dataset to conduct the experiments. The results of the experiments show that count vectorization and TF-IDF are more promising in improving classifiers' performances on the SM dataset. The best performance was seen on SVM with an F1-score of 82% under the count vectorization. TF-IDF also improve the performances of the classifiers with the best improvement on RF with an F1-score of 78%.This research concluded based on the experiment that F1-score is the best evaluation metric for a dataset with an imbalanced class distribution. This study will serve as a good reference document for both new and old researchers in the field of text mining and opinion mining to be specific.

**Keywords:** Machine learning, text vectorization**,** Text feature extraction, Cyber-hate speech, Feature vectors space, Natural language processing

**Introduction**

Cyber-hate speech has been on the rise in the last few decades (Munn, 2020). Detection of cyber-hate and offensive comments on social media (SM)using natural language processing (NLP) and machine learning (ML) is a well-known research area (Modha et al., 2020). All offensive content identification is first modelled asa text categorisation task. Text categorisation has been researched extensively in the domain of text mining because of its significant roles in many text-related applications such as sentiment analysis (SA), claims investigations, fake news detection, hate speech detection and spam filtering systems among others(Arcila-Calderón et al., 2021; Kalsnes&Ihlebæk, 2021)

The need for feature extraction becomes imminent as text cannot be processed directly by statistical learning algorithms on the computer system because these algorithms depend on statistical principles. Feature extraction is a critical feature engineering stage in the text classification pipeline in general. All the processes involved in making text-based data usable (machine-understandable format) by the learning algorithmsare called feature engineering (Kulkarni &Shivananda, 2019). The performance accuracy of any MLA (shallow or deep learning) depends significantly on the type of feature engineering technique applied. These feature engineering techniques include text pre-processing and feature extraction techniques. Among these feature engineering techniques, feature extraction is the most important.

Feature extraction generally will improve the detection accuracy of the machine learning algorithm (MLA) and also shorten the learning time of the algorithm (Plaza-del-Arco et al., 2021). As researchers quest for the improvement of detection accuracy as ever before, the feature extraction stage must be treated with utmost sincerity. At this phase of the classification pipeline, an n-dimensional feature vector is generated and serves as input to the algorithm to process. The significance of feature extraction in data science, especially text categorization, cannot be over-emphasized. This is a stage where texts are converted to numerical data and mapped into decision vector space. The optimality of this phase is directly proportional to the final output of the model (Liang et al., 2017).

Different TFE techniques have been proposed in the literature which can be used with traditional MLAsfor text classification problems. Determining which TFE technique can

improve the performance of the classifier is a common problem among researchers. Most researchers nowadays use SM datasets for hate speech detection simulation experiments, which generally share common features.The most important among the features shared by the SM dataset is the shortness of the text, which suffers from sparsity problems in the vector space. Therefore, it is important to conduct experiments to recommend which TFE technique is likely to improve hate speech detection accuracy when the SM dataset is to be used. To address this problem, the researchers used the Kaggle dataset to test widely used TFE techniques for cyber-hate speech detection in the literature.

Many TFE methods are available for use,whichinclude BoW, TF-IDF, n-grams, one-hot encoding, Count Vectorization, co-occurrence matrix, and hash vectorization. Among the TFE techniques, these researchersdentify the most widely applied.Researchers need to know which one is suitable and most likely to improvethe model's performancefor cyber-hate detection on SM data. Considering these facts and the need to improve the model efficiency, it becomes vital to examine the many possible approaches for TFE for hate speech detection.

This study is guided by the following research questions (R.Qs) in anattempt to address the research's main objective:

**R.Q.1:** What are the TFE techniques and classifiersmostly employed for cyber-hate speech detection tasks in the literature?

**R.Q.2:** Which of these TFE techniquesis likely to improve hate speech detection algorithm performance?

**R.Q.3:** Does the training time of the classifier vary under different text feature extraction (TFE) techniques with the same dataset?

**R.Q. 4:** Which evaluation metric is suitable for evaluating the true performance of a given model?

This study will add the following contributions to the body of knowledge in the domain of opinion mining:

- This research helps identify TFE techniques and classifiers mostly employed by researchers.

- This study helpsthe researchers to identify among the widely used TFE techniques that can likely improve the hate speech detection classifier performance using the SM dataset.
- Theexperiment conducted will help researchers to have an idea or estimatethe training time of the classifiers under different TFE techniques when using the SM dataset.
- This study will help researchers know the suitable evaluation metric when evaluating theirtrained model.

This study is organised in the following ways: sections 2 and 3 address motivation and related works. Sections 4, 5, and 6, discusseddata and methodology, experimental result and discussion, and conclusion and future work respectively.

**Motivation**

The study is motivated by the low prediction accuracy of ML models for hate speech detection in SM datasets. The investigation of human interactions within societies was historically the sole responsibility of the social sciences (Wagner, et al, 2021). Human societal interactions nowadays are more in virtual communities, which are algorithmically driven. This made the social scientists incapacitated. Now that societies are immersed in the virtual world driven by algorithmic and computer-powered, computer scientists are responsible for policing the virtual society. Therefore, these virtual communities which are full of offensive and hate speech need virtual policing for a safer virtual world. Detecting any form of offensive content in the virtual society is normally modelled as a text classification problem. Therefore, the more accurate the text classifier, the better the virtual society.

From an industry perspective,an estimation of over 80% of the data generated is in an unstructured form such as text, image, video or audio (Kulkarni &Shivananda, 2019). These text data are constantly being generated due to application-to-application and human-to-application interactions. Human-to-application interactions generated most of the data such as text messaging, e-commerce, surfing the internet, internet-based calls, video streaming, and video conferencing among others. Text data form the majority of the unstructured data generated with over 50% out of 80% of the unstructured data (Kulkarni &Shivananda, 2019).

Sources of these text data include chats on SM platforms such as Twitter, Facebook or YouTube. Other sources of text data are blogs, news apps, customers' reviews on products and

services and medical records. Recent developments have speech-to-text apps such as Dragon Anywhere[1], Google Assistant[2], Transcribe[3], Windows dictation[4], SpecchTexter[5] and Voice Notes[6], among others. SM is playing a significant role in our daily lives and most interactions are done through text messages. Therefore, getting rid of hate text completely will make SM a safer place for everyone to socialise with each other at all times.

**Related Work**

SM adoption and penetration are no longer news, however, the negative impacts on individuals, disadvantaged groups and communities of minorities arepoints of concern. This leads to growing concern over the inability of SM platforms to promptly and accurately detect hate speech. Twitter is one of the most used SM platforms for research such as opinion mining, political debate, natural disaster analysis and pandemic research among others (Modha et al., 2020). Cyber-hate speech detection using NLP and MLA methods is relatively new,therefore, the number of research articles in this area is limited (Mullah & Zainon, 2021). Especially articles dedicated to evaluating TFE techniques are out-rightly very limited. This makesresearchers find it difficult to conclude which TFE techniques improve a classifier's performance for hate speech detection on SM data. The catastrophic impact of SM posts is the driving force for research on cyber-hate speech detection in recent times.

Text classification has been researched extensively in the domain of text mining because of its significant roles in many text-related applications such as sentiment analysis, topic search, claims investigations, query recommendation, fake news detection, hate speech detection and spam filtering systems (Ji et al., 2019). The conversion of text data to numerical data is a significant breakthrough inthe text classification task. This conversion of text data to numerical data is called text feature extraction (TFE).TFE techniques play important roles as components of the text classifications processing pipeline using NPL. The significance of this stage is the fact that computers do not accept text data as input and therefore, must be converted to numerical

[1]https://apps.apple.com/us/app/dragon-anywhere/id1024652126

[2] https://apps.apple.com/us/app/google-assistant/id1220976145

[3] https://apps.apple.com/us/app/transcribe-speech-to-text/id1241342461

[4] https://support.microsoft.com/en-us/windows/use-dictation-to-talk-instead-of-type-on-your-pc-fec94565-c4bd-329d-e59a-af033fa5689f

[5]https://play.google.com/store/apps/developer?id=SpeechTexter

[6]https://play.google.com/store/apps/details?id=com.SouthernPacificOceanFisher.VoiceToText_memo&hl=en_US

data first. The feature extraction in the text classification also determines to a large extent the performance of the learning algorithm used. However, there islittle work done on this important subject matter.

Research conducted by Kwok & Wang (2013) attempted to identify racist posts on Twitter using Naive Bayes classifier. Racism is a subtype of hate speech that targets a particular race, and in this case, the black race was the target. Unigram or BoW was used as feature extraction along with the naive classifier for tweet categorization. The problem was modelledas a binary classification task, to group the labelled tweets as racist or non-racists tweets. A macro average accuracy of 76% was achieved with a 10-fold cross-validation technique.

Related research was conducted by Burnap & Williams (2015) which proposed a voting ensemble method to categorise tweets as hateful or antagonistic tweets with an emphasise on ethnicity, race and religion. Logistic regression (LR), support vector machines (SVM), random forest(RF), and decision tree(DT) were used as the base models. The unigrams and bigrams were employed as the TFE techniques for the research. The ensemble achieved the best F1-score (F1) of 95%.Waseem &Hovy (2016) also conducted a similar study to identify hate speech on Twitter. LR was employed as the classifier and character n-gram and word n-gram techniques were used as the TFE methods. Extra-linguistic features such as location and gender were also analysed. Sexism and racism were the focus of the content. In this research, character n-gram (73.89%) outperforms word n-gram (64.58%) with a difference of 9.31%.

A highly cited research conducted by Davison et al. (2017) proposed a multi-class classifier to distinguish hate speech from other offensive content on Twitter. In the research, crowd-sourcing was used to label the collected tweets as hate speech, offensive and neither. The research also concluded that homophobic and racist posts were mostly categorised as hate speech, while sexist tweets were seen as offensive by the model. TF-IDF with word n-gram (unigram, bigram and trigram) were used as TFE techniques. LR, naïve bayes (NB), DT, RF and SVM classifiers were used.LR perform best with an F1-score of 0.90.

A related study was also conducted by Watanabe et al. (2018) which intended to automatically detect and filter any hate content on Twitter. Unigram features were used along with RF, SVM, J48graft as classifiers. In this research, two approaches were proposed, binary and multi-class

classification. The binary classifier was meant to classify the tweets as offensive or non-offensive. The multi-class on the other hand was designed to categorise tweets as hateful, offensive and clean.The binary classifier reached an accuracy of 0.874 while the multi-class achievedan accuracy of 0.784.Ombui et al. (2019) researched to fill the gap of identifying hate speech in a codeswitched text post on Twitter. In the research, count vectors, TF-IDF (both word-level and character level) were used as the TFE techniques. Classifiers such as NB, LR, SVM, k-nearest neighbours (KNN), DT, and RF have been tested on the 25K annotated dataset. In this study, SVM has proven superior over other classifiers with an F1 of 0.825.

Recent research conducted by Laaksonen et al., (2020) published by Frontiers, aimed to monitor Finnish 2017 municipal elections. In the study, the SM accounts of the aspirants were monitored using a technical infrastructure for hate speech posts. Various supervisedML methods were tested in this research with SVM, NB, and RF as classifiers. The BoW was used as the main TFE technique and SVM as the classifier for the final result in the research.A related and most recent study conducted by Plaza-del-Arco et al.(2021) published with Elsevier aimed to address Spanish hate speech on SM. In the research, three NLP approaches were used, shallow learning, deep learning and transfer learning techniques. In the shallow method, classifiers such as LR and SVM were used along with the TF-IDF as the TFE technique. For deep learning, CNN, LSTM and BilSTM were used along with pre-trained models. Both monolingual and multilingual pre-train models show promising results for detecting hate speech in Spanish texts. Transfer learning outperformed both shallow and deep learning approaches.

**Data and Methodology**

Dataset description

For this research, we useda publicly available dataset provided by Kaggle[7] to conduct our experiments to help address our research objective. The dataset is made up of training and testing data. The training data were all labelled and the statistics are shown in Table 1.

[7]https://www.kaggle.com/arkhoshghalb/twitter-sentimentanalysis-hatred-speech

**Table 1**

Dataset statistics.

| Class label | quantity | Percentage (%) |
|---|---|---|
| 0 | 29720 | 93 |
| 1 | 2242 | 7 |
| *Total* | 31962 | 100 |

We used the training dataset for both training and testing for this study. The dataset was split in the ratio of 70:30, for training and testing respectively.

**Data Pre-processing**

Text pre-processing is a necessary step to guarantee a successful text analysis implementation (Ramya et al., 2016). However, this text pre-processing can be tasking and time-consuming, especially for SM data. SM data when collected are generally messy and full of noise, therefore, pre-processing is necessary to clean it up. We used the sklearn library to carry out the following pre-processing steps: removalof special characters and punctuation marks, empty spaces, numerical figures, emojis, URL, hashtags, cashtags, Twitter handles, and RT.

After pre-processing, we carried out text normalization. Text normalization is the conversion of words into their standard/canonical form. For instance, drive, driving and drove, should be normalised to “drive” and should be seen as one word. This is an important operation in short text analysis because of the unstandardized ways of writing on SM platforms. Commonly used normalization operations in Python libraries from sklearn(Pedregosa et al., 2011) are: stop words removal, tokenization, lower casing, stemming and lemmatization. We also used a regular expression to clean up some noise such as non-English words, for instance, two-letter words that do not belong to the stop words list.

When the data was cleaned and normalised, TFE techniques were employed. The application of the TFE technique on the dataset or corpus is to convert the text data into its equivalent in a numerical dataset ready for use by the algorithm. After the application of the TFE techniques,

the training of the MLA for the classification task takes place. Lastly, the evaluation process of the model is next in the pipeline. This experimental investigation researchis focused on the TFE section which determines the efficiency of the MLA.

Data Exploration

In data science research generally, scientists always explore the dataset to understand the relationship in the data to guide them in making an informed decision regarding further analysis operations. This can also help the researchers to determine the type of data cleaning technique for further cleaning and better analysis.

In the analysis of text data, it is necessary to identify the words that constitute the dataset. The exploration is significant because the meaning or semantics of the text can be interpreted by analysing the words in the dataset. For example,we used “word cloud”, and “n-grams” to help us visualize our data.See Fig.1 and Fig.2 for the wordcloud and bigrams respectively of our dataset.


**Fig. 1.**Word cloud for the dataset.

From Fig. 1, one can see obviously that the data contains more cheerful words than hate. This is in line with the class label counts, where 93% of the data is non-hate tweets. We explore further with Bigram to see the relationship that exists in the pairs and come up with Fig. 2.

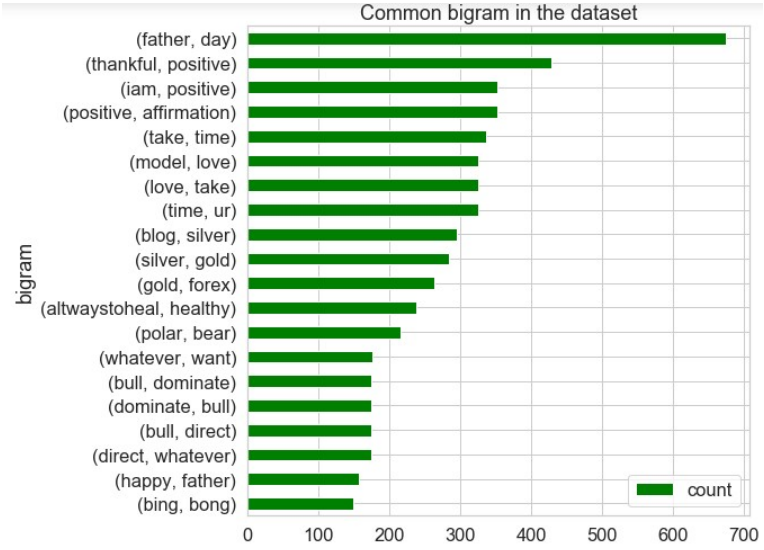


**Fig.2.** Bigram counts.

Fig.2 shows the bigram relationship in the dataset. The pair with the highest frequency is "Father day" with over 600 pairsco-occurrence. Most of the pairs make sense in real-life scenarios, for instance, "Iam positive", "polar bear", "happy father" and many others.

**Feature Representation/ TFE**

Text analysis is one of the important application domains for MLAs. Though text data cannot be fed directly to the MLA, because algorithms use numerical feature vectors having fixed size lengths and not data with varying size lengths. Many TFE techniques have been proposedto address this challenge. Different IDEs have developed libraries to simplify the process of solving the problem. The widely used library is the Python sklearn library (Pedregosa et al., 2011).

sklearn provides utilities that help in extracting numerical features from the text document. These include tokenizing (splitting tokens and assigning ID for each) counting (frequencies of each token occurrence in the document) normalizing (and weighting with diminishing importance). The general process of converting a set of text documents to numerical feature vectors is called vectorization. Each process (tokenization, counting and normalizing) is termed

the Bag of Words (BoW) or Bag of n-grams representation. Documents are characterized by word occurrences while entirely disregarding the relative location information of the words in the document.

Python as a programming language contains different modules. A module in Python is a file made up of Python definitions and statements. Modules usually define classes, functions and variables. sklearn feature extraction is an important example of a module for NLP. This module contains four functions that are very significant in TFE operations worth mentioning as described in Fig. 3.

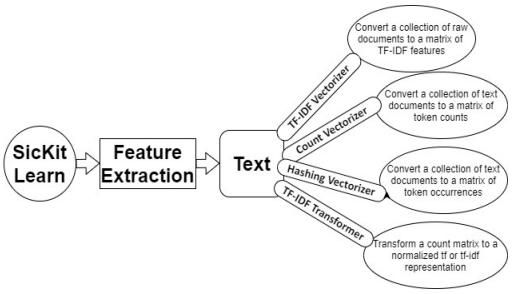


**Fig.3.** TFE functions are provided by sklearn library.

The accuracy of MLAs for text classification tasks depends greatly on the type of TFE method used. TFE methods help to transform text data to a machine-understandable format which is suitable for classifier consumption. This research aims to identify the most suitable TFE techniques for SM classification for hate and non-hate content. We identified some commonly used TFE techniques such as TF-IDF, count vectorization, word n-gram, and character n-gram for hate speech detection

TFE is an important stage in any text classification task for two reasons: It transforms the text data into feature vectors andcan improve the performance of the detection classifier (Liang et al., 2017). Text data normally contain large features and consequently, lead to a curse of dimensionality. The method of selecting some important features to reduce the dimensions of the feature space in the dataset is called feature selection. Feature selection techniques are incorporated in most feature extraction methods. In carrying out the feature extraction, data that cannot add value to the training of the algorithm will be discarded. The feature extraction method usually does this carefully such that the original meaning of the text data is not distorted.

Given this high volume and unstructured nature of SM text data, to gain significant insight, we need NLP alongside ML (both shallow and Deep Learning (DL). Unstructured data simply means the data that do not conform to the traditional relational database. However, machines and algorithms do not understand character or text data. Therefore, it is paramount to convert the characters into text for the machines and algorithms consumption or machine-understandable formats. The machine-understandable formats are numbers that the algorithms can use to perform any type of analysis it is programmed to do on the text data. The analysis can be text classification or text interpretation among others. Technically, this process is referred to as the transformation of text data into feature vectors which arerepresentations of the original dataset. This process of making machines alongside algorithms to understand and interpret human natural language in text form is called NLP.

TFE emanated from Vector space models (VSMs) (Turney & Pantel, 2010) in computing science and also distributional models (DMs) (Erk, 2012) in linguistics. The VSM aimed to present a document in a collection as a point in space. A point in space is referred to as a vector in a vector space (Turney & Pantel, 2010). DM portray a word as a point in high-dimensional space in which each dimension represents a contextual item, and a word's coordinates reflect its contextual frequencies(Erk, 2012; Liang et al., 2017)**.** Those words that are nearer to each other in the vector space are considered to be similar in meaning while points that are not close to each other are seen as having a different meaning. DM represents a word through it context in which it has been seen. The similarity in meaning of a word can be predicted using DM via distributed hypothesis. This hypothesis states that "two words that occur in similar contexts tend to have similar meanings (Erk, 2012)." For instance, the meaning of the word "Lion"is closer to the meaning of the word "Buffalo" than the meaning of the "Computer".

The success story of VSMs for information retrieval motivated researchers to extend the VSMs idea to other semantic-related tasks in NLP, suchas hate speech detection, opinion mining, fake news detection, and topic modelling among others. This implies that based on the labelled keywords, the weights of the words in the document can be computed using some methods to get a digital vector, which represents the feature vector of the text (Erk, 2012; Liang et al., 2017).Different techniques have been developed to help achieve this. Text data can be transformed into feature vectors using the following techniques TF-IDF Vectors as features

(Word level, N-Gram level, Character level), Count Vectors as features, word embedding as features, text / NLP-based features and Topic Models as features.

**Performance Evaluation Metrics**

The evaluation of text classifiers is generally a tricky one. Most SM datasets used for hate speech detection suffer from an imbalanced class distribution problem (Jang et al., 2021). On this note, the performance evaluation metric employed for assessing any model trained with this type of dataset requires special attention tothe evaluation procedure. The evaluation metric is used to determine how good a model is after the training has been conducted.In literature, recall (R), precision ($P_r$), F1-score (F1) and accuracy (A)evaluation metrics have been used extensively. Each of these is computed using a confusion matrix (CM). CM contains True-negative (TN), true-positive (TP), false-positive (FP), and false-negative (FN) values. The CM can simply be represented as shown in Fig. 4 for a binary classification task.

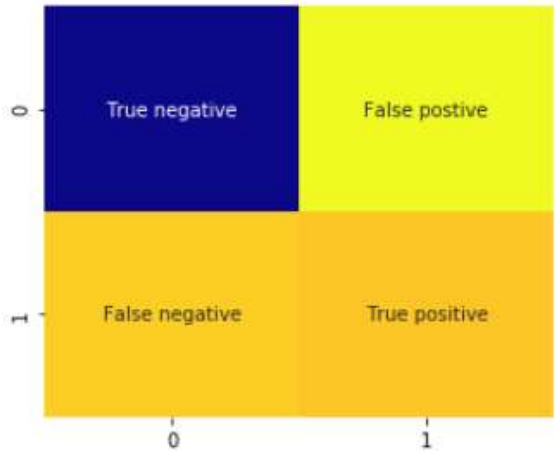


**Fig.4**. Confusion matrix.

Fig. 4 is typical of a binary classification task showing all the corresponding cells for each of TP, TN, FN, and FP in the CM. In every experiment, researchers attempted to maximise the values of TP and TN. On the other hand, researchers always try to minimise FN and FP. The R, $P_r$, F1 and A metricscan be computed by the relations in Eq. (1), Eq. (2), Eq. (3) and Eq. (4) respectively (Plaza-del-Arco et al., 2021):

$$R = \frac{TP}{(FN + TP)} \quad (1)$$

$$P_r = \frac{TP}{(TP + FP)} \quad (2)$$

$$F1 = 2 * \frac{P_r * R}{(P_r + R)} \quad (3)$$

$$A = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (4)$$

**Experimental Results and Discussion**

To answer R.Q. 1, these researchers searched comprehensively in the most reputable databases for the different TFE techniques used in hate speech detection studies. We considered only the work in which the researcher clearly stated the type of TFE techniques used in the research. We also limit the search to those studies that made use of traditional ML methods for hate speech detection. We summarised these TFE methods and the commonly used classifiers in Table 3 to enable usto answer some of our research questions. We selected sixteen articles that conformed to our inclusion criteriafrom reputable databases,those that used classical ML with clear TFE methods. We captured the table under reference, TFE technique, classifier, evaluation metric, and journal database. The analysis is shown in Table 3.

Table 3 shows the different types of TFE techniques commonly applied for hate speech detection on SM datasets. Some were used more often than others. Researchers used different nomenclatures to call different TFE techniques. For instance, unigram and BoW mean the same. TF-IDF and word vectors refer to the same technique. When most researchers mentioned n-gram, they are referring to the word n-gram method. An n-gram is a contiguous series of n-elements (character, word, phrase, sentence, or paragraph) from a sample of text dataset or corpus. When character n-gram is employed, it is always explicitly specified. Different n-gram

ranges can be used based on the type of application, the problem to be solved and the dataset. N-gram may be confusing, therefore we give a further explanation in the next paragraph.

Let the lower boundary of the range of n-values be $n_x$ and the upper boundary of the range of n-values be $n_y$. Where $n_x n_y \in n$ for word and character n-gram combinations, for example. Therefore, ngram_range of ($n_x$, $n_y$), means all values of n in the inequality range $n_x \leq n \leq n_y$ are to be extracted. For instance, the wordngram_range(1,3) means all unigrams, bigrams and trigrams in the corpus or dataset are to be considered for the analysis. Also, when n-gram_range(1,1) is used, it means the same as BoW in the implementation sense. That is individual words only will be considered. N-gram_range(2,5); this means all 2,3,4,5grams will be considered.

To clearly explain Table 3, we represent the TFE techniques and their frequencies in the 16 articles analysed in the form of a graph as shown in Fig. 5.From Fig.5, it is obvious that TF-IDF is the most patronised text vectorization method for cyber-hate detection. BoW and word n-gram are the second most used TFE techniques. Count vector is also used relatively well. This has answered the first part of R.Q.1. To answer the second part of R.Q.1, we can summarise the classifiers used in the articles for hate speech detection as represented in Fig. 6.Out of the sixteen articles, at least ten used either RF or SVM in their study. SVM is the most favoured classifier as twelve out of sixteen articles used it for their research, and this agrees with the research conducted by (Plaza-del-Arco et al., 2021). NB is also used considerably well. LR is gaining more attention with eight experiments out of sixteen articles analysed. Based on this, we can conclude that SVM is the most used, followed by the RF in the state-of-the-art for hate speech detection.

To answer research question 2, the researchers captured the timing for both training and prediction for each classifier in all the TFE methods commonly used as shown in Table 4.From Table 4, the timing ranges from 0.04s in character n-gram (CharNG) to 20007.69s inthe count vectorization technique. It took SVM to train under the count vectorization 20007.69s while SVM training lasted for 343.126s in the character n-gram technique. For LR, it lasted 209.319s to train using count vectorization while it took LR 0.041s to train under character n-gram. SVM and LR, and count vectorization and character n-gram produced extreme cases. Other classifiers under different TFE techniques show different training times, with count vectorization highest

to least in character n-gram. To answer the second part of R.Q. 2, we tabulate the performance of each classifier under different TFE methods as shown in Table 5.

**Table 2**

Analysis of TFE techniques used for hate speech detection using ML.

| S/N | Reference | TFE technique | Classifier | Evaluation metric | Journal Database |
|---|---|---|---|---|---|
| 1 | (Yadav et al., 2021) | unigram, bigram | LR, SVM, NB | $P_r$, R, F1, A | Springer |
| 2 | (Plaza-del-Arco et al., 2021) | TF-IDF | LR, SVM | $P_r$, R, F1 | Elsevier |
| 3 | (Khan et al., 2021) | Count vectors, TF-IDF, Char ngram | NB, LR, RF, SVM | $P_r$, R, F1, A, CM | ACM |
| 4 | (Laaksonen et al., 2020) | TF-IDF | SVM, GNB, MNB, RF | $P_r$, R, F1, A, AUC | Frontiers |
| 5 | (Aljarah et al., 2020) | BoW, TF-IDF | SVM, NB, DT, RF | $P_r$, R, A, GM | Sage |
| 6 | (Kumar et al., 2020) | Count Vectors | RF | $P_r$, R, F1, A | Open review |
| 7 | (Mulki et al., 2019) | uni-bi-tri-grams | SVM, NB | $P_r$, R, F1, A | Research gate |
| 8 | (Pereira-Kohatsu et al., 2019) | POS tags, unigrams | SVM, RF, QDA, LDA | $P_r$, R, F1, A, AUC | MDPI |
| 9 | (Ousidhoum et al., 2019) | BoW | LR | Macro-F1 Micro-F1 | arXiv |
| 10 | (Nugroho et al., 2019) | Count vectors | RF | $P_r$, R, F1, A, CM | IEEE |
| 11 | (Ombui et al., 2019) | TF-IDF | NB, LR, SVM, KNN, DT, RF | A, CM | IEEE |
| 12 | (Watanabe et al., 2018) | Unigrams | RF, SVM, J48graft | $P_r$, R, F1, CM | IEEE |
| 13 | (Martins et al., 2018) | N-gram | SVM, NB & RF | $P_r$, R | IEEE |
| 14 | (Davidson et al., 2017) | n-grams, TF-IDF | RF, LR, NB,SVM, DT | CM | arXiv |
| 15 | (Zia et al., 2016) | TF-IDF | NB, SVM, KNN | $P_r$, R, F1 | Research gate |
| 16 | (Waseem &Hovy, 2016) | char n-grams, word n-grams | LR | $P_r$, R, F1 | Semantic Scholar |

AUC=Area under the curve, CM=confusion matrix, GM=Geometric mean

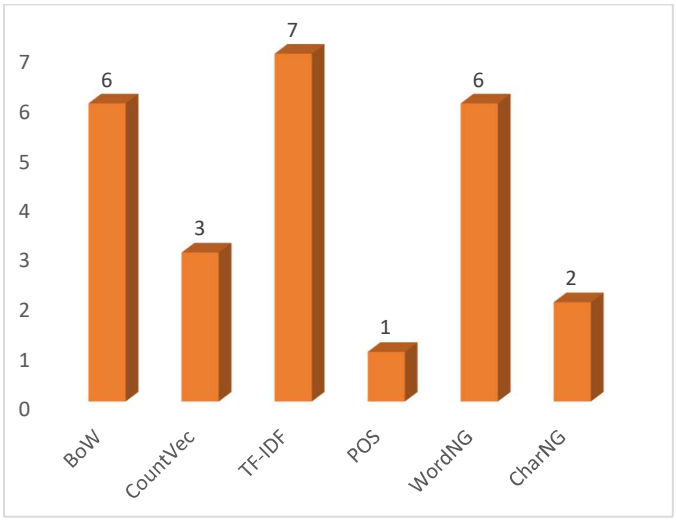


**Fig.5.** Frequencies of TFE techniques are commonly used in literature.

Using Tables 4 and 5, we make the following analysis. It took SVM a longer time to train (see Table 4) on the datasetwith one of the best F1 0.82 (see Table 5) using the count vectorization technique as TFE, while it took the LR 0.041s to train on the same dataset with an F1 of 68% using charNG. Based on this, it is obvious that the longer it takes the algorithm to train, the better the performance. Therefore, based on this experiment, count vectorization has the longest training time and produces one of the best performances as well.

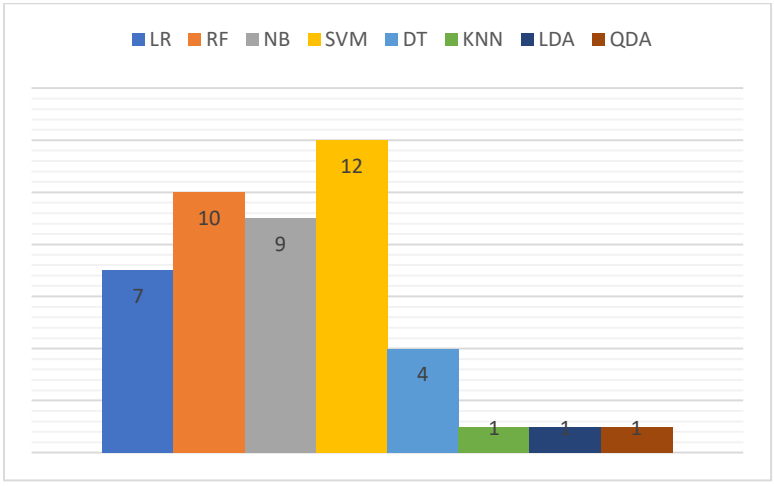


**Fig.6.**Hate speech detection classifiers and frequency of use.

From Table 5, we plotted F1 for each classifier in the TFE techniques, see Fig. 7. From Fig. 7, one cannote that count vectorization outperforms all other TFE techniques testedon the five classifiers used. SVM gave the best result of 82% followed by LR with 80% while RF & DT take the third position with 79% and MNB last, with 77% using the count vectorization technique. The second promising TFE technique is the TF-IDF. It happened to be the most widely used in the literature, though our result did not justify this. The best of TF-IDF was seen in RF with 78%. Both count vectors and TF-IDF perform relatively well across the five classifiers, with F1 ranges from (71% - 82%). This will help us answer R.Q.3, which seeks to find out the TFE technique (s) that is likely to improve hate speech detection in SM datasets. Our results are very similar to the research conducted by Ombui et al. (2019) with little variation. Ombui et al. (2019) found out that SVM performed best under TF-IDF at the character level and LR performed best under the count vectorization. The little variation could be a result of different datasets used and pre-processing techniques employed. This shows that both are good TFE techniques to go for when classifying the SM datasets as hate and non-hate.

Knowing which evaluation metric to use for the evaluation of text classification models is another concern among researchers. New researchers in the field of text mining can get worried about which evaluation metric can better describe their models. F1-score (F1) and accuracy (A) have been used quite a lot in the literature (see Table 3). But which one will tell us the true performance of our model? To answer this question, we plotted F1 alongside the A as shown in Fig. 8. This is to help us make an informed conclusion on the better evaluation metric for our dataset.

The F1 for the mostly applied ML classifiers (MNB, LR, SVM, RF and DT) ranges from 52% to 82% in TFE techniques used (Count Vectorization, TF-IDF,WordNG and CharNG). The A ranges from 92% to 96% for all the TFE techniques applied. The A metric shows no significant difference across the five classifiers under the four TFE techniques. The highest A was 96% and the least was 92%, a difference of 4%. The A metric has a high value across all the classifiers with little variation because $P_r$ and R were not factored in the computation of A (see Eq.(4)**)**. $P_r$ and R for minority classes are usually low due to the inability of the algorithm to learn sufficiently about it. In the case of the F1, the highest was 82% and the least was 52%,

a difference of 30%. That means the F1 can tell better about the performance of a model compared to an accuracy metric. Another interesting thing about the F1 is that it is a harmonic mean of precision and recall (See Eq. (3)). Accuracy will be suitable when the class distribution of a dataset is equal or not skewed to one class. For instance, if a dataset has both hate speech and non-hate class at 50% in each case.In this case, A becomes the preferred evaluation metric. The A metric can be misleading with the dataset that has a skewed class distribution (Jang et al., 2021). Our findings concur with other findings in the literature as stated by (Zhang & Luo, 2018). This has answered our R.Q.3

**Table 3**

Training time (TT) and prediction time (PT) of the selected classifier under ContVec, TF-IDF, WordNG and CharNG.

| Classifier | CountVec | | TF-IDF | | WordNG | | CharNG | |
|---|---|---|---|---|---|---|---|---|
| | TT | PT | TT | PT | TT | PT | TT | PT |
| MNB | 95.37 | 115.864 | 3.68 | 0.12 | 0.278 | 0.094 | 0.631 | 0.11 |
| LR | 209.31 | 54.095 | 5.32 | 0.089 | 2.982 | 0.084 | 2.218 | 0.041 |
| RF | 3620.677 | 1100.16 | 427.3 | 2.789 | 357.6 | 3.842 | 39.44 | 1.23 |
| SVM | 20007.69 | 1507.50 | 541.3 | 33.799 | 427.7 | 62.20 | 343.126 | 19.15 |
| DT | 7715.86 | 511.783 | 407.0 | 27.16 | 182.7 | 27.968 | 65.672 | 12.87 |

TT stands for training time and PT for prediction time

**Table 4**

Precision ($P_r$), Recall (R), F1-score (F1) and Accuracy of the selected classifier under ContVec, TF-IDF, WordNG and CharNG.

| CountVec | TF-IDF | WordNG | CharNG |
|---|---|---|---|

| Clf | P | R | F1 | A | P | R | F1 | A | P | R | F1 | A | P | R | F1 | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MNB | 79 | 75 | 77 | 95 | 92 | 65 | 71 | 95 | 90 | 59 | 64 | 94 | 95 | 52 | 52 | 93 |
| LR | 91 | 74 | 80 | 96 | 90 | 67 | 73 | 95 | 96 | 58 | 63 | 94 | 88 | 62 | 68 | 95 |
| RF | 93 | 73 | 79 | 96 | 89 | 72 | 78 | 96 | 88 | 60 | 64 | 94 | 97 | 60 | 66 | 95 |
| SVM | 86 | 79 | 82 | 96 | 90 | 69 | 75 | 95 | 95 | 59 | 64 | 94 | 93 | 62 | 67 | 95 |
| DT | 82 | 76 | 79 | 95 | 74 | 74 | 74 | 93 | 86 | 60 | 64 | 94 | 68 | 69 | 69 | 92 |

Clf for the classifier.

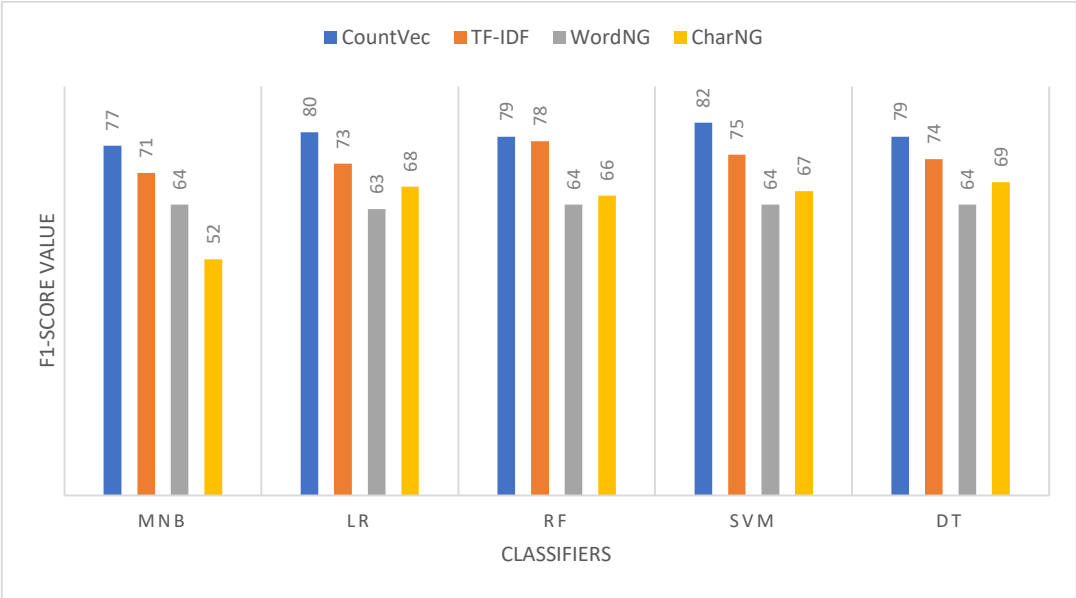


**Fig.7.**F1-score for each classifier in the four TFE techniques.

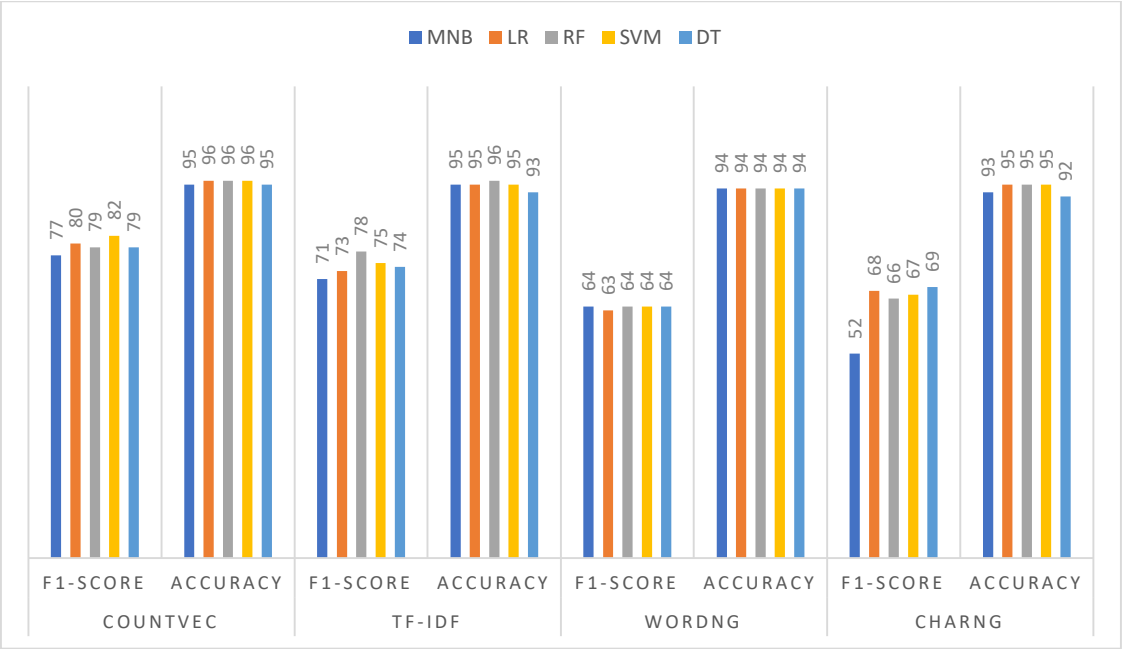


**Fig.8.**Precision and accuracy performance plot.

*Remember the dataset we used is grossly skewed in the class distribution. Out of 31,962 tweets, only 2,242 were labelled as hate, which is 7% of the total tweets. That means 93% of the tweets were non-hate tweets, with a total of 29,720 tweets. However, a lot of researchers still employ different evaluation metrics such as A, $P_r$, and R despite the imbalanced class nature of their datasets. You can see some of the different metrics used in Table 3.*

**Conclusion and Future Work**

In this article, the researchers identified the four most widely applied TFE techniques in state-of-the-art for hate speech detection studies, which include count vectorization, TF-IDF, wordNG and charNG. The researchersalso identified five commonly used traditional ML classifiers for hate speech detection on SM datasets such asSM, LR, SVM, DT and RF.The researchers used the publicly available Kaggle dataset which is labelled as hate speech and non-hate speech. The study carried out the necessary cleaning and normalization operations on the dataset using the four commonly identified TFE techniques on the dataset to extract features. The study then applied the extracted features to train the five classical MLAs identified in the literature.

Count vectorization as a TFEmethod shows a remarkable improvement in the performance of the five models. The best was seen on SVM with an F1-score of 82%. TF-IDF is a TFE technique that also improves the performance of the classifiers with the best improvement on RF with an F1 of 78%. wordNG and charNG both do not improve the performance of our classifiers significantly. The time taken for the training of the selected classifiers ranges from 20007.69s to 0.04. SVM took the longest time of the 20007.69s to train using count vectorization as the feature extraction technique.

SVM also produced one of the best performances of 82% in F1. LR on the other hand was trained in the shortest time of 0.04s using charNG as the feature extraction method. Based on this experiment, this research concludes that the longer it takes a classifier to train, the better the performance. Count vectorization and TF-IDF are better options to try on SM datasets as TFE techniques to improve the classifier performance. SVM is the most used classifier in the literature and these results justify the reason for the high patronage.

In the future, the researchers will experiment using different deep-learning text feature extraction methods to study the behaviour in different scenarios. The reseachers will also experiment with TFE techniques that are rarely used such as one Hot encoding, co-occurrence matrix, and hash vectorization.

**References**

Aljarah, I., Habib, M., Hijazi, N., Faris, H., Qaddoura, R., Hammo, B., Abushariah, M., & Alfawareh, M. (2020). Intelligent detection of hate speech in Arabic social network: A machine learning approach. *Journal of Information Science*. https://doi.org/10.1177/0165551520917651

Arcila-Calderón, C., Blanco-Herrero, D., Frías-Vázquez, M., & Seoane, F. (2021). Refugees welcome? Online hate speech and sentiments in twitter in spain during the reception of the boat aquarius. *Sustainability (Switzerland)*, *13*(5), 1–17. https://doi.org/10.3390/su13052728

Burnap, P., & Williams, M. L. (2015). Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy and Internet*, *7*(2), 223–242. https://doi.org/10.1002/poi3.85

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). Automated hate speech detection and the problem of offensive language. *Proceedings of the 11th International Conference on Web and Social Media, ICWSM 2017*, 512–515.

Erk, K. (2012). Vector Space Models of Word Meaning and Phrase Meaning: A Survey. *Linguistics and Language Compass*, *6*(10), 635–653. https://doi.org/10.1002/lnco.362

Jang, J., Kim, Y., Choi, K., & Suh, S. (2021). Sequential targeting: A continual learning approach for data imbalance in text classification. *Expert Systems with Applications*, *179*(April), 115067. https://doi.org/10.1016/j.eswa.2021.115067

Ji, L., Wang, Y., Shi, B., Zhang, D., Wang, Z., & Yan, J. (2019). Microsoft Concept Graph: Mining Semantic Concepts for Short Text Understanding. *Data Intelligence*, *1*(3), 238–270. https://doi.org/10.1162/dint_a_00013

Kalsnes, B., & Ihlebæk, K. A. (2021). Hiding hate speech: political moderation on Facebook. *Media, Culture and Society*, *43*(2), 326–342. https://doi.org/10.1177/0163443720957562

Khan, M. M., Shahzad, K., & Malik, M. K. (2021). Hate Speech Detection in Roman Urdu. *ACM Transactions on Asian and Low-Resource Language Information Processing*, *20*(1), 1–19. https://doi.org/10.1145/3414524

Kulkarni, A., & Shivananda, A. (2019). *Natural Language Processing Recipes*. Apress. https://doi.org/10.1007/978-1-4842-4267-4

Kumar, S., Ratn Pranesh, R., & Chandra Pandey, S. (2020). TweetBLM: A Hate Speech Dataset and Analysis of Black Lives Matter-related Microblogs on Twitter. *Zeneoo.Org*. https://doi.org/10.5281/zenodo.4000539

Kwok, I., & Wang, Y. (2013). Locate the Hate: Detecting Tweets against Blacks. *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, *360*(17), 3266–3270. https://doi.org/10.1002/adsc.201800642

Laaksonen, S.-M., Haapoja, J., Kinnunen, T., Nelimarkka, M., & Pöyhtäri, R. (2020). The

Datafication of Hate: Expectations and Challenges in Automated Hate Speech Monitoring. *Frontiers in Big Data*, *3*(February), 1–16. https://doi.org/10.3389/fdata.2020.00003

Liang, H., Sun, X., Sun, Y., & Gao, Y. (2017). Text feature extraction based on deep learning: a review. *Eurasip Journal on Wireless Communications and Networking*, *2017*(1), 1–12. https://doi.org/10.1186/s13638-017-0993-1

Martins, R., Gomes, M., Almeida, J. J., Novais, P., & Henriques, P. (2018). Hate speech classification in social media using emotional analysis. *IEEE Proceedings - 2018 Brazilian Conference on Intelligent Systems, BRACIS 2018*, *April 2019*, 61–66. https://doi.org/10.1109/BRACIS.2018.00019

Modha, S., Majumder, P., Mandl, T., & Mandalia, C. (2020). Detecting and visualizing hate speech in social media: A cyber Watchdog for surveillance. *Expert Systems with Applications*, *161*, 113725. https://doi.org/10.1016/j.eswa.2020.113725

Mulki, H., Haddad, H., Bechikh Ali, C., & Alshabani, H. (2019). L-HSAB: A Levantine Twitter Dataset for Hate Speech and Abusive Language. *Third Workshop on Abusive Language Online*, 111–118. https://doi.org/10.18653/v1/w19-3512

Mullah, N. S., & Zainon, W. M. Na. W. (2021). Advances in Machine Learning Algorithms for Hate Speech Detection in Social Media : A Review. *IEEE Access*, *9*, 88364–88376. https://doi.org/10.1109/ACCESS.2021.3089515

Munn, L. (2020). Angry by design: toxic communication and technical architectures. Humanities and Social Sciences Communications, 7(1), 1–11. https://doi.org/10.1057/s41599-020-00550-7

Nugroho, K., Noersasongko, E., Purwanto, Muljono, Fanani, A. Z., Affandy, & Basuki, R. S. (2019). Improving random forest method to detect hatespeech and offensive word. *IEEE International Conference on Information Technology and Communication Technology*, 514–518. https://doi.org/10.1109/ICOIACT46704.2019.8938451

Ombui, E., Muchemi, L., & Wagacha, P. (2019). Hate Speech Detection in Code-switched Text Messages. *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, 1–6.

Ousidhoum, N., Lin, Z., Zhang, H., Song, Y., & Yeung, D.-Y. (2019). Multilingual and Multi-Aspect Hate Speech Analysis. *ArXiv*, 4667–4676. https://doi.org/10.18653/v1/d19-1474

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grise, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., & Cournapeau, D. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *19*(1), 2825–2830. https://doi.org/10.1145/2786984.2786995

Pereira-Kohatsu, J. C., Quijano-Sánchez, L., Liberatore, F., & Camacho-Collados, M. (2019). Detecting and monitoring hate speech in twitter. *Sensors (Switzerland)*, *19*(21), 1–37. https://doi.org/10.3390/s19214654

Plaza-del-Arco, F. M., Molina-González, M. D., Ureña-López, L. A., & Martín-Valdivia, M. T. (2021). Comparing pre-trained language models for Spanish hate speech detection. *Expert Systems with Applications*, *166*(September 2020), 114120. https://doi.org/10.1016/j.eswa.2020.114120

Ramya, R. S., Venugopal, K. R., Iyengar, S. S., & Patnaik, L. M. (2016). Feature extraction and duplicate detection for text mining: A survey. *Global Journal of Computer Science and Technology: C Software & Data Engineering*, *16*(5), 1–20.

Turney, P. D., & Pantel, P. (2010). From Frequency to Meaning: Vector Space Models of Semantics Peter. *Journal of Artificial Intelligence Research*, *9*, 141–188.

Waseem, Z., & Hovy, D. (2016). Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. *NAACL-HLT 2016*, 88–93. https://doi.org/10.18653/v1/n16-2013

Watanabe, H., Bouazizi, M., & Ohtsuki, T. (2018). Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection. *IEEE Access*, *6*, 13825–13835. https://doi.org/10.1109/ACCESS.2018.2806394

Yadav, N., Kudale, O., Rao, A., Gupta, S., & Shitole, A. (2021). Twitter Sentiment Analysis Using Supervised Machine Learning. *.." In Intelligent Data Communication Technologies and Internet of Things: Proceedings of ICICI 2020*, *57*(March), 631–642. https://doi.org/10.1007/978-981-15-9509-7_51

Zhang, Z., & Luo, L. (2018). Hate speech detection: A solved problem? The challenging case of long tail on Twitter. *ArXiv*, *10*(5), 925–945. https://doi.org/10.3233/SW-180338

Zia, T., AKRAM, S., NAWAZ, S., SHAHZAD, B., ABDULLATIF, A. M., MUSTAFA, R. U., & LALI, I. (2016). Identification of hatred speeches on twitter. *The IRES International Conference,November*, 1251–1255. https://doi.org/10.1109/ASONAM.2016.7752398