International Journal of Computing and Artificial Intelligence

E-ISSN: 2707-658X
P-ISSN: 2707-6571
IJCAI 2021; 2(1): 01-05
Received: 05-11-2020
Accepted: 10-12-2020

Francisca O Oladipo
Federal University Lokoja, Nigeria

Sunday P Afolabi
Federal University Lokoja, Nigeria

Ezendu Ariwa
University of Bedfordshire, UK

Corresponding Author:
Francisca O Oladipo
Federal University Lokoja, Nigeria

A dataset of abusive comments on the Nigerian web

Francisca O Oladipo, Sunday P Afolabi and Ezendu Ariwa

DOI: https://doi.org/10.33545/27076571.2021.v2.i1a.20

Abstract
As the online comments keep growing in an exponential manner, the need for effective filtering or detecting abusive language in generated web textual content becomes increasingly important. In addition, the geometric increase in the textual content of the web has made the use of methods like word-bag and pattern matching for detection of hate speech less effective. The purpose of this work is to develop a corpus of online users' comments annotated for abusive language and develop a classification machine learning based model to detect hate speech on the Nigerian web. The model is evaluated on real-time comments and in different settings to further increase the learning rate of our model.

Keywords: Naive Bayes, Logistic Regression, XG Boost and Support Vector Machine.

Introduction
Social networking companies have done wonderful work in the globe today by making communication among physically distanced people possible (Al-Hassan & Al-Dossari, 2019) [2]. However, internet users have also seen it as a platform where they could rain abusive comment on others which denigrates them (Ibrohim & Budi, 2019) [5]. Offensive comments also referred to as hate speech (de Pelle & Moreira, 2014) [7] is defined as "any communication that disparages person or a group on some characteristics such as race, color, ethnicity, gender, sexual orientation, nationality, religion or other characteristics" (Brown, 2017) [3]. Resolving a problem like this is very difficult because of the irregularity in the context of the words, phrases or clauses used in these comments (Ibrohim & Budi, 2019) [5].
Machine Learning is the conjunction of the statistical theorems and artificial intelligence techniques which tend to make computer program capable of learning from an observational data in deduction of an insightful pattern which could be used to making decisions of the intended goals (Agarwal, 2013) [1]. Sentiment analysis is one of the applications of Natural Language Processing (NLP), which deals with the motive behind human languages or expression whether written or oral. According to Edureka (one of the online Data Science eLearning platforms), NLP can be split into two: Natural Language Understanding and Natural Language Generation. In this research, we are going to mine text data from social media such as Twitter using text mining technique. Recent work shows that, effort has been channeled as regard to hate speech detection on the web. But, the first datasets become publicly available in 2016 which centered on Indonesians' language by (Pelle & Moreira, 2014) [7]. With 470 tweets in German (Malmasi & Zampieri, 2017) [6] classified as to whether they contain hate speech against refugees. (Gao *et al*., 2017) [4] contains 115K Wikipedia discussion comments in English annotated as to whether the comment has a personal attack. (Waseem, 2016) [9] has 7K tweets in English annotated for hate speech. Also, the Kaggle website has a dataset with about 4K English tweets classified as offensive or not. None of these datasets cover the Nigerian web, and we present what is, to the best of our knowledge, the first initiative in this direction.

Review of related research
Netizens' safety and security in social media has become an inevitable concern, in the view of this, there are been quite a lot of papers that had addressed this challenge on social media, however, limited to the observed country. They applied the approaches of Machine Learning, Natural Language Processing etc. which are related to our research. One very close paper to our work, is the research carried out by de Pelle and Moreira (2014) [7], who apply n-gram Natural Language Processing Technique on randomly selected 1,250 instances out of 10,336 generated comments on the Brazilian Web scraped from 115 News websites.

Prior to n-gram technique, they manually annotate these datasets of offensive comments for Portuguese and these were trained using SVM and Naïve Bayes. But they have in their report that, n-gram turns out to be better feature in detecting hate speech for SVM than Naïve Bayes Classifier. Also, Malmasi and Zampieri (2017) [6], present a paper that explores hate speech detection on website methods, though, distinguishing this from the typical profanity. In that research, they applied supervised machine learning methods with the recently observed dataset for the aim of their research. In extracting features, n-grams, word n-grams, and word skip-grams were expended with their model accuracy of 78%, SVM classifier result. Here is another research work which further unveils another technique used in NLP for the ease of our research. The works of (Razavi *et al*., n.d.), (Ibrohim & Budi, 2019) [5] and (Al-Hassan & Al-Dossari, 2019) [2] also contribute to our research in term of cross validation of the achieved accuracies in their work in contrast to ours. However, in our work we consider the first four most used classification algorithm, that are suitable for text classification, sentiment analysis which are aforementioned above. Our accuracy for each of these classifiers is stated later in the research as you read.

Methodology

This section of the project describes the sequence stages in which we adopt to achieve the stated objectives of the proposed system. The selected methodology for this paper is the Cross Industry Standard Process for Data Mining due to its sequential and iterative approach to problem solving applying data science and machine learning algorithms which is relevant to activities carried out in this research. We systematically employ the scientific methods identified with this methodology.

The research is developed basically with two programming languages which include Python Programming Language and Java Programming Language for the data analysis and utilizing inference from the developed model on an android mobile app called 'Text Classifier' for demonstration respectively.

The system developed in this research work however, is divided into two modules: Registration Module, and Real-time Abusive Text Detection Module. We use the Python programming language for the real-time abusive text detection module and the Java Programming language to create the system to use the trained model (Registration Module and Chat Activity Module).

1. Data Collection and Description

Data in the form of raw tweets is retrieved by using the Python library called 'tweepy' which provides a package for real time twitter streaming API. The API requires us to register a developer account with Twitter and fill in parameters such as consumer Key, consumer Secret, access Token Access, and token Secret. This API allows to get all random tweets or filter data by using keywords. Filters supports to retrieve tweets which match a specific criterion defined by the developer. We used this to retrieve tweets related to specific keywords which are taken as input from users. The input array of keywords is provided as an argument to Streaming filter. For instance, on inputting multiple keywords like, 'Atiku', 'Buhari', 'Sowore', the output we obtained from this live stream of tweets associated with these keywords.

2. Data preprocessing

Here, we carry out some cleaning techniques which help us to have a cleaned text, as it is known generally the users of the web tend to mix up their comments sometimes with some character such as '#wow', '@home' etc. which if not cleaned, would cause our model acts abnormally. Data processing involves case swapping, removal of special character, tokenization, stop words. And in cleaning our retrieved tweets, we consider the following enumerated procedures:

i) Removal of non-alphabetical characters and URLs

Non-alphabetical characters are set symbols that are not letters contain in English Alphabet such as "@$£^&~#¬`+-%><? /.," etc. which add no semantical meaning in the content of language where they are been used. Numbers hold no relevance in case of sentiment analysis and are removed using pattern matching, '[0-9]'. To remove this, here regular expression technique is deployed as shown in the figure below.

```
raw_data.head()
```

	text	label	sentiment
0	@OmasoroO The worst kind of insidious self-loa...	positive	1
1	Beto is so mediocre. It's painful to watch.	positive	1
2	Oh Beto. ??????? https://t.co/XLJL40BCzQ	positive	1
3	The world would be a much better place if only...	positive	1
4	Basically, how far are we willing to go to ach...	positive	1

```
#raw_data['label'].value_counts()

#use regular expression to replace email addresses, urls, phone numbers, other numbers, symbols
import re

#replace email addresses with 'emailaddr'
raw_data['processed'] = raw_data.text.str.replace('^.+@[^\.].*\.[a-z]{2,}$', 'emailaddr')

#replace urls with 'webaddress'
raw_data.processed = raw_data.processed.str.replace('^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',

#replace money symbols with 'moneysymb'
```

Fig 1: Code snippet for removing digit and special characters used within the tweet (Source: extracted from our code)

ii) Change case
Sequence of letters is said to change case if a given string is upper case than convert it into lower case and vice versa. This is very important in text classification, in order to reduce the ambiguity of character encoding. For instance, Beto is so mediocre. It?s painful to watch.' Convert to 'beto is so mediocre. it?s painful to watch.'

iii) Removal of stop words
The process of converting text data into something a computer can understand is referred to as text preprocessing**.** One of the major forms of text preprocessing is to filter out useless data. In natural language processing, useless words (data), are referred to as stop words which do not make any difference in determining the polarity of a sequence of characters wherein, they are contained. A stop word is a commonly used word (such as *"the", "a", "an", "in"*) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.

iv) Tokenization
In Natural Language Processing, NLP, tokenization must be completed before any processing can occur, this is to split the textual data into smaller units. At a higher level, the text is initially divided into paragraphs and sentences. As a consequence of the length limitation of 140 characters imposed by twitter, it is seldom the case that a tweet will contain more than a paragraph. In these regards, the research aim at this step is to correctly identify sentences. This can be done by interpreting the punctuation marks such as a period mark ".", within the text analyzed. The next step is to extract the words (tokens) from sentences. The challenge at this step is to handle the orthography within a sentence. Consequently, spelling errors have to be corrected, URLs and punctuation shall be excluded from the resulting set of tokens. Then, the result of tokenization phase of text preprocessing is an array containing a set of strings.

v) Text stemming
The aim of stemming is to reduce inflectional forms and derivatives of word to its fundamental form. For instance, "loving", "loved", "lover", "loves" these words will have the same root, which is "love". But there is a consider when using stemming technique in normalizing document, that is, it chops the end of words, so that only the base form is kept.

vi) Text lemmatization
In solving or mitigating the effect of stemming technique when reducing words that have the same base form to their root, we use lemmatization. Lemmatization uses the morphological analysis of the words, and returns their dictionary form (base), this is referred to as "lemma".

3. Features extraction
This is another important aspect in Natural Language Processing (NLP), as the target variable (defined class) depends on the independent variables i.e features. The following are the approach normally use in text features extraction.

i) Vector count
This is a technique used in which every unit word or either n-gram word is been converted into numerical value count matrix since machine learning algorithms take in numerical value(s). Vector count produces a sparse representation of the counts. And to do this, we using Count Vectorizer (also known as One-Hot Encoding) class which is found in feature extraction. text as it is available in python sklearn library. For instance,

	almighty	and	are	around	christ	do	doing	done	everyone	father	...
0	1	0	0	0	0	0	0	0	0	1	...
1	0	0	0	0	1	0	0	0	0	0	...
2	0	0	0	0	0	0	0	0	0	0	...
3	0	1	0	1	0	0	0	1	1	0	...
4	0	1	1	1	0	0	1	0	0	0	...
5	0	0	0	0	0	1	0	0	0	0	...

	me	son	spirit	still	thank	that	those	what	will	you
0	0	0	0	0	0	0	0	0	0	1
1	0	1	0	0	0	0	0	0	0	1
2	0	0	1	0	0	0	0	0	0	1
3	2	0	0	0	1	0	0	1	0	1
4	2	0	0	1	1	1	1	1	0	1
5	1	0	0	0	1	0	0	1	1	2

Fig 2: The Result of Count Vectorizer (Source: extracted from our code)

ii) Term frequency – inverse document frequency (TF-IDF)
This is a statistical technique which compute the importance of a word is to a document in a collection of words or corpus. TFIDF takes the output of Count Vectorizer i.e. for TFIDF to be computed, the text data must be vectorized [conversion of the text data to numerical value]. Furthermore, TF-IDF value increases proportionally as the number of frequents a word occurs in the given document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. Here is the mathematical formula used to compute TF-IDF,

$$\mathrm{tf}(w, d_i) = \frac{count(w)}{\sum_{i=0}^{len(d)} w_i}$$

$$\mathrm{idf}(t_w, D) = \log\left(\frac{count_{total}(w)}{\{d \in D : t \in d\}}\right)$$

$$\mathrm{tfidf}(t_w, d, D) = \mathrm{tf}(w, d_i) * \mathrm{idf}(t_w, D)$$

Where,
w – word, such as, "prejudice"
d_i – document in the word Document
count (w) – return the frequency of times word, w, occurs in d or D as the case may be.
len(d_i) – total number of documents in the whole Document
D – all the documents in a collection or corpus

```
from sklearn.feature_extraction.text import TfidfVectorizer
tf=TfidfVectorizer()
text_tf= tf.fit_transform(raw_data.processed)

print(text_tf)
```

```
  (0, 11650)    0.37741910144087254
  (0, 17748)    0.2798851222729091
  (0, 8754)     0.23321879437196807
  (0, 7770)     0.34082746239032213
  (0, 14342)    0.26199346587811206
  (0, 9439)     0.37741910144087254
  (0, 16196)    0.2654682242592032
  (0, 10196)    0.2402844389154561
  (0, 1985)     0.361227043213742
  (0, 5024)     0.23268275237364155
  (0, 3225)     0.18197546273069365
  (0, 12755)    0.2272173453357487
  (1, 1735)     0.6419720868855717
```

Fig 3: The Result of TF-IDF (Source: extracted from our code)

For sentiment analysis to be performed, data manipulation (text manipulation) is required through processing chain. The detect abusive comment module is capable of integrating and testing the following components:

i) Data Collection
ii) Data preprocessing
iii) Pattern mining
iv) Sentiment Classifier

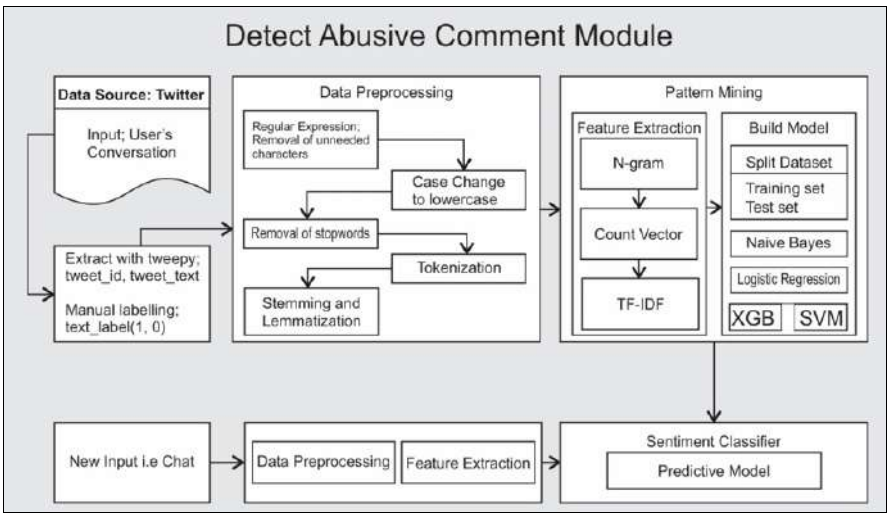

Fig 4: Abusive Comment Detection Model Architecture Design

Baseline results

The table below shows the results gotten from each of the above-mentioned Classification Algorithms using TF-IDF.

Table 1: Model Performance Evaluation.

Algorithm		Precision	Recall	F1 score	support
Naïve Bayes	0	0.97	0.56	0.71	656
	1	0.81	0.99	0.89	1216
Support Vector Machine	0	0.93	0.77	0.84	656
	1	0.88	0.97	0.92	1216
Logistic Regression	0	0.96	0.58	0.72	656
	1	0.81	0.99	0.89	1216
XGBoost	0	0.91	0.26	0.41	656
	1	0.71	0.99	0.83	1216
		Naïve Bayes	**SVM**	**Logit Reg**	**XG Boost**
Accuracy		0.840	0.900	0.843	0.730

For a baseline, we use a simple positive and negative word counting method to assign sentiment to a given tweet. We use the Opinion Dataset of positive and negative words to classify tweets. In cases when the number of positive and negative words are equal, we assign positive sentiment. Using this baseline model, we achieve maximum accuracy obtained in the research after comparing all the four classifiers' accuracies is 0.900.

Conclusion

This research is concerned with the use of Natural Language Processing and Machine Learning algorithm for the extraction of features from users' tweets generated from Twitter, social media in order to detect the abusiveness in the user's comment and filter it beforehand.
Other systems have been built already, but most of these existing systems are expert systems, and are often complex and hard to operate. Furthermore, they are vulnerable, and

not accurate in their prediction when considering large volume of data i.e. users' comments and hence are not so reliable, coupled with the fact that only social media agents operate it periodically.

However, this research has given birth to a novel abusive comment detection model and a system that is more reliable and easier to understand. The system is also an app based.

Table 2: Accuracies and Confusion Matrices

MultinomialNB Accuracy: 0.840277777

		predicted	
		positive	negative
actual	positive	369	287
	negative	12	1204

Support Vector Machine Accuracy: 0.897435

		predicted	
		positive	negative
actual	positive	503	153
	negative	39	1177

Logistic Regression Accuracy: 0.842948

		predicted	
		positive	negative
actual	positive	379	277
	negative	17	1199

XGBoost Accuracy: 0.7323717948717948

		predicted	
		positive	negative
actual	positive	173	483
	negative	18	1198

Contribution

We are enabled to contribute the following in carrying out this research work

i) Create a dataset of 9,360 tweets with 2 columns (text, and label)
ii) Prepare the dataset for training and testing on the Naïve Bayes, SVM, Logistic Regression, and XGBoost classifiers.
iii) Create corresponding annotations for all textual data in the dataset.
iv) Train and test the hate speech detection model on the custom dataset created and prepared for the research.
v) Visualize the network structure of the resulting model.

Future work

Due to not willingness of the judges and the time constraint stipulated for this research work, we could only manually annotate 9,360 tweets out 130,000 mined from Twitter and also some preprocessing techniques which could have enhance the accuracy of our model were left out. However, we intend to keep making improvements on the following:

i) The accuracy of the model gotten in this research work by annotating more of the tweets and also adding more from other social media platforms.
ii) Cluster these tweets into different categories of abusiveness, that is, strong, medium, and weak.
iii) Due to limitation stated above we could not integrate our model into the social mobile app developed which is an aspect of the research, we intend to further work on that aspect and some features which could have made it competitive to currently use social media app were not built in it, so we intend to keep working on it.

References

1. Agarwal U, Data Mining and Data Warehousing SK. Kataria & Sons 2013.
2. Al-Hassan A, Al-Dossari H. Detection of Hate Speech in Social Networks: a Survey on Multilingual Corpus. February 2019;83-100. https://doi.org/10.5121/csit.2019.90208
3. Brown A. What is Hate Speech? Part 1: The Myth of Hate. Law and Philosophy 2017;36(4):419-468. https://doi.org/10.1007/s10982-017-9297-1
4. Gao L, Huang R. Detecting Online Hate Speech Using Context Aware Models. 2015 2017;260-266. https://doi.org/10.26615/978-954-452-049-6_036
5. Ibrohim MO, Budi I. Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter 2019;46-57. https://doi.org/10.18653/v1/w19-3506
6. Malmasi S, Zampieri M. Detecting hate speech in social media. International Conference Recent Advances in Natural Language Processing, RANLP-Septe 2017;467-472. https://doi.org/10.26615/978-954-452-049-6-062
7. De Pelle RP, Moreira VP. Offensive Comments in the Brazilian Web: A Dataset and Daseline Results. 2014;510-519.
8. GP Pokhariyal. Importance of moderating and intervening variables on the relationship between independent and dependent variables. Int J Stat Appl Math 2019;4(5):01-04.
9. Waseem Z. Are You a Racist or Am I Seeing Things? Annotator Influence on Hate Speech Detection on Twitter 2016;138–142. https://doi.org/10.18653/v1/w16-5618
10. Razavi AH, Inkpen D, Uritsky S, Matwin S. (n.d.). Offensive Language Detection Using Multi-level Classification.
11. Raju Sake, P Mohammed Akhtar. Fitting of modified exponential model between rainfall and ground water levels: A case study. Int J Stat Appl Math 2019;4(4):01-06.