A Graph Based Approach for Cyberbullying Classification Using Machine Learning Algorithms

Khadija Khedraoui
LAROSERI Laboratory
Chouaib Doukkali University
El Jadida, Morocco
khadija.khedraoui@gmail.com

Khalid Zine-Dine
Mohammed V University in Rabat
Faculty of Sciences
Rabat (FSR), Morocco
khalid.zinedine@fsr.um5.ac.ma

Abdellah Madani
LAROSERI Laboratory
Chouaib Doukkali University
El Jadida, Morocco
madaniabdellah@gmail.com

2023 14th International Conference on Intelligent Systems: Theories and Applications (SITA) | 979-8-3503-0821-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/SITA60746.2023.10373699

***Abstract*— Using Social Network Services becomes a huge necessary, especially for teenagers. They can get closer in the real time. With more than 237 million daily active users [1], Twitter allows sharing short messages named « tweets ». Some users can employ offensive words, pictures or videos against another person via electronic devices. This phenomenon is calling cyberbullying. Because of timid, fear or weak cyberbullying attacks teens in darkness. Victims suffer in silence and can be driven to lower self-esteem, depression or suicide. In that aims many researches in computer science, sociology and psychiatry take the challenge and many initiatives are developed in order to detect offensive, abusive and negative comments.**

Because of the huge number of users, the tweets make a raw dataset containing a lot of information that must be purified. Some information are misspelled, missing, duplicated or inuseless. Our work is based on Natural Language Processing; we will discuss the importance of the data pre-processing and feature engineering before proceeding to the text classification using Machine Learning Algorithms.

Keywords*— *Twitter, Cyberbullying, Text analysis, Data annotation, Data pre-processing, Feature Engineering.

I. INTRODUCTION

The United Nations International Children's Emergency Fund (UNICEF) defines the cyberbullying as “bullying with the use of digital technologies. It can take place on social media, messaging platforms, gaming platforms and mobile phones. It is repeated behavior, aimed at scaring, angering or shaming those who are targeted”[2]. “The effects can last a long time and affect a person in many ways: 1- Mentally: feeling upset, embarrassed, stupid, even afraid or angry; 2- Emotionally: feeling ashamed or losing interest in the things you love; 3- Physically: tired (loss of sleep), or experiencing symptoms like stomachaches and headaches”[2].

Our motivation for that work is based on the disaster caused by the cyberbullying. L’UNICEF and the UN Special Representative of the Secretary-General (SRSG) on Violence against Children affirms that: “One in three young people in 30 countries said they have been a victim of online bullying, with one in five reporting having skipped school due to cyberbullying and violence” [3]. Against that cybercrime, we could be the first line of defense.

The way we use Twitter as an application domain is that Twitter restricts the number of the characters in one tweet to 280 character [4]. These short documents represent a big challenge because usually the techniques adopted are more performing with long texts.

Few studies focus on the effect of pre-processing method on the performance of Twitter sentiment analysis. This paper is exploring various pre-processing and feature engineering methods in order to get better results. The detection of the bully traces and definition of the polarity of the tweets requires not only an efficient classifier but also a very good dataset. To get a purified dataset we should reduce the irrelevant and noisy information. In addition, we have to correct the imbalancement of the text in question. This work will prove how the dataset treatment can improve the performance of the machine learning algorithms and speed up the classification process.

This paper is organized as follows: section 2 recapitulates works from literature. Section 3 gives definition of what we will evoke in the upcoming sections. Section 4 defines the purpose and gives an overview detailed of the steps followed in our actual work. Section 5 shows results in number with explanation to each result. The last section concludes while specifying future work.

II. RELATED WORK

Cyberbullying detection has powerful backgrounds in literature. Generally, it is a supervised-classification task having as objective the building of smart detection models.

In [5], the authors affirm that building a good learning model does not mean always developing new algorithm. The focus should be on filtering the dataset by adjusting, transforming, extracting or creating new features. The article presents a list of feature engineering techniques; this list is the most popular and successful but can’t be limited. The authors discuss PCA(Principal Component Analysis) which can reduce the data dimensionality by creating new features. A Linear Discriminant Analysis called LDA can find an optimal linear transformation for the data. To evaluate the nodes importance in the networks, centrality measures are used; degree and PageRank algorithms as an example in wide networks. In small networks, betweenness and closeness are used. Otherwise, time series data are treated differently; the time series are transformed in first into a feature-vector representation and this new representation will be fed into a traditional predictive model. We end this summary by recall the feature engineering techniques for unstructured data such as multimedia (audio, video, and image), geospatial, network and text dataset. Different techniques can interpret the text data meaning; Bag-of-Words as traditional approach, which can count the appearance of a word in a document. In order to quantify the importance of a word in a corpus, the TF-IDF can be used as second method. The word embedding models appear to handle the inefficiency of the traditional methods and propose a vector representation of the word, we can respectively talk about and in ascending order: word2vec,

GloVe (Global Vectors for Word Representation), fastText, ELMO and BERT.

In [6], the authors work on detecting cyberbullying from tweets. An existing benchmark dataset was used containing 9093 tweets; the dataset was stored in a CSV file with two columns 'tweet' and 'class'. As the dataset used contain a raw data, pre-processing steps are required to handle this issue. The data pre-processing steps include (noise removal, punctuation removal, lower case converting, tokenization, stemming and detokenization). The cleaned dataset was splitted into train dataset 80% and to test dataset 20%. The dataset imbalancement need a several cross-validation techniques such as K-Fold and Stratified K-Fold to be validated. The authors use Bag of Words and TF-IDF as features extraction techniques. The result obtained was 94% accuracy for SLE (Single Level Ensemble Model) in accordance with TF-IDF ('Word' and 'Unigram') as a features extractor. And for DLE (Double Level Ensemble Model), the accuracy achieves 93% using TF-IDF ('Character'). Moreover, when using cross-validation techniques with SLE, the accuracy attends 96% with TF-IDF ('Unigram') concerning K-fold and Stratified Shuffle Split. As to DLE, the accuracy achieves 96% using TF-IDF ('Word', 'Character', 'Unigram') concerning K-fold and Stratified K-fold. In effect, please refer to the illustration below "Fig.1" to understand SLE and DLE.

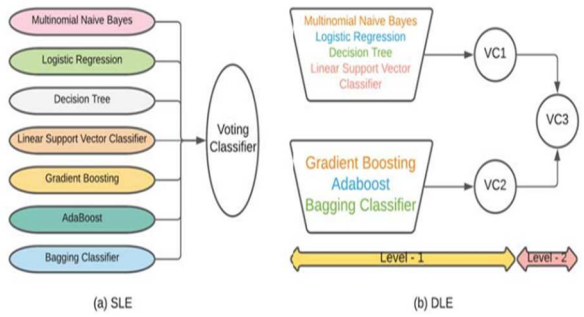

Fig. 1. Figure of proposed SLE and DLE models.

In [7], the authors used the Formspring dataset. The dataset imbalancement was corrected using Adaptive Synthetic ADASYN oversampling algorithm and some pre-processing steps were applied in order to get a cleaned dataset. Four word embedding techniques were established (Formspring word2vec, Glove, Reddit and ELMo). Three deep learning algorithms were experimented achieving a great progress on NLP: GRU, LSTM and Bidirectional LSTM (BLSTM). The authors figure out that ELMO outperforms comparing to the other methods especially when using BLSTM. What makes ELMO a powerful technique is its capacity to give an embedding based on the context where the word is used. Alternatively, the prediction could be improved by having datasets that contain more instances of cyberbullying.

In [8], the authors use machine-learning techniques to detect cyberbullying in social media. The choice of the dataset was from kaggle. The authors proposed an approach based on three main steps: Pre-processing, Features extraction and Classification. In the pre-processing step, the data was cleaned using: Tokenization, Lowering text, Stop words & encoding cleaning and Word Correction. In the second step, the text was transformed into numerical form using TF-IDF and the polarity was detected using Text Blob library. In addition, and for a better evaluation of the model, NGram were used to test the different words combination. In the last step, two classifiers were used: SVM (Support Vector Machine) and Neural Network. With an accuracy of 90.3%, the SVM classifier attend the highest percentage using 4-Gram. An accuracy of 92.8% were achieved using NN and 3-Gram. The average accuracy of all n-gram models of NN achieves 91.76% and the average accuracy of all n-gram models of SVM achieves 89.87%.

In [9], the authors propose a CNN-CB (Convolutional Neural Network-Content Based) algorithm which outperform on the classical machine learning models. The CNN-CB model is made of four layers which are embedding, convolutional, max pooling and dense layers. The proposed model comes with a smart idea; it eliminates the pre-processing steps (feature determination, feature extraction & feature selection) and replace them with a word embedding representation that can be used directly in a convolutional layer. A convolutional layer has, as a main role, the extraction of features by compressing the original and preserving the worthy ones. Then comes the turn of the max-pooling layer; its combination with the convolutional layer makes the CNN algorithm a robust solution within its proficiency to deal with data complexity. A dense or fully connected layers show up as a last step; it provides the dataset shaping. A comparison between CNN-CB and traditional cyberbullying approach (SVM) comes to support the deep learning approach and demonstrates its capacity to provide better predictions. The experiment showed an accuracy of 81% with SVM while it shows 95% after 10 epochs with CNN-CB.

In [10], when looking in the literature, the authors are convinced that deep learning algorithms are rarely used in text classification problems despite on their good reputation against the traditional machine learning techniques. In that case, three deep learning algorithms were established: a simple Neural Network, a hybrid CNN layer followed by LSTM layer and finally a mix of CNN, LSTM and DNN. Three different sources were used: Formspring, Google-News and Twitter with the word embedding technique "word2vec". The two cases balanced and imbalanced dataset were treated. SVM and Logistic Regression were crushed by CNN and C-LSTM. Some hyper-parameters were used to optimize the results.

III. Metrics, Feature Engineering and Global Definitions

In general, the classifiers evaluation is based on some evaluation metrics using confusion matrix. The most famous classification metrics are: Accuracy, Precision, Recall and F-score. Each have its own equation as follow:

$$\boldsymbol{Accuracy} = \frac{TP+TN}{TP+FN+TN+FP} \quad (1)$$

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

$$\boldsymbol{Recall} = \frac{TP}{TP+FN} \quad (3)$$

$$\boldsymbol{F1\ Score} = 2 * \frac{\boldsymbol{Precision * Recall}}{\boldsymbol{Precision + Recall}} \qquad (4)$$

Where TP is the number of True Positive, TN is the number of True Negative, FP is the number of False Positive, and FN is the number of False Negative.

In order to skim well the article and skip all kind of ambiguity; a list of metrics, feature engineering techniques and other global definitions was established.

- Accuracy: is an evaluation metric that calculate the percentage of true predictions. Accuracy measures the number of correct predictions in relation to the general number of predictions. The true predictions are divided by the total of all predictions.
- Precision: is an evaluation metric that calculate the percentage of true positive predictions. Precision measures the number of positive correct predictions in relation to the general number of positive predictions.The true positive predictions are divided by the total of positive predictions.
- Recall: is an evaluation metric that calculate the percentage of true positive predictions. Recall measures the number of positive correct predictions in relation to everything that should have been predicted as positive. The true positive predictions are divided by the total of what is supposed be positive.
- F1 Score or F-measure: is also an evaluation metric helping the good evaluation of the Machine Learning model performance. It is the combination between Precision and Recall into one single score.
- TF-IDF (Term Frequency-Inverse Document Frequency): is numerical statistic approach that calculate the importance of a word in a corpus. This technique gives the importance of a word in a document or corpus by quantify it. TF-IDF is a combination between two parts: TF and IDF.

  The Term Frequency (TF) for a word inside a paragraph is calculated according to the equation (5). The result constitutes a matrix of TF values.

$$TF(word) = \frac{(Number\ of\ times\ term\ appears\ in\ a\ document\ )}{(Total\ number\ of\ terms\ in\ the\ document)} \qquad (5)$$

  The Inverse Document Frequency (IDF) is calculated according to the equation (6). The result constitutes a matrix of IDF values.

$$IDF(word) = log\frac{(Total\ number\ of\ documents)}{(Number\ of\ documents\ with\ term\ in\ it)} \qquad (6)$$

  For each word, its score TF-IDF is the multiplication of TF score and IDF score; the result should be like the equation (7):

$$TF - IDF = TF(word) \times IDF(word) \qquad (7)$$

- Stemming: is reducing a word to its root by removing affixes. The related algorithm is used to extract the base form of a word. Otherwise, cutting down the tree branches to its spindle.
- Stop-words: are a set of commonly used words in a specific language, it could be English or others. Usually used in Natural Language Processing (NLP) and Text Mining, stop-words remove words that gives less information.
- Tokenization: is splitting the original text into small parts (sentences or words). The tokenization result is calling tokens and helps understand the sequence of the words.
- Pre-processing: several mandatory tasks helping preparing the raw dataset and making it ready for a machine-learning model. The purpose is to get a cleaned dataset ready to be implemented.
- Centrality Algorithms: is finding the important nodes within a graph. The important node could have many incoming connections, could be connected to many important nodes, and could attend other nodes with few leaps or being on the shortest way of couple of nodes. In our work, we use PageRank as a special case.
- PageRank: a link analysis algorithm providing a weight to each element in the purpose to measure the weight (importance) within the global set. This weight can determine the node importance in the whole graphical representation. An oriented graph is made by nodes and edges; each edge is oriented towards an other node or the opposite. PageRank algorithm takes into account only the incoming edges; the more a node have more incoming relations the more it is famous and the more it is well noticed. PageRank is a function bringing a solution to the next equation[11]:

$$PR(A) = (1-d) + d(\frac{PR(T1)}{C(T1)} + \cdots + \frac{PR(Tn)}{C(Tn)}) \qquad [11]$$

  Where:

  A is a page; A has pages $T1$ to Tn pointing to it.

  d is a damping factor $\in$ [0,1[. Its score is usually equal to 0.85.

  $C(A)$ is the outgoing links count of a page A.
- Cypher: a query language for graph developed by Neo4j. It is simple as SQL and can collect data from graphs. Cypher is easy to learn and gives a visual representation of the data; by writing a query, you design a graph across your data.
- CSV File: a file format known by its Comma-Separated Values, allowing saving tabular data. Very useful for huge datasets.
- Graph Theory: is the study of graphs; and graphs are mathematical representations who connect a set of points using connections called edges. It is very useful for creating relationships in pairwise between objects.
- Neo4j: is a graph database management system; its data elements are stored in nodes and have connections called edges. Each node have some properties such as Id and Label. The scheme in "Fig.2" illustrates Neo4j data structure [12].

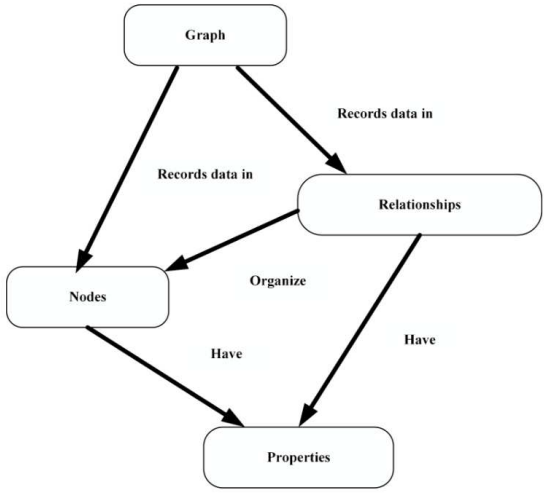

Fig. 2. The Data Structure of Neo4j.

- PorterStemmer: it is reducing a word to its root. There is actually many types of stemmers but the most popular one is PorterStemmer. It is famous regarding its simplicity and speediness; giving then a best result with less error rate.
- Word Embedding: is a word/text representation under a vector form. It helps catching the semantic of o word in the whole corpus. Some techniques provide the same word representation for the word even if the word have different meaning according to the sentence. Instead, many word embedding techniques have the potential to represent the word differently according to its signification in the corpus.
- Numpy: a mandatory scientific package for any type of mathematical operation. Used also for add arrays and matrices with a large dimensionality.
- Matplotlib: this library need to be supported with a sub-library "pyplot". As a 2D plotting library, it can plot any type of charts in python.
- Pandas: an open-source data manipulation library, useful for managing and importing datasets in data science and machine learning.
- Nltk: a source of text processing libraries; responsible for stemming, tokenization, classification and semantic tasks.
- Sklearn: an open-source, simple and efficient library for data analysis. It provides regression, classification, model selection, pre-processing and clustering. It was built on NumPy, SciPy, and matplotlib.
- SMOTE (Synthetic Minority Oversampling Technique): is a machine learning technique aims to fix the imbalancement of the dataset by generating new instances. The algorithm works by generating new samples from existing minority ones.

  Having an element Xi, we try to have its copy using k nearest-neighbors algorithm. The idea is looking into Xi nearest samples inside the blue circle where we can see three points "Fig.3"; one of those neighbors Xzi is selected and $Xnew$ is produced following the equation:

$$Xnew = Xi + \lambda * (Xzi - Xi) \quad (8)$$

Where λ is a randomize number between 0 and 1.

Be informed that SMOTE should not be applicable on a full dataset. SMOTE shall be used on the training part after splitting the dataset of course. Normally, SMOTE have to produce better result with your model; if it is not the case you have to review your model.

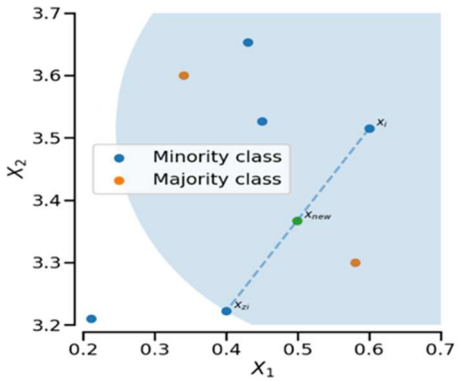

Fig. 3. Xnew creation illustration using SMOTE method.

IV. Our Approach

In this article, we have two primary goals: the first one is proving the importance of pre-processing techniques on classification problems and the second goal is developing an efficient detection approach using a centrality algorithm. For this, we developed two methods: the first one by using TF-IDF and the second one by using PageRank. Keeping a good computational cost and time to the minimum is a mandatory need.

In first at all, we choose a Boolean annotated dataset from Kaggle website containing 60K Tweets [13]. The tweets are annotated as "0" for the bully tweets and "1" for the non-bullying category. The dataset is in CSV form.

Some predefined python libraries are required to perform data pre-processing and doing some specific jobs; it is a good help during the pre-processing process. In our work, we use multiple libraries as: Numpy, Matplotlib, Pandas, Nltk and Sklearn.

Our dataset is raw and noisy, contain a plenty of unuseless information and can seriously damage the efficiency of our classification model by taking more execution time, more extra memory space and could drive to misunderstanding of the real meanings. A pre-processing techniques have been used in order to get a cleaned dataset. The pre-processing workflow includes removing additional information (digits, links, symbols, duplication and whitespaces). It includes also the transformation of the document into lower case for more text consistency. The removable of stop words is also mandatory since words tokenization outperform and gives good results. The overall pre-processing steps are shown in "Fig.4".

For more details, stop words were removed using Nltk and Sklearn libraries; it was tested that a lonely one was not enough to remove all stop words. Also, and in order to

decrease the types of words, we convert the tokens to their root form using "PorterStemmer". Actually, there is multiple techniques for stemming/lemmatization; the choice of PorterStemmer was beneficial in our case.

According to our actual experience, the Untokenized step is very important in our pre-processing workflow. Well, the text in the beginning is a corpus of sentences. After purifying the tweets they were transformed to tokens. We realise that if we work with tokens, the TF-IDF is not well calculated. Contrariwise, when we untokenize the final result it is much more better and gives outstanding outcomes.

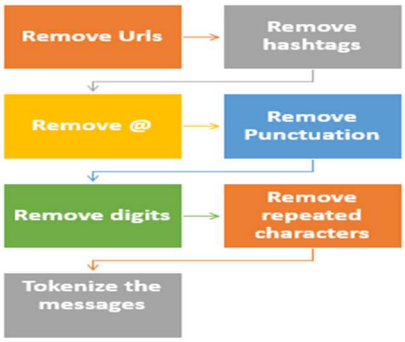

Fig. 4. The Pre-processing steps.

Our Dataset is readable and cleaned from any kind of noises. It is ready to be trained, evaluate and at last, we can conclude what kind of machine-learning model is more adopted for our classification problem.

A. *First method*

The first method is using the TF-IDF algorithm with its parameter "max_features". The purpose here is to trust the TF-IDF logic on choosing the best 5 000 features. The "best" in our work must have a good weight in the dataset. Otherwise, the first important features are selected and will be trained. The whole process is illustared in the workflow of the "Fig.5".

Many word-embedding techniques are already discussed in the literature. Previous literature researches found that TF-IDF is a useful and a powerful algorithm in text analysis and classification concern. The use of TF-IDF to extract features in term of unigram is a best solution in our work. The idea behind using TF-IDF is to give a weight to each word while respecting the document; the value of a word increases proportionally after each count. Each word is transformed to word vector and filled in a data frame. The parameter max_features of the TfidfVectorizer, a powerful tool from sklearn library, is a sort of feature selection where some features are selected ; in our actual work only 5 000 were needed.

In order to demonstrate the importance of working with a balanced dataset, we train the both cases. Well, the imbalancement of the dataset can influence the good training of the data; the result could be driven on behalf of the dominate category. We can realize that the number of the non-bullying tweets is much more important than the bullying ones. Well, our dataset should be balanced in order to give acceptable performance. Balancing the dataset can be done using one of the proposed models:

- Under sampling majority class: that means that if you have a group of 90 and a group of 10 you will keep only 10 of the majority data and we train it with the minority data. This method have a big inconvenient is that we throw much data and we could be conducted to create a not efficient model.
- Over sampling minority class by duplication: that means that we generate new samples from current sample "the minority class" by duplicating them. The inconvenient of that method is that the simple copying of the tweets cannot help the built of a good model.
- Over sampling minority class using SMOTE (the method adopted in our work): that method generates synthetic examples using K nearest neighbors algorithm in order to get the same number in both categories.

Finally, the extracted features are fed into multiple classification algorithms to be trained. The training Dataset is used to fit the machine-learning model with a ratio of 80% and the test Dataset is used to evaluate the fit machine-learning model with a ratio of 20%. We used three classifiers Logistic Regression, Naïve Bayes and Random Forest to measure the performance of each model.

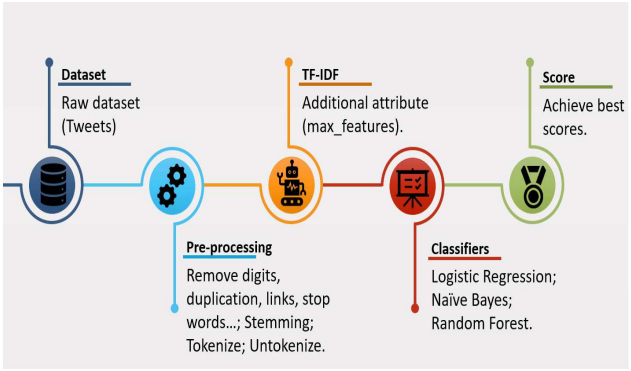

Fig. 5. The first classification method using TF-IDF.

B. *Second method*

The second method is using PageRank algorithm by giving a numerical weight to each feature. Since PageRank belongs to a mathematical algorithm based on the webgraph, it would be very convenient to use Neo4j as a graph database management. The idea behind the representation of the dataset under a graphical network is to help the quick analysis of a big quantity of information. The good understanding of data helps making good classification. Graphical visualization can easily illustrate relationships between items; it's actually a robust solution in big data analytics. The whole process of the second methods is illustared in the workflow of the "Fig.8".

The concept now is by extracting tokens from the cleaned dataset and feed them in Neo4j. A graph of nodes=tokens will be implemented; each node is connected to some nodes if they have a relationship. The connection between nodes is called "edge". In our work, a CSV file was extracted from the cleaned dataset. Once we obtain the bi-gram list and fill it in a CSV file we import it into Neo4j environment. We drew a graph using cypher query. If a node "A" has a relationship with a node "B" they will be represented as two circles joined by an arrow from "A" to "B"; "B" is then the successor of "A". The Graph result is oriented, contains 34061 nodes and

257547 edges. Due to the huge number of nodes and edges, Neo4j cannot represent the whole dataset. A small projection is required, in that case, as shown in “Fig.6”.

In Neo4j Graph Data Science (GDS) some centrality algorithms are already implemented, include PageRank. A PageRank score was affected to each node using cypher language.

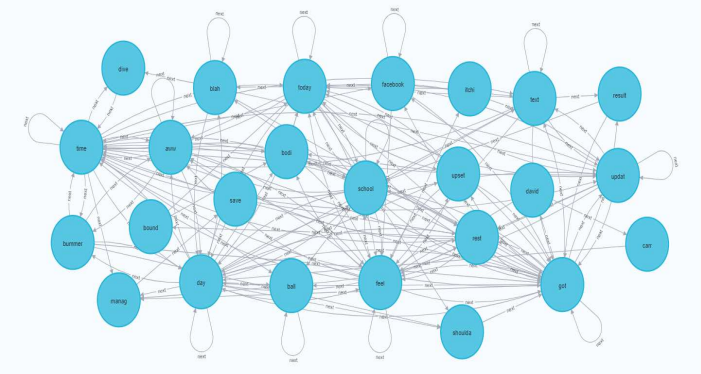

Fig. 6. The Dataset illustration in Neo4j (reduced projection).

The word “GOT”, as an example in the graph, have its score PageRank conforming to its importance in the graph. PageRank calculates the importance in accordance with the incoming relationships from other nodes in the same graph. The “Fig.7” gives an overview of token properties in Neo4j.

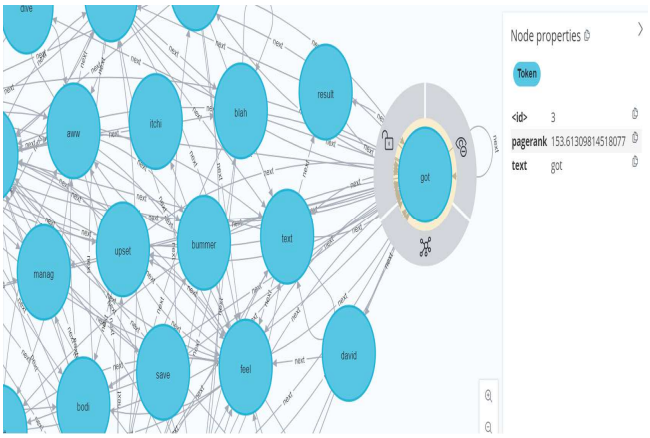

Fig. 7. The Dataset in Neo4j & PageRank Score.

Such as in TF-IDF method, when only 5000 tokens were selected, also with PageRank method we will choose the first 5000 best-classified items. The file was exported in Jupyter environment and the same steps as in TF-IDF method were applied; we train 80% of the set and the test is made on the 20% left. The same classifiers are used to calculate the efficiency of each one.

Obviously, we will train the model with balanced and imbalanced dataset to see the difference.

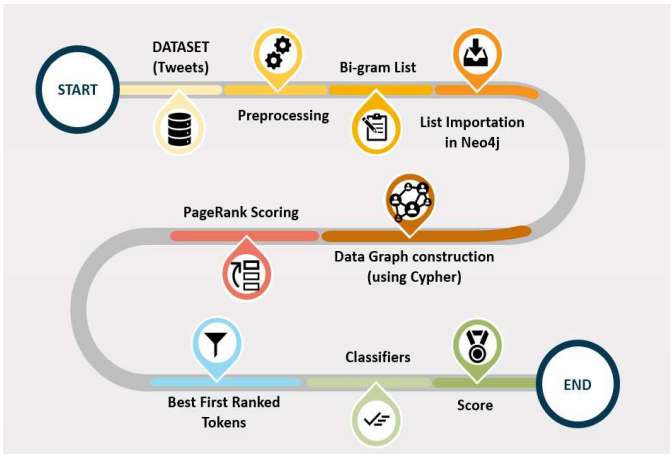

Fig. 8. The second classification method using PageRank.

V. Experiment Results and Discussion

Text classification helps mainly on how to automatically categorize words in a corpus using machine learning algorithms and good techniques. There is no better method; it depends on how you manage your data and how to choose your better classifier. A good choice can easily add value to your analytics.

A. Dataset Description

Cyberbullying is widespread in the last years. Unfortunately, this is a bad phenomenon having a negative impact on persons especially teenager. Various datasets exist on multiple media websites, we pick up one containing tweets including offensive words. The dataset we used was collected from the Kaggle website and contain almost 60K tweets. The dataset is made as a table of two columns: the first one is dedicated to messages “tweets” and the second one for the label. The label of the tweet is a Boolean tag; it is either “0” or “1”. “0” means that the message belongs to bullying category and “1” must refer to non-bullying category. The table I shows the number of the tweets before and after balancing.

TABLE I. Tweet Dataset Details.

	Before balancing	After balancing
Bullying (0)	6 133	53 574
No-Bullying (1)	53 574	53 574

Fig. 9. The dataset details before and after balancing.

Usually, in order to use the dataset it should be a CSV file. It may be in another format as HTML or xlsx.

B. Results of the First Method “Fig.10”

The cleaned dataset has been split into two groups. The features matrix is divided into random train and test subsets. Train set (80%) will be input as training samples and Test set (20%) will be considered as testing samples.

As mention before, we built a model based on TF-IDF algorithm having as objective the good classification of a dataset containing bullying and non-bullying messages. Table II “Fig.10” gives an overview of what we got as results, which are very inconsistent. As first two primary remarks, we note: the precision, recall and F1-score show very bad results when using an imbalanced dataset. The second remark is regading

the accuracy; well that one gives good scores even with imbalanced dataset.

Well, we got a very good accuracy whether the dataset was balanced or not. A score of 98% was achieved using Random Forest classifier. Accuracy can be considered delucive when using as evaluation metric on an imbalanced dataset. The accuracy cannot evaluate the performance of the model when it is about many cases:

- Dataset imbalancement (as in our work): our dataset is divided into two categories but one category is dominant with a pourcentage of 90%. The prevailing class drives the result on its favor while ignoring the minority population by giving a bad performance.
- Misclassification costs: the misclassification should not be treated equally, some errors are more critical than others. Using accuracy could impact the model's prediction negatively and not reflect the reality.
- Outliers: the accuracy cannot handl the outliers due to their deviation beside the other dataset values. That issue drives to a bad performance evaluation.
- Overfitting: the ability of well detecting the training data and failing on the test ones have a negative impact. The model then can achieve a good accuracy ignoring the specifity of the overfitting fact.

In such cases, focusing solely on accuracy might not reflect the true impact of the model's predictions. This is why and to exceed this limitation, it is useful to use other adequate evaluation metrics as precision, recall and F1-score. Those metrics consider multiples factors that accuracy alone might ignores.

Regarding the metrics values, except accuracy, the results are very bad when it's about working with an imbalanced dataset. We got a best precision of 50,6% using Random Forest, a recall of 72,0% using Naïve Bayes and a F1-score of 64,5% using Naïve Bayes.

Otherwise, we got a very good results when the dataset is balanced. A precision and recall of 93,5% were achieved using Logistic Regression. A recall of 89,0% was achieved using Naïve Bayes. And at last, a precision and recall of 97,5% were achieved using Random Forest.

We conclude then that the dataset balancement influences positively the performance of the machine-learning model. Well, when working with a dataset non balanced we get scores reflecting a poor detection of the bullying tokens. The reason is that the major population hide the minority one and dominates at last. The use of an algorithm to balance the dataset is the best solution. When the dataset is balanced all elements have the same chance to arise. In consequence, the models are well trained and predict well the future results.

Without any doubt, there is a big difference between scores when working on balanced dataset and the opposite case.

C. Results of the Second Method "Fig.11"

In the second method, we built a model based on PageRank algorithm having as objective the good classification of a dataset containing bullying and non-bullying messages. Table III "Fig.11" gives an overview of what we got as results.

As in TF-IDF algorithm, the accuracy shows good results with balanced and imbalanced dataset. A score of 98% was achieved using Random Forest.

For the same reasons as before, and when working with imbalanced dataset, we got a precision of 50,2% using Random Forest, a recall of 84,1% using Naïve Bayes and a F1-score of 79,1% using Naïve Bayes.

Otherwise, we got a very good results when the dataset is balanced. A precision and recall of 98,0% were achieved using Random Forest. And at last, a F1-score of 98% was achieved using Random Forest.

At last, a comparison between the second method and the first one is necessary. Getting the best results using centrality algorithms is a big success for us. Well, using the PageRank algorithm as a graph approach gives the best result with the Random Forest method comparing to the result with TF-IDF.

The representation of a dataset under a graph gives more chances to give a best visualization of the components and their weight. The more a token have a great weight the more it's well qualified in the graph. The most it's famous in the graph, the most a centrality algorithm as PageRank can easily catch it and present it as a wanted token.

Our approach gives the best result as hoped from the beginning.

VI. Conclusion and Future Work

In this paper, we proposed a solution to detect cyberbullying in Twitter as a social media network. We made a comparison between two classification methods: TF-IDF and PageRank. Each one use its proper logic in order to classify the inputs; the attribute max_features vs PageRank scoring gives us an overview about classification techniques.

Of course, we faced many issues; we used a raw dataset that should be purified by using pre-processing techniques. Also we had to fix the dataset imbalancement to get the best performance.

As a conclusion, the best a dataset is well cleaned and organized the easy we can do better detection and classification.

In our future work, deep learning techniques will be established in order to make a comparison between machine learning and deep learning algorithms.

TABLE II. TF-IDF APPROACH PERFORMANCE RESULTS.

First Method (TF-IDF)	Classifier	Tweets Category	Total Number	Accuracy (%)	Precision (%)	Recall (%)	F1-score
Imbalanced Dataset	Logistic Regression	1	53 574	0.96	0.498	0.603	0.558
		0	6 133				
	Naïve Bayes	1	53 574	0.93	0.487	0.720	0.645
		0	6 133				
	Random Forest	1	53 574	0.98	0.506	0.535	0.521
		0	6 133				
Balanced Dataset	Logistic Regression	1	53 574	0.93	0.935	0.935	0.930
		0	53 574				
	Naïve Bayes	1	53 574	0.89	0.890	0.890	0.890
		0	53 574				
	Random Forest	1	53 574	0.97	0.975	0.975	0.970
		0	53 574				

Fig. 10. Machine Learning Performance Results Using TF-IDF.

TABLE III. PAGERANK APPROACH PERFORMANCE RESULTS.

Second Method (PageRank)	Classifier	Tweets Category	Total Number	Accuracy (%)	Precision (%)	Recall (%)	F1-score
Imbalanced Dataset	Logistic Regression	1	53 574	0.95	0.496	0.620	0.567
		0	6 133				
	Naïve Bayes	1	53 574	0.91	0.487	0.841	0.791
		0	6 133				
	Random Forest	1	53 574	0.98	0.502	0.532	0.517
		0	6 133				
Balanced Dataset	Logistic Regression	1	53 574	0.95	0.955	0.955	0.955
		0	53 574				
	Naïve Bayes	1	53 574	0.93	0.975	0.930	0.925
		0	53 574				
	Random Forest	1	53 574	0.98	0.980	0.980	0.980
		0	53 574				

Fig. 11. Machine Learning Performance Results Using PageRank.

REFERENCES

[1] https://www.statista.com/statistics/970920/monetizable-daily-active-twitter-users-worldwide/

[2] https://www.unicef.org/end-violence/how-to-stop-cyberbullying

[3] https://www.unicef.org/press-releases/unicef-poll-more-third-young-people-30-countries-report-being-victim-online-bullying

[4] https://developer.twitter.com/en/docs/counting-characters#:~:text=In%20most%20cases%2C%20the%20text,as%20more%20than%20one%20character.

[5] Verdonck, T., Baesens, B., Óskarsdóttir, M., Broucke, S., Special issue on feature engineering editorial. Mach Learn (2021). https://doi.org/10.1007/s10994-021-06042-2

[6] Alam, K.S., Bhowmik, S., Prosun, P.R.K. Cyberbullying detection: An ensemble based machine learning approach, Proceedings of the 3rd International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, ICICV 2021, 2021, pp. 710–715, 9388499.

[7] Al-Hashedi, M., Soon, L.-K., Goh, H.-N., Cyberbullying Detection Using Deep Learning and Word Embeddings: An Empirical Study, ACM International Conference Proceeding Series, 2019, pp. 17–21.

[8] John Hani1 , Mohamed Nashaat2 , Mostafa Ahmed3 , Zeyad Emad4 , Eslam Amer5, Ammar Mohammed, Social Media Cyberbullying Detection using Machine Learning, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 10, No. 5, 2019.

[9] Al-Ajlan, M.A., Ykhlef, M., Deep learning algorithm for cyberbullying detection, International Journal of Advanced Computer Science and Applicationsthis link is disabled, 2018, 9(9), pp. 199–205.

[10] Rosa, H., Matos, D., Ribeiro, R., Coheur, L., Carvalho, J.P., A 'Deeper' Look at Detecting Cyberbullying in Social Networks, Proceedings of the International Joint Conference on Neural Networks, 2018, 2018-July, 8489211.

[11] https://neo4j.com/docs/graph-data-science/current/algorithms/page-rank/

[12] https://journals.plos.org/plosone/article/figure?id=10.1371/journal.pone.0207595.g003

[13] https://www.kaggle.com/datasets/syedabbasraza/suspicious-tweets