The Impact of Pre-processing and Feature Extraction on Tweet classification

Karamjot Singh
Registration No.- 2006510
Computer Science and Electronic Engineering
University of Essex
Colchester, Essex
ks20809@essex.ac.uk

***Abstract*—Numerous research in NLP (Natural Language Processing) focuses on identifying techniques to extract meaningful information from raw text data. Tweet classification poses a major challenge because of the informal nature of the text in tweets, and the lack of methods to properly model such sentences. The goal of this paper is to thoroughly investigate the influence of preprocessing and feature extraction on tweet classification using Logistic Regression as a baseline model. The impact of different combinations of techniques was analyzed in the paper and discovered that there is no fixed strategy that would guarantee an optimal classification on every task. Three benchmark datasets were utilized in the paper where stemming and stopwords were the two preprocessing techniques that enabled a significant improvement. Our experimental results also show that in comparison to not using any data sampling, random oversampling improves classification performance. Furthermore, feature extraction methods such as Glove, TF-IDF and Bag-of-Words based features on word-word co-occurrence, word relevance and word count respectively, revealed their significance on different datasets. It is evident from the findings that, depending on the role and the domain, various combinations of pre-processing and feature extraction should be considered.**

***Index Terms*—Logistic Regression; Glove; TF-IDF; Bag-of-Words; Word2Vec**

I. Introduction

Twitter is a gold mine of text data that presents both an opportunity, and a challenge; the opportunity to analyse numerous exploratory and predictive tasks, whereas the challenge to mine (extract information) such a massive volume of data. Tweets are one of the key sources for the government and organizations to monitor public sentiments towards them. In general, tweets can be classified based on topic, sentiment and intent; however, this is usually not straightforward as they are high dimensional, unstructured and turbulent, making it a demanding task. Due to the limited amount of contextual cues available in Twitter data, the process of classification demands multiple pre-processing and feature extraction techniques [1]. Therefore, the paper investigates the influence of these techniques on Tweet classification.

Three TweetEval datasets are utilized and processed using combinations of pre-processing (tokenization, stemming, lemmatization removing stopwords) and feature extraction (word-embeddings) methods with a baseline model to assess the performance. TweetEval is a standardized test bed for seven tweet classification tasks [1]. These are sentiment analysis, emotion recognition, offensive language detection, hate speech detection, stance prediction, emoji prediction, and irony detection [1]. For the scope of this paper, we have considered three classification tasks: (i) sentiment analysis, (ii) hate speech detection, (iii) offensive language detection.

The motivation behind this work is to discover the finest set of processing techniques for the datasets to achieve a higher score in the Leaderboard of TweetEval.

II. Literature Review

In paper [2], the importance of domain-specific stop words is analyzed. Tweets are full of stopwords (I, you, him, no, not) which may affect the negation of the sentence and lead to wrong entailment decisions. In this paper [3], Logistic Regression as a baseline approach is considered in natural language processing due to its close relationship with neural networks. Thus the classifier is used as a baseline model in our paper to compare the experiments. In this paper [4], the authors have compared feature extraction such as Bag of Words, TF-IDF, word embedding on the SS-Tweet dataset and found TF-IDF using N-gram features the most optimal by considering F-Score, Accuracy, Precision and Recall performance parameters and concluded the importance of TF-IDF (unigram) as compared to N-gram. Authors in this paper [5] suggested that the combination of two as compared to three or more pre-processing techniques enhance the performance of the classification, for instance specifically using lemmatization and punctuation splitting, lemmatization and lowering, and lowering and punctuation giving them positive effects. In the paper [6], the authors conducted experiments using different word embeddings and deep learning architectures and found Glove embedding the best on the CrisisNLP dataset. They have fine-tuned the Glove by transferring the knowledge from the initial corpus where the embedding was built to the domain dataset.

Analyzing all the above researches gave us a basis to explore it further into selecting different processing techniques for our problem.

III. Methodology

The section outlined the path that was used in conducting the research. The proposed path explains the preprocessing and feature extraction using Logistic Regression for Baseline Approach. The major purpose of this section is to decide the optimal parameters for the final model. It consists of the following modules:

A. Loading Dataset

Three labelled datasets of different categories were selected from [7]: (i) sentiment, (ii) offensive, (iii) hate containing 2 columns i.e. tweet and label. Each dataset is provided with a Train, Validation Test set. As can be seen in the following pie charts, the distribution of the label class is imbalanced.

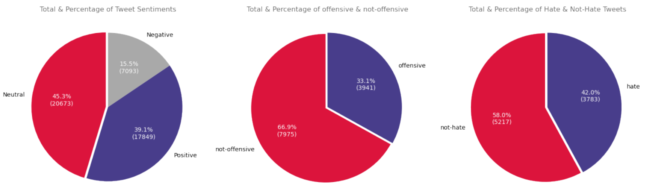


Fig. 1. Distribution of Train Datasets

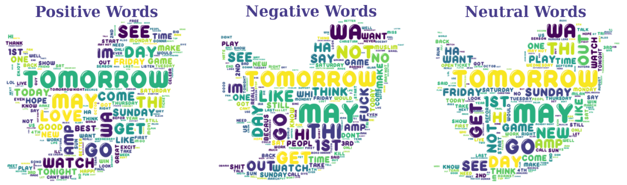


Fig. 2. These are WordClouds to check the frequency of the words, an example of a visualization technique used for the paper which can be seen in the code

B. Pre-Processing

Before beginning the analysis, some data preprocessing is required as tweets carry noises that have to be removed such as hashtags ‘’, link ‘@’, ‘RT’ (retweets), punctuations numbers. Following are the numerous preprocessing techniques that were applied to the text without changing its context:

- Normalization - It includes the removal of the punctuations, numbers and converting the text to lowercase. This process maintains homogeneity in each text.
- Remove stopwords - These are the common words that don’t add any meaning to the sentences. Two approaches to stop words were used in the paper: (i) stop words from nltk (ii) stop words without negative words.
- Lemmatization - This is a process of converting text to their base form such as words like (works, working worked) to work. It reduces the dimensionality of the corpus which increases the computational time of the model training.
- Stemming - It is similar to lemmatization but unlike changing the words to base form its converts to their stem form for instance words like (works, working worked) to wor. It also reduces the dimensionality of the corpus.

TABLE I
Size of train dataset before and after sampling

Task	***Label***	***Before***	***After***
Sentiment	3	45615	62019
Hate	2	9000	10434
Offensive	2	11916	15950

Apart from text cleaning, the imbalance dataset was handled by Random Oversampling so that the classifier will not bias towards the majority class that can be seen in the Table 1.

C. Feature Extraction

Once the preprocessing was done, we identified and extracted features from the tweets, which is a requirement for the given classification problem, as it is directly proportional to the accuracy. We used word embedding as a feature extraction strategy for converting raw texts to a vector representation of words. Following are the word embedding techniques that we applied on a baseline model:

- **CountVectorizer** is one of the easiest techniques, which transforms the text into a bag of words such that the frequency of words in a sentence is taken into account. While sequence and grammar are lost, the frequency of the words remains unchanged.
- **TF-IDF** stands for term frequency-inverse document frequency is an information retrieval technique that can be used to determine the importance of words in sentences in relation to their context. This method can also be viewed as a form of the Bag of Words model since it does not take grammar or order into consideration.
- **N-Gram** - is a technique of combining words as features for supervised machine learning algorithms. These are n sequence of words from the text. For instance, a sentence like - “Colchester is the northeast of London” and for n = 3, the features will be “Colchester is the”, “is the northeast”, “the northeast of” and so on.
- **Glove** stands for global vectors for word representation. It is an unsupervised learning algorithm for obtaining vector representations for words. The embedding is a well recognised pre-trained word embedding developed by the authors [6]. It is a freely accessible 100-dimensional embedding that is similar to social media texts such as tweets [8]. Glove embedding works on transfer learning which represents transferring the built-in knowledge to the domain dataset.
- **Word2Vec** is a two-layer neural net that processes text by “vectorizing” words. It takes a text corpus as an input and outputs a set of vectors. The purpose and usefulness of Word2vec is to group the vectors of similar words in vector-space [9]. It works on the principle of cosine similarity.

D. Baseline Model

In natural language processing, logistic regression is the baseline supervised machine learning algorithm for classification, and also has a very close relationship with neural networks [3]. Furthermore, in addition to its use as a classifier, logistic regression in NLP and many other fields is widely used as an analytic tool for testing hypotheses about the effect of various explanatory variables [3]. It uses the softmax function to compute probabilities. We used it as a baseline model to discover the behaviour of the above mentioned preprocessing and feature extraction techniques on the provided datasets.

IV. Experiments and Evaluation

We conducted 3 experiments using different word embeddings on three datasets using Logistic Regression. The performance of the models was measured in terms of macro-average Recall for sentiment and macro-average F1 for Hate and Offensive. For the frequency-based embedding such as CountVectorizer and TFIDF, different combinations of unigram, bigram and trigram were used but only the maximum score was given in the tables. Word2Vec was used by a default vector size and a minimum word count of 350 and 2 respectively. We have fine-tuned Glove on our training, validation and test datasets to extract features from them and then implemented the baseline model. The experiments were shown and described in the results section with the tables.

V. Results

In this section, we report the experimental findings of the performance of the classifier. The goal is to evaluate and compare the pre-processing and feature extraction techniques with the baseline model. Table 2 shows the results with no pre-processing.

TABLE II
Results without any pre-processing on imbalanced dataset

Word Embeddings	***Sentiment***	***Hate***	***Offensive***
CountVec	0.527 Bigram	0.42 Bigram	0.70 Unigram
TF-IDF	0.547 Bigram	0.41 Unigram	0.67 Unigram
Word2Vec	0.47	0.35	0.634
Glove	0.52	0.46	0.610

Table 3 shows the results on the balanced dataset with removed stop words and lemmatization. After implementing oversampling, there was a drastic change in performance.

TABLE III
Results with removed stopwords and lemmatization on balanced dataset

Word Embedding	***Sentiment***	***Hate***	***Offensive***
CountVec	0.576 Unigram	0.47 Bigram	0.726 Trigram
TF-IDF	0.595 Trigram	0.442 Unigram	0.713 Unigram
Word2Vec	0.537	0.40	0.64
Glove	0.572	0.49	0.67

TABLE IV
Results with removed stopwords and lemmatization on balanced dataset

Word Embedding	***Sentiment***	***Hate***	***Offensive***
CountVec	0.589 Bigram	0.484 Unigram	0.74 Unigram
TF-IDF	0.618 Trigram	0.454 Unigram	0.746 Trigram
Word2Vec	0.539	0.405	0.69
Glove	0.568	0.54	0.66

Table 4 shows the results on the balanced dataset with removed stop words (excluding negative words) and stemming. The negative words (“not”, “won’t”) from the stopword list of nltk library were ignored in order to maintain the negation of the tweets.

The best accuracy result was achieved by the Glove on the Hate dataset, and for the sentiment and Offensive dataset, TF-IDF with Trigram outperforms all other embedding methods. These results are with balanced training datasets, selected stop words and stemming as pre-processing. It is clear from the results that different combinations of pre-processing and feature extraction should be considered depending on the task and the domain as it can abruptly affect the classification.

VI. Conclusion and Future Work

This work contributes to the importance of the combination of multiple pre-processing and feature extraction techniques in terms of accuracy. We discovered significant improvements in all three datasets relative to findings in Table 2. The extensive experimental analysis revealed that negation handling plays an important part in Tweet classification. Moreover, we concluded that on the smallest dataset i.e. the hate speech dataset the Glove embedding outperforms all other embeddings because of its transfer learning ability. The results indicate the potential of transfer learning models and because of that Transformer-based models will be considered for future work to beat the accuracy of TweetEval on the given datasets.

References

[1] F. Barbieri, J. Camacho-Collados, L. Neves, and L. Espinosa-Anke, “TWEETEVAL: Unified Benchmark and Comparative Evaluation for Tweet Classification

[2] V. Parisi Baradad and A.-M. Mugabushaka, “Corpus Specific Stop Words to Improve the Textual Analysis in Scientometrics

[3] J. Daniel and J. Martin, “Speech and Language Processing”.

[4] R. Ahuja, A. Chug, S. Kohli, S. Gupta, and P. Ahuja, “The Impact of Features Extraction on the Sentiment Analysis,” Procedia Computer Science, vol. 152

[5] H. Woo, J. Kim, and W. Lee, “Validation of Text Data Preprocessing Using a Neural Network Model,” Mathematical Problems in Engineering, vol. 2020.

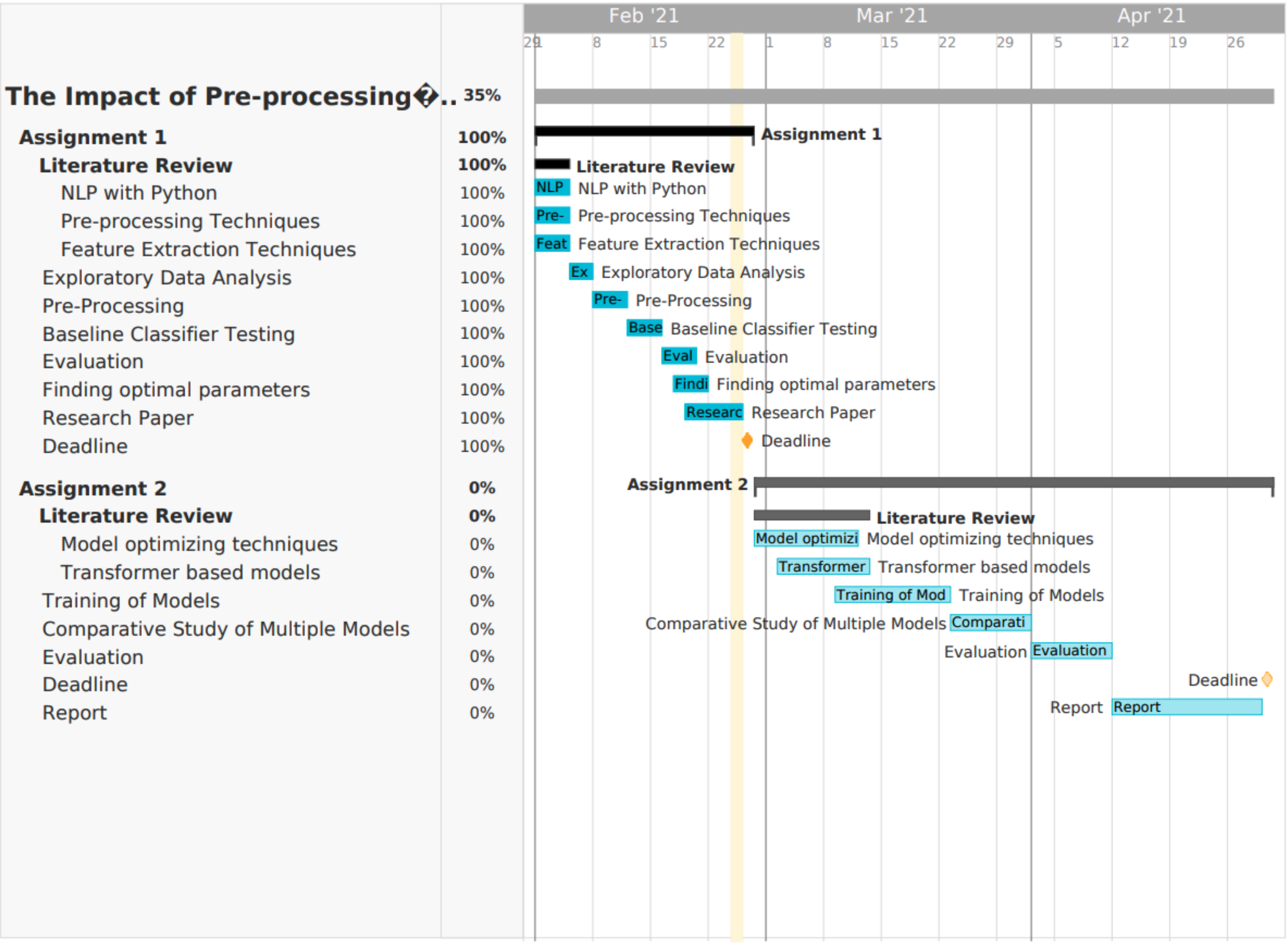
[6] J. Pennington, R. Socher, and C. Manning, “GloVe: Global Vectors for Word Representation,” Association for Computational Linguistics, 2014.

[7] cardiffnlp, “cardiffnlp/tweeteval,” GitHub, Dec. 17, 2020. https://github.com/cardiffnlp/tweeteval/tree/main/datasets

[8] R. Alrashdi and S. O’keefe, “Deep Learning and Word Embeddings for Tweet Classification for Crisis Response,.”

[9] Munesh Lakhey, “Word2vec Made Easy - Towards Data Science,” Medium, Apr. 16, 2019. https://towardsdatascience.com/word2vec-made-easy-

Planning of the Project (Gantt Chart)



The Impact of Pre-processing�..	**35%**
Assignment 1	**100%**
Literature Review	**100%**
NLP with Python	100%
Pre-processing Techniques	100%
Feature Extraction Techniques	100%
Exploratory Data Analysis	100%
Pre-Processing	100%
Baseline Classifier Testing	100%
Evaluation	100%
Finding optimal parameters	100%
Research Paper	100%
Deadline	100%
Assignment 2	**0%**
Literature Review	**0%**
Model optimizing techniques	0%
Transformer based models	0%
Training of Models	0%
Comparative Study of Multiple Models	0%
Evaluation	0%
Deadline	0%
Report	0%