ISSN 1870-4069

Feature Analysis for Stress Detection on Text Posts

Erick Barrios-González

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación,
Mexico

erick.barrios@alumno.buap.mx

Abstract. This paper examines stress detection in social networks, specifically focusing on the Dreaddit corpus. The study utilizes a Naive Bayes classifier with tagging from the Spacy tool. Two grid searches are performed to identify optimal features for the classifier. The evaluation of results using the F_1 metric shows superior performance compared to other Naive Bayes models. Features based on n-grams, POS tagging, lemma, and stem were analyzed. Useful features were found from an approach where their frequency of occurrence in the corpus was evaluated, and, likewise, other features were discarded.

Keywords: Stress detection, naive bayes, N-grams.

1 Introduction

Stress is defined as the reaction to pressures, existing demands, and future demands [1]. It is the natural response of the human being to situations of fear, tension, or danger [8]. Excessive stress can be harmful to the mind and body. Stress is a normal part of our lives, and in small amounts, it can have positive effects. However, excessive stress can cause negative alterations in our organism and mind [9], making the individual prone to physical and psychological illnesses.

Every day, social media networks are becoming more common in our daily lives, and it is increasingly normal for people to continuously turn to social media platforms like Twitter and Reddit to share their feelings and express their stress. This interest in sharing feelings on social media is the main reason why analyzing texts posted on social media is useful for stress detection.

The main objective of stress detection on social media is to determine which users may be suffering from stress, to have more information about people with this condition, or to implement solutions that can help individuals with their stress levels.

Reddit is a social media platform where users post in specific topic communities (subreddits), and other users comment and vote on these posts. The extensive nature of these posts makes Reddit an ideal source of information for studying the nuances of phenomena like stress [2].

The forthcoming sections are arranged as follows: Related work, Speech emotion recognition algorithm, experiments, and conclusions.

2 Related Work

Stress detection in social media, especially on Reddit, is not as well-explored as depression detection. Therefore, there are few corpora available for this task. The most widely used corpus for stress detection is Dreaddit [2]. However, some works create their corpora for this task, such as [10, 11].

Dreaddit is a corpus collected from Reddit with the purpose of facilitating the development of models for stress detection. Dreaddit consists of a set of posts annotated by humans as either stress or non-stress. This corpus collects posts from various subreddits where stress-related topics could be discussed [2]. Additionally, this corpus provides an extensive set of features, primarily based on lexical diversity using the categories of Linguistic Inquiry and Word Count (LIWC). The posts in this corpus range from 3 to 300 words, with the majority of posts being above 26 words.

Several works address stress detection using the Dreaddit corpus. The creators of Dreaddit evaluated various baseline models and obtained the best result (BERT-base) with an F_1 score of 0.8065. On the other hand, [12] evaluated multiple models and achieved an F_1 score of 0.84 with the MentalRoBERTa^{FT} model (which is the model from [3] with features from [12]), while [3] achieved an F_1 score of 0.819 with MentalRoBERTa.

BERT-based models have achieved the best results, with an F_1 score above 0.8. However, several approaches have achieved values between 0.75 and 0.80 in F_1 score. For example, [12] implements two Bayesian models (Bernoulli NB and Multinomial NB) with F_1 scores of 0.75 and 0.76, respectively. Additionally, [6] and [13] apply logistic regression algorithms, obtaining F_1 scores between 0.77 and 0.7980. Furthermore, [13] implements a Random Forest classifier, resulting in an F_1 score of 0.78.

In the literature on stress detection in social media, Bayesian approaches have not been fully explored. For instance, [12] implements two Naive Bayes models with features based on their proposed Monte Carlo Tree Search, which allows for targeted keyword searching. Another study [12] implements two Naive Bayes models with TFIDF and BERT-based features, with F_1 scores below 0.69.

2.1 Contribution

In this work, we propose to explore features for a Naive Bayes classifier, such as publication time, subreddit in which the post was made, n-grams for tokenized text, n-grams for POS tagging, n-grams for lemmatization, and n-grams to stem words. Furthermore, with this exploration, we aim to identify features that can be used for training in other models.

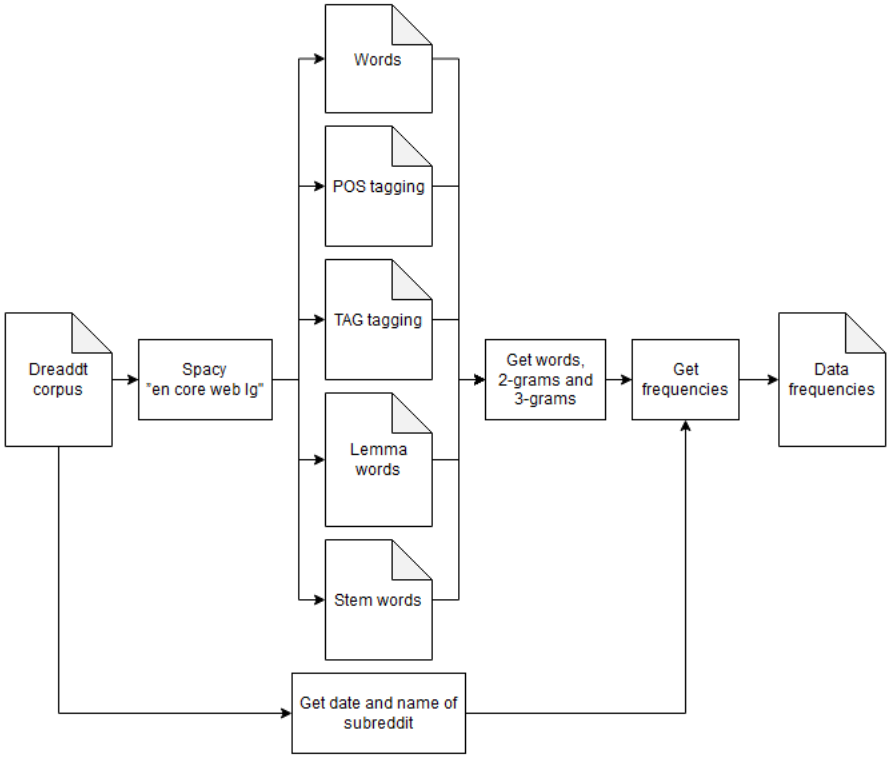

Fig. 1. Pre-processing used to obtain the features and data frequencies.

3 Stress Detection Algorithm

In this section, the process that was carried out to perform stress detection will be described, the following points will be described: pre-processing, feature selection, and classification with Naive Bayes algorithm.

3.1 Pre-processing

The pre-processing involved tokenizing the words in each post, POS tagging, lemmatizing, and stem words. Additionally, only the hour was extracted from the publication dates, and the subreddit name in which the post was made was extracted. Later, 2-grams and 3-grams were created from the tokenized words, POS tags, lemmatized words, and stem words, subsequently, the frequency of appearance of each n-gram and each word was obtained, the frequency in which each word or n-gram appears will be used to build the Naive Bayes model (calculating the probabilities). In Fig. 1 the previously mentioned process is observed.

The corpus pre-processing was programmed with Python 3.9, and the Spacy tool (using the "en_core_web_lg" pipeline for english, this "lg" version is the most complete that spacy can offer us for labeling tasks).

3.2 Feature Selection

To find the best features, a grid search has been implemented considering different combinations between different types of features. The combinations are made considering combinations between 7 types of data (Timestamp, subreddit, words, POS labels, TAG labels, lemmas and stems), in total there are 4096 combinations, that are the product of the lists of characteristics shown below:

- Timestamp (2 features): "social timestamp" (consider the time and date of publication), "without timestamp" (no date and time).
- Subreddit (2 features): "Name subreddit" (probability per name), "without subreddit" (name is not considered).
- Words (4 features): "One word" (probability per word), "2-grams" (probability per 2-gram), "3-grams" (probability per 3-gram), "without words" (words are not considered).
- POS labels (Simple part-of-speech tag, 4 features): "One word" (probability per label), "2-grams" (probability per 2-gram), "3-grams"(probability per 3-gram), "without POS" (POS labels are not considered).
- TAG labels (Detailed part-of-speech tag, 4 features): "One word" (probability per label), "2-grams" (probability per 2-gram), "3-grams" (probability per 3-gram), "without TAG" (TAG labels are not considered).
- Lemma (4 features): "One word" (probability per word), "2-grams" (probability per 2-gram), "3-grams" (probability per 3-gram), "without lemma" (lemma words are not considered).
- Stem (4 features): "One word" (probability per word), "2-grams" (probability per 2-gram), "3-grams" (probability per 3-gram), "without stem" (stem words are not considered).

For the grid search, combinations of 7 elements each were made (4096 combinations in total). In the Naive Bayes algorithm's main processing load lies in calculating the frequencies of occurrence during the pre-processing stage, for that reason the grid search only performed sum of probabilities to calculate the results for each combination. The approximate execution time was 2 seconds per combination, that is, about 8,192 seconds (136.53 minutes), running on a single core with a Ryzen 7 5800X processor.

Subsequently, to improve the results, other grid search was conducted to select the best TAG labels, filtering out the most relevant labels for stress detection.

In the labeling of the posts, 17 different labels were found (for TAG labels). To create combinations of these labels, combination sizes from 2 (136 combinations), 3 (680 combinations), 4 (2,380 combinations), 5 (6,188 combinations), 7 (12,376 combinations), 8 (24,310 combinations), 9 (24,310 combinations) and 10 (19,448

combinations) were considered, with the purpose of discarding tags that do not provide any information for classification. It is important to mention that combinations with more than 10 elements were not performed because the evaluation of results started to decrease.

3.3 Classification

As mentioned previously, a Naive Bayes classifier will be used for classification. To perform this classification, the probabilities were calculated in the following ways:

In equation 1, the calculation of the probability of a specific hour appearing in a class x ("Stress" and "Not stress") is shown. Here, FA represents the frequency of that hour appearing in class x, IC is the number of instances in class x, and H is the total number of hours in a day (i.e. 24 hrs.):

$$P = (FA + 1)/(IC + H). \tag{1}$$

In equation 2, the calculation of the probability of a specific subreddit appearing in a class x ("Stress" and "Not stress") is shown. FA represents the frequency of that subreddit appearing in class x, IC is the number of instances in class x, and N is the total number of subreddits considered in the corpus:

$$P = (FA + 1)/(IC + N). \tag{2}$$

In equation 3, the calculation of the probability of a specific word, tag, lemma, stem, or n-gram appearing in a class x ("Stress" and "Not stress") is shown. FA represents the frequency of that word appearing in class x, VC is the vocabulary size in class x, and VL is the vocabulary size of the specific feature being calculated (word vocabulary, POS tag vocabulary, n-gram vocabulary, etc.):

$$P = (FA + 1)/(VC + VL). \tag{3}$$

4 Experiments and Evaluation

In this section will be shown, the datasets used, the metrics used for evaluation, and the cross-validation process are described.

4.1 Dataset

The corpus used for evaluation is Dreaddit, which has a binary labeling with the tags "Stress" and "Not stress". It consists of 3,553 instances, out of which 1,696 are labeled as "Not stress" and 1,857 as "Stress". Table 1 shows the distribution of instances per subreddit.

As observed in the Table 1, instances have been counted for each subreddit thread. The threads with fewer instances are Food pantry, Stress, and Almost homeless. To ensure that each fold contains at least 8 instances of "Not stress" from the "Food pantry" subreddit, two folds were created for the cross-validation experiment.

Table 1. Number of instances in Dreaddit corpus by subreddit.

Subreddit	Label	Instances	Total
Relationships	Not stress	387	694
	Stress	307	
Anxiety	Not stress	234	650
	Stress	416	
PTSD	Not stress	297	711
	Stress	414	
Assistance	Not stress	229	355
	Stress	126	
Homeless	Not stress	139	220
	Stress	81	
Almost homeless	Not stress	40	99
	Stress	59	
Domestic violence	Not stress	139	388
	Stress	249	
Survivors of abuse	Not stress	172	315
	Stress	143	
Stress	Not stress	33	78
	Stress	45	
Food pantry	Not stress	26	43
	Stress	17	

4.2 Evaluation

For the evaluation of the system, the main metric used was F_1. This metric allows for a proper comparison with related works, as studies using the Dreaddit corpus also present their results using these metrics. The following are the cases used to calculate the metrics of recall, precision, and F_1:

- True positives (TP): Correct detection of the "Stress" label.
- True negatives (TN): Correct detection of the "Not stress" label.
- False positives (FP): Incorrect detection of the "Stress" label.
- False negatives (FN): Incorrect detection of the "Not stress" label.

In equation 4, the formula for calculating recall is shown, while in equation 5, the formula for calculating precision is shown. These two metrics are necessary to calculate the F_1 score:

$$Recall = (TP)/(TP + FN), \tag{4}$$

$$Precision = (TP)/(TP + FP). \tag{5}$$

In equation 6, the formula for calculating the F_1 score is presented, which combines precision and recall measurements into a single value:

Table 2. Number of created instances per set.

Set Set	"Stress" instances	"Not stress" instances	Total instances
Test	367	344	711
Fold 1	734	687	1,421
Fold 2	734	687	1,421

Table 3. Best five results F_1, mean and standard deviation, for the different combinations of characteristics.

Fold 1 F_1	Fold 2 F_1	Mean	Test F_1	Features
0.7518	0.7582	0.7550	**0.7605**	words, TAG, lemma
0.7524	**0.7584**	**0.7554**	0.7581	words, TAG, lemma, stem, subreddit
0.7517	**0.7584**	0.7550	0.7581	words, TAG, lemma, stem, hour, subreddit
0.7524	0.7572	0.7548	0.7581	words, TAG, lemma, stem
0.7512	0.7576	0.7544	0.7581	words, TAG, lemma, stem, hour

$$F_1 = 2((PrecisionRecall)/(Precision + Recall)). \quad (6)$$

For the evaluation of the experiments, the corpus was divided into 80% for creating two folds for cross-validation, and the remaining 20% was used as the final test set. Table 2 shows the number of instances for the test set and each created fold.

During the second implementation of grid search for TAG label filtering, folds 1 and 2 were used together and split into 80% for training and 20% for testing, with the aim of finding an improvement in the results. However, in the final evaluations, the previously described cross-validation approach was continued to be used.

5 Results

In this section, the results obtained with the folds and the test set will be shown.

In Table 3, the results with the F_1 metric can be observed. The results are ordered based on the combinations of features that have the best F_1 results. As seen in the table, the model with the best performance in terms of the mean is the second result, with an F_1 score of 0.7584. On the other hand, the model that achieved the best result on the test set is the first result, with an F_1 score of 0.7605, which also has the fewest implemented features. Another detail to note is that the features "words," "TAG," and "lemma" are constant in all the best models.

Additionally, an experiment was conducted to improve the results obtained in Table 3. This experiment involved applying a grid search to filter out less relevant

Table 4. Results F_1, mean and standard deviation in folds, for the best combinations of characteristics, filtering tags.

Fold 1 F_1	Fold 2 F_1	Mean	Features
0.7545	**0.7565**	**0.7554**	words, TAG (filtered), lemma, stem, subreddit
0.7534	**0.7560**	0.7550	words, TAG (filtered), lemma, stem, hour, subreddit
0.7520	0.7569	0.7544	words, TAG (filtered), lemma, stem, hour
0.7513	0.7573	0.7543	words, TAG (filtered), lemma, stem
0.7523	0.7539	0.7531	words, TAG (filtered), lemma

Table 5. Results for the test set, in each model filtering tags.

Precision	Recall	F_1	Features
0.7259	**0.8333**	**0.7759**	words, TAG (filtered), lemma
0.7149	0.8225	0.7650	words, TAG (filtered), lemma, stem, hour, subreddit
0.7159	0.8198	0.7644	words, TAG (filtered), lemma, stem, subreddit
0.7102	0.8172	0.7600	words, TAG (filtered), lemma, stem
0.7102	0.8172	0.7600	words, TAG (filtered), lemma, stem, hour

TAG labels. The grid search used different combination sizes, the combination that yielded the best results was of size 4, and the labels it contained were as follows: 'NNPS' (noun, proper plural), 'UH' (interjection), 'MD' (verb, modal auxiliary), and 'NFP' (superfluous punctuation).

Table 4 presents the results of this experiment on the folds. The results were sorted from highest to lowest using the average, and it can be observed that the lowest result corresponds to the model with fewer features, while the second position in Table 3 now takes the first place.

When we consider the results with tag filtering on the test set in Table 5, we can observe that the best model is still the one with fewer features. Another notable detail is the improvement in the test results.

Finally, in Table 6, a comparison of the proposed model with most of the models seen in the literature can be observed. The proposed model is better than other Naive Bayes-based approaches. It is worth noting that the proposed model achieved higher precision than other Naive Bayes algorithm-based models. However, the proposed model does not manage to position itself among the top-performing models.

6 Conclusions

In this paper, the stress detection task in social networks was reviewed specifically for the Dreaddit corpus. The task was addressed using a Naive Bayes classifier, and the tagging provided by the Spacy tool for Python was utilized. Additionally, two grid searches were conducted to find the best features for this type of classifier.

Table 6. F_1 scores of the most relevant models reviewed in the literature.

Model	Paper	Precision	Recall	F_1
MentalRoBERTaFT	[12]	0.780	0.900	0.840
KC-Net	[1]	0.841	0.833	0.835
MentalRoBERTa	[3]	0.821	0.818	0.819
RoBERTa	[4]	0.812	0.813	0.813
EMO_INF	[5]	0.817	0.817	0.817
Random Forest (BERT)	[13]	0.720	0.850	0.780
Naive Bayes	**Proposed**	**0.725**	**0.833**	**0.775**
LR+Features	[6]	0.735	0.810	0.770
Logistic Reg.	[13]	0.750	0.790	0.770
n-grams + features*	[2]	0.747	0.794	0.770
Multinomial NB	[12]	0.680	0.870	0.760
Bernoulli NB	[12]	0.690	0.840	0.750
BiLSTM_Att	[7]	0.727	0.720	0.720
Naive Bayes (TFIDF)	[13]	0.650	0.740	0.690

The results were evaluated using the F_1 metric, which allowed for a comparison with the works found in the literature. Moreover, the obtained results surpassed those achieved by other models that employed the Naive Bayes algorithm.

Useful features were found from an approach where their frequency of occurrence in the corpus was evaluated, and, likewise, other features were discarded. The use of n-grams to identify potential stress patterns in detection was discarded. Similarly, the use of simple POS tagging from Spacy was also discarded. Instead, the use of detailed tagging (TAG) is suggested, particularly with the identified tags ('NNPS', 'UH', 'MD', and 'NFP'), as they have proven to be useful for stress detection.

Furthermore, as future work, exploring dependency parsing in Spacy is recommended to identify common dependency pairs in texts expressing stress. Additionally, using the discovered features to experiment and investigate if models in the literature can further improve their results.

References

1. Yang, K., Zhang, T., Ananiadou, S.: A mental state Knowledge–aware and Contrastive Network for early stress and depression detection on social media. Information Processing & Management **59**(4), 1–16 (2022)
2. Turcan, E., McKeown, K.: Dreaddit: A Reddit Dataset for Stress Analysis in Social Media. In: Proceedings of the Tenth International Workshop on Health Text Mining and Information Analysis (LOUHI 2019), pp. 97–107 (2019)
3. Ji, S., Zhang, T., Ansari, L., Fu, J., Tiwari, P., Cambria, E.: MentalBERT: Publicly Available Pretrained Language Models for Mental Healthcare. In: Proceedings of the Thirteenth Language Resources and Evaluation Conference, pp. 7184–7190 (2022)

4. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, Y.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692. (2019)
5. Turcan, E., Muresan, S., McKeown, K.: Emotion-Infused Models for Explainable Psychological Stress Detection. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2895–2909 (2021)
6. Tadesse, M., Lin, H., Xu B., Yang, L.: Detection of Depression-Related Posts in Reddit Social Media Forum. IEEE Access **7**, 44883–44893 (2019)
7. Ren, L., Lin, H., Xu, B., Zhang, S., Yang, L., Sun, S.: Depression Detection on Reddit With an Emotion-Based Attention Network: Algorithm Development and Validation. JMIR Med Inform **9**(7), 1–13 (2021)
8. Capdevila, N., Segundo, M.: Estrés. Offarm: farmacia y sociedad **24**(8), 96–104 (2005)
9. Calcia, M.A., Bonsall, D.R., Bloomfield, P.S.: Stress and neuroinflammation: a systematic review of the effects of stress on microglia and the implications for mental illness. Psychopharmacology 233, 1637–1650 (2016)
10. Gong, C., Saha, K., Chancellor, S.: The Smartest Decision for My Future: Social Media Reveals Challenges and Stress During Post-College Life Transition. In: Proceedings of the ACM on Human-Computer Interaction, pp. 1–29 (2021)
11. Rastogi, A., Liu, Q., Cambria, E.: Stress Detection from Social Media Articles: New Dataset Benchmark and Analytical Study. In: 2022 IEEE International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2022)
12. Swanson, K., Hsu, J., Suzgun, M.: Monte Carlo Tree Search for Interpreting Stress in Natural Language. In: LTEDI 2nd Workshop on Language Technology for Equality, Diversity and Inclusion, pp. 107–119 (2022)
13. Selvadass, S., Malin Bruntha, P., Priyadharsini, K.: Stress Analysis in Social Media using ML Algorithms. In: 4th IEEE International Conference on Smart Systems and Inventive Technology (ICSSIT), pp. 1502–1506 (2022)