Hate Speech and Offensive Language Detection: A New Feature Set with Filter-Embedded Combining Feature Selection

Noor Azeera Abdul Aziz
School of Computing, Faculty of Engineering, Universiti Teknologi
Malaysia, Johor, Malaysia
zera167@gmail.com

Mohd Aizaini Maarof
School of Computing, Faculty of Engineering, Universiti Teknologi
Malaysia, Johor, Malaysia
aizaini@utm.my

Anazida Zainal
School of Computing, Faculty of Engineering, Universiti Teknologi
Malaysia, Johor, Malaysia
anazida@utm.my

2021 3rd International Cyber Resilience Conference (CRC) | 978-1-6654-1844-7/20/$31.00 ©2021 IEEE | DOI: 10.1109/CRC50527.2021.9392486

***Abstract*— Social media has changed the world and play an important role in people lives. Social media platforms like Twitter, Facebook and YouTube create a new dimension of communication by providing channels to express and exchange ideas freely. Although the evolution brings numerous benefits, the dynamic environment and the allowable of anonymous posts could expose the uglier side of humanity. Irresponsible people would abuse the freedom of speech by aggressively express opinion or idea that incites hatred. This study performs hate speech and offensive language detection. The problem of this task is there is no clear boundary between hate speech and offensive language. In this study, a selected new features set is proposed for detecting hate speech and offensive language. Using Twitter dataset, the experiments are performed by considering the combination of word n-gram and enhanced syntactic n-gram. To reduce the feature set, filter-embedded combining feature selection is used. The experimental results indicate that the combination of word n-gram and enhanced syntactic n-gram with feature selection to classify the data into three classes: hate speech, offensive language or neither could give good performance. The result reaches 91% for accuracy and the averages of precision, recall and F1.**

Keywords—Twitter, hate speech, offensive language, word n-gram, syntactic n-gram, feature selection, machine learning.

I. INTRODUCTION

As increasing number of social media platforms have caused content overload. Unfortunately, not all contents are relevant, and some might harm people. This happen when there are people who misuse the platforms to propagate hate. Though difficult to achieve, identifying hate speech becomes an essential task in order to simultaneously provides freedom of speech and prevent hate speech content [1]. There are numerous studies focus on hate speech detection [2]. Theoretically, most of the theorist distinguish hate speech from merely offensive language [3]. Contrasting to the theorist, many studies overlook the offensive language. The studies focus more on the binary classification between hate speech and non-hate speech [4], [5] and fine-grained detection of various types of hate speech [6], [7]. Concatenate both classes, hate speech and offensive language has caused overset limits of hate speech that lead to false positive [8], [9]. The reputation of social media platforms can be tarnished when users feel frustrated as many non-hate speech contents are mistakenly detected as hate [10].

Although no formal definition of hate speech is universally been accepted, Fortuna and Nunes [11] has concluded the definition of hate speech from the numerous definitions as "*language that attacks or diminishes, that incites violence or hate against groups, based on specific characteristics such as physical appearance, religion, descent, national or ethnic origin, sexual orientation, gender identity or other, and it can occur with different linguistic styles, even in subtle forms or when humor is used.*" For example, the tweet "*We agree... do you? http://t.co/4diz5NKYMN” F*CK YES, I DO! Send those illegal, wetbacks home!!!*" is obviously contains racial insult towards immigrants as a target.

Offensive language is "*posts which are degrading, dehumanizing, insulting an individual, threatening with violent acts*" [12]. As example, this tweet "*and you look like hmmm a n*gger*" contains offensive language content but does not incite violence or hatred to any groups based on specific characteristics.

Nowadays, offensive words are commonly used in social media platforms. People tend to use offensive words in different context such as to express emotions like anger, frustration, or surprise [13]–[15]. For example:

- Hate speech: "*This **n*ggah** pharrell or whatever this **n*ggah** name is do not deserve **sh*t** for that white *ss g*y song*"
- Offensive language: "*Lame **n*ggaz** wait in line, wait for b*tches, wait to create, wait for someone else to do **sh*t** for them.* "
- Neither: *"I really need to take my rose colored glasses off though. I gotta stop thinking everybody does **sh*t** with good intentions.*"

As shown in the example, offensive words "sh*t" can be found in all classes. The presence of offensive words in various aspects make detection becomes difficult. Although offensive words often considered rude and offensive or to emphasize emotion, offensive words should not to be taken lightly as the words can denote hate speech [16].

Other than offensive words, pejoratives appear in various classes as the use of "n*" pejorative in hate speech and offensive language in the previous example. These conditions make hate speech and offensive language detection becomes more challenging and has resulted the grey boundary between the classes as the characteristics are almost similar [9], [17]. Thus, instead of looking at the appearance of specific word, the detection approach needs to analyse how words are related to each other in bringing the context of the sentence.

978-1-6654-1844-7/21/$31.00 ©2021 IEEE

Sentence is a form of words that are syntactically related. Figure 1 shows the dependency structures for different sentences with the same semantics. The number on each arch indicates the dependency distance between words. Local dependencies are often a dependency between two words that share the same syntactic rule as shown by the relation between "knocks" and "out" in Sentence 1. In Sentence 2, long-distance dependency occurs between "knock" and "out" when this words far apart in a sentence but can be syntactically

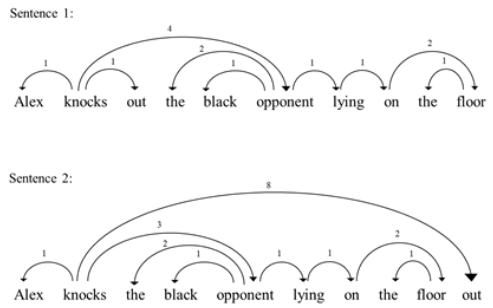

Fig. 1. Dependency structures

dependent.

Extracting feature representation is a crucial step in most machine learning tasks. The more in transforming meaningful information from raw text to feature representations usually would improve the classification performance [18]. An effective method that handles the difficulties of retrieving long-distance dependencies as well as local dependencies is needed to extract more information from raw text and thus, capture features that contribute most to the prediction model. Therefore, as highlighted by [9], this study will investigate how to extract more meaningful features to understand the sentence context and thus, solve the problem of multiclass classification between hate speech, offensive language and neither class.

The remainder of this paper is organized as follows: Section II describes all related works. Section III presents the proposed method for hate speech and offensive language detection, features extraction methods and feature selection methods. Section IV explains the implementation, and the results are discussed in section V. Section VI concludes the paper and recommends the future works.

II. Related Work

The related research on hate speech detection shows that most studies consider binary classification task and fine-grained classification of various types of hate speech. Besides these detection tasks, hate speech detection can predict the content according to the level of hateful [19]. This can ensure only those that are truly hateful need to be justified as prohibited and not rigid to hate speech and non-hate speech only. Although the quantity not as much as the other detection tasks, there are several studies focus on the ternary classification of hate speech, offensive language and neither.

Davidson *et al.* [9] weighted each unigram, bigram, and trigram features with Term Frequency–Inverse Document Frequency (TF-IDF), constructed unigrams, bigrams and unigrams of Part-of-Speech (POS) tag, derived readability scores from Flesch-Kincaid Grade Level and Flesch Reading Ease scores, included binary and count indicators for hashtags, mentions, retweets, and URLs, the number of characters, words, and syllables in each tweet to differentiate between hate speech and offensive language.

The various levels of surface n-grams (word and character) and word skip grams are used as features in [20]. Using a Davidson dataset that is publicly available, the authors had attempted to distinguish between hate speech and offensive language with a linear Support Vector Machines (SVM) classifier. The study has performed feature selection for feature reduction and has reported that these features were not significance in distinguishing between hate and offensive language.

Extending the work in [20], Malmasi and Zampieri [21] has distinguished general offensive language from hate speech in social media by considering n-grams, skip-grams and clustering-based word representation as features. The Radial basis function (RBF) kernel SVM was reported as more suitable for data with a smaller number of features compared to the linear SVM.

Gaydhani *et al.* [22] has combined three difference datasets to make hate speech class more balanced and has reported a good result for Logistic Regression (LR) classifier with L2 normalization. The authors had extracted the n-gram features from the tweets and weight each feature with TF-IDF values.

In [17], the author proposed a pragmatic approach to extract unigram, sentiment feature, semantic feature and pattern feature. Although the study applied the same way as in [22] to balance the dataset, the result is lower compared to the result in [22].

Madukwe and Gao [23] proposed typed dependency features in detecting hate speech and offensive language. Good accuracy is achieved with a smaller feature set when applying embedded feature selection. Typed dependency features able to capture long-distance dependencies between two words based on the dependency relation. In contrast, the enhanced syntactic n-gram features of this study able to capture long-distance dependencies more than the relation between two words by traversing the dependency parse tree.

III. Proposed Method

The methodology for this study is illustrated in Figure 2.

Machine learning approach is fed with set of features extracted from training dataset to perform the classification.

A. Features extraction method

There are two text-based features are extracted in this study. The combination of these features is assumed to give a beneficial contribution to this study.

1) Word n-grams: N-grams have successfully used in many domains and turn out to be effective in hate speech detection [9], [22]–[26]. For this study, by following other works [9], [21]–[23] word unigrams, bigrams and trigrams are extracted from each tweet. Stop words are removed while extracting word n-grams.

2) Syntactic n-grams: Differ from the linear manner in n-grams, syntactic n-grams extracted by traversing path in dependency parse tree can capture long-distance dependency between word [27], [28]. The extraction of syntactic n-grams are divided into three steps:

a) Dependency relation generation: In this step, all tweets are parsed into the enhanced++ dependencies of Stanford Parser. Enhanced++ dependencies are more semantic compared to other dependencies and easier to begin with [29]. The punctuation dependency relations for example "punct(got-1, .-10)" are removed from the set of dependency relations. The example set are listed in Table I.

TABLE I. Dependency Relations Set

Tweet	Dependency Relations
*This n*ggah pharrell or whatever this n*ggah name is do not deserve sh*t for that white *ss g*y song."*	ROOT(ROOT-0, pharrell-3) det(pharrell-3, this-1) amod(pharrell-3, niggah-2) cc(pharrell-3, or-4) advmod(deserve-12, whatever-5) det(name-8, this-6) amod(name-8, niggah-7) nsubj(deserve-12, name-8) aux(deserve-12, is-9) aux(deserve-12, do-10) neg(deserve-12, not-11) conj:or(pharrell-3, deserve-12) dobj(deserve-12, shit-13) case(song-19, for-14) det(song-19, that-15) amod(song-19, white-16) compound(song-19, ass-17) amod(song-19, gay-18) nmod:for(deserve-12, song-19)

b) Subtrees extraction: The dependency parse trees are generated based on the set of dependency relations in step *2(a)*. Tweet may contains more than one sentence per tweet or sometimes just a word. Unlike the work in [30] that allows only one sentence to be processed at one time, this study processes more than one sentence per time to preserve the sentence semantic. Besides processing multiple sentences per time, a function is added to extract unigram of each tweet to handle tweet with a word and to gain more meaningful features. The first step in subtrees extraction is conducting breadth-first search of the tree and finds all subtrees of height equal to 1 as shown in Figure 3. For example, the output of this step will be "0[1], 0[2], 0[1,2]" at level 0 and "2[3], 2[4], 2[3,4]" at level 1. Then, the tree is traversed in pre-order traversal and the node occurrence in a subtree is replaced with the subtrees from higher levels where the node is the root.

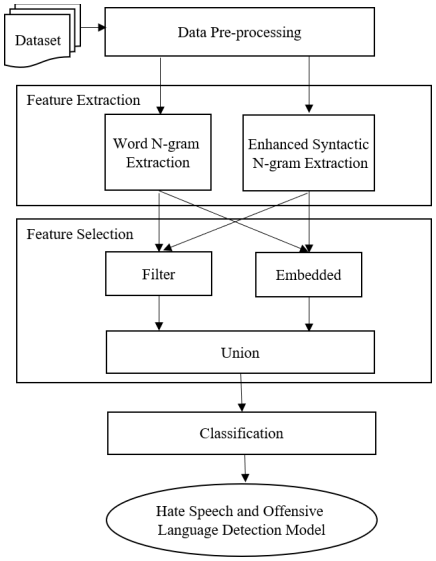

Fig. 2. Proposed methodology

The outputs for this step are "0[2[3]], 0[2[4]], 0[2[3,4]], 0[1,2[3]], 0[1,2[4]], 0[1,2[3,4],]." Unfortunately, the word orders of the extracted syntactic n-grams are not preserved when using the pre-order traversal. This can impact the semantic of the sentence as word order is important for construction of grammar [31], [32]. Therefore, the word reordering process is needed to get the same order as the input tweet.

c) Words Reordering: In this process, the extracted syntactic n-grams are reordered based on the word index of the original input provided in dependency relations. Same as word n-grams, all stop words are removed while extracting the features.

Table II shows the extracted features for tweet "*This n*ggah pharrell or whatever this n*ggah name is do not deserve sh*t for that white *ss g*y song.*" Word n-grams and syntactic n-grams extract very difference features. This is accredited to how the features are generated. In word n-grams, the features are extracted by the combinations of adjacent words of length n in sentence. Differ from word n-grams, syntactic n-grams obtained the features by traversing the dependency parsing tree. In particular, the adjacent word in syntactic n-grams based on the dependency relation between words. Similar with syntactic n-grams, enhanced syntactic n-grams follow the same extracted approach use in syntactic n-grams except with the addition of processing multiple sentences per time, unigram extraction and word reordering step at the end of the extracting process to maintain the sentence semantic.

Thus, word n-grams and enhanced syntactic n-grams can complement each other for extracting more meaningful information to understand the hidden context of each tweet when considering more deep semantic information that is able to capture local dependency and long-distance dependency between words, respectively.

TABLE II. Extracted Text-based Features

Features	Features Extraction Method		
	Word n-grams	*Syntactic n-grams*	*Enhanced syntactic n-grams*
Unigram	[niggah, pharrell, niggah, deserve, shit, white, ass, gay, song]	-	[niggah, pharrell, niggah, deserve, shit, white, ass, gay, song]
Bigram	[niggah pharrell, pharrell niggah, niggah deserve, deserve shit, shit white, white ass, ass gay, gay song]	[deserve shit, song ass, deserve name, pharrell deserve, song white, name niggah, song gay, deserve song, pharrell niggah, deserve whatever]	[deserve shit, ass song, name deserve, pharrell deserve, white song, niggah name, gay song, deserve song, niggah pharrell, whatever deserve]
Trigram	[niggah pharrell niggah, pharrell niggah deserve, niggah deserve shit, deserve shit white, shit white ass, white ass gay, ass gay song]	[deserve whatever shit, pharrell deserve name, deserve song gay, deserve name niggah, pharrell deserve whatever, song ass gay, deserve shit song, deserve name shit, song white ass, deserve name song, pharrell deserve shit, deserve song ass, pharrell deserve song, deserve song white, pharrell niggah deserve, deserve whatever song, song white gay, deserve whatever name]	[whatever deserve shit, pharrell name deserve, deserve gay song, niggah name deserve, pharrell whatever deserve, ass gay song, deserve shit song, name deserve shit, white ass song, name deserve song, pharrell deserve shit, deserve ass song, pharrell deserve song, deserve white song, niggah pharrell deserve, whatever deserve song, white gay song, whatever name deserve]

B. Feature selection method

Most of the feature extraction methods suffer from high dimensionality of the feature space. The problems of the extracted features are noisy, redundant and not all featuresare significant for prediction model may influence the performance of the classifier [33], [34]. Eliminating these problems indirectly reduce the feature space and thus improve the efficiency and accuracy of classifier. In hate speech detection, feature selection has proven to enhance the detection accuracy by selecting the features that contribute most to the predictive model [23], [35], [36].

Instead of using one method, this study applied the combination of feature selection methods. Combining multiple feature selections can overcome the limitation of each method [37], [38] and thus, improve the accuracy of hate speech and offensive language detection. In this study, filter-embedded combination feature selection is used.

1) Filter: Filter method is performed without specific type of predictive model and select the best features according to the score of feature ranking. In this study, the SelectKbest of Scikit-learn is used to construct f_regression function.The numbers of features are reduced by removing the least significant features.

2) Embedded: Embedded method depends on a specific learning algorithm that performs the feature selection in the process of classifier construction. LR is used as embedded feature selection for this study.

IV. Implementation

The aim of this study is to classify each tweet of dataset into one of three classes which are hate speech, offensive language and neither.

A. Data Pre-processing

Each tweet is pre-processed to reduce noise. The details of each processes listed as follow:

1) HTMLs removal: All HTML encoding that are not being converted to text, such as “&amp” and “&quot” are removed.

2) URLs removal: All URL links are removed as the links do not add any value in building the model.

3) Twitter ID and Hashtag Replacement: Twitter ID (@mention) and hashtag (#) are replaced to USERNAME and HASHTAG, respectively. Sometimes hate speech target or offended user can be identified by the appearances of Twitter ID in tweet and people use hashtag to represent some words. Both tokens are remains because removing the tokens can change the sentence structure and thus effects the classification result.

4) Consecutive punctuation marks removal: The consecutive punctuation marks are removed and leave only one of the punctuations. This study does not remove all punctuation marks because removing all marks will change the structure of the sentence and affects the meaning of the sentence.

5) Unnecessary symbol or whitespace removal: The remaining symbols or extra whitespaces appear on tweet for example in the beginning of tweet are deleted since the tokens does not give any impact to the hate speech and offensive language detection model.

B. Features Combination

Proposed features set for this study is the combination of word n-grams and enhanced syntactic n-grams with grams in range (1,3). The extracted features for each tweet are transformed into a document-term matrix by calculating the term frequency weight for each word n-grams and enhanced syntactic n-grams.

C. Feature selection

Important features are selected based on p-value of f_regression function. K is set to 30 000 in selectKbest function. For LR embedded method, penalty is set to L2 and the threshold value is set to 1.25. Union method is applied to aggregate the different reduced feature sets. For example, all the selected feature set in word n-grams (W) and enhanced syntactic n-grams (E) are used in the union of reduced feature sets W and E (i.e., $W \cup E$).

D. Classification

LR with L2 regularization is used for the final model. Each model is tested using 5-fold cross validation and One-

Vs-Rest approach is used for the multiclass classification problem.

E. Evaluation Metrics

Since this study handling multiclass problem, the performance evaluation described in terms of accuracy and averaging for precision (P), recall (R) and F1-score (F1) following other hate speech detection studies as stated by [39]. Due to imbalance dataset, the averaging of macro scores is used to show the real performance of the minority class which is hate speech class for this study. For the comparison with other baselines, this study reports the performance result using micro-average in the same way as reported by other existing hate speech and offensive detection.

V. Results

This study utilized a Twitter dataset by [9]. Each tweet on this dataset was manually annotated by three or more people in CrowdFlower (CF) with 92% of intercoder-agreement score. The dataset consists of 24 783 tweets in which 1430 hate tweets, 19 190 offensive tweets, and 4163 neither tweets. 30% of the dataset is used to evaluate the performance of the proposed model.

There are several features sets are used for the experiment but only the best results for four different features sets are reported for this study as shown in Table III. These features sets are word n-gram, syntactic n-gram, enhanced syntactic n-gram and the proposed feature set which the combination of word n-gram and enhanced syntactic n-gram.

TABLE III. Classification Result for Different Features Sets

Features	Classification Result			
	Macro-P	*Macro-R*	*Macro-F1*	*Accuracy*
Word n-gram	0.751	0.677	0.689	**0.898**
Syntactic n-gram	0.744	0.661	0.670	0.888
Enhanced syntactic n-gram	**0.759**	0.673	0.688	0.896
Proposed feature set	0.749	**0.686**	**0.699**	**0.898**

The proposed features set produced the highest score for macro-R and macro-F1 which are 0.686 and 0.699, respectively. For accuracy, proposed result reaches 0.898 similar with the accuracy score of word n-gram features. Unfortunately, with 0.759, the best result for macro-P is belong to the enhanced syntactic n-grams features set. As discussed in the previous section, the used of all features in features set has resulted high dimensionality of the features space. Therefore, features selection is used to reduce the features space. The classification result for proposed set with different types of features selection methods are shown in Table IV.

TABLE IV. Classification Result for Selected Proposed Features Set

Feature Selection Methods	Classification Result			
	Micro-P	*Micro-R*	*Micro-F1*	*Accuracy*
F-regression	0.897	0.897	0.897	0.897
LR embedded method	0.901	0.901	0.901	0.901
F-regression-LR combination	**0.910**	**0.910**	**0.910**	**0.910**

Table IV shows that the classification result is improved after applying feature selection. The advantage of the combination of f-regression and LR embedded method is proven to be beneficial when reaches the highest result for all scores. This result is then compared with other studies that used the same dataset and traditional machine learning. The comparison result is given in Table V. The result shows that the proposed approach has obtained better performance compared to other approaches for all scores which is 0.91.

TABLE V. Comparison of Best Classification Result of Proposed Approach with Baselines

Approaches	Classification Result			
	Micro-P	*Micro-R*	*Micro-F1*	*Accuracy*
Davidson *et al.* [9]	0.91	0.90	0.90	-
Madukwe and Gao [23]	0.90	0.90	0.90	0.90
Proposed approach	**0.91**	**0.91**	**0.91**	**0.91**

Obviously, the improvement of proposed approach is only 1%. There is 9% out of the test data are being misclassified. Many of the hate speech tweet were misclassified as offensive language class compared to the neither class. For offensive language class and neither class, majority of the tweets belong to these classes were misclassified to each other compared towards the hate speech class. The used of dependency parser as a base of syntactic n-gram features extraction could influence the classification result. Tweets are suffering from the spelling and grammatical error. Therefore, the potential errors occur during the dependency relations generation process. More investigation is needed to reduce the noise in tweet and to identify the effective dependency parser in handling noisy data.

VI. Conclusion

In this study, a new feature set with the combination of filter and embedded features selection is proposed. The selected word n-gram and enhanced syntactic n-gram features set is used to classify tweets into hate speech, offensive language and neither. The proposed approach achieved 0.91 accuracy as well as the averages of micro-P, micro-R and micro-F1.

In future, improvements can be done by considering other dependency parser and explore other traversing tree approaches to extract syntactic n-grams.

Acknowledgment

This study was funded by UTM Transdisciplinary Research Grant, Graph-Based Knowledge Representation for Threat Intelligence, PY/2018/03477.

References

[1] Article 19, ' Hate Speech' Explained A Toolkit. 2015.

[2] A. Schmidt and M. Wiegand, "A Survey on Hate Speech Detection using Natural Language Processing," in Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, 2017, pp. 1–10.

[3] P. Billingham and M. Bonotti, “Introduction: Hate, Offence and Free Speech in a Changing World,” Ethical Theory Moral Pract., vol. 22, no. 3, pp. 531–537, 2019.

[4] A. Bisht, A. Singh, H. Bhadauria, J. Virmani, and Kriti, “Detection of Hate Speech and Offensive Language in Twitter Data Using LSTM Model,” in Recent Trends in Image and Signal Processing in Computer Vision, Springer, 2020, pp. 243–264.

[5] C. Nobata, J. Tetreault, A. Thomas, and Y. Mehdad, “Abusive language detection in online user content,” Proc. 25th, 2016.

[6] W. Alorainy, P. Burnap, H. Liu, and M. L. Williams, “‘The Enemy Among Us’: Detecting Cyber Hate Speech with Threats-Based Othering Language Embeddings,” ACM Trans. Web, vol. 13, no. 3, Jul. 2019.

[7] Z. Waseem, “Are You a Racist or Am I Seeing Things? Annotator Influence on Hate Speech Detection on Twitter,” in Proceedings of2016 EMNLP Workshop on Natural Language Processing and Computational Social Science, 2016, pp. 138–142.

[8] T. Gröndahl, L. Pajola, M. Juuti, M. Conti, and N. Asokan, “All you need is ‘love’: Evading hate speech detection,” Proc. ACM Conf. Comput. Commun. Secur., pp. 2–12, 2018.

[9] T. Davidson, D. Warmsley, M. Macy, and I. Weber, “Automated hate speech detection and the problem of offensive language,” Proc. 11th Int. Conf. Web Soc. Media, ICWSM 2017, pp. 512–515, 2017.

[10] M. Mozafari, R. Farahbakhsh, and N. Crespi, “A BERT-Based Transfer Learning Approach for Hate Speech Detection in Online Social Media,” in Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019, 2019, vol. 881, pp. 928–940.

[11] P. Fortuna and S. Nunes, “A survey on automatic detection of hate speech in text,” ACM Computing Surveys, vol. 51, no. 4. 2018.

[12] T. Mandl et al., “Overview of the HASOC track at FIRE 2019: Hate speech and offensive content identification in Indo-European languages,” ACM Int. Conf. Proceeding Ser., pp. 14–17, 2019.

[13] R. Stephens and A. Zile, “Does Emotional Arousal Influence Swearing Fluency?,” J. Psycholinguist. Res., vol. 46, no. 4, pp. 983–995, 2017.

[14] S. O. Sood, E. S. Street, J. Antin, and E. F. Churchill, “Profanity Use in Online Communities 4301 Great America Parkway,” Advances, pp. 1481–1490, 2012.

[15] J. Timothy and K. Janschewitz, “The pragmatics of swearing,” J. Politeness Res., vol. 4, no. 2, pp. 267–288, 2008.

[16] P. L. Teh, C. Bin Cheng, and W. M. Chee, “Identifying and categorising profane words in hate speech,” ACM Int. Conf. Proceeding Ser., pp. 65–69, 2018.

[17] H. Watanabe, M. Bouazizi, and T. Ohtsuki, “Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection,” IEEE Access, vol. 6, pp. 13825–13835, 2018.

[18] Z. Liu, Y. Lin, and M. Sun, Representation Learning for Natural Language Processing. Springer Nature, 2020.

[19] B. Ross, M. Rist, G. Carbonell, B. Cabrera, N. Kurowsky, and M. Wojatzki, “Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis,” vol. 2016, no. September, pp. 0–4, 2017.

[20] S. Malmasi and M. Zampieri, “Detecting hate speech in social media,” Int. Conf. Recent Adv. Nat. Lang. Process. RANLP, vol. 2017-Septe, pp. 467–472, 2017.

[21] S. Malmasi and M. Zampieri, “Challenges in discriminating profanity from hate speech,” J. Exp. Theor. Artif. Intell., vol. 30, no. 2, pp. 187–202, 2018.

[22] A. Gaydhani, V. Doma, S. Kendre, and L. Bhagwat, “Detecting Hate Speech and Offensive Language on Twitter using Machine Learning: An N-gram and TFIDF based Approach,” 2018.

[23] K. J. Madukwe and X. Gao, “The Thin Line Between Hate and Profanity,” in Australasian Joint Conference on Artificial Intelligence, 2019, pp. 344–356.

[24] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, “Deep Learning for Hate Speech Detection in Tweets,” in InProceedings of the 26th International Conference on World Wide Web Companion, 2017, no. 2, pp. 759–760.

[25] P. Burnap and M. L. Williams, “Us and them: identifying cyber hate on Twitter across multiple protected characteristics,” EPJ Data Sci., vol. 5, no. 1, 2016.

[26] Z. Waseem and D. Hovy, “Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter,” in Proceedings of the {NAACL} Student Research Workshop, 2016, pp. 88–93.

[27] G. Sidorov, F. Velasquez, and E. Stamatatos, “Syntactic Dependency-based N-grams as Classification Features,” MICAI ’12 Adv. Artif. Intell., no. Cic, pp. 1–11, 2012.

[28] G. Sidorov, F. Velasquez, E. Stamatatos, A. Gelbukh, and L. Chanona-Hernández, “Syntactic N-grams as machine learning features for natural language processing,” Expert Syst. Appl., vol. 41, no. 3, pp. 853–860, 2014.

[29] S. Schuster and C. D. Manning, “Enhanced English universal dependencies: An improved representation for natural language understanding tasks,” Proc. 10th Int. Conf. Lang. Resour. Eval. Lr. 2016, pp. 2371–2378, 2016.

[30] G. Sidorov, G. Helena, I. Batyrshin, and E. Mirasol-m, “Algorithm for Extraction of Subtrees of a Sentence Dependency Parse Tree,” vol. 14, no. 3, pp. 79–98, 2017.

[31] R. Van Trijp, “A computational construction grammar for English,” AAAI Spring Symp. - Tech. Rep., vol. SS-17-01-, no. Ogden 1968, pp. 266–273, 2017.

[32] J. Kuningas and J. Leino, “Word Orders and Construction Grammars,” SKY J. Linguist., vol. 19, no. SUPPL, pp. 301–309, 2006.

[33] H. Liu and L. Yu, “Toward integrating feature selection algorithms for classification and clustering,” IEEE Trans. Knowl. Data Eng., vol. 17, no. 4, pp. 491–502, 2005.

[34] F. Sebastiani, “Machine Learning in Automated Text Categorization,” ACM Comput. Surv., vol. 34, no. 1, pp. 1–47, 2002.

[35] Z. Zhang, D. Robinson, and J. Tepper, “Hate Speech Detection Using a Convolution-LSTM Based Deep Neural Network,” Eurpoean Semant. Web Conf., pp. 745–760, 2018.

[36] D. Robinson, Z. Zhang, and J. Tepper, “Hate speech detection on twitter: Feature engineering v.s. feature selection,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2018, vol. 11155 LNCS, pp. 46–49.

[37] Y. C. Saw, Z. I. M. Yusoh, A. K. Muda, and A. Abraham, “Ensemble Filter-Embedded Feature Ranking Technique (FEFR) for 3D ATS drug molecular structure,” Int. J. Comput. Inf. Syst. Ind. Manag. Appl., vol. 9, pp. 124–134, 2017.

[38] C. W. Chen, Y. H. Tsai, F. R. Chang, and W. C. Lin, “Ensemble feature selection in medical datasets: Combining filter, wrapper, and embedded feature selection results,” Expert Syst., vol. 37, no. 5, pp. 1–10, 2020.

[39] Z. Zhang and L. Luo, “Hate speech detection: A solved problem? The challenging case of long tail on Twitter,” Semant. Web, vol. 10, no. 5, pp. 925–945, 2019.