11-2020

Leveraging profanity for insincere content detection: A neural network approach

Swapna GOTTIPATI
Singapore Management University, SWAPNAG@smu.edu.sg

Annabel TAN
Singapore Management University, annabel.tan.2017@sis.smu.edu.sg

David Jing Shan CHOW
Singapore Management University, david.chow.2017@sis.smu.edu.sg

Joel Wee Kiat LIM
Singapore Management University, joel.lim.2017@sis.smu.edu.sg

Citation

GOTTIPATI, Swapna; TAN, Annabel; CHOW, David Jing Shan; and LIM, Joel Wee Kiat. Leveraging profanity for insincere content detection: A neural network approach. (2020). *2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON): November 4-7, Virtual: Proceedings*. 41-47.

Available at: https://ink.library.smu.edu.sg/sis_research/5666

Leveraging Profanity for Insincere Content Detection - A Neural Network Approach

Swapna Gottipati, Annabel Tan, David Chow Jing Shan, Joel Lim Wee Kiat
School of Information Systems
Singapore Management University
*swapnag,annabel.tan.2017,david.chow.2017,joel.lim.2017@smu.edu.sg

***Abstract*—Community driven social media sites are rich sources of knowledge and entertainment and at the same vulnerable to the flames or toxic content that can be dangerous to various users of these platforms as well as to the society. Therefore, it is crucial to identify and remove such content to have a better and safe online experience. Manually eliminating flames is tedious and hence many research works focus on machine learning or deep learning models for automated methods. In this paper, we primarily focus on detecting the insincere content using neural network-based learning methods. We also integrated the profanity features as profanity is correlated with honesty according to psychology research. We tested our model on the questions datasets from CQA platform to detect the insincere content. Our integrated neural network model enabled us to achieve a high performance of F1-score, 94.01%, compared to the standard machine learning algorithms.**

Keywords-Social media, insincere content, profanity, neural networks, classification.

I. INTRODUCTION

Social media platforms are rich sources of knowledge and entertainment [1]. These platforms include social networks such as; kiwibox.com (teen magazines), Raverly (arts), and cellufun (gaming), encyclopedias such as; Wikipedia, Britannica, and community question answering (CQA) platforms such as; Quora (community topics) and Stack Overflow (programmers). They are the sources of information on a variety of topics useful for various groups of users. As with any social media websites, these knowledge resources are vulnerable to flames or toxic content threats such as fake or insincere content [2, 3], the emergence of hate and conflict posts [4, 5, 6], and obscene, profanity or illegal language [7].

As a crowd-sourced service, such platforms rely on their users for monitoring and flagging content that violates community rules. Users can report plagiarism, harassment, spam, and factually incorrect articles, etc. The common wisdom is to eliminate the users who receive many flags of violation of rules. According to Kayes et al, a mature Q&A site showed that users with many flags may still contribute positively to the community [8]. On the other hand, users who never get flagged are found to violate community rules and get their accounts suspended. This raises the dire need of automated techniques to flag the undiscovered toxic content and aid the site managers to improve the quality of the online content in community social media.

Several researchers provided NLP, machine learning, and deep learning based techniques to detect toxic content [4, 9, 10]. In this paper, we focus on detecting insincere content from social media data. The current works used the basic features of the language to detect and applied the ML models for classification. We argue that combining external features such as profanity will enable us to provide higher performance. We adopted the idea from Zhang et al. [11] and Poria et al. [12] who applied ML for irony or sarcasm detection by leveraging on features such as sentiments and emotions. De Vries et al. found that there is a positive relationship between profanity and dishonesty [13]. According to their study swear words are often associated with deceit.

In this paper, we propose the neural network based classifier that leverages the profanity to discover the insincere content. We evaluated our model on the CQA platform Quora dataset of questions [14]. Quora is one of the most popular question forum and plays a key role in motivating people to learn from each other. At the same time, the site managers must make sure it is safe for people from all over the world to share knowledge following the rules of integrity. However, the problem of insincere content is common and they created the dataset of insincere questions. These are questions that are created under false ground and not indented to look for helpful answers. In classifying the content to sincere and insincere, our model has provided F1-score of 74.2% without using the profanity features. Integrated with the profanity features, the model has achieved an F-score 95.07%.

The rest of the paper is organized as follows. Section II presents the related work on toxic content detections and models used by various researchers. In section III, we describe our solution model along with the justification of the choice of the techniques at each stage. Section IV describes the experiments at each stage, analysis of results, and limitations of the model, and we conclude in Section V.

II. LITERATURE SURVEY

Content in social media can be categorized into regular text or flame text. Flame text can be very generic or context-based. In Debatepedia, we might encounter an aggressive language regarding sociopolitical topics [29]. However, given the context, one might argue that it is acceptable to social media users. Therefore, it becomes crucial to have a clear definition of flames text and the context before studying the auto detection models. In this survey, we first present the detection of flames in a generic context followed by a more specific context.

In a general context, research on flames in social media aims to detect any offensive language with the standard scores for the vocabulary. The offensive language aims to mock or insult somebody or a group of people. The common attacks include aggression against some culture, a subgroup of the society, race or ideology. Detecting offensive language in a general context such as obscene, pejoratives, profanity, etc., can be achieved with lexicon methods [15] or classification methods [16].

According to the above lexicons, profanities are labeled as strongly offensive. Pejoratives and obscenities receive the label of strongly offensive with certain conditions. In classification methods, the language aspects such as syntactic and lexical features and sentence aspects such as structure and style play role in high performance. Based on this study, we choose profanity as an indicator to improve our model performance.

In a specific context, flames in social media include, site content-based or stakeholder-based. Stakeholders like adolescents or females are attacked with the flames text and detecting such language in the given sensitive context is crucial. The flames detection methods involve combining factors related to age, gender, culture, etc., to the machine learning algorithms [16].

On the other hand, site content includes; news, reviews, speeches, question & answers, etc. Our work is related to site content-based. We specifically study the content in the question & answers community sites,

Fake content in news and reviews are being promoted on social media platforms to deceive the public for ideological or financial gain. Issues related to fake information such as stories, news, pictures, and its impact in the digital environment is a key concern to public debate due to the internet's role in modern societies. It can be categorized as fact-based (news) and opinion-based (reviews) [2]. Thota et al. applied a neural network approach for fake news detection together with simple NLP and TF-IDF based neural network achieved an accuracy of 94.31% [9]. With deep learning techniques applied to various review datasets, Shahariar et al. achieved an accuracy of more than 95%. In our solution, we adopted similar ideas and compared them with the standard ML techniques for analysis [17].

Anonymity in social media attribute to the hate speech and eventually hate crime. Hate speech is categorized into main classes such as race, colour, ethnicity, gender, sexual orientation, nationality, religion, or other social characteristics. Detecting hate content has been a very popular research area due to its impact on society and the researchers employ semantic content analysis techniques built on Natural Language Processing (NLP) and Machine Learning (ML) methods. Lexical methods also play an important role in hate detection. Using neural networks, together with the semantic and sentiment features, Zhang & Luo, achieved an accuracy of 94% in hate content detection [18]. However, even with techniques of combining POS features and sentiment lexicons, the performance of hate detection is lower when sub-categorizing it from the offensive language content [10].

Insincerity and spam content is a growing concern in community based social media sites such as CQA. To handle spam or offensive answers LDA based expert systems are used by Riahi et al [19]. The goal is to use the experts who are reliable to handle new questions and thus preventing the offensive answers or spam generation. However, users do post several insincere questions and should be filtered before directing to the expert systems to maintain the quality of such community social media sites.

Current models focused only on the ML or deep learning methods and basic text features for insincerity detection [3]. De Vries et al. from psychology explored the relationship between profanity and honesty [13]. Dishonesty involves the conscious attempt by a person to convince others of a false reality. According to their study with lie and impression management scales profanity was negatively associated with less lying and deception at the individual. In our solution, we employ the neural network-based algorithms together with the profanity features to detect the insincere content.

III. SOLUTION APPROACH

Figure 1 shows the overall solution approach for insincere content detection. Ensembling or stacking methods are procedures designed to increase predictive performance by blending or combining the predictions of multiple machine learning models. The idea is to stack them up to produce a final prediction[1].

Three main machine learning components are combined/integrated into the solution design for better prediction performance of the tool; sincerity classifier, profanity classifier, and profanity based sincerity classifier.

[1] https://www.kdnuggets.com/2017/02/stacking-models-imropved-predictions.html

The input to the model is the training dataset of sincere and insincere questions. To develop an efficient sincerity classifier, we evaluate multiple machine learning models and choose the best model. To develop a profanity classifier, we depend on the existing work [26] which generates high performance using the support vector machines. These two intermediary or level-1 classifiers generate probability scores on the sincerity and the profanity of the given input documents. These probability scores are the inputs features to train the integrated classifier, profanity based sincerity classifier. The output from the solution design is the integrated classifier that can detect sincere and insincere questions on the new dataset. The blue color boxes indicate the three classifiers and we also conducted the performance of each classifier in our experiments. We now describe each classifier and the algorithm used to build the classifier with the motivation.

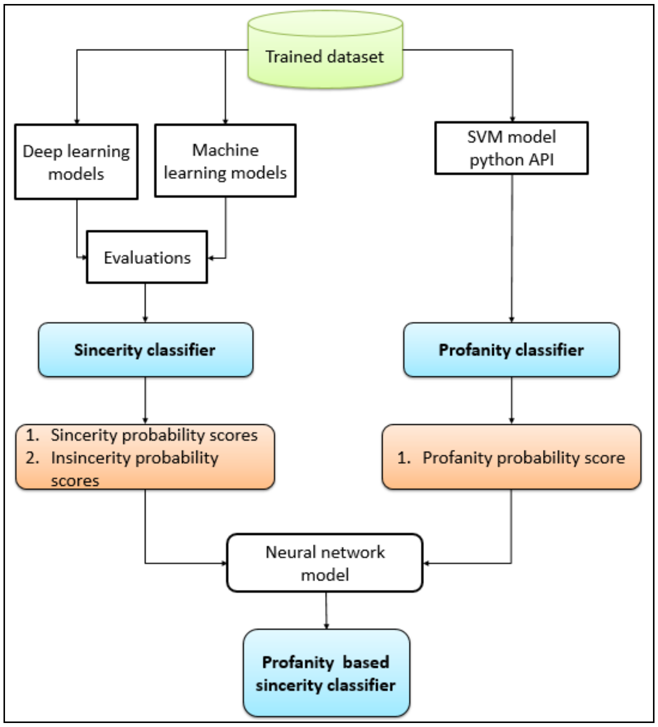


Figure 1. Solution model for the insincerity content detection

A. *Sincerity Classifier*

The sincerity classifier is trained on the set of sincere and insincere content. For a given document, it generates the below for each document;

- Sincerity probability score
- Insincerity probability score

To create sincerity classifier, we adopted ideas from the previous research on toxic content extraction and propose machine learning and neural network based learning models for classification. We choose the best classification model and this model aids in generating the probability scores for the next stage.

1) *Naïve Bayes*

Naïve Bayes models are popular in machine learning applications due to their simplicity in allowing each attribute to contribute towards the final decision equally and independently from the other attributes [20]. This simplicity equates to computational efficiency, which makes Naïve Bayes techniques attractive and suitable for many domains. The probability of a label value *c* given a document *d* is computed as;

$$P(c|d) = \frac{P(c)\prod_{w\epsilon d} P(w|c)^{n_{wd}}}{P(d)} \tag{1}$$

where $P(c|d)$ refers to the probability of document *d* belonging to class *c*, n_{wd} is the number of times word w occurs in document *d*, $P(w|c)$ is the probability of observing word w given class *c*, $P(c)$ is the prior probability of class *c*, and $P(d)$ is a constant that makes the probabilities for the different classes sum to one. $P(c)$ is estimated by the proportion of training documents pertaining to class *c* and $P(w|c)$.

2) *Log Regression*

Logistic regression statistical method is used for analyzing the dataset and produces a binary outcome [21]. It is a specific category of regression and it is used in the best way to predict the binary and categorical output. Similar to the previous method, the probability of a document *d* belonging to a class *c*, can be obtained using the following equation;

$$P(c|d) = \frac{1}{Z(d)} exp(\textstyle\sum_i \gamma_{i,c}\, f_{i,c}\,(d,c)) \tag{2}$$

where $f_{i,c}(d, c)$ is the feature or class function for feature f_i and class *c*, $\gamma_{i,c}$ is the parameter to be estimated and $Z(d)$ is the normalizing factor. In order to use log regression, a set of features is needed to be selected. For text classification purposes, word counts are considered as features.

3) *Stocastic Gradient Descent*

Stochastic Gradient Descent (SGD) is a simple yet very efficient approach to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines (SVM) and logistic regression [21]. This method is used when the training data size is observed to be large.

In SGD, instead of computing the gradient, each iteration estimates the value of the gradient and updates model parameters with a learning rate η. If η decreases slowly, the parameter estimate decreases equally slowly; but if the rate decreases too quickly, the parameter estimate takes a significant amount of time to reach the optimum point.

4) Neural network based classifier - fastText

fastText is an open-source, free, lightweight library that allows users to learn text presentation and text classifiers [22]. It works on standard, generic hardware and it is a library for efficient learning of word representations and sentence classification [23]. It is a library for the learning of word embeddings and text classification by Facebook's AI Research lab [22, 24]. For linear models such as SGD classifier and logistic regression, sentences are represented as a bag of words that is invariant to word order before being used as input. However, linear classifiers do not share parameters among features and classes. This can potentially limit the generalization in the context of output.

Meanwhile, for fastText, it uses a bag of *n*-gram as additional features to record some partial information about the local word order. Each word is represented as a bag of character *n*-grams in addition to the word itself, so the overall word embedding is a sum of these character *n*-grams [25].

Given a dictionary of n-grams of size G. Given a word w, let us denote by $G_w \subset \{1 \ldots G\}$ the set of *n*-grams appearing in w. A vector representation Z_g is associated to each n-gram g. A word is represented by the sum of the vector representations of its n-grams. Finally, the scoring function is obtained as follows;

$$s(w, c) = \sum_{g \epsilon G_w} z_g^T \, v_c \tag{3}$$

where c is the context and v_c is the context vector. Using the score for a pair (w. c), scores are assigned to a set of words for a context. Figure 2 shows the example word vectors generated for two words.

asparagus 0.46826 -0.20187 -0.29122 -0.17918 0.31289 -0.31679…

yellow -0.39965 -0.41068 0.067086 -0.034611 0.15246 -0.12208...

Figure 2. Sample word vectors generated by fastText model.

As a result, fastText can generate better and reliable word embeddings for rare words. Vectors for words (out of vocabulary words) that do not appear in training corpus can also be constructed from their respective character *n*-grams. These word representations will then be fed into a softmax regression model which is a generalized multinomial logistic regression [25]. The settings of the fastText in our experiments are explained in Section IV.

B. Profanity Classifier

Instead of thoughtfulness in the posts, users usually tend to fall towards profanity. Thoughtful comments can be detected using various text mining methods [30]. Similarly to detect profanity, NLP and text mining methods can be applied. Profanity-Check is a fast-robust Python library to check for profanities or offensive languages in strings [26]. It uses a linear SVM model trained on a dataset of 200k human labeled data. Profanity-Check classifier uses a bag of words model to vectorize input strings before feeding them to train the linear classification model. The input to the model is a document and the output from the function, *predict_prob*() is the probability score of profanity.

C. Profanity based Sincerity Classifier

Keras[2] neural network library is to provide high-level building blocks for developing deep learning models. Keras provides three backend implementations: TensorFlow, Theano, and CNTK. In this paper, we use the Keras neural network library with TensorFlow backend and build the sequential model with a linear stack of layers to the constructor.

The activation argument is typically applied after each convolutional layer. In the Keras neural network library, there are ten available activation types. R*elu* is a variant of the nonlinear rectified linear unit. We used this due to its simplicity and the ability for fast training. The inputs to the sequential model are the three probability scores from the previous classifiers. The other settings of this network are discussed in Section IV.

IV. EXPERIMENTS & RESULTS

In this section, we first describe the dataset and the pre-processing steps. We evaluate the results of each model for the sincerity classifier. We use the standard F-score to compare the models and evaluate the performance. Finally, we present discussions of our findings and limitations of the work.

A. Dataset

The dataset for model evaluations is collected from Kaggle's competition: Quora insincere questions classification [14]. We train our text analytics learning algorithm on the training set, train.csv, which consists of the following three columns: qid – unique question identified, question_text – Quora question text and target – question labeled "insincere" having a value of 1, otherwise 0.

We use the train.csv dataset provided by Quora to train, validate, and test our models. This is because the testing dataset from Quora has no labels provided to train or test our models. The dataset was split into 60-20-20, which represent

[2] https://keras.io/

the percentages for training, testing, and validation, respectively.

We train our classifier using this labeled training dataset and test the performance of the trained classification model by predicting labels of the testing dataset. In addition to evaluating the sincerity classifier in the first stage, we evaluate the total model with the validation dataset.

B. *Data sampling*

We first perform a simple Exploratory Data Analysis (EDA) on the dataset for statistical data analysis. We observe that there is a high imbalance in the two classes before sampling as shown in Table I. We have approximately 6% of data that is classified as insincere.

Uneven distribution of sincere and insincere questions may affect the performance of the models [27]. Therefore, we choose to use the up-sampling technique to handle this situation. Without up-sampling, the unevenly distributed data may affect the learning of our model, making it bias at predicting label 0 but not label 1 as it was trained on more sincere questions. Thus, the classification of sincere and insincere questions would be inaccurate.

Up-sampling is the process of randomly duplicating observations from the minority class in order to reinforce its signal. Table I shows the final dataset statistics after up-sampling using sklearn's resample API [27].

TABLE I. DATASET STATISTICS

Sampling status	**#Sincere docs**	**#Insincere docs**
Before up-sampling	734881	48792
After up-sampling	734881	734881

We observe that the final dataset has more observations than the original. We split this data into training, testing, and validation as described in Section IV. A for evaluation out solution model.

C. *Pre-processing*

Besides up-sampling the data, common pre-processing steps such as removal of stop words and lemmatization were not performed on the dataset at this stage. This is because pre-processing the words might cause the potential and valuable meaning of a sentence to be lost [28]. In our preliminary experiments, we observed this phenomenon.

D. *Sincerity Classifier Results*

We trained our four models mentioned in Section III with the same up-sampled training dataset. Since the results were not conclusive and reliable, we have conducted a round of validation on the validation dataset, separated as 20% from the training dataset.

For fastText, we set epochs=100 and ngrams=bigrams. Since we are looking for a balance of precision and recall, we apply F1-Score evaluations for our study. Using F1-score as the evaluation metric across the different models, the validation results are depicted in Table II.

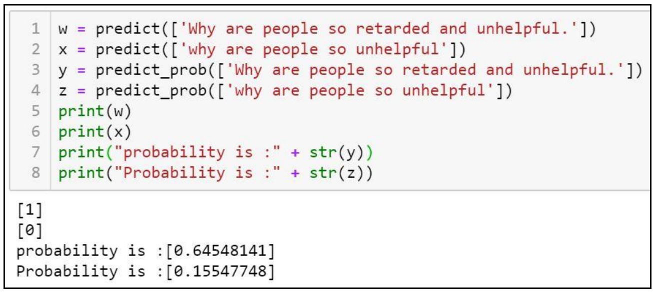
TABLE II. SINCERE CLASSIFIER EVALUATIONS

Model	**F1 Score**
Naïve Bayes	0.359
Logistic Regression	0.591
Stochastic Gradient Descent (SGD)	0.461
fastText	**0.742**

Recall that we didn't apply other language features of the data and this resulted in a lower F1-score compare to other similar works [3, 13]. From Table II, we observe the fastText has the best performance compared to other models and these results are consistent with other similar works.

E. *Profanity Classifier*

Recall that the profanity-check classifier has an ability to generate the probability scores. Figure 3 shows the output of the classifier for two sentences.



```
w = predict(['Why are people so retarded and unhelpful.'])
x = predict(['why are people so unhelpful'])
y = predict_prob(['Why are people so retarded and unhelpful.'])
z = predict_prob(['why are people so unhelpful'])
print(w)
print(x)
print("probability is :" + str(y))
print("Probability is :" + str(z))
```

```
[1]
[0]
probability is :[0.64548141]
Probability is :[0.15547748]
```

Figure 3. Profanity classification evaluations

From Figure 3, we observe that the word "retarded" is classified as profane and hence the profanity probability score is higher than the second sentence. Further, we also argue that profanity plays a key role in detecting insincerity content [13]. Therefore, we leverage on the profanity scores for our final model.

F. *Profanity based Sincerity Classifier*

From the sincerity classifier based on fastText, we obtain the sincere probability and insincere probability scores for each question. Together with the profanity probability score generated from the profanity classifier, we proceed to train a neural network. After several trials and errors to obtain the best result, we have finally implemented a Keras neural network that consists of 4 layers with the below settings for better performance.

1. Input layer with three input nodes
2. Two hidden layers
    - First hidden layer with 12 nodes

- Second hidden layer with 8 nodes output layer uses a sigmoid function as its activation function

We tested our classifier with the validation dataset. Figure 4 shows the experiment design.

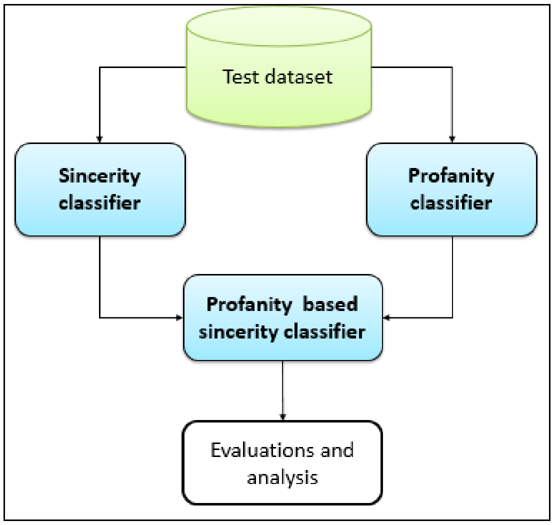


Figure 4. Experiment model to evaluate profanity based classifier on the new dataset

Our model was trained with 100 epochs and the model performed with the F-score of **0.9507**.

G. Discussions

Sincere probability scores are higher compared to the previous research works due to the up-sampling technique and at the same time, fastText handles two limitations for the linear classifiers; a bag of words and unigrams representation. This is the main reason why our model with fastText performs better than the other models, with logistic regression leading next.

For the detection of profanity in our dataset, we observe that Profanity-Check is able to detect insincere words. However, since the profanity classifier uses the bag of words model, it does not take into consideration the context of the sentence. This may affect the quality of the classification model as a question, for example, “What is the effect of a retardation curve?” would be classified as vulgar even though it is not. The word “retardation” does not indicate negativity, but it means slowing or going down. Hence, the profanity feature generated from the profanity classifier may not be accurate and it may affect the final result negatively when it was used for classification.

Another potential reason why the profanity classifier may not be fully reliable is because of new words that are not listed in the lexicon. The dataset used was from 2018, and there could be new words that are considered insincere today and specific to the context. Therefore, if a document containing new vulgar or inappropriate words is introduced to the model, the chances that it may be classified wrongly are high. A suggestion to improve this is to add newer words to the profanity classifier and retrain the classifier.

To improve the performance, one may argue to use other features tested by previous works such as sentiments, emotions, language aspects, etc. Training such models and applying in real scenarios is affected by the context and has a big impact on the time. Therefore, we argue to choose context-related and impactful features. Our model provides a simple yet powerful example of context-based feature implementation. For example, in the context of Quora, we would like to include other additional feature which would classify a question as insincere when it contains religious content. This is because, in our human analysis, religious content seems to be a topic with flames, and combining this observation in the model may aid the model performance.

V. Conclusion

To detect the insincere content, we proposed a profanity based deep learning model which not only provided high accuracy but also outperformed other commonly used machine learning algorithms. The psychology studies show the positive correlation between profanity and dishonesty and we explored this finding to build a profanity integrated sincerity detection classifier based on neural network models where the word embeddings play an important role in improving the performance of the tool. Our model tested on the community social media data, CQA, performed with an F-score of 0.9507 and outperformed the machine learning algorithms. Our study also shows how the scores generated from stage one classifiers can be integrated to train a stage two neural network model, to classify textual content. Moreover, fastText is a dedicated classification algorithm which is faster for training the model compared to the deep learning classifiers.

Acknowledgment

We thank Kymberlyn Ng Ting Ting, Peh Anqi, Thin Xin Hui for the assistance in this research study

References

[1] Zembik, M. (2014). Social media as a source of knowledge for customers and enterprises, Online Journal of Applied Knowledge Management 2(2): 132–148

[2] Kumar, Srijan & Shah, Neil. (2018). False Information on Web and Social Media: A Survey.

[3] Mediratta, Deepshi & Oswal, Nikhil. (2019). Detect Toxic Content to Improve Online Conversations.

[4] N. Bauwelinck and E. Lefever. (2019) Measuring the impact of sentiment for hate speech detection on Twitter, in Proceedings of HUSO 2019, The fifth international conference on human and social analytics, Italy, pp. 17–22.

[5] Van Hee, Cynthia & Lefever, Els & Verhoeven, Ben & Mennes, Julie & Desmet, Bart & Pauw, Guy & Daelemans,

Walter & Hoste, Veronique. (2015). Automatic detection and prevention of cyberbullying. HUSO , 2015

[6] Kulaszewicz, Kassia E.. (2015). Racism and the Media: A Textual Analysis. Retrieved from Sophia, the St. Catherine University repository website: https://sophia.stkate.edu/msw_papers/477

[7] Yar, Majid (2018). A Failure to Regulate? The Demands and Dilemmas of Tackling Illegal Content and Behaviour on Social Media, International Journal of Cybersecurity Intelligence & Cybercrime: 1(1), 5-20.

[8] Kayes, Imrul & Kourtellis, Nicolas & Quercia, Daniele & Iamnitchi, Adriana & Bonchi, Francesco. (2015). The Social World of Content Abusers in Community Question Answering. 10.1145/2736277.2741674.

[9] Thota, Aswini; Tilak, Priyanka; Ahluwalia, Simrat; and Lohia, Nibrat (2018) "Fake News Detection: A Deep Learning Approach," SMU Data Science Review: Vol. 1 : No. 3 , Article 10.

[10] Thomas Davidson, D. W. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. Retrieved from Cornell University: https://arxiv.org/pdf/1703.04009.pdf

[11] Zhang, Shiwei & Zhang, Xiuzhen & Chan, Jeffrey & Rosso, Paolo. (2019). Irony detection via sentiment-based transfer learning. Information Processing & Management. 56. 1633-1644. 10.1016/j.ipm.2019.04.006.

[12] Poria, S., Cambria, E., Hazarika, D., & Vij, P. (2016). A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks. ArXiv, abs/1610.08815.

[13] De Vries, R. E., Hilbig, B. E., Zettler, I., Dunlop, P. D., Holtrop, D., Lee, K., & Ashton, M. C. (2018). Honest People Tend to Use Less-Not More-Profanity: Comment on Feldman et al.'s (2017) Study 1. Social psychological and personality science, 9(5), 516–520.

[14] Quora. (2019). Quora Insincere Questions Classification. Retrieved from Kaggle: https://www.kaggle.com/c/quora-insincere-questions-classification/data

[15] A. Razavi, D. Inkpen, S. Uritsky, and S. Matwin. (2010) Offensive language detection using multi-level classification, Advances in Artificial Intelligence, vol. 6085/2010, pp. 16-27.

[16] Y. Chen, Y. Zhou, S. Zhu, and H. Xu. (2012) Detecting offensive language in social media to protect adolescent online safety. In Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference. IEEE and 2012

[17] Shahariar, G. & Biswas, Swapnil & Omar, Faiza & Shah, Faisal & Hassan, Samiha. (2019). Spam Review Detection Using Deep Learning. IEE IEMCON.

[18] Ziqi Zhang and Lei Luo. (2018). Hate speech detection: A solved problem? the challenging case of long tail on twitter. CoRR, abs/1803.03662

[19] Riahi, Fatemeh & Zolaktaf, Zainab & Shafiei, Mahdi & Milios, Evangelos. (2012). Finding expert users in Community Question Answering. WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web Companion.

[20] T. Mitchell. (1997) Machine Learning. McGraw-Hill Science/Engineering/Math, 1st edition, 1997.

[21] Christopher M. Bishop. (2006) Pattern Recognition and Machine Learning.

[22] Joulin A, Grave E, Bojanowski P, Mikolov T. (2017) Bag of tricks for efficient text classification. 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, p. 427–431.

[23] fastText. (9 November, 2015). fastText. Retrieved from fastText: https://fasttext.cc/

[24] Piotr Bojanowski and Edouard Grave and Armand Joulin and Tomas Mikolov. (2016). Enriching Word Vectors with Subword Information https://www.aclweb.org/anthology/Q17-1010.pdf

[25] Abu-Rmileh, A. (2019). How does FastText classifier work under the hood? Retrieved from Towards Data Science: https://towardsdatascience.com/fasttext-bag-of-tricks-for-efficient-text-classification-513ba9e302e7

[26] PyPI. (2018). profanity-check 1.0.3. Retrieved from PyPI: https://pypi.org/project/profanity-check/

[27] Lemaître, Guillaume & Nogueira, Fernando & Aridas, Christos. (2016). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. 18.

[28] Vallantin, L. (2019). Why is removing stop words not always a good idea. Retrieved from Medium: https://medium.com/@limavallantin/why-is-removing-stop-words-not-always-a-good-idea-c8d35bd77214

[29] Gottipati, S., Qiu, M., Sim, Y., Jiang, J., & Smith, N.A. (2013). Learning Topics and Positions from Debatepedia. EMNLP.

[30] Gottipati Swapna and Jing Jiang. Finding Thoughtful Comments from Social Media. (2012). Proceedings of COLING 2012: 24th International Conference on Computational Linguistics, 8-15 December 2012, Mumbai, India. 995-1010.