A comparison of deep learning models for hate speech detection

Eglė Kankevičiūtė[1,2,*], Milita Songailaitė[1,2,*], Justina Mandravickaitė[1,2,*], Danguolė Kalinauskaitė[1,2,*] and Tomas Krilavičius[1,2,*]

[1] Vytautas Magnus University, Faculty of Informatics, Vileikos street 8, LT-44404 Kaunas, Lithuania
[2] Centre for Applied Research and Development, Lithuania

Abstract

Hate speech is a complex and non-trivial phenomenon that is difficult to detect. Existing datasets used for training hate speech detection models are annotated based on different definitions of this phenomenon, and similar instances can be assigned to different annotation categories based on these differences. The goal of our experiment is to evaluate selected hate speech detection models for English language from the perspective of inter-annotator agreement, i.e. how the selected models "agree" in terms of annotation of hate speech instances.

For model comparison we used English dataset from HASOC 2019 shared task and 3 models: BERT-HateXplain, HateBERT and BERT. Inter-annotator agreement was measured with pairwise Cohen's kappa and Fleiss' kappa. Accuracy was used as additional metric for control. The experiment results showed that even if the accuracy is high, the reliability, measured via inter-annotator agreement, can be low. We found that the best accuracy in hate speech detection was achieved with BERT-HateXplain model, however, Cohen's kappa metric for the results of this model was close to 0, meaning that the results were random and not reliable for real life use. On the other hand, comparison of BERT and HateBERT models revealed that annotations are quite similar and they have the best Cohen's kappa score, suggesting that similar neural network architectures can deliver not only high accuracy, but also correlating results and reliability. As for Fleiss' kappa, a comparison of expert annotations and three selected models gave an estimate of a slight agreement, confirming that high accuracy can go together with low reliability of the model.

Keywords

Hate speech, deep learning, model comparison, HASOC 2019 dataset, English language

1. Introduction

Hate speech is a complex and non-trivial phenomenon that is difficult to detect. Online hate speech is assumed to be an important factor in political and ethnic violence such as the Rohingya crisis in Myanmar [1], [2]. Therefore, media platforms are pressured to timely detection and elimination of hate speech occurrences [3]. This tendency led to increasing efforts in terms of hate speech detection, and a number of hate speech detection models have been developed.

Existing datasets used for training hate speech detection models are annotated based on different definitions of this phenomenon, and similar instances can be assigned to different annotation categories based on these differences in perception of what constitutes hate speech. Analysis of the effects of definition on the annotation reliability led to the conclusion that hate speech phenomenon requires a stronger and more uniform definition [4]. Also, it was found out that most of the publicly available datasets are incompatible due to different definitions attributed to similar concepts [5]. Moreover, hate speech datasets can have very similar labels, so some studies merge them together into one class to reduce class imbalance [10]. However, this practice could make a negative impact on research as distinction between classes is necessary. For example, merging the offensive language and hate speech classes of [6] dataset in [3] and [12] or the racist language and sexist language classes of [11] dataset in [13] and [14]. In hate speech research abusive language or toxic comments can cover several paradigms [10], therefore following available definitions is very important. Similarly, it was suggested that offensive language is not the same as hate speech and therefore they should not be merged [6].

Following other authors, such as [6], [7], [8] and [9], the summarised definition of hate speech is the following: **hate speech** describes negative attributes or deficiencies of groups of individuals because they are members of a particular group. Hateful comments occur toward groups because of race, political opinion, sexual orientation, gender, social status, health condition, etc. As it was suggested in [6] and [9], offensive comments could be attributed to separate class and offensive language could be defined as an attempt of degrading, dehumanizing, insulting an individual and / or threatening with violent

IVUS 2022: 27th International Conference on Information Technology, May 12, 2022, Kaunas, Lithuania

*Corresponding author.
egle.kankeviciute@stud.vdu.lt (E. Kankevičiūtė);
milita.songailaite@stud.vdu.lt (M. Songailaitė);
justina.mandravickaite@vdu.lt (J. Mandravickaitė);
danguole.kalinauskaite@vdu.lt (D. Kalinauskaitė);
tomas.krilavicius@vdu.lt (T. Krilavičius)

acts.

As one of the reasons why it is difficult to detect hate speech is varied definitions in different studies [4],[5], a comparison of different hate speech detection models not in terms of performance but in terms of what is marked as hate speech could contribute to more comprehensive understanding of the phenomenon and its timely identification. Following the latter notion, the goal of this experiment is to evaluate selected hate speech detection models for English language from the perspective of inter-annotator agreement, i.e. how the selected models "agree" in terms of annotation of hate speech instances.

Section II presents methods used as well as experimental setup, Section III describes the data used in the experiment, Section IV reports the results, and Section V ends this paper with conclusions and future plans.

2. Methods and experimental setup

For our experiment we selected 3 popular hate speech detection models for English language and tested them on HASOC 2019 dataset. Our setup consisted of 4 "annotators" - results provided by aforementioned 3 models and annotations presented in HASOC 2019 dataset. The annotations mentioned were treated as "gold standard".

In the following sections, methods of data representation are presented (it was important for selecting hate speech detection models), and hate speech detection models as well as inter-annotator metrics used in our experiment are introduced.

2.1. Basic word embeddings

Perception of natural language from textual data is an important area of artificial intelligence. As images can be perceived as pixels for a computer, language also needs a way to be represented as textual data in a way that can be processed automatically. For example, the sentence *The cat sat on the mat* cannot be directly processed or understood by a computer system. One of the best methods to represent this for a computer is to convert the words into real numeric vectors - word embeddings [16]. Word embeddings associate each word in the vocabulary (a set of words) with a real-valued vector set in a predefined N-dimensional space (Fig. 1). After transforming the words or sentences into their embeddings, it is possible to model the semantic importance of a word in numerical form and thus to carry out mathematical operations [35].

This vector mapping can be learned using unsupervised methods such as statistical document analysis or by using supervised techniques, for example, neural network model developed for tasks such as sentiment analysis or document classification [38].

Figure 1: Projection of word and phrase insertions showing that words of similar meaning in space are adjacent [17]

The simplest way to represent words in numeric values is with One-Hot Encoding [24]. This method is one of the most popular and works well when there are not many different categories (up to 15 works best, although in some cases it may work poorly with fewer).

The single hot encoding is a method which creates new binary columns of categorical variables, where value of 1 indicates that the original data row belongs to that category. For example, we have the original data: red, red, red, red, yellow, green, yellow. For each possible value a separate column is created and where the initial value is red, we enter 1 in the corresponding column, while in the other columns 0s are inserted (Fig. 2) [36].

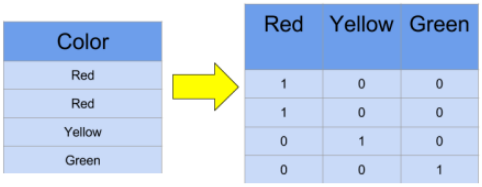

Figure 2: Example of a single hot encoding [18]

Although this method is simple and easy to learn, it has major drawbacks. Because we only give our computer ones and zeros, it cannot interpret any meaning from this data (calculating cosine similarity will always result in zero or near-zero values). This is where pre-trained word embeddings and BERT embeddings help and that is why they have become popular in variety of natural language processing tasks, including hate speech detection.

2.2. Pre-trained word embedding models

It is often an optimal solution to use pre-trained models for deep learning tasks. A pre-trained model is developed and trained by someone to solve a specific problem based on chosen data [37]. Using pre-trained models saves time spent on training the model or in search of efficient neural network architecture. Two main ways to use a pre-

trained model is fixed feature extraction or fine-tuning of the model and adapting it to the problem at hand [19].

The fine-tuning of the model is done in one step. Fig. 3 represents process, where each user-generated comment for hate speech detection is classified according to a fine-tuned BERT model[20].

The feature-based approach involves two steps. First, each text, for example, a user-generated comment, is represented as a sequence of words or subwords, and each word or the insertion of each subword is calculated using fastText or BERT models. Second, this sequence of insertions will form the input to the neural network (NN) classifier, where the final decision regarding label of the input text will be made (Fig. 3) [20]. For this task a variety of deep neural network (DNN) architectures can be used, for example, deep recurrent neural network (RNN) [31], deep convolutional neural network (CNN) [33], gated recurrent unit (GRU) [3], long short-term memory (LSTM) [34], etc. The most suitable architecture usually is selected via experiments and by combining more than one architecture for the task.

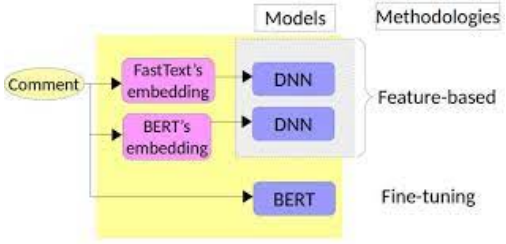

Figure 3: Illustrative explanation of the feature-based and fine-tuning methodologies [20]

2.3. BERT embeddings

BERT - Bidirectional Encoder Representations from Transformers, released in 2018 by Google AI Language researchers. BERT features the state-of-the-art performance on most NLP problems [25]. BERT word embeddings can take one or two sentences as input and use a special token [SEP] to separate them. The [CLS] token is always placed at the beginning of the text and is a characteristics of classification tasks. These characters are always required, even if we have only one sentence or if we are not using BERT model for classification tasks [35] as it helps the algorithm to distinguish between different sentences.

Thus, for BERT model to be able to distinguish between words, there are normally three main steps. First, as mentioned above, the [SEP] and [CLS] characters are added at the beginning and at the end of the sentence. Next, an index is specified for each word and, finally, for sentences that are shorter than the longest sentence, zeros are added, i.e. the lengths of the sentences are made equal. This step is called padding [25]. Next, word embeddings are used - taking each word for each of them one specific vector is assigned. Each value of these vectors represents one aspect of the words (Fig. 4).

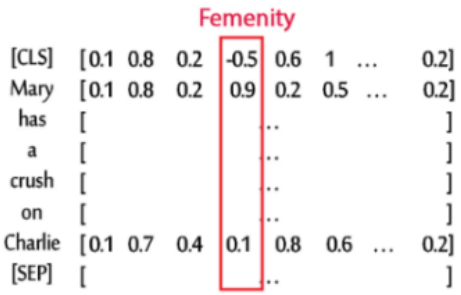

Figure 4: Three steps before word embeddings [21]

BERT is based on the transformer architecture, therefore it uses the attention mechanism. Attention is a way of looking at the relationship between the words in each sentence, and it allows for BERT to take into account a very large amount of context of a concrete size, both from the left and the right of a particular word [20]. By examining the working principle of BERT word embeddings, it can be seen that when inputting an English word with an ambiguous meaning, for example, *crush*, BERT model can understand that this is a word with several different meanings (each word is inserted according to the context in which it was used). On the other hand, in Word2Vec or fastText based models every word has a single meaning (it specifies only one vector for all the different meanings of this word) [36].

In addition, BERT uses tokenization of word parts or subwords. For example, the English word *singing* can be represented as two strings: *sing* and *ing*. The advantage of this is that when a word is not in the BERT dictionary, it can be split into parts to produce rare word embeddings [20]. This type of embeddings was used in all 3 chosen hate speech detection models.

2.4. Selected hate speech detection models

For our experiment we selected three BERT models which were differently pre-trained for the hate speech recognition task:

- BERT-HateXplain.[1]

[1]Available at https://github.com/hate-alert/HateXplain.

- HateBERT.[2]
- BERT.[3]

The selected models were trained on different datasets and used for classifying texts as either hate speech, offensive or non-hate. BERT model was trained using tweets from Twitter [30], BERT-HateXplain also was trained using Twitter and, additionally, Gab[4]. Moreover, Human Rationales were included as part of the training data to boost the performance [29]. HateBERT model was trained using RAL-E: the Reddit Abusive Language English dataset [30].

2.5. Inter-annotator agreement

In linguistics inter-annotator agreement is a formal means of comparing annotator performance in terms of reliability [26]. The annotation guidelines define a correct annotation for each relevant instance. As the actual annotations are created by the annotators, there is no reference dataset against which to check if the annotations are correct. Therefore, common practice is to check for reliability of the annotation process, assuming that if the annotation process is not reliable, then annotations cannot be expected to be correct.

For our experiment, we chose inter-annotator agreement to evaluate how the selected models for hate speech detection "agree" in terms of annotation of hate speech instances. We selected Cohen's kappa, Fleiss' kappa and Accuracy metrics.

Accuracy is one of the metrics for evaluating classification models. Having more than two classes, the targets are calculated as part of the correctly predicted sample in the test set, divided by all predictions made in the test set hey [39]

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \quad (1)$$

Cohen's kappa is commonly used for measuring the degree of agreement between two raters on a nominal scale. This coefficient also controls for random agreement [28]. Cohen's kappa has value 1 for perfect agreement between the raters and value 0 - for random agreement. As we compared more than 2 models ("annotators"), we used pairwise Cohen's kappa (2). Fleiss' kappa (3) is used for analyzing agreement between more than two raters rating nominal categories [27] and its value for perfect agreement is 1, while 0 marks random agreement.

$$cohen(\kappa) = \frac{p_0 - p_e}{1 - p_e} \quad (2)$$

$$fleiss(\kappa) = \frac{\overline{P} - \overline{P}_e}{1 - \overline{P}_e} \quad (3)$$

3. Data

For model comparison we used English dataset from HASOC 2019 shared task[5]. The data source is Twitter, and the data was sampled using keywords or hashtags relevant for hate speech [15]. All the tweets were annotated by 2 annotators. When there was a mismatch in the annotation between annotators, the tweet was assigned to a third annotator. The dataset has been labelled with 5 classes:

- NOT - Non Hate / Non Offensive Content: posts with no hate, profane or offensive content.
- HOF - Hate Speech and Offensive Language: posts with hate, offensive or profane content.
- HATE - Hate Speech: posts contain hateful content.
- OFFN - Offensive Language: posts contain offensive content.
- PRFN - Profane Language: posts contain profane words but hate or offensive content is absent.

We have chosen 3 of these classes for evaluation, namely, NOT, HATE and OFFN, as these were the classes our selected models were trained to identify. PRFN (profane language) class was merged with NOT (non hate / non offensive content) as it did contain neither HATE (hate speech) nor OFFN (offensive language) content [32]. The number of records assigned to each class is shown in Table 1.

The dataset has 2 subsets - training subset (5852 posts) and test subset (1153 posts). We performed evaluation on these subsets with different models separately.

Table 1
Distribution of classes

Data Subset	NOT posts	HATE posts	OFFN posts
Training	4042	1443	667
Testing	958	124	71

4. Results

We used *disagree*[6] library developed for the Python programming language. It was used to calculate the number of disagreements between three models and expert annotations. This makes it easier to understand how hate

[2] Available at https://github.com/tommasoc80/HateBERT.
[3] Available at https://github.com/google-research/bert.
[4] Gab is American microblogging and social networking service. Available at https://gab.com.
[5] Available at https://hasocfire.github.io/hasoc/2019/dataset.html.
[6] Available at https://github.com/o-P-o/disagree/.

speech is treated by each of the selected models. After reviewing the data, it was found that the most coincidences are in those comments that are marked as NOT (non hate speech or non offensive language). For models and experts, it is easier to distinguish these types of comments because of the large amount of comments with this label present in the dataset. The biggest discrepancies are observed where the content contains hate speech (HATE) (Table 2).

Table 2
Disagreements in annotations

Data Subset	All match	2 do not match	3 do not match
Training	2919	2556	377
Testing	802	298	52

After calculating Accuracy of the models, it was observed that BERT-HateXplain model has the highest estimate, which reaches almost 68 percent using the training subset. Accuracy becomes even greater when using the testing subset, in this case accuracy stands at nearly 82 percent. However, all models do not differ by a large percentage, as HateBERT model reached 77 percent and BERT model with 75 percent had the lowest Accuracy score using the testing subset (Fig. 5 and Fig. 6).

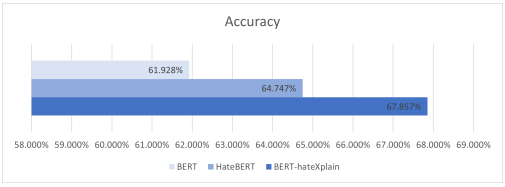

Figure 5: Accuracy of training subset

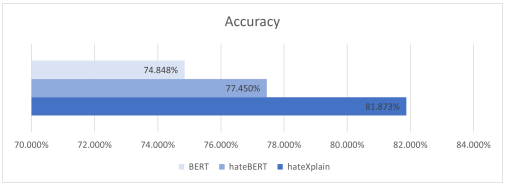

Figure 6: Accuracy of testing subset

From the results obtained, it can be seen that with larger amount of data Accuracy percentage drops down. It is also important to note that there is a small amount of OFFN (offensive) and HATE (hate speech) comments in the test data subset and for that reason it is easier for the model to achieve higher accuracy.

Cohen's kappa coefficient is a quantitative measure of two evaluators (annotators) evaluating the same thing, a measure of reliability adjusted for how often annotators agree. A coefficient value of 0 means that the consensus of the evaluators is random, and 1 means that the evaluators fully agree [26]. It is possible for the statistic to be negative, which can occur by chance if there is no relationship between the ratings of the two raters, or it may reflect a real tendency of the raters to give differing ratings [22].

When this metric is applied to the models, the best result was obtained between BERT and HateBERT models. BERT-HateXplain model has a coefficient of almost 0 (0.007), indicating that most consensus is random and that the model is not reliable, even though Accuracy is high. However, all models have a relatively low Cohen's kappa coefficient (Fig. 7 and Fig. 8), therefore it would be incorrect to rely on the results of these models for automated hate speech detection without taking into account their limitations.

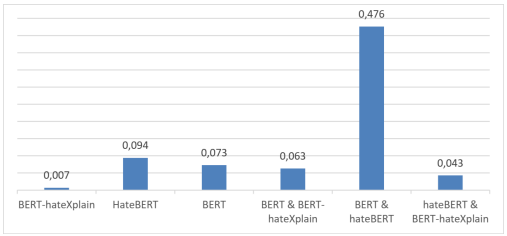

Figure 7: Cohen's kappa for training subset

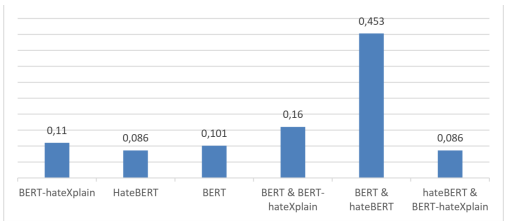

Figure 8: Cohen's kappa for testing subset

We have also calculated Fleiss' kappa coefficient, which is defined as extended the case of Cohen's kappa, where the annotations of more than two evaluators can be compared. A comparison of expert annotations and 3 selected models gave an estimate of 0.122 using training subset and 0.163 for testing subset. According to [23], such Fleiss' kappa ratio refers to a slight agreement.

The results showed that the selected models, namely, BERT, HateBERT and BERT-HateXplain, which are trained on English datasets, are not very reliable. Although selected models are popular in hate speech re-

search, when evaluated against selected inter-annotator agreement metrics, it can be seen that their performance is not enough to solve the hate speech detection tasks.

5. Conclusions and future plans

In this paper, we presented an inter-annotator agreement for hate speech detection tasks between three different BERT models using HASOC 2019 dataset. The experiment results showed that it is not correct to rely only on Accuracy metric, even if Accuracy percentage is high, because the reliability could be low. To check if the model is reliable we chose Cohen's kappa and Fleiss' kappa. In our selected models we found that the highest Accuracy was achieved with BERT-HateXplain model, even so, when calculating the Cohen's kappa metric the estimate was almost 0, which means that model's results were random and were not reliable for real life use. However, comparing BERT and HateBERT models we saw that annotations are quite similar, and their Cohen's kappa metric result suggests that similar neural network architectures can deliver not only high accuracy, but also correlating results and reliability. As for Fleiss' kappa, a comparison of expert annotations and three selected models gave an estimate of a slight agreement (0.122 for training subset and 0.163 for testing subset), confirming that high Accuracy can go together with low reliability of the model.

Our future plans include wider model testing with different annotation schemes (e.g. distinguish profane language, sexist language, misogyny, etc.) and data sources as well. We also plan to test models for different languages, e.g. Russian, Spanish, German, French, etc. We plan to use the knowledge gained from this experiment for developing hate speech detection model for Lithuanian language as well.

References

[1] Reuters. 2018. Why Facebook is losing the war on hate speech in Myanmar. https://www.reuters.com/investigates/special-report/myanmar-facebook-hate/. (Accessed on 10/03/2022).

[2] M. A. Rizoiu, T. Wang, G. Ferraro, & H. Suominen, "Transfer learning for hate speech detection in social media," arXiv preprint arXiv:1906.03829, 2019.

[3] Z. Zhang, D. Robinson, and J. Tepper, "Detecting hate speech on Twitter using a convolution-gru based deep neural network," In European semantic web conference, pages 745–760. Springer, 2018.

[4] B. Ross, M. Rist, G. Carbonell, B. Cabrera, N. Kurowsky, & M. Wojatzki, "Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis," arXiv e-prints, arXiv-1701, 2017.

[5] P. Fortuna, J. Soler, & L. Wanner, "Toxic, hateful, offensive or abusive? what are we really classifying? An empirical analysis of hate speech datasets," In Proceedings of the 12th language resources and evaluation conference, pp. 6786–6794, 2020.

[6] T. Warmsley, M. Macy, I. Weber, "Automated hate speech detection and the problem of offensive language," In Proceedings of the eleventh international conference on web and social media, AAAI, pp 512–515, 2017.

[7] A. Schmidt, & 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, Association for Computational Linguistics (ACL) pp. 1–10, 2017.

[8] P. Fortuna, & S. Nunes, "A survey on automatic detection of hate speech in text," ACM Computing Surveys (CSUR), 51(4), pp. 1–30, 2018.

[9] S. Modha, T. Mandl, G. K. Shahi, H. Madhu, S. Satapara, T. Ranasinghe, & M. Zampieri, "Overview of the hasoc subtrack at fire 2021: Hate speech and offensive content identification in English and Indo-Aryan languages and conversational hate speech," In Forum for Information Retrieval Evaluation, pp. 1–3, 2021.

[10] K. Madukwe, X. Gao, & B. Xue, "In data we trust: A critical analysis of hate speech detection datasets," In Proceedings of the Fourth Workshop on Online Abuse and Harms, pp. 150–161, 2020.

[11] Z. Waseem, & D. Hovy, "Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter," In Proceedings of the North American chapter of the association for computational linguistics: Human language technologies 2016, Association for Computational Linguistics (ACL), pp. 88–93, 2016.

[12] Z. Zhang, & L. Luo, "Hate speech detection: A solved problem? the challenging case of long tail on Twitter," Semantic Web, 10(5), pp. 925–945, 2019.

[13] H. Watanabe, M. Bouazizi, & T. Ohtsuki, "Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection," IEEE access, 6, pp. 13825–13835, 2018.

[14] M. Wiegand, J. Ruppenhofer, & T. Kleinbauer, "Detection of abusive language: the problem of biased datasets," In Proceedings of the 2019 conference of the North American Chapter of the Association for Computational Linguistics: human language technologies, volume 1 (long and short papers), pp. 602–608, 2019.

[15] T. Mandl, S. Modha, P. Majumder, D. Patel, M. Dave,

C. Mandlia, & A. Patel, "Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in Indo-European languages," In Proceedings of the 11th forum for information retrieval evaluation, pp. 14–17, 2019.

[16] Y. Li and Y. Tao, "Word embedding for understanding natural language: a survey," Guide to big data applications. Springer, Cham, 2018. 83–104. 2018.

[17] A. Pogiatzis, "NLP: Contextualized word embeddings from bert," Medium, 20-Mar-2019. [Online]. Available: https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b. [Accessed: 25-Mar-2022].

[18] D. Becker, "Using categorical data with one hot encoding," Kaggle, 22-Jan-2018. [Online]. Available: https://www.kaggle.com/dansbecker/using-categorical-data-with-one-hot-encoding. [Accessed: 25-Mar-2022].

[19] Z. Yichu and S. Vivek, "A Closer Look at How Fine-tuning Changes BERT," 2021. Available: https://arxiv.org/pdf/2106.14282.pdf. [Accessed: 25-Mar-2022].

[20] A. G. D'Sa, I. Illina, and D. Fohr, "Bert and fasttext embeddings for automatic detection of toxic speech," 2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA), 2020.

[21] M. Mirshafiee, "Step by step introduction to word embeddings and Bert Embeddings," Medium, 07-Oct-2020. [Online]. Available: https://mitra-mirshafiee.medium.com/step-by-step-introduction-to-word-embeddings-and-bert-embeddings-1779c8cc643e. [Accessed: 25-Mar-2022].

[22] J. Sim and C. C. Wright, "The Kappa statistic in Reliability Studies: Use, interpretation, and sample size requirements," Physical Therapy, vol. 85, no. 3, pp. 257–268, 2005.

[23] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," Biometrics, vol. 33, no. 1, p. 159, 1977.

[24] P. Rodríguez, M. A. Bautista, J. Gonzàlez, & S. Escalera, "Beyond one-hot encoding: Lower dimensional target embedding," Image and Vision Computing, 75, 21–31, 2018.

[25] W. Zhang, W. Wei, W. Wang, L. Jin, & Z. Cao, "Reducing BERT computation by padding removal and curriculum learning," In 2021 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) (pp. 90–92), 2021.

[26] R. Artstein, "Inter-annotator agreement," In Handbook of linguistic annotation (pp. 29-7-313). Springer, Dordrecht, 2017.

[27] L. Bartok and M. A. Burzler,"How to assess rater rankings? A theoretical and a simulation approach using the sum of the pairwise absolute row differences (PARDs)," Journal of Statistical Theory and Practice 14.3 pp. 1–16, 2020.

[28] A. De Raadt, M. J. Warrens, R. J. Bosker, and H. AL Kiers, "Kappa coefficients for missing data," Educational and psychological measurement 79, no. 3, pp. 558–576, 2019.

[29] B. Mathew, P. Saha, S. M. Yimam, C. Biemann, P. Goyal, and A. Mukherjee, "Hatexplain: A benchmark dataset for explainable hate speech detection," arXiv.org, 18-Dec-2020. [Online]. Available: https://arxiv.org/abs/2012.10289. [Accessed: 29-Mar-2022].

[30] T. Caselli, V. Basile, J. Mitrović, and M. Granitzer, "Hatebert: Retraining bert for abusive language detection in English," arXiv.org, 04-Feb-2021. [Online]. Available: https://arxiv.org/abs/2010.12472. [Accessed: 29-Mar-2022].

[31] R. Alshaalan and H. Al-Khalifa, "Hate speech detection in Saudi Twittersphere: A deep learning approach," In Proceedings of the Fifth Arabic Natural Language Processing Workshop, pp. 12–23. 2020.

[32] S. Malmasi and M. Zampieri, "Challenges in discriminating profanity from hate speech." Journal of Experimental & Theoretical Artificial Intelligence 30, no. 2, pp. 187–202, 2018.

[33] M. A. Bashar, and R. Nayak, "QutNocturnal@ HASOC'19: CNN for hate speech and offensive content identification in Hindi language," arXiv preprint arXiv:2008.12448, 2020.

[34] G. L. De la Pena Sarracén, R. G. Pons, C. E. M. Cuza, and P. Rosso, "Hate speech detection using attention-based lstm," EVALITA evaluation of NLP and speech tools for Italian 12, pp. 235–238, 2018.

[35] K. Ethayarajh, "How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings," arXiv preprint arXiv:1909.00512, 2019.

[36] F. K. Khattak, S. Jeblee, C. Pou-Prom, M. Abdalla, C. Meaney, & F. Rudzicz, "A survey of word embeddings for clinical text," Journal of Biomedical Informatics, 100, 100057, 2019.

[37] M. Mosbach, M. Andriushchenko, and D. Klakow, "On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines," arXiv preprint arXiv:2006.04884, 2020.

[38] M. K. Dahouda, and J. Inwhee, "A Deep-Learned Embedding Technique for Categorical Features Encoding," IEEE Access 9: 114381–114391, 2021.

[39] Google Developers, "Classification: Accuracy," Google. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy. [Accessed: 14-Jun-2022].