Large Comparative Study of Recent Computational Approach in Automatic Hate Speech Detection

Wesam Shishah [1], Ricky Maulana Fajri [2]

[1]*College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia*
[2]*Dept of Computer Science, University of Indo Global Mandiri, Palembang, 30129, Indonesia*

Abstract **– Social media has become a constant in our everyday life. However, its steady growth has increased the hate speech and hostile content problem. To curb this, hate speech detection and recognition is required, but it is faced to two major challenges - laws and enforcement, and automatic computerized hate speech detection. Although many studies are already implemented in detecting hate content, many of these are done in a single setting showing a single dataset in comparison to machine learning or deep learning models. Thus, there is no comparison between previous approaches and recent inventions such as transformer model. Therefore, in this work we explored and compared recent advanced approaches in automatic hate speech detection. Our aim is to analyze the influence different approaches in detecting hate content and its applicability in the real world. Several experiments were conducted on eight real hate speech datasets from recent studies. We present the results of each comparison which shows that the recent transformer model approach is able to outmatch many of the previous hate speech recognition models by significant G-Means and F1 scores.**

To the author's knowledge, this paper is the first attempt to present a large comparative study of approaches in hate speech detection.

DOI: 10.18421/TEM111-10
https://doi.org/10.18421/TEM111-10

Corresponding author: Wesam Shishah,
College of Computing and Informatics Saudi Electronic University, Riyadh, Saudi Arabia.
Email: w.shishah@seu.edu.sa

Received: 24 September 2021.
Revised: 07 December 2021.
Accepted: 14 December 2021.
Published: 28 February 2022.

Keywords **- Hate speech, transformer model, machine learning and deep learning.**

1. Introduction

Today, there is rapid improvement on the internet and social media communication. This significant improvement provides new platforms for communication, sharing ideas and products advertisement. It unlocks a virtually infinite space for people to openly express themselves and this is sometimes done anonymously. Although many channels of social media are changing over time, the pathways for group-based aggression remain the same. Social media networking sites, including Twitter, or Facebook, have introduced new social models of discursive engagement and their users contribute to the propagation of racism, fake news and hostility toward immigrants or other disadvantaged groups on daily basis.

Several studies have found that the majority of internet users report having experienced hate speech or received hate messages online [1]. While laws have been introduced, enforcing social media sites to delete hate expression, fake news, and illicit content in a limited period such as the time when the illicit content has been identified, however this may not include all of the hate speech content. Since, regulation on its own cannot address hate speech recognition. This has proven that computerized and automatic detection has to be carried out in order to identify hate speech. Therefore, there is a need to incorporate a machine learning or a deep learning algorithm to perform automated hate speech recognition [2], [3]. This campaign is supported by the efficiency of deep learning and machine learning approaches to identify hate speech in social media. In addition, the latest invention of the transformer model indicates recent advancements in solving automated text classification compared to previous approaches. Thus, several scientists are seeking to apply this method in the resolution of hate speech identification. However, many of these studies are

performed on single hate speech dataset and using one machine learning model. Thus, the comparison of each model is under studied.

The purpose of this study is to perform a large comparison study between the traditional approach and recent transformer model in detecting hate speech content. We perform large bodies of experiments with several hate speech datasets known for hate speech identification. Furthermore, our experiments are equipped with various datasets which show a variety of source of dataset, class distribution and hate speech domain. We follow standard machine learning and deep learning preprocessing and experiment procedures, thus the result achieved in this comparative study will show both the weakness and the strength of each model in automatic hate speech detection. This paper consists of 5 sections, section 2 illustrates several previous and recent approaches in hate content detection, while section 3 explains the methods that are being compared in this paper and illustrates the various hate content datasets, with the comparative results illustrated in section 4. Finally, the conclusion of the several findings made from the comparative study will be presented in section 5.

2. Literature Review

The methodology for hate speech detection can be divided into three types - traditional machine learning, artificial neural network, and transformer model.

2.1. Traditional Machine Learning

2.1.1. Logistic Regression

Logistic Regression is a part of the classic machine learning classifiers. It is fast and commonly used for machine learning model benchmarking. A Recent study of logistic regression on hate speech detection was performed by Ginting et al., [4]. This study conducted multinomial logistic regression for Indonesian hate speech detection. They implemented Term Frequency-Inverse Document Frequency (TFIDF) for the text representation. More recently, Reich et al. [5] implemented logistic regression to obtain a more secure hate-related text classification. In their study, they used an existing logistic regression protocol with logistic regression models for the secure classification of feature vectors. The logistic regression protocol used only additions and multiplications over a fixed field. The result showed a more secure approach of hate speech detection between two parties.

2.1.2. Support Vector Machine

The Support Vector Machine (SVM) was proposed by Cortes & Vapnik [6] and is a supervised machine learning method. SVM works by separating the input vectors by constructing a separation line in a high dimensional space, thus a vector that is closer to the separating line will be classified accordingly. SVM is one of the most commonly used classifiers for hate speech detection. For example, Perello et al. [7] implemented the SVM with linear kernel equipped with bag-of-n-gram as the features for twitter hate speech detection. Specifically, they designed an SVM classifier to detect multilingual hate speech against women and immigrants on twitter. A similar study was also conducted by Florio et al. [8], where they also implement linear SVM kernel for another task i.e., Italian hate speech detection. The difference from previous study is that Florio et al. [8] implemented Term Frequency-Inverse Document Frequency (TFIDF) as the word feature representation to be fed to the SVM model. Another study that implemented the linear SVM kernel was also conducted by Basile et al. [9], this study also performed a similar task – to detect hate speech against women and immigrants on twitter. However, they used another kind of feature representation which is bag-of-words. A study by Indurthi et al. [10] suggested other types of SVM Kernel such as the Radial Basis Function (RBF) to perform a similar task as Perello et al. [7]. They perform pre-trained universal sentence embeddings for features extraction for SVM with RBF kernel. In their study, they found that pre-trained universal sentence embedding, performed significantly well on the hate speech detection against women and immigrants on twitter task.

2.1.3. Random Forest

Random forest is a machine learning model derived from decision tree. The random forest is constructed by a substantial amount of individual decision-making trees that operate as a unit. Each tree in the random forest speculates on the class prediction, and the class with the majority vote becomes the model prediction. A study from Nughroho et al. [11] implemented the random forest to detect hate speech. This study used standard text preprocessing as the random forest classifier preprocessing. In the experiment they showed that the random forest outmatches artificial neural network.

2.2. Artificial Neural Network

Artificial Neural Network (ANN) or deep learning is one of the recent state-of-the art approaches in classification. It includes a set of weighted input variables which are immensely connected in parallels. ANN is available in variations and its usage includes areas such as detection, pattern recognition and classification tasks. For hate speech detection, ANN has different types such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Long Term Shot Memory (LSTM)

2.2.1. Convolutional Neural Network

Convolutional Neural Network (CNN) has its fame in solving image recognition problem and has not been recognized in solving text classification problem until the breakthrough work from Yoon Kim [12]. In his work, Yoon Kim [12] implemented one convolution layer on a pre-trained word embedding on hundred billion words of Google News trained by Mikolov et al. [13]. Thus, CNN become one of the neural network approaches in addressing text classification problem from sentiment analysis to hate speech detection. In hate speech recognition, early works that implement CNN for the main approach including Gamback & Sikdar [14]. In their work, they implemented 4 combinations of word embedding with CNN such as ,1. random vectors + CNN, 2. word2vec + CNN, 3. Character n-grams + CNN and 4. word2vec + character n-grams + CNN. Their experiment showed that random vectors + CNN has a better precision compared to other approaches, however word2vec +CNN has a better F1 Score. Similarly, Winter & Kern [15] deployed CNN to perform hate speech detection. In their work, they take combinations of network settings such as number of pooling layer and activation function of CNN. They showed similar results to the previous study where is F1 score of CNN is better compared to other baselines. Another study conducted by Kamble & Joshi [16] deployed 1D CNN for detecting Hindi-English hate speech. They compared their approach with LSTM and BiLSTM, which showed the significant performance of 1D CNN for detecting Hindi-English hate speech.

2.2.2. Recurrent Neural Network

Another commonly used neural network for text classification is Recurrent Neural Network (RNN). A RNN is an artificial neural network variant that enables previous output layers and feedback processes as an input to the next successive layers. Although well known for addressing text classification problems, many works prefer Long Short-Term Memory (LSTM). LSTM is a form of RNN architecture with a stronger capability on learning long-term sequence-to-sequence dependencies such as text classification. LSTM has shown promising results in answering hate speech detection problems, like the work of Miok et al [17]. They designed LSTM with Monte Carlo drop out. The Monte Carlo dropout is used as a regularization technique by capturing the prediction uncertainty. Thus, the regularization will be fed to the LSTM parameter to improve the predictive abilities of the model. Recently, Bisht et al. [18], proposed a single LSTM layer as an underlying model for detecting offensive language and hate speech in twitter data. The study used pre-trained word2vec for input to one layer LSTM. They found that word2vec+LSTM performed slightly better compared to word2vec+Bidirectional LSTM (BiLSTM). Another study performed by Modha et al. [19], proposed LSTM and BiLSTM with combination of pre-trained word vectors as the distributed word representation. In their work, they point out that BiLSTM has a better F1 score for predicting hate content. Although they also argue that BiLSTM performed poorly due to limited number of training data. Another BiLSTM based approach for hate speech classification was done by Bosco et al. [20]. They proposed multiple combination of BiLSTM such as 1-layer bidirectional LSTM, and 2-layer BiLSTM to be compared to linear LSTM and SVM. The experiment indicated that 2-layer BiLSTM performed better compared to the aforementioned approach.

Table 1. Methods Overview

Approach Category	Approach	Text Representation
Machine Learning Approach	Logistic Regression	BOW Logistic Regression
		TFIDF Logistic Regression
	Support Vector Machine (SVM)	BOW SVM
		TFIDF SVM
Artificial Neural Network	Convolutional Neural Network (CNN)	GloVe CNN
		Word2Vec CNN
		FastText CNN
	Long Short Term Memory (LSTM)	GloVe LSTM
		Word2Vec LSTM
		FastText LSTM
	BiDirectional Long Short Term Memory (BiLSTM)	GloVe BiLSTM
		Word2Vec BiLSTM
		FastText BiLSTM
Transformer Approach	Transformer Model	Bert Cased
		Bert Uncased
		Roberta
		XLNet

2.3. Transformer Model

Pre-trained vector representations of word embeddings mined from large bodies of text data attracts a lot of research and have been present in most language-based tasks with encouraging results. Recently, natural language processing community had another breakthrough that outperformed pre-trained vector representation such as ULMFit [21], Elmo [22], Open Ai Generative pre-trained Transformer (GPT) [23] and Google BERT [24]. BERT is a language model that learns deep bidirectional representations of words as a result of pre-training with a large corpus. In Devlin et al. [24], the BERT model performed significantly better than ELMo and OpenAI GPT in a chain of downstream tasks carried out in NLP. Since it is a novel breakthrough, the BERT effect on improving hate speech detection is still under studied.

3. Experimental Methodology

3.1. Methods Overview

Table1 shows the selected approaches for this study. 17 approaches will be compared in three experiments. The approaches are Logistic Regression, Support Vector Machines Cortes & Vapnik [6], Convolutional Neural Network [12], Long-Short Term Memory and Bidirectional Long Short-Term Memory [25] and Transformer Model such as BERT [24], RoberTa [26] and XLNet [27]. The first experiment compares Machine Learning Approaches (BOW Logistic Regression, TFIDF Logistic Regression, BOW SVM and TFIDF SVM). The second experiment compares Deep Learning Approaches GloVe [28] CNN, Word2Vec [13] CNN, FastText [29] CNN, GloVe LSTM, Word2Vec LSTM, FastText LSTM, GloVe BiLSTM, Word2Vec BiLSTM and FastText BiLSTM) the third experiment compares Transformer Approaches (Bert Cased, Bert Uncased, Roberta [26] and XlNet [27]).

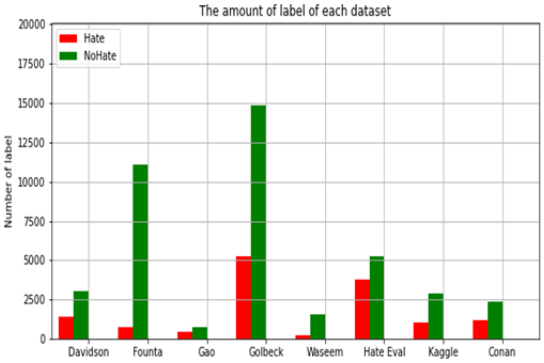


Figure 1. The ratio of hate and non-hate label

3.2. Dataset Exploration

In this paper, we use several recent hate speech datasets from different hate speech recognition studies. The various datasets have been widely used and investigated in hate speech detection problems thus presenting an interesting benchmark hate speech evaluation. Table 2 illustrates the amount and percentage of the text in each dataset. While Fig. 1 and Fig. 2 illustrates the class ratio and the text length with respect to frequency of the dataset respectively.

3.3. Dataset Pre-processing

We perform standard text cleaning and processing which is suitable for text classification problem. We use take each sentence and clean them using Natural Language Tool Kit (NLTK)[1]1. Since mostly we are working with twitter text data, we remove hashtags, mentions, and links in the text document. During all the experiments, we use several text preprocessing techniques such as Tokenization, Lowercasing, Remove Punctuation, Stop words removal, and Contraction replacement.

Table 2. The characteristics of Dataset

Dataset	#Instance	#Hate	#NoHate	%Hate	%NoHate
Davidson [30]	5233	1376	3857	26%	74%
Founta [31]	11863	737	11126	6%	94%
Gao [32]	1127	423	704	37%	63%
Golbeck [33]	20324	5240	14836	25%	75%
Waseem [34]	1811	229	1582	12%	88%
Hate Eval [9]	9000	3783	5217	42%	58%
Kaggle[2]	3947	1049	2898	26%	74%
Conan [35]	3600	1200	2400	33%	67%

3.4. Evaluation Metric

In this work, we implement several metrics to evaluate each model's performance in detecting hate content, such as accuracy, F1 score, AUC score, and G-Means. We illustrate each criterion below

- Accuracy

Accuracy score is a standard machine learning metric used to evaluate the performance of a model. It is straightforward and trivial to be implemented on binary classification.

[1] Nltk.org
[2] https://www.kaggle.com/c/detecting-insults-in-social-commentary

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (5)$$

- F1 Score

The F1 score gives the weighted average of precision and recall. Hence, it considers both the false positive and the false negative. In an unbalanced class distribution, the F1 score gives a better evaluation when compared to the accuracy score. The formula for the F1 score can be defined as follows:

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

$$F1Score = \frac{2\ X\ Precision\ X\ Recall}{Precision + Recall} \quad (8)$$

- AUC score of ROC

The Receiver Operator Characteristic (ROC) curve is an evaluation metric that is used for binary classification problems.

It plots the TPR (True Positive Rate) against FPR (False Positive Rate) at various threshold values on a probability curve and basically separates the 'signal' from the 'noise'. The Area Under the Curve (AUC) measures the ability of a classifier to differentiate between classes. This is used to summarize the ROC curve. The TPR and FPR can be calculated as follows:

$$TPR = \frac{True\ Positive}{All\ Positive} \quad (9)$$

$$FPR = \frac{False\ Positive}{All\ Negative} \quad (10)$$

- G-Means

G-Means is the geometric mean of the accuracies of both the minority and the majority classes. It is calculated as follows:

$$G - Means = \sqrt{Minority\ Accuracy\ X\ Majority\ Accuracy}$$

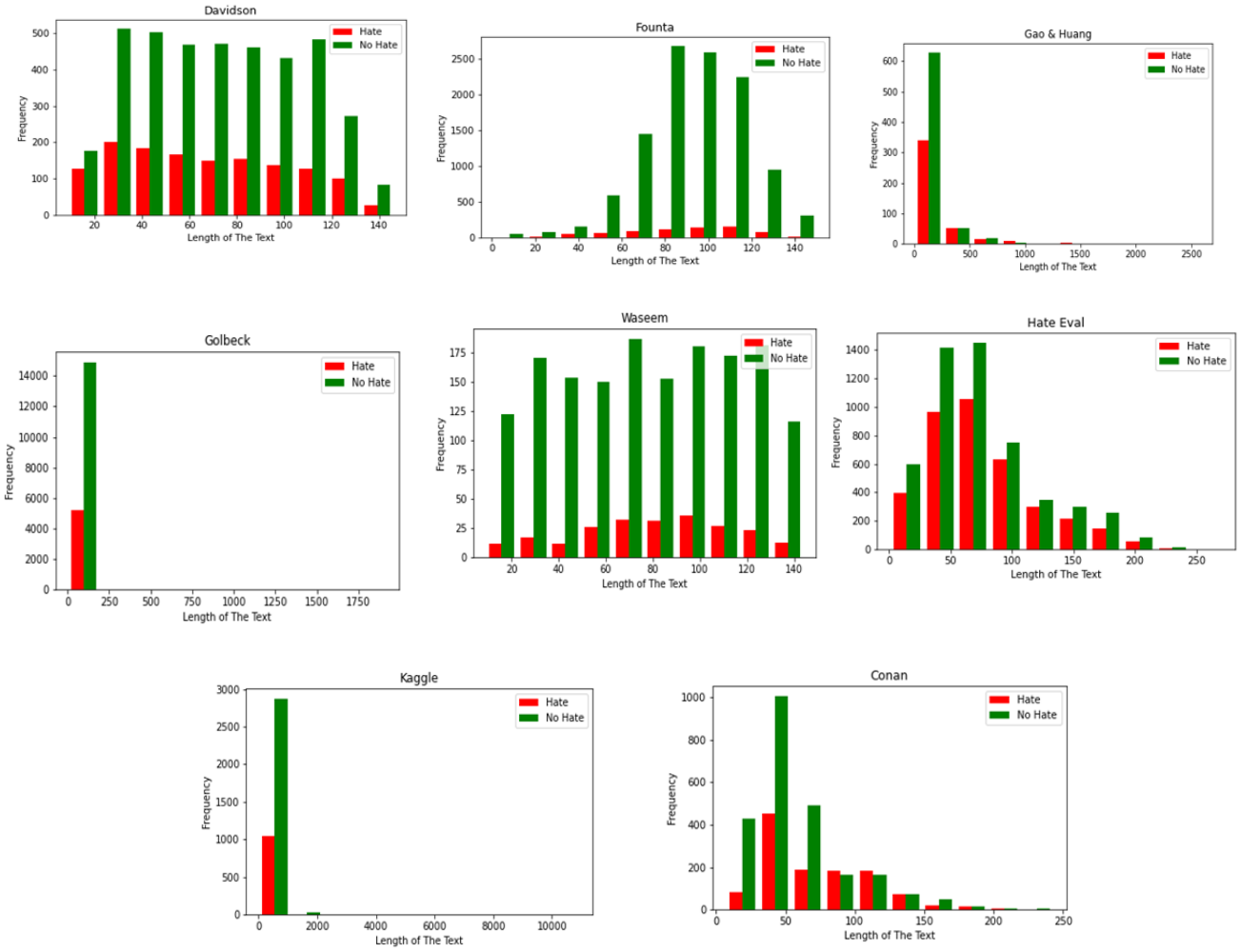


Figure 2. Text length and frequency

4. Results

The aim of our experiments is to compare several state-of-the-art models in hate speech detection. For the experiment, we fed the datasets to the corresponding model. The dataset was split into test and training data at 70% and 30% respectively. We divided the experiments into 3 types. Firstly we compared the traditional feature representation such as bag-of-words (BOW) and Term Frequency Inverse Document Frequency (TFIDF) combined with SVM and Logistic regression. Secondly, we compared word embedding approach such as Word2Vec [13], GloVe [28] and FastText [29] with deep learning model and finally, we compared the recent transfer learning model i.e., bert, roberta and xlnet. We ran all the experiments on a standard I7 dual core machine, with 32 GB of RAM and 4 GB GPU. However, we ran the BERT, roberta and XLNet on Google Collaboratory environment with 32 GB GPU. We ran each model 10 times and took the average.

4.1. Traditional Feature Representation with Machine Learning

In this experiment we compared the combination of BOW and TFIDF with the machine learning approach. We used 5000 as the maximum features for TFIDF representation. Table 3 presents the results of BOW and TFIDF representation combined with SVM and Logistic regression. From the table, TFIDF with SVM have a better evaluation score for almost all the datasets, except in Golbeck dataset. The table also infers that BOW as the traditional text representation is inferior compared to TFIDF, however it is a fast and simple representation to be implemented. TFIDF reached its maximum performance in this task, thus better representation is needed to improve the evaluation score. Looking closer at the result, it shows that BOW representation cannot outmatch TFIDF in both machine learning models. It is understandable since BOW is the most trivial and traditional feature representation techniques. Interestingly, BOW was able to get a high AUC score in Founta dataset which is the most imbalanced dataset used in the experiment

Table 3. Traditional Feature Representation with Machine Learning

Methods	Metrics	Datasets							
		Davidson	Founta	Gao	Golbeck	Waseem	HateEval	Kaggle	Conan
BOW SVM	G-Means	0.40	0.28	0.43	0.36	0.29	0.48	0.33	0.44
	F1 Score	0.23	0.07	0.27	0.21	0.11	0.38	0.17	0.31
	AUC Score	0.60	0.84	0.68	0.73	0.81	0.55	0.66	0.66
	Accuracy	0.48	0.49	0.51	0.52	0.50	0.52	0.49	0.55
BOW Logistic Regression	G-Means	0.28	0.18	0.21	0.45	0.12	0.47	0.21	0.32
	F1 Score	0.07	0.05	0.10	0.30	0.03	0.36	0.07	0.17
	AUC Score	0.84	0.90	0.62	0.71	0.87	0.56	0.07	0.64
	Accuracy	0.49	0.49	0.50	0.53	0.45	0.52	0.49	0.51
TFIDF SVM	G-Means	**0.90**	**0.60**	**0.87**	0.36	**0.61**	**0.74**	**0.68**	**0.98**
	F1 Score	**0.89**	**0.51**	**0.86**	0.22	**0.52**	**0.70**	**0.58**	**0.98**
	AUC Score	**0.95**	**0.95**	**0.91**	**0.75**	**0.90**	**0.75**	**0.82**	**0.98**
	Accuracy	**0.91**	**0.67**	**0.90**	0.55	**0.68**	**0.74**	**0.71**	**0.98**
TFIDF Logistic Regression	G-Means	0.80	0.42	0.79	**0.39**	0.32	0.71	0.63	0.95
	F1 Score	0.80	0.34	0.76	**0.25**	0.19	0.67	0.56	0.94
	AUC Score	0.86	0.94	0.85	**0.75**	0.88	0.75	0.79	0.96
	Accuracy	0.84	0.56	0.84	**0.56**	0.57	0.73	0.69	0.95

4.2. Word Embedding with Deep Learning

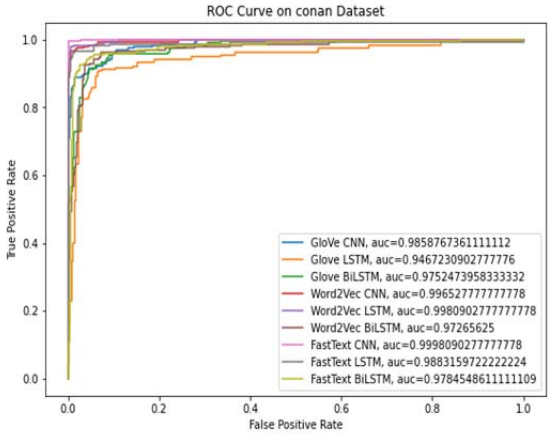
In the next experiment we combined recent advanced word embedding with a recent deep neural network model. We implemented this combination because, many studies showed word embedding performed better when combined with deep neural network such as CNN and LSTM. We compared several pre-trained word embedding approaches from the literature such as Word2Vec, FastText and GloVe. For word2vec embedding we used pre-trained vectors that had been trained on part of Google News dataset (about hundred billion words) [13]. This model contains 300-dimensional vectors for three million words and phrases. In GloVe we used 6B tokens trained on Wikipedia 2014, and in FastText we use 1-million-word vectors trained on Wikipedia.

Table 4. Performance of pre-trained word embedding with deep learning model

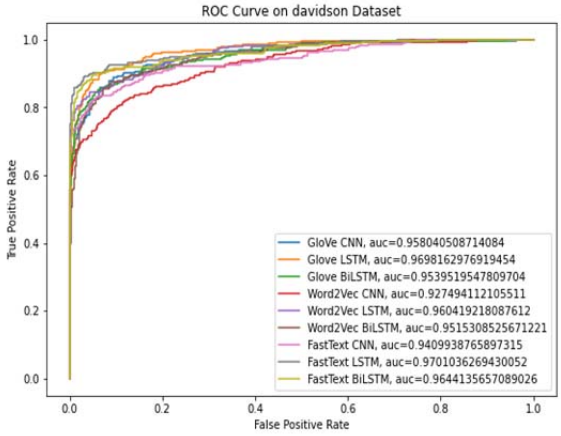
Methods	Metrics	Dataset							
		Davidson	Founta	Gao	Golbeck	Waseem	Hate Eval	Kaggle	Conan
GloVe CNN	G-Means	0.87	0.53	0.38	0.36	0	0.68	0.69	0.93
	F1 Score	0.94	**0.87**	0.28	0.23	0	0.62	0.57	0.92
	Accuracy	0.91	0.94	0.67	0.75	0.87	0.71	0.78	0.94
	AUC Score	0.83	0.41	0.78	0.64	0.65	0.79	0.83	0.98
GloVe LSTM	G-Means	0.88	0.47	0.20	0.11	0.38	**0.73**	**0.75**	0.94
	F1 Score	**0.95**	0.81	0.07	**0.30**	0.25	0.70	0.65	0.91
	Accuracy	0.92	0.94	0.62	0.75	0.88	0.72	0.82	0.94
	AUC Score	0.84	0.37	0.60	0.65	0.66	0.08	0.84	0.95
GloVe BiLSTM	G-Means	0.78	0.63	0.81	0.40	0.20	0.72	0.64	0.93
	F1 Score	0.80	0.48	0.78	0.26	0.08	0.68	0.55	0.90
	Accuracy	0.87	0.94	0.85	0.75	0.87	0.73	0.81	0.93
	AUC Score	0.92	**0.87**	0.90	0.66	0.76	0.80	0.83	0.97
Word2Vec CNN	G-Means	0.81	0.63	0.90	0.37	0.56	**0.73**	0.69	0.97
	F1 Score	0.76	0.57	0.88	0.24	0.42	0.68	**0.97**	0.97
	Accuracy	0.88	**0.95**	0.91	0.76	0.88	**0.75**	**0.98**	0.98
	AUC Score	0.91	0.85	0.95	0.65	0.83	**0.81**	**0.99**	**0.99**
Word2Vec LSTM	G-Means	0.90	0.61	**0.91**	**0.43**	**0.75**	0.69	0.71	**0.98**
	F1 Score	0.86	0.49	**0.89**	**0.30**	0.60	0.64	0.63	**0.98**
	Accuracy	0.92	**0.95**	0.92	0.76	**0.90**	0.74	0.83	**0.98**
	AUC Score	0.97	0.85	**0.96**	0.66	**0.91**	**0.81**	0.82	**0.99**
Word2Vec BiLSTM	G-Means	0.86	0.56	0.34	0.38	**0.60**	**0.73**	0.65	0.94
	F1 Score	0.84	0.44	0.20	0.25	0.50	0.69	0.56	0.92
	Accuracy	0.92	**0.95**	0.66	0.75	**0.90**	0.72	0.81	0.94
	AUC Score	0.95	0.85	0.80	0.64	0.89	**0.81**	0.83	**0.99**
FastText CNN	G-Means	0.84	0.53	0.90	0.35	0.67	0.71	0.65	**0.98**
	F1 Score	0.80	0.42	**0.89**	0.21	0.53	0.66	0.52	0.97
	Accuracy	0.90	**0.95**	0.92	0.76	0.89	0.73	0.76	0.98
	AUC Score	0.93	0.82	0.95	0.66	0.86	0.80	0.79	**0.99**
FastText LSTM	G-Means	**0.93**	**0.65**	0.89	0.40	0.66	0.72	**0.75**	0.96
	F1 Score	0.92	0.52	0.85	0.27	0.51	0.68	0.69	0.96
	Accuracy	0.96	**0.95**	0.88	**0.77**	0.89	0.73	0.85	0.97
	AUC Score	**0.98**	0.85	0.94	**0.68**	**0.91**	0.80	0.88	**0.99**
FastText BiLSTM	G-Means	0.91	0.60	0.61	0.39	0.73	0.67	0.69	0.93
	F1 Score	0.89	0.48	0.50	0.25	0.59	0.62	0.60	0.91
	Accuracy	**0.94**	**0.95**	**0.95**	0.76	**0.90**	0.72	0.82	0.94
	AUC Score	0.96	0.87	0.85	0.66	0.85	0.78	0.87	0.97

2017, UMBC webbase corpus and statmt.org news dataset resulting 16B tokens. Table 4 shows the overall results of the word embedding approaches. It is clear that word embedding representation offers a better feature representation compared to traditional BOW and TFIDF. The result shows superior performance across all the datasets. However, the combination of each pre-trained word embedding with deep neural network shows a similar performance. There is no real absolute winner in the experiment. For example, Word2vec + CNN has a better accuracy and AUC score on kaggle and conan dataset, however it loses to Glove + LSTM for G-Means in the Davidson dataset. In one case i.e., conan dataset, Word2Vec + LSTM won over all the evaluation metrics; however, this performance is only marginally above FastText + CNN. Thus, any word embedding techniques combined with deep learning will achieve reasonably good performance. Furthermore, this performance only needs a little computational overhead compared to traditional BOW and TFIDF while pre-trained transformer model approach required higher computational resources. We further analyzed the performance of

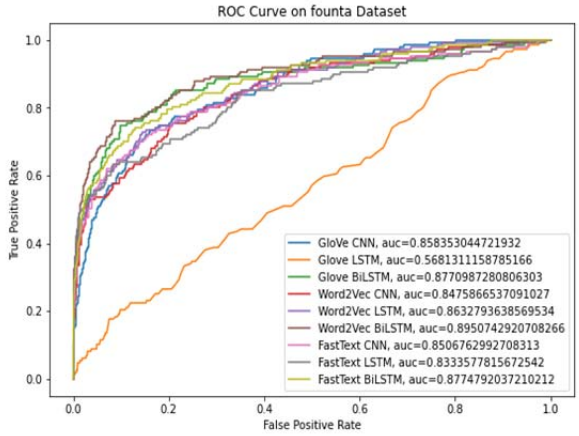
each methods using ROC curve. Fig. 3 shows the ROC curve of the word embedding approaches on every dataset. We summarized that every word embedding approach combined with deep neural network performs significantly well on several datasets. In fact, from these experiments, we inferred that there is no absolute winner from all the approaches mentioned above. The differences between each model are only marginal. The vast differences could be seen in the Founta and the Waseem dataset. In both datasets, GloVe with LSTM had the lowest performance, while the winner was the combination of Word2Vec with LSTM and BiLSTM.



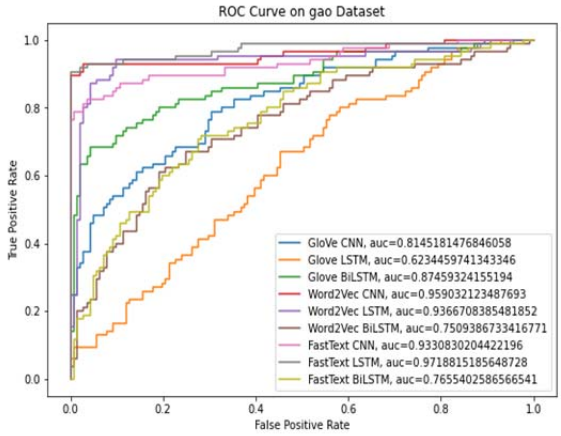
(a). Conan



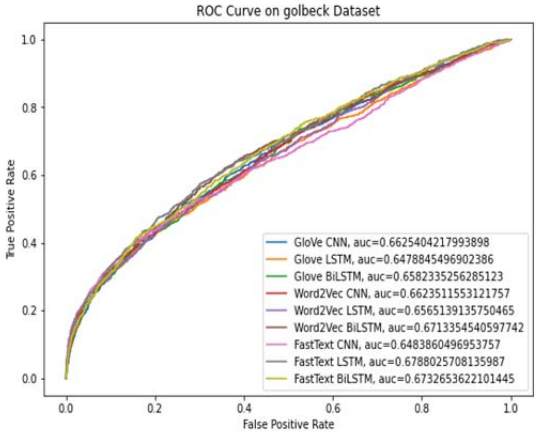
(b). Davidson



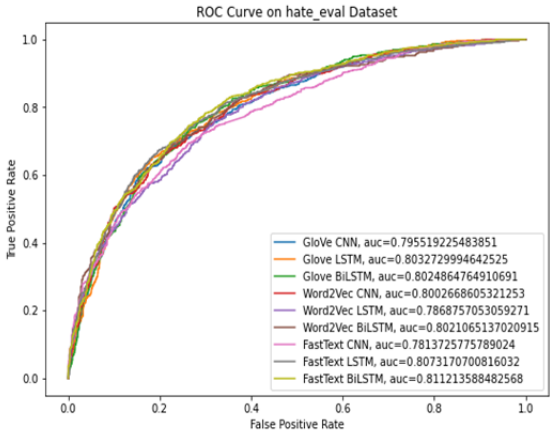
(c). Founta



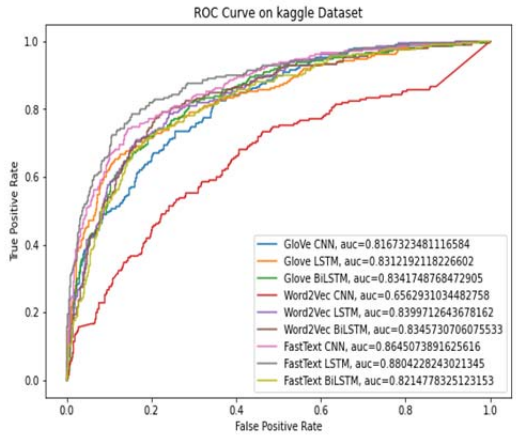
(d). Gao & Huang



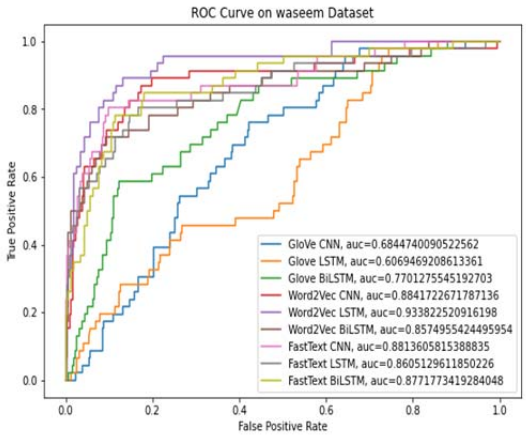
(e). Golbeck



(f). Hate Eval



(g). Kaggle



(h). Waseem

Figure 3. AUC of ROC of Every Dataset

4.3. Comparison on Transformer Approach

Table5 compares the BERT model (cased and uncased), Roberta and XLNet model. For each of the models we used its related tokenizer to tokenize the text in the dataset. From the table we summarized that the models exhibit similar performance. Most of the models perform well in other tasks such as machine translation, sentiment analysis, and hate speech. For comparison purposes, we showed that XLNet outmatched most of the transfer learning approaches, it performed the best in balanced and imbalanced class distribution.

Table 5. Performance Comparison of Transformer Approach

Methods	Metrics	Dataset							
		Davidson	Founta	Gao	Golbeck	Waseem	Hate Eval	Kaggle	Conan
Bert Cased	G-Means	**0.96**	0.73	0.92	0.57	0.73	0.73	0.75	0.96
	F1 Score	**0.95**	0.57	0.91	0.42	0.58	0.58	0.66	0.96
	Accuracy	0.96	**0.95**	0.92	0.74	0.90	**0.90**	0.83	0.97
	AUC Score	0.96	0.76	0.92	0.62	0.76	0.76	0.76	0.96
Bert Uncased	G-Means	**0.96**	0.74	0.91	0.57	0.73	0.75	**0.80**	0.98
	F1 Score	**0.95**	0.61	0.90	0.42	**0.60**	0.71	0.73	**0.98**
	Accuracy	**0.97**	**0.95**	0.92	0.72	**0.91**	0.76	0.86	**0.98**
	AUC Score	0.96	0.77	0.91	0.61	0.76	0.75	0.81	0.98
Roberta	G-Means	0.93	0.71	**0.93**	0.49	0.72	**0.76**	0.79	**0.98**
	F1 Score	0.90	**0.86**	0.92	0.36	0.54	**0.73**	0.70	0.92
	Accuracy	0.95	**0.95**	0.93	**0.77**	0.89	0.77	0.83	**0.98**
	AUC Score	0.93	0.74	0.93	0.60	0.75	0.76	0.79	**0.99**
XLNet	G-Means	0.95	**0.95**	**0.93**	**0.71**	**0.77**	0.74	0.63	0.97
	F1 Score	0.91	0.70	**0.93**	**0.58**	0.58	0.70	0.56	0.97
	Accuracy	0.89	**0.95**	0.91	0.44	0.88	0.75	0.82	**0.98**
	AUC Score	0.97	**0.88**	**0.98**	0.67	0.88	**0.83**	0.86	**0.99**

4.4. Time Execution Analysis

Finally, we performed computational time analysis to further assess the models' performance with respect to execution time. For this experiment we employed the Golbeck dataset since it is the largest dataset, we used in this work thus it required high computational time. It is unfair to compare the execution time of bag-of-words + logistic regression to BERT because BERT requires many extra computational resources therefore BERT would require more execution time. For this reason, we divided the experiment into three types, traditional feature representation with machine learning, word embedding with deep learning and transformer approach. Table 6 illustrates the computational time needed for each of the models. It is intuitive that the traditional feature representation and a machine learning approach will have the lowest time execution due to its computational simplicity, while

the transformer-based model will have higher computational cost. The experiment shows that the absolute winner is XLNET with a 0.71 G-Means score which outmatched all the aforementioned approaches with the cost of 30 Minutes and 54 S in required computational execution time.

Table 6. Computational Time Analysis

Methods	Time	G-Means
Feature Representation + Machine Learning		
BOW Logistic Regression	**0 Min 2.7 Seconds**	0.45
BOW SVM	2 Min 3 Seconds	0.36
TFIDF Logistic Regression	0 Min 0.13 Seconds	0.39
TFIDF SVM	1 Min 41 Seconds	0.36
Word Embedding + Deep Neural Network		
GloVe CNN	6 Min 57 Seconds	0.36
GloVe LSTM	4 Min 56 Seconds	0.11
Glove BiLSTM	8 Min 0.6 Seconds	0.40
Word2Vec CNN	5 Min 58 Seconds	0.37
Word2Vec LSTM	16 Min 45 Seconds	0.43
Word2Vec BiLSTM	45 Min 15 Seconds	0.43
FastText CNN	5 Min 45 Seconds	0.35
FastText LSTM	17 Min 24 Seconds	0.40
FastText BiLSTM	42 Min 56 Seconds	0.39
Transformer Model		
Bert Cased	25 Min 44 Seconds	0.57
Bert Uncased	13 Min 44 Seconds	0.57
Roberta	34 Min 44 Seconds	0.49
XLNet	30 Min 54 Seconds	**0.71**

5. Conclusion

In this work we compared several machine learning, deep learning and transformer models in detecting hate content. We used the dataset which provided different imbalanced class distribution making an interesting and comprehensive analysis. In a balanced setting most of the approaches worked well for example in the Davidson, Kaggle and Conan all methods were able to get high G-Means performance except for the traditional feature representation such as Bag of Words with either Logistic Regression or Support Vector Machine. In the Conan dataset, TFIDF with Logistic Regression and SVM had a 0.98 G-Means. Surprisingly, this result is almost similar to the word embedding with deep learning and transformer model. Thus, it can be summarized that datasets also play an important role in providing good quality data for a computational approach in performing automatic recognition. We performed this error analysis to investigate where the model fails to work. In our experiment it showed that most of the model fails where human also fails such as in the case where there is context needed to determine whether a sentence is a hate speech. Thus, we recommend feeding high quality datasets to the model. We also used several metrics to analyze the performance of each model. While many works implemented classification accuracy to measure the model performance, we showed that the accuracy might be high but it not showing the real performance score. Finally, in our time execution analysis, the transformer model had the highest performance compared to other approaches, however the differences are only marginal with extra cost in computational time.

References

[1]. Matamoros-Fernández, A., & Farkas, J. (2021). Racism, Hate Speech, and Social Media: A Systematic Review and Critique. *Television & New Media*, *22*(2), 205-224.

[2]. Raufi, B., & Xhaferri, I. (2018, September). Application of machine learning techniques for hate speech detection in mobile applications. In *2018 International Conference on Information Technologies (InfoTech)* (pp. 1-4). IEEE.

[3]. Ayo, F. E., Folorunso, O., Ibharalu, F. T., & Osinuga, I. A. (2020). Machine learning techniques for hate speech classification of twitter data: State-of-the-art, future challenges and research directions. *Computer Science Review*, *38*, 100311.

[4]. Ginting, P. S. B., Irawan, B., & Setianingsih, C. (2019, November). Hate speech detection on twitter using multinomial logistic regression classification method. In *2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)* (pp. 105-111). IEEE.

[5]. Reich, D., Todoki, A., Dowsley, R., De Cock, M., & Nascimento, A. C. (2019). Privacy-preserving classification of personal text messages with secure multi-party computation: An application to hate-speech detection. *arXiv preprint arXiv:1906.02325*.

[6]. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

[7]. Perelló, C., Tomás, D., Garcia-Garcia, A., Garcia-Rodriguez, J., & Camacho-Collados, J. (2019, June). UA at SemEval-2019 task 5: setting a strong linear baseline for hate speech detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 508-513).

[8]. Florio, K., Basile, V., Polignano, M., Basile, P., & Patti, V. (2020). Time of your hate: The challenge of time in hate speech detection on social media. *Applied Sciences*, *10*(12), 4180.

[9]. Basile, V., Bosco, C., Fersini, E., Debora, N., Patti, V., Pardo, F. M. R., ... & Sanguinetti, M. (2019). Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation* (pp. 54-63). Association for Computational Linguistics.

[10]. Indurthi, V., Syed, B., Shrivastava, M., Chakravartula, N., Gupta, M., & Varma, V. (2019, June). Fermi at semeval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 70-74).

[11]. Nugroho, K., Noersasongko, E., Fanani, A. Z., & Basuki, R. S. (2019, July). Improving random forest method to detect hatespeech and offensive word. In *2019 International Conference on Information and Communications Technology (ICOIACT)* (pp. 514-518). IEEE.

[12]. Kim, Y. (2014). *Convolutional Neural Networks for Sentence Classification*. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), 1746–1751.

[13]. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111-3119).

[14]. Gambäck, B., & Sikdar, U. K. (2017, August). Using convolutional neural networks to classify hate-speech. In *Proceedings of the first workshop on abusive language online* (pp. 85-90).

[15]. Winter, K., & Kern, R. (2019, June). Know-center at SemEval-2019 task 5: multilingual hate speech detection on Twitter using CNNs. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 431-435).

[16]. Kamble, S., & Joshi, A. (2018). Hate speech detection from code-mixed hindi-english tweets using deep learning models. *arXiv preprint arXiv:1811.05145*.

[17]. Miok, K., Nguyen-Doan, D., Škrlj, B., Zaharie, D., & Robnik-Šikonja, M. (2019, October). Prediction uncertainty estimation for hate speech classification. In *International Conference on Statistical Language and Speech Processing* (pp. 286-298). Springer, Cham.

[18]. Bisht, A., Singh, A., Bhadauria, H. S., & Virmani, J. (2020). Detection of hate speech and offensive language in Twitter data using LSTM model. In *Recent trends in image and signal processing in computer vision* (pp. 243-264). Springer, Singapore.

[19]. Modha, S., Majumder, P., & Patel, D. (2019, June). DA-LD-Hildesheim at SemEval-2019 task 6: tracking offensive content with deep learning using shallow representation. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 577-581).

[20]. Bosco, C., Felice, DO, Poletto, F., Sanguinetti, M., & Maurizio, T. (2018). Pregled zadatka otkrivanja govora mržnje evalita 2018. U *EVALITA 2018. – Šesta evaluacijska kampanja alata za obradu prirodnog jezika i govora za talijanski* (Vol. 2263, str. 1-9). CEUR.

[21]. Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

[22]. Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

[23]. Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. Retrieved from: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf [accessed: 10 August 2021].

[24]. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[25]. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

[26]. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

[27]. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding. *Advances in Neural Information Processing Systems*, *32*, 5753-5763.

[28]. Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

[29]. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2017). Advances in pre-training distributed word representations. *arXiv preprint arXiv:1712.09405*.

[30]. Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017, May). Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media* (Vol. 11, No. 1).

[31]. Founta, A. M., Djouvas, C., Chatzakou, D., Leontiadis, I., Blackburn, J., Stringhini, G., ... & Kourtellis, N. (2018, June). Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*.

[32]. Gao, L., & Huang, R. (2017). Detecting online hate speech using context aware models. *arXiv preprint arXiv:1710.07395*.

[33]. Golbeck, J., Ashktorab, Z., Banjo, R. O., Berlinger, A., Bhagwan, S., Buntain, C., ... & Wu, D. M. (2017, June). A large labeled corpus for online harassment research. In *Proceedings of the 2017 ACM on web science conference* (pp. 229-233).

[34]. Chung, Y. L., Kuzmenko, E., Tekiroglu, S. S., & Guerini, M. (2019). CONAN--COunter NArratives through Nichesourcing: a Multilingual Dataset of Responses to Fight Online Hate Speech. *arXiv preprint arXiv:1910.03270*.

[35]. Waseem, Z. (2016, November). Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the first workshop on NLP and computational social science* (pp. 138-142).