Bournemouth University

Faculty of Media and Communication

HATE SPEECH AND OFFENSIVE LANGUAGE DETECTION

Project Report Submitted by:-

Gagandeep Singh Rehal
S5532166
MSc Artificial Intelligence for Media
14th August 2023

Supervised by:-

Prof. Hammadi Nait Charif

Contents

Acknowledgment

First and foremost, I want to thank my Supervisor, **Prof. Hammadi Nait Charif**, for their continuous support, essential assistance, and beneficial feedback over the course of this project. Their knowledge and perspectives were invaluable in the progress of this project.

My deepest gratitude goes to my family for their constant love, patience, and faith in my skills. Their encouragement has been my rock, and this accomplishment is as much theirs as it is mine. Finally, I want to thank everyone who generously offered their time and thoughts to make this project possible.

Abstract

Distinguishing hate speech from other types of abusive language on social media is a serious job. Techniques based on lexical detection frequently lack accuracy since they classify all messages containing specified terms as hate speech. Traditional supervised learning methodologies have failed to separate the two groups. This paper makes use of the **Twitter Hate Speech and Offensive Language** dataset, which categorizes messages as either containing hate speech, just offensive language, or neither. This paper uses **BERT** to classify between these classes. According to the predictions, tweets with racist and homophobic connotations are frequently labeled as hate speech, whereas sexist tweets are usually judged offensive. The categorization of tweets that do not contain explicit hate-related language proved more difficult. This paper focuses on the application of **BERT** to combat hate speech and offensive language, as well as a detailed discussion of other innovative approaches to this problem.

Keywords: Hate Speech, Offensive Language, NLP, BERT, Tweets.

1. Introduction

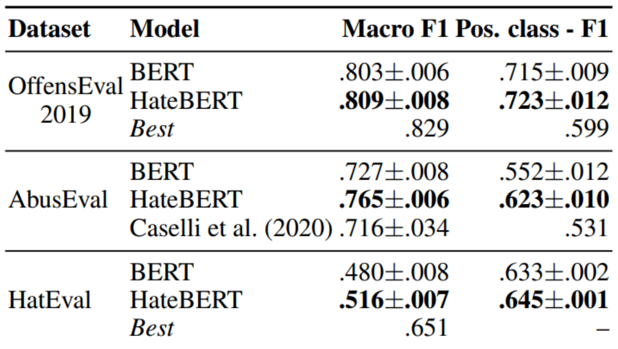
Hate speech is defined by Davidson et al., 2017 as "language that is used to express hatred towards a targeted group or is intended to be derogatory, humiliating, or insulting to the members of the group." Saha et al., 2019, assert that in the context of social networks, abusive language refers to content that contains any type of unacceptable speech, whether in a post or a remark. This sort of language is classified into three categories: hate speech, offensive language, and profanity. Hate speech consists of insulting words directed at a wide group, impacting traits such as ethnic origin, racial origin, or gender. In contrast, offensive language, while comparable to a demeaning statement, is directed at a specific person. Profanity is defined as the use of inappropriate words without regard for a particular person or organization. Among them, profanity is seen to be the least destructive, while hate speech is thought to have the most severe negative influence on society.

According to Mozafari et al., 2019, the increasing number of particular individuals posting offensive and harmful information on social media platforms has prompted researchers to devote significant attention to the challenging task of recognizing such content. This effort necessitates not just a competent automated system for identifying hate speech that employs complex techniques in machine learning and natural language processing, but also a large store of labeled data for model training. The scarcity of labeled hate speech data and existing biases is a major barrier in this field of study. As per the paper by Sanh et al.,2020, the use of large-scale pre-trained models using transfer learning is gaining popularity in the field of Natural Language Processing (NLP). However, executing these massive models in constrained or edge computing settings during training or evaluation stages is a substantial difficulty. This paper is centered around the implementation of **BERT** (Devlin et al.,2019), which can be fine-tuned on a wide range of tasks such as Hate Speech and Offensive Language detection without significant task-specific architectural changes. As per Devlin et al.,2019, BERT is both theoretically and experimentally simple. It achieves new cutting-edge outcomes on eleven natural language processing tasks, including raising the **GLUE score to 80.5%**.

2. Recent Works

2.1. HateBERT: In the paper by Caselli et al., 2021, HateBERT was first introduced. This model is a variant of the typical BERT model (Devlin et al.,2019) that has been tweaked particularly for detecting offensive words in English. HateBERT was trained using RAL-E, a large dataset of English Reddit comments from groups that were banned due to provocative, abusive, or hateful content. The dataset, which was collected and made available for continuing study, was an important aspect of this endeavor. Using posts from banned groups, the study compares the performance of a generic pre-trained language framework with a variant fine-tuned to recognize abusive content. Caselli et al.,2021 utilized the **Masked Language Model** technique to modify the English BERT base-uncased model using **1,478,348** messages from the **RAL-E** dataset. Researchers set aside **14,932** messages for testing purposes. The retraining procedure was carried out across **100** epochs, with **64** samples processed in batches. Each example may contain up to **512** sentence pieces tokens. During this retraining, the **Adam optimization** algorithm with a learning rate of **5e-5** was also applied. To test HateBERT's suitability for identifying offensive and abusive language patterns, researchers compared 3 English datasets designed to detect rude, abusive, and hateful words. The 3 datasets are stated below:

- **OffensEval:** (Zampieri et al., 2019) There are **14,100** tweets in the dataset that have been flagged as having inappropriate language. According to the task's criteria, a tweet is considered offensive if it contains any type of unsuitable or undesirable language, whether the offense is veiled or explicit. This dataset is separated into two sections: training and testing. The training segment comprises **13,240** posts, while the testing phase includes **860**. Of these, **4,400** posts in the training set and **240** in the testing set were categorized as positive for having inappropriate language.
- **AbusEval:** (Caselli et al.,2021) The dataset was developed by adding an extra layer of annotations for abusive language to OffensEval. It has the same size as OffensEval, with a total of **14,100** tweets, and the split between training and test sets stays similar, with **13,240** and **860** posts, respectively. The overall distribution within the positive category distinguishes this dataset, resulting in **2,749** cases in the training section and **178** in the test section.
- **HatEval:** (Basile et al., 2019) There are **13,000** tweets in the dataset's English part that have been carefully marked to recognize hate speech directed against migrants and women. **10,000** messages are earmarked for training, whereas **3,000** are intended for testing. The distribution of messages in both the training and test sets is balanced in terms of the goals, with **5,000** in the training and **1,500** in the test for each of the categories. This symmetry, however, does not extend to the distribution of the positive class, where the training set contains **4,165** messages and the test set has **1,252**.



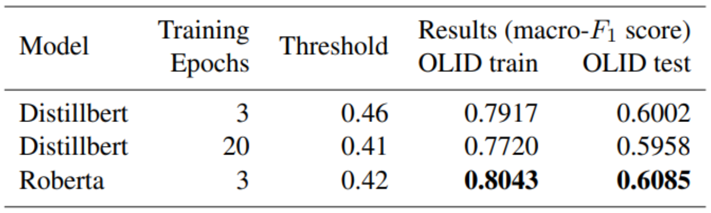
Dataset	Model	Macro F1	Pos. class - F1
OffensEval 2019	BERT	.803±.006	.715±.009
	HateBERT	**.809±.008**	**.723±.012**
	Best	.829	.599
AbusEval	BERT	.727±.008	.552±.012
	HateBERT	**.765±.006**	**.623±.010**
	Caselli et al. (2020)	.716±.034	.531
HatEval	BERT	.480±.008	.633±.002
	HateBERT	**.516±.007**	**.645±.001**
	Best	.651	–

Table 1: HateBERT vs BERT. Caselli et al.,2021

HateBERT consistently outperforms a generic BERT across many abusive language phenomena as shown in Table 1, including offensive language (OffensEval), abusive language (AbusEval), and hate speech (HatEval). HateBERT generates precise representations of each harmful language phenomenon for which it has been fine-tuned, according to cross-dataset studies.

2.2. RoBERTa: RoBERTa is a version of the BERT model created by Liu et al., 2019 and researchers outline a systematic set of tweaks and refinements to BERT in their research paper, leading to significant improvements in performance across a number of Natural Language Processing (NLP) tasks. The researchers opted to extend the training time inside the RoBERTa framework, which encourages a more thorough convergence of the model. Unlike BERT (Devlin et al.,2019), which uses a set pattern for masking tokens in the pretraining phase, RoBERTa employs a **variable masking scheme**. The masking configurations shift between different training epochs, preventing the model from acclimating to certain masked configurations. Alonso et al., 2020, employed RoBERTa in their research for detecting hate speech and offensive words. Regarding the data utilized, for training, researchers used the Offenseval (Zampieri et al., 2019) dataset, while for development and validation, researchers utilized the OLID dataset (Zampieri et al., 2019). The researchers used English language data from the two datasets to detect hate speech in English.

- **Offenseval:** (Zampieri et al., 2019) The dataset consisted of **9,089,140** tweets. For annotation, two metrics were used: the average and the standard deviation. Because the training set lacked actual labels, one of the challenges was determining an acceptable threshold for the predictions. This threshold point was necessary in order to manually categorize the entry's outcomes. Following that, the chosen threshold was cross-validated using a test batch of **3,887** tweets for which the dataset owners provided accurate classifications.
- **OLID:** (Zampieri et al., 2019) The dataset was used as a validation tool, primarily to determine the best threshold level. This collection includes messages with insults, frightening language, and phrases including filthy or obscene terms, all of which are labeled as improper. A total of **13,241** elements, or tweets, are accounted for in the OLID data. Around **4,400** of these, or roughly 35%, are categorized as obnoxious with the remaining classified as non-offensive.



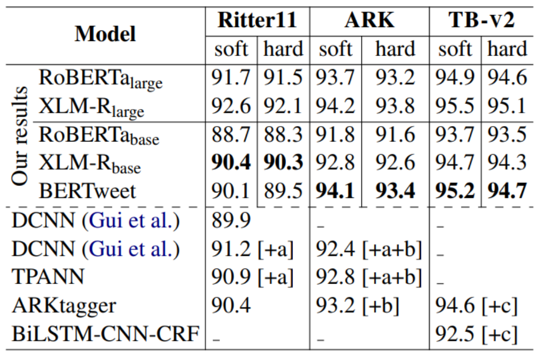
Model	Training Epochs	Threshold	Results (macro-F_1 score) OLID train	OLID test
Distillbert	3	0.46	0.7917	0.6002
Distillbert	20	0.41	0.7720	0.5958
Roberta	3	0.42	**0.8043**	**0.6085**

Table 2: RoBERT vs DistilBERT Alonso et al., 2020

In the paper by Alonso et al., 2020, the model was started with a preprocessing stage to obtain refined data using the OffenseEval dataset (Zampieri et al., 2019). One million initial samples were taken from the training part, and the latest 3,000 tweets were assigned to a developing subset, keeping the remainder for training. RoBERTa (Liu et al., 2019) was used for training, with specific hyper-parameter modifications such as defining "**early stopping patience**" as **3,** setting the "**learning rate**" at **1e-5**, and fixing "**evaluate throughout training steps**" at **2000**. Two DistilBert iterations (3 and 20 epochs) were trained, and RoBERTa was trained for 3 epochs, concentrating primarily on the mean score for regression purposes. The last phase involves converting the predicted scores into category labels, with each model having its own threshold based on the OLID (Zampieri et al., 2019) training data. The best threshold was determined for each model, representing the best potential result obtained from the OLID (Zampieri et al., 2019) database training segment. RoBERTa displayed its most efficient classification on both the OLID training and testing phases as shown in Table 2.

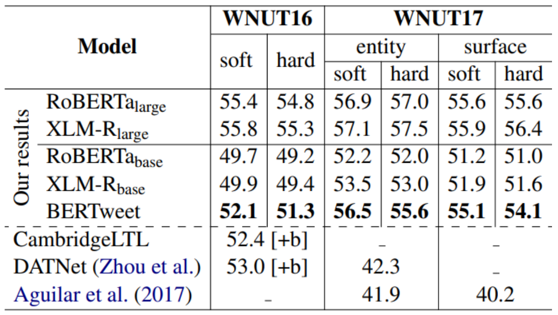
2.3. BERTweet: Nguyen et al., 2020 developed BERTweet, a pre-trained language model particularly tailored to recognize the intricate details of English tweets. BERTweet's architecture is based on the BERT (Devlin et al.,2019) base model's fundamental structure, and it employs the Transformer framework for textual sequence encoding. Its pre-training method is influenced by RoBERTa (Liu et al., 2019), a variant that improves the resilience and performance of BERT's pre-training methodology. The researchers used an **80GB** pre-training sample of uncompressed texts, which included **850 million** Tweets, or **16 billion** word tokens. These Tweets were limited to no less than **10** and no more than **64**-word tokens. The researchers also collected a massive Twitter Stream, including **4TB** of Tweet data, which was broadcast from Twitter between January 2012 and August 2019. The English Tweets in this set were tokenized and normalized using the **NLTK** toolkit's "**TweetTokenizer**" function. The translation of user mentions and web/URL links into specialized tokens denoted as *@USER* and *HTTP URL*, respectively, was part of the normalization process. The RoBERTa implementation from the **Fairseq** library (Ott et al., 2019) was used, with a maximum sequence length of **128**. This setup produced around **166 million** sequence blocks, computed as **850 million times 25 divided by 128**. The model was optimized with Adam (Kingma and Ba, 2014), as directed by Liu et al., 2019, with a batch size of **7,000** over 8 V100 GPUs (each with 32GB). A maximum learning rate of **0.0004** was employed. BERTweet's pre-training lasted **40 epochs** over a 4-week period, equating to about **950,000** training steps (166 million times 40 divided by 7,000).

Using benchmark Tweet datasets, researchers analyze and compare BERTweet's performance against outstanding baselines on three downstream NLP tasks: Part-of-Speech tagging, Named Entity Recognition, and text categorization. Three datasets were used for POS tagging: **Ritter11-T-POS** (Ritter et al., 2011), **ARK-Twitter** (Owoputi et al., 2013), and **TWEEBANK-V2** (Liu et al., 2018). Following the instructions established by Owoputi et al., 2013, the ARK-Twitter set was further separated into individual files with unique divisions for training, validation, and testing. Datasets for NER were obtained from the **WNUT16 NER** (Strauss et al., 2016) and **WNUT17** shared task on a novel and emerging entity recognition (Derczynski et al., 2017). Two datasets were used for text categorization. The 3-class sentiment analysis dataset was obtained from **SemEval2017** (Rosenthal et al., 2017), while the 2-class irony detection dataset was obtained from **SemEval2018** (Van Hee et al., 2018). For separating the datasets, many methodologies were used. The TWEEBANK-V2, WNUT16, and WNUT17 datasets kept their original divides, however, the SemEval2017 and SemEval2018 datasets were divided **90%** for training and **10%** for validation because they were only given training and test sets. Finally, throughout the experimental datasets, Nguyen et al., 2020 used two different normalization procedures. A "**soft**" technique converted certain word tokens such as user mentions and web/URL links into special tokens such as *@USER* and *HTTP URL*, and emotion icons into equivalent strings. Using lexical normalization dictionaries, a "**hard**" technique was used to further normalize word tokens in Tweets (Han et al., 2012). This technique shows a thorough way to deal with diverse datasets and tasks in natural language processing, offering important insight into data pre-processing and utilization in this subject.



	Model	Ritter11		ARK		TB-v2	
		soft	hard	soft	hard	soft	hard
Our results	RoBERTa$_{large}$	91.7	91.5	93.7	93.2	94.9	94.6
	XLM-R$_{large}$	92.6	92.1	94.2	93.8	95.5	95.1
	RoBERTa$_{base}$	88.7	88.3	91.8	91.6	93.7	93.5
	XLM-R$_{base}$	**90.4**	**90.3**	92.8	92.6	94.7	94.3
	BERTweet	90.1	89.5	**94.1**	**93.4**	**95.2**	**94.7**
	DCNN (Gui et al.)	89.9		–		–	
	DCNN (Gui et al.)	91.2 [+a]		92.4 [+a+b]		–	
	TPANN	90.9 [+a]		92.8 [+a+b]		–	
	ARKtagger	90.4		93.2 [+b]		94.6 [+c]	
	BiLSTM-CNN-CRF	–		–		92.5 [+c]	

Table 3: Scores achieved by BERTweet on Ritter11, ARK, and TB-v2 datasets for POS tagging. Nguyen et al., 2020



	Model	WNUT16		WNUT17			
				entity		surface	
		soft	hard	soft	hard	soft	hard
Our results	RoBERTa$_{large}$	55.4	54.8	56.9	57.0	55.6	55.6
	XLM-R$_{large}$	55.8	55.3	57.1	57.5	55.9	56.4
	RoBERTa$_{base}$	49.7	49.2	52.2	52.0	51.2	51.0
	XLM-R$_{base}$	49.9	49.4	53.5	53.0	51.9	51.6
	BERTweet	**52.1**	**51.3**	**56.5**	**55.6**	**55.1**	**54.1**
	CambridgeLTL	52.4 [+b]		–		–	
	DATNet (Zhou et al.)	53.0 [+b]		42.3		–	
	Aguilar et al. (2017)	–		41.9		40.2	

Table 4: Scores achieved by BERTweet on WNUT16 and WNUT17 datasets for NER tagging. Nguyen et al., 2020

Tables 3 and 4 illustrate the results obtained for BERTweet and other initial models using both "soft" and "hard" normalization approaches. The researchers discovered that "soft" normalization scores were frequently higher than their "hard" equivalents. This suggests that using lexical normalization dictionaries to standardize word tokens in tweets does not typically improve the performance of pre-trained language models for future tasks. BERTweet outperformed its main competitors, RoBERTa and XLM-Rbase, in all datasets examined, except for a modest underperformance relative to XLM-Rbase on the Ritter11-T-POS dataset. When compared to RoBERTa and XLM-Rlarge, which use far bigger model configurations, it was discovered that, while they performed better in POS tagging and NER, BERTweet performed better on the two datasets for text classification.

3. Suggested Solution

BERT (Bidirectional Encoder Representations from Transformers)

In their research paper, Devlin et al., 2019 introduced BERT. BERT is intended to pre-train deep bidirectional representations from the unlabeled text by conditioning on both the left and right context at all levels. As a result, the pre-trained BERT model may be finetuned with just one extra output layer to generate state-of-the-art models for a wide variety of tasks, such as question answering and language inference, without requiring significant task-specific architectural adjustments. BERT is both theoretically simple and experimentally powerful. It achieves new state-of-the-art outcomes on eleven natural language processing tasks, including increasing the **GLUE** score to **80.5%**, **MultiNLI** accuracy to **86.7%**, **SQuAD v1.1** question answering Test **F1** to **93.2**, and **SQuAD v2.0** Test **F1** to **83.1**. BERT employs the Transformer architecture, which Vaswani et al., 2017 introduced in their research paper. The Transformer model uses self-attention processes to assess the value of various words in a phrase in relation to a specific word. The BERT architecture comprises two stages of operation: **pre-training** and **fine-tuning**. The model learns from data with no labels across several training tasks during the first training phase. The model begins the fine-tuning with the parameters generated from the first training. Using labeled data related to the end tasks, these parameters are then changed and optimized.

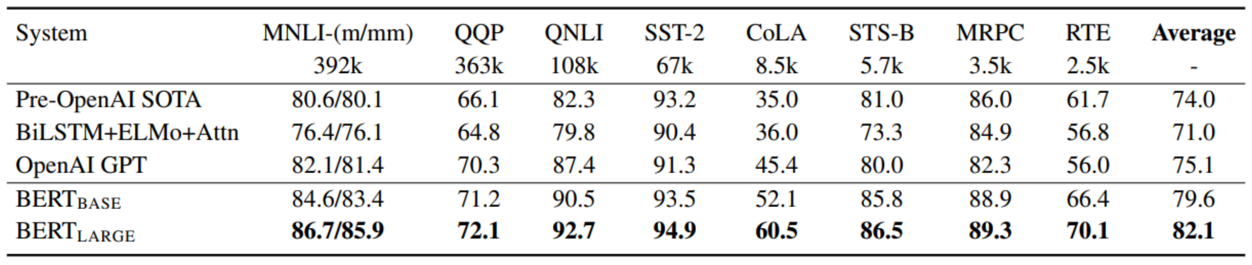
3.1. Pre-Training BERT

- **Masked Language Model:** As per Devlin et al., 2019, the MLM technique tries to train language models with a deep bidirectional representation. MLM proposes an approach for bidirectional training in which a particular proportion of input tokens are masked at random and the model is trained to predict these masked tokens. **15%** of all WordPiece tokens in a given sequence are masked at random. The model, predicts just the masked words and not the complete input. However, because the token is not utilized in fine-tuning, this creates a discrepancy between pre-training and fine-tuning. To solve this, researchers do not always use the token when masking words for training. **80%** of the masking tokens are replaced with the token, **10%** with a random token, and **10%** stay untouched.

- **Next Sentence Prediction:** The NSP task is presented to assist models in understanding the link between two phrases, which is required for subsequent tasks like Question Answering (QA) and Natural Language Inference (NLI). When picking sentences A and B for pre-training in the NSP task, B is 50% of the time the real subsequent sentence to A, and 50% of the time it is a random sentence. This short pre-training activity boosts both QA and NLI greatly. However, unlike prior research that simply transferred embeddings of sentences to subsequent tasks, BERT transmits all of its properties.

3.2. Fine-Tuning BERT

As per Devlin et al., 2019, the Transformer's self-attention mechanism lets it tackle a variety of tasks requiring single or paired texts, making fine-tuning the BERT model efficient. Rather than individually encoding text pairings and then applying bidirectional cross-attention, BERT encodes concatenated text pairs with self-attention, which automatically integrates bidirectional cross-attention. BERT incorporates appropriate inputs and outputs based on the job, and all characteristics are fine-tuned end-to-end. Paraphrasing, entailment, question answering, text categorization, and sequence tagging are examples of such tasks. Token representations, for example, are utilized for tasks such as sequence tagging, whereas the [CLS] token representations are used for classification tasks.



System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT$_{BASE}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{LARGE}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 5: GLUE Test outcomes. Devlin et al., 2019

4. Dataset

Twitter Hate Speech and Offensive Language Dataset

Davidson et al., 2017 collected tweets containing hate speech terms using a crowd-sourced hate speech lexicon. Researchers utilize crowdsourcing to categorize a sample of these tweets into three categories: those including hate speech, those having merely foul language, and those containing neither. Davidson et al., 2017 gathered tweets containing suspected hate speech using keywords from Hatebase.org. Hatebase.org is a crowdsourced repository that categorizes hateful words used on the internet. Because of the dependence on this vocabulary, the dataset is focused on probable instances of hate speech or objectionable language rather than random tweets. Using this hate speech vocabulary enabled the authors to collect a large quantity of data, with **85 million** tweets giving a rich corpus from which to generate a diversified sample. Annotators were provided rules to assist them make decisions. As per Davidson et al., 2017, Hate speech was defined as rhetoric that was detrimental or supported damage toward certain groups based on factors such as race, religion, ethnic origin, and so on. Tweets that were profane or featured swear words but did not demonstrate the damages described in hate speech were labeled as offensive. If a tweet did not fit into either category, it was placed in the 'Neither' category. The distribution of these groupings in the dataset was a remarkable observation. Out of **24,802** tweets, **5%** were classified as **hate speech**, **77%** as **offensive language**, and **18%** as **neither**.

How the dataset is labeled?
As per Davidson et al., 2017, there are several columns in the dataset: **count**, **hate_speech**, **offensive_language**, **neither**, **class**, and **tweet**. The column **'tweet**' holds the content of the assessed tweet. Multiple annotators evaluated each tweet in the collection. The '**count**' column represents how many annotators labeled a particular tweet. Each tweet was typically reviewed by three annotators to achieve a balanced perspective and minimize individual biases. The number of annotators who classified a specific tweet into one of the three categories is shown by the columns **Hate_Speech**, **Offensive_Language**, and **Neither**. For example, if a tweet was labeled as "hate speech" by two annotators and "offensive language" by one, the hate_speech column would have a value of 2, the offensive_language column a value of 1, and neither column a value of 0 for that specific tweet. Davidson et al., 2017 defined **Hate Speech** as tweets that exhibited a clear **bias**, **hostility**, or **threat** against individuals or groups based on characteristics such as race, religion, ethnic origin, sexual orientation, handicap, or gender. Annotators were particularly taught to use this designation only when a hate-filled motive was clear. Tweets that contain **harsh** or **vulgar language**, potentially **swear words**, but do not express outright hatred or threats against specific groups or persons are considered **offensive**. Tweets that did not fit into either of these categories were labeled as **neither**.

5. Implementation of BERT
The implementation begins with the import of the necessary libraries and the reading of the dataset. To comprehend class distribution, a graphical representation is created as shown in Table 6.

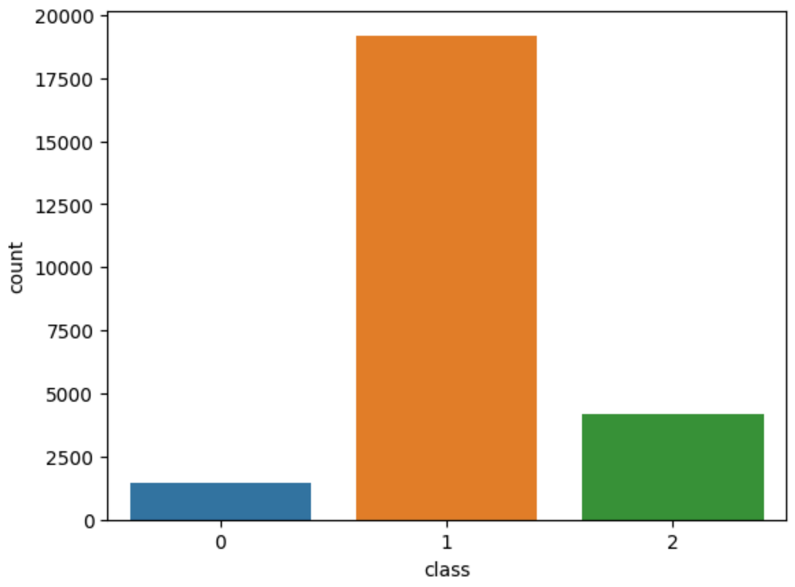
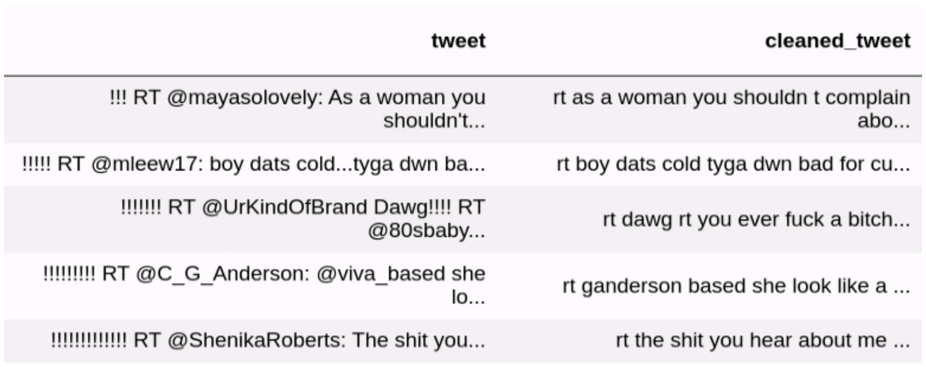


Table 6: Countplot shows the distribution of classes.

Countplot depicting the breakdown of classes in the dataset. 0 denotes hate speech, 1 denotes offensive language, and 2 denotes neither.

5.1. Data Preparation & Preprocessing: Data preprocessing is a critical step because it removes undesirable characters and patterns from individual words or strings, making them more suitable for model training. The first step is to use the **lower()** function to transform the word or string to lowercase, maintaining consistency in the text data. The next stage uses a regular expression to delete any content included between square brackets, a pattern that can be seen in tweets or textual datasets. The function then substitutes any non-word character with a space, essentially deleting special characters while maintaining word boundaries. Following that, URLs beginning with 'HTTP' or 'www' are removed, guaranteeing that any links in the tweets are removed (Liu et al., 2019).

Another regular expression is used to remove material within angle brackets, which might be beneficial for filtering out HTML tags if they are present. The following step removes punctuation, which is a common feature of written text. New line characters, denoted by 'n,' are also eliminated to maintain a continuous flow of content. Finally, any word involving numbers is removed, which is notably effective for removing terms such as '1st', '2nd', or alphanumeric combinations. The final stage in data preprocessing is to clean the tweets by deleting identities that begin with '@' and putting the tweets in an additional column called '**cleaned_tweet**' as shown in Table 7.



tweet	cleaned_tweet
!!! RT @mayasolovely: As a woman you shouldn't...	rt as a woman you shouldn t complain abo...
!!!!! RT @mleew17: boy dats cold...tyga dwn ba...	rt boy dats cold tyga dwn bad for cu...
!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...	rt dawg rt you ever fuck a bitch...
!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...	rt ganderson based she look like a ...
!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...	rt the shit you hear about me ...

Table 7: Tweets before and after preprocessing.

5.2. Model Definition and Compilation: The **TFBertForSequenceClassification.from_pretrained("bert-base-cased", num_labels=3)** method is used to initialize a BERT (Devlin et al., 2019) model particularly built for sequence classification tasks. This loads a pre-trained BERT model with case-sensitivity **("bert-base-cased")** and configures it for a classification job with 3 output labels **(num_labels=3)**. Following that, an Adam optimizer (Kingma and Ba., 2015) with a learning rate of **5e-5** is created to update the parameters of the model during training. The model is then compiled, indicating that it is ready for training, using the Adam optimizer, a loss function created specifically for categorical tasks **(SparseCategoricalCrossentropy)**, and a metric **(SparseCategoricalAccuracy)** to assess its performance throughout the training process. The **from_logits=True** input assures that the loss function will internally apply a **softmax** function to the unnormalized output of the model, transforming it to probabilities.

- **SparseCategoricalCrossentropy:** (Tensorflow Documentation) This is a loss function that is used to calculate the disparity between the observed class label distribution and the anticipated probability for multi-class classification. If there are three distinct classes and a particular instance relates to class 2, its label would be [0, 1, 0]. SparseCategoricalCrossentropy enables the use of integer labels (e.g., 0, 1, 2), which are substantially more memory-efficient.
- **SparseCategoricalAccuracy:** (Tensorflow Documentation) This is a statistic used to assess categorization accuracy. In multi-class classification problems, it computes how frequently predictions match the integer labels. SparseCategoricalAccuracy, like SparseCategoricalCrossentropy, is intended to deal with integer tags directly rather than one-hot encoded vectors. This is very helpful and effective when the total amount of classes is huge.

5.3. Handling Imbalance: As per Bartosz Krawczyk, 2016, by computing class weights, the model handles the possible imbalance of classes in the dataset. The scikit-learn **compute_class_weight** function is used to calculate the weights for each class, ensuring that underrepresented classes receive greater weight. The approach is set to '**balanced**,' which means that the weights are inversely proportional to the frequency of the classes. **np.unique(df['class'])** is used to find the unique class labels, and **y=df['class']** is used to transmit the actual class distribution. Finally, the calculated weights are translated into a dictionary format in which the keys represent class labels and the values represent class weights. This dictionary may then be utilized during model training to provide more weight to underrepresented classes, preventing the model from becoming biased toward the dominant class.

5.4. Training and Validation: The model is trained for 5 epochs. There is a considerable increase during the training period, as indicated by lowering training loss (from **0.5982** to **0.3110**) as shown in Table 8, and rising accuracy (from **0.7940** to **0.8681**). The validation results show a similar pattern, with the loss decreasing (from **0.5581** to **0.3829**) as shown in Table 8, and the accuracy increasing (from **0.8464** to **0.8623**). Two callback techniques are used to minimize overfitting and improve training efficiency: **Early Stopping** and **ReduceLR**. If the validation loss fails to increase for 4 consecutive epochs, Early Stopping stops training and restores the best model weights. This approach is well-known for its ability to prevent overfitting (Lutz Prechelt, 2012). Meanwhile, ReduceLR dynamically changes the learning rate by 0.2 if the validation loss peaks for 4 epochs, guaranteeing that the model converges quicker and avoids local minima (Leslie N. Smith, 2017).

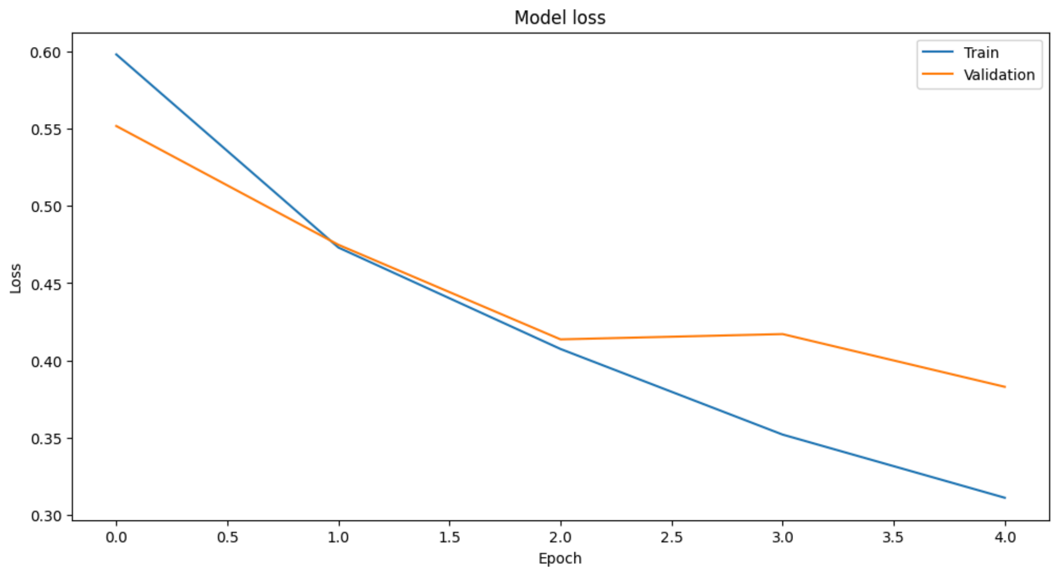


Table 8: Graph depicting Train and Validation loss for BERT.

5.5. Evaluation

The model has a **test accuracy** of **0.8360**, showing that it is capable of making valid predictions on unknown data. High accuracy frequently means that a model has effectively learned the fundamental patterns that exist in the data. The F1 score, which is a harmonic mean of accuracy and recall, is, on the other hand, **0.5670**. While not as high as accuracy, an F1 score at this level is nevertheless decent, especially in situations where there may be class mismatches (Sokolova and Lapalme, 2009). The mismatch between accuracy and F1 score implies that, while the model is generally correct, it may fail to strike a balance of false positives and false negatives in some classes (David M. W. Powers, 2020).

5.6. Improvements

The gap between the high accuracy and the moderate F1 score, which suggests possible concerns with false positives and false negatives, is one of the key areas for improvement. Improving false positives necessitates ensuring that the model does not wrongly classify samples. This may be accomplished by using approaches like data augmentation, which increases the variety of the training set and therefore makes the model more discriminating (Wang and Perez, 2017). Addressing false negatives, on the other hand, entails ensuring that all instances of a specific class are accurately detected. A proposed approach is cost-sensitive learning, in which the misclassification of minority groups is penalized more severely than the misclassification of majority groups (Charles Elkan, 2001). BERT's performance can be improved further by fine-tuning its hyperparameters. Changes in parameters such as learning rate and batch size may be made to examine how the model performs.

6. BERT Vs DistilBERT

Knowledge Distillation, as defined by Hinton et al. 2015, is a compression approach in which a compact model - the student - is trained to mimic the behavior of a bigger model - the teacher - or an ensemble of models. Sanh et al.,2020, describe in their work a strategy for pre-training a more compact general-purpose language representation model known as DistilBERT, which can subsequently be finetuned to perform well on a broad range of tasks like its bigger equivalents. Researchers use knowledge distillation during the pre-training phase to demonstrate that the BERT model may be reduced in size by **40%** while preserving **97%** of its language comprehension skills and being **60%** quicker.

As per Sanh et al.,2020, the general architecture of DistilBERT is the same as that of BERT. The pooler and token-type embeddings are deleted, and the number of layers is decreased by a factor of two. In modern linear algebra frameworks, most of the computations used in the Transformer architecture are highly optimized, and investigations revealed that variations in the tensor's last dimension have a smaller impact on computation efficiency than variations in other factors such as the number of layers. As a result, the emphasis is on minimizing the total amount of layers.

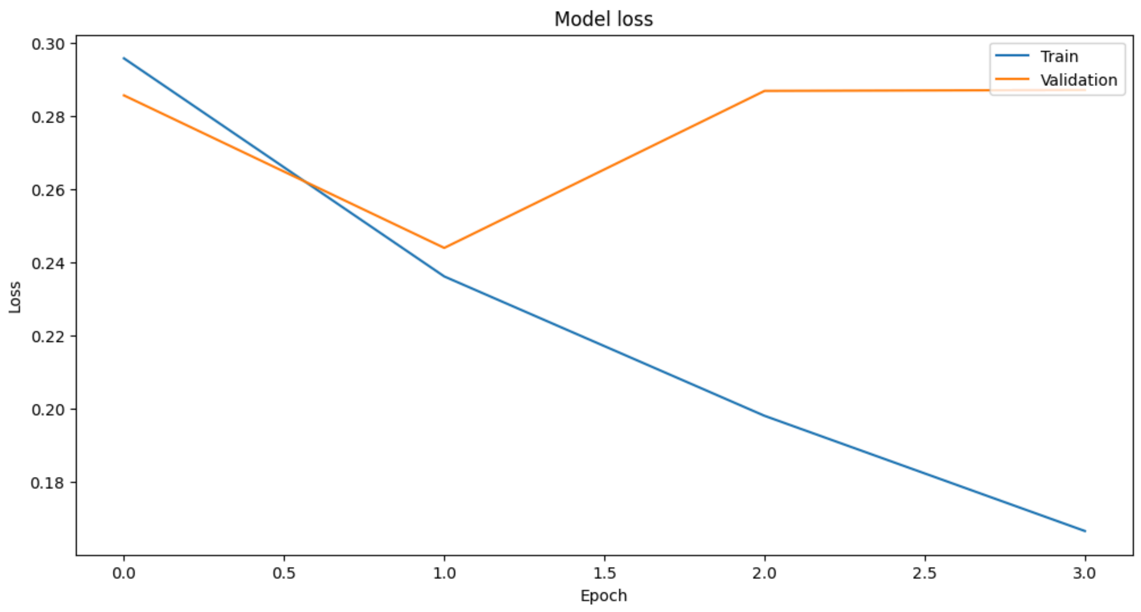


Table 9: Graph depicting Train and Validation loss for DistilBERT.

DistilBERT was also utilized to address the issue of Hate Speech and Offensive Language using the same Dataset as BERT, although BERT performed better. BERT obtained a test accuracy of **0.8360** after **5** epochs, with an **F1 score** of **0.5670**, indicating a significant increase in terms of training loss and validation accuracy from the first to the fifth epoch as shown in Table 8, whereas, DistilBERT (Sanh et al.,2020) starts off with favorable outcomes in the early epochs, as evidenced by the training performance of **0.8983** in the first epoch. However, despite a great training accuracy of **0.9381**, the validation accuracy suggests a probable overfitting problem by the fourth epoch as shown in Table 9. This mismatch between training and validation results implies that the model is fitting too closely to the training data, which might be owing to its lower model capacity compared to BERT, which makes it more prone to memorizing training samples. Overfitting in transformer models, especially models such as DistilBERT, is complex. Overfitting can occur for a variety of reasons, including the model's capacity being too great for the supplied data to a lack of proper regularisation (Zhang et al., 2016). As per Alonso et al., 2020, high accuracy does not always convert to a strong F1 score, which assesses the model's precision and recall, and this might be due to an imbalanced dataset or data biases.

7. Why do people spread Hate/Offensive speech online?

According to John Suler, 2004, one of the primary reasons people participate in hate speech on the internet is the supposed anonymity of the internet. According to the Online Disinhibition Effect, people are more prone to engage in antisocial behavior online due to a lack of face-to-face connection and a sense of invisibility. People may speak things they would not normally utter if there were no imminent real-world implications. As per Buckels et al. 2014, the design of social media platforms favors attention-seeking behavior. Trolls, as they are commonly referred to, may utilize hate speech since it consistently evokes significant emotions. Online trolls frequently exhibit sadistic tendencies, meaning that some people gain pleasure from bringing others misery.

Vicario et al., 2015 say that social media networks sometimes serve as echo chambers, exposing users to viewpoints that are similar to their own. This can lead to group polarisation, in which individuals in a group adopt a more radical stance after debating a topic than they did before the conversation. In these circumstances, hate speech and harsh language might grow. According to Zannettou et al., 2019, gaining likes, shares, or nice remarks can confirm someone's beliefs, no matter how extreme. Positive reinforcement can lead to more nasty speech. Furthermore, repeated exposure to hate speech might result in desensitization, making it appear more acceptable or usual.

8. Future Works

Integrating Optical Character Recognition (OCR) with Natural Language Processing (NLP) and Computer Vision to combat Hate Speech and Offensive Language in Memes and images.

Detecting hate speech and offensive language in memes and photos presents particular challenges, owing to the fact that memes use both textual and visual information to convey a message. This combination may be very powerful and sophisticated, making detection more difficult than when dealing with text or images individually. As per Clausner et al., 2019, multimodal content integrates many modalities of information, such as text and images. One issue with such content is that many of them feature superimposed text that cannot be handled directly by normal NLP techniques. This text may be extracted using OCR technologies. Advanced NLP approaches may be used to detect hate speech after the text has been retrieved. Memes, on the other hand, aren't only about language. The visual content and environment in which text appears can have a significant impact on its meaning. For example, a statement may be safe on its own but becomes insulting when combined with a certain image as shown in Table 10. Combining NLP approaches with computer vision can thus provide a more comprehensive analysis.

Pre-trained models have generic knowledge and have been trained on massive volumes of data. Memes, on the other hand, have their own language, style, and cultural allusions that may or may not be included in typical training datasets. In their paper named "The Hateful Memes Challenge," Kiela et al., 2021 propose a method for detecting hate speech in multimodal memes. These algorithms' performance in detecting hate speech within memes may be greatly improved by fine-tuning them using meme-specific datasets.

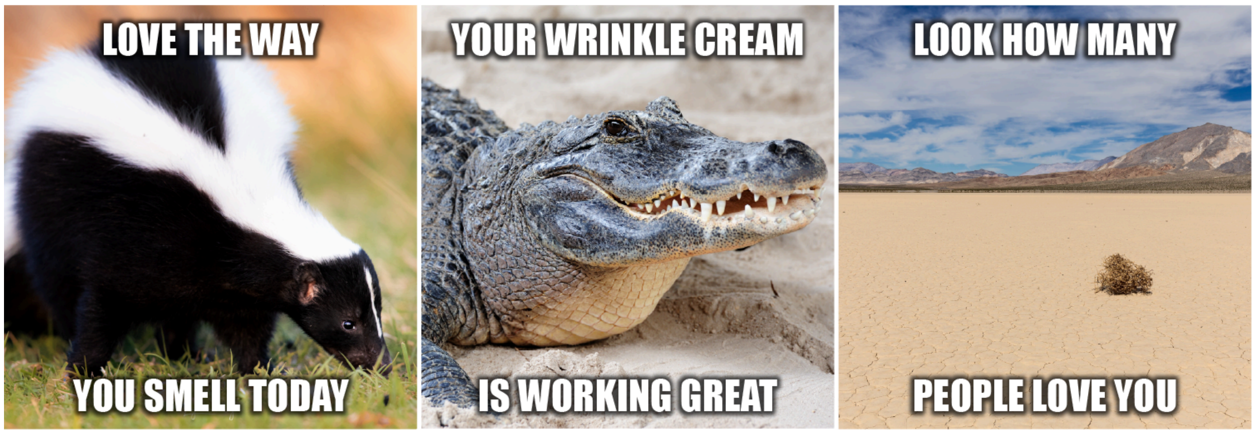


Table 10: Text when combined with the image may look offensive.
https://hatefulmemeschallenge.com/

9. Conclusion

The future areas of hate speech detection are fraught with difficulties but yet bursting with technical potential. As attempts to reduce internet hatred escalate, maintaining a balance is critical. To protect free speech, an essential component of a democratic society, it is necessary to ensure that overzealous or incorrectly deployed algorithms do not suppress free speech. The next generation of detection technologies must include subtlety and context to protect both our online communities and our inherent freedom to express ourselves.

REFERENCES

- Thomas Davidson, 2017. Automated Hate Speech Detection and the Problem of Offensive Language [online]. *In:* Dana Warmsley, Michael Macy, Ingmar Weber, eds., *Computer Vision and Pattern Recognition,* 2017. Available from: https://arxiv.org/pdf/1703.04009.pdf [Accessed 21 July 2023].
- Punyajoy Saha, 2019. HateMonitors: Language Agnostic Abuse Detection in Social Media [online]. *In:* Binny Mathew, Pawan Goyal, Animesh Mukherjee, eds., *Computer Vision and Pattern Recognition,* 2019. Available from: https://arxiv.org/pdf/1909.12642v1.pdf [Accessed 21 July 2023].
- Marzieh Mozafari, 2019. A BERT-Based Transfer Learning Approach for Hate Speech Detection in Online Social Media [online]. *In:* Reza Farahbakhsh, Noel Crespi, eds., *Computer Vision and Pattern Recognition,* 2019. Available from: https://arxiv.org/pdf/1910.12574v1.pdf [Accessed 22 July 2023].
- Victor Sanh, 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper, and lighter [online]. *In:* Lysandre Debut, Julien Chaumond, Thomas Wolf, eds., *Computer Vision and Pattern Recognition,* 2020. Available from: https://arxiv.org/pdf/1910.01108.pdf [Accessed 23 July 2023].
- Jacob Devlin, 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [online]. *In:* Ming-Wei Chang, Kenton Lee, Kristina Toutanova, eds., *Computer Vision and Pattern Recognition,* 2019. Available from: https://arxiv.org/pdf/1810.04805.pdf [Accessed 24 July 2023].
- Tommaso Caselli, 2021. HateBERT: Retraining BERT for Abusive Language Detection in English [online]. *In:* Valerio Basile, Jelena Mitrovic, Michael Granitzer, eds., *Computer Vision and Pattern Recognition,* 2021. Available from: https://arxiv.org/pdf/2010.12472v2.pdf [Accessed 24 July 2023].
- Marcos Zampieri, 2019. Identifying and Categorizing Offensive Language in Social Media (OffensEval) [online]. *In:* Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, Ritesh Kumar, eds., *International Workshop on Semantic Evaluation,* 2019. Available from: https://aclanthology.org/S19-2010.pdf [Accessed 25 July 2023].
- Valerio Basile, 2019. Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter [online]. *In:* Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Range, Paolo Rosso, Manuela Sanguinetti, eds., *International Workshop on Semantic Evaluation,* 2019. Available from: https://aclanthology.org/S19-2007.pdf [Accessed 25 July 2023].
- Yinhan Liu, 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach [online]. *In:* Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov, eds., *Computer Vision and Pattern Recognition,* 2019. Available from: https://arxiv.org/pdf/1907.11692.pdf [Accessed 25 July 2023].

- Pedro Alonso, 2020. Hate Speech Detection using RoBERTa [online]. *In:* Rajkumar Saini, Gyorgy Kovacs, eds., *International Workshop on Semantic Evaluation,* 2020. Available from: https://aclanthology.org/2020.semeval-1.292.pdf [Accessed 26 July 2023].
- Marcos Zampieri, 2019. Predicting the Type and Target of Offensive Posts in Social Media [online]. *In:* Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, Ritesh Kumar, eds., *International Workshop on Semantic Evaluation,* 2019. Available from:https://aclanthology.org/N19-1144.pdf [Accessed 26 July 2023].
- Dat Quoc Nguyen, 2020. BERTweet: A pre-trained language model for English Tweets [online]. *In:* Thanh Vu, Anh Tuan Nguyen, eds., *Computer Vision and Pattern Recognition,* 2020. Available from: https://arxiv.org/pdf/2005.10200v2.pdf [Accessed 26 July 2023].
- Myle Ott, 2019. FAIRSEQ: A Fast, Extensible Toolkit for Sequence Modeling [online]. *In:* Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, Michael Auli, eds., *Computer Vision and Pattern Recognition,* 2019. Available from: https://arxiv.org/pdf/1904.01038.pdf [Accessed 26 July 2023].
- Diederik P. Kingma and Jimmy Lei Ba, 2017. ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION [online]. eds., *Computer Vision and Pattern Recognition,* 2017. Available from: https://arxiv.org/pdf/1412.6980.pdf [Accessed 27 July 2023].
- Alan Ritter, 2011. Named Entity Recognition in Tweets: An Experimental Study [online]. *In:* Sam Clark, Mausam Etzioni, Oren Etzioni, eds., *International Workshop on Semantic Evaluation,* 2011. Available from: https://aclanthology.org/D11-1141.pdf [Accessed 27 July 2023].
- Olutobi Owoputi, 2013. Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters [online]. *In:* Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, Noah A. Smith, eds., *International Workshop on Semantic Evaluation,* 2013. Available from: https://aclanthology.org/N13-1039.pdf [Accessed 27 July 2023].
- Yijia Liu, 2018. Parsing Tweets into Universal Dependencies [online]. *In:* Yi Zhu, Wanxiang Che, Bing Qin, Nathan Schneider, Noah A. Smith, eds., *International Workshop on Semantic Evaluation,* 2018. Available from: https://aclanthology.org/N18-1088.pdf [Accessed 27 July 2023].
- Benjamin Strauss, 2016. Named Entity Recognition Shared Task [online]. *In:* Bethany E. Toma, Alan Ritter, Marie-Catherine de Marneffe, Wei Xu, eds., *International Workshop on Semantic Evaluation,* 2016. Available from: https://aclanthology.org/W16-3919.pdf [Accessed 27 July 2023].
- Leon Derczynski, 2017. Novel and Emerging Entity Recognition [online]. *In:* Eric Nichols, Marieke van Erp, Nut Limsopatham, eds., *International Workshop on Semantic Evaluation,* 2017. Available from: https://aclanthology.org/W17-4418.pdf [Accessed 27 July 2023].

- Sara Rosenthal, 2017. Sentiment Analysis in Twitter [online]. *In:* Noura Farra, Preslav Nakov, eds., *International Workshop on Semantic Evaluation,* 2017. Available from: https://aclanthology.org/S17-2088.pdf [Accessed 27 July 2023].
- Cynthia Van Hee, 2018. Irony Detection in English Tweets [online]. *In:* Els Lefever, Veronique Hoste, eds., *International Workshop on Semantic Evaluation,* 2018. Available from: https://aclanthology.org/S18-1005.pdf [Accessed 27 July 2023].
- Bo Han, 2012. Automatically Constructing a Normalisation Dictionary for Microblogs [online]. *In:* Paul Cook, Timothy Baldwin, eds., *International Workshop on Semantic Evaluation,* 2018. Available from: https://aclanthology.org/D12-1039.pdf [Accessed 27 July 2023].
- Ashish Vaswani, 2017. Attention Is All You Need [online]. *In:* Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin, eds., *Conference on Neural Information Processing Systems,* 2017. Available from: https://papers.nips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf [Accessed 28 July 2023].
- Tensorflow Documentation. tf.keras.metrics.SparseCategoricalCrossentropy [online]. Available from: https://www.tensorflow.org/api_docs/python/tf/keras/metrics/SparseCategoricalCrossentropy [Accessed 28 July 2023].
- Tensorflow Documentation. tf.keras.metrics.SparseCategoricalAccuracy [online]. Available from: https://www.tensorflow.org/api_docs/python/tf/keras/metrics/SparseCategoricalAccuracy [Accessed 28 July 2023].
- Bartosz Krawczyk, 2016. Learning from imbalanced data: open challenges and future directions [online]. eds., *Computer Vision and Pattern Recognition,* 2016. Available from: https://link.springer.com/article/10.1007/s13748-016-0094-0 [Accessed 28 July 2023].
- Lutz Prechelt, 2012. Early Stopping — But When? Universität Karlsruhe, Germany. Pg-53-54. [Accessed 01 August 2023].
- Leslie N. Smith, 2017. Cyclical Learning Rates for Training Neural Networks [online]. eds., *Computer Vision and Pattern Recognition,* 2017. Available from: https://arxiv.org/pdf/1506.01186.pdf [Accessed 01 August 2023].
- Marina Sokolova and Guy Lapalme, 2009. A systematic analysis of performance measures for classification tasks [online]. eds., *Computer Vision and Pattern Recognition,* 2009. Available from: https://www.researchgate.net/publication/222674734_A_systematic_analysis_of_performance_measures_for_classification_tasks [Accessed 02 August 2023].
- David M. W. Powers, 2020. EVALUATION: FROM PRECISION, RECALL, AND F-MEASURE TO ROC, INFORMEDNESS, MARKEDNESS & CORRELATION [online]. eds., *Computer Vision and Pattern Recognition,* 2020. Available from: https://arxiv.org/ftp/arxiv/papers/2010/2010.16061.pdf [Accessed 03 August 2023].

- Wang and Perez, 2017. The Effectiveness of Data Augmentation in Image Classification using Deep Learning [online]. eds., *Computer Vision and Pattern Recognition,* 2017. Available from: https://arxiv.org/pdf/1712.04621.pdf [Accessed 04 August 2023]
- Geoffrey Hinton, 2015. Distilling the Knowledge in a Neural Network [online]. *In:* Oriol Vinyals, Jeff Dean, eds., *Computer Vision and Pattern Recognition,* 2015. Available from: https://arxiv.org/pdf/1503.02531.pdf [Accessed 05 August 2023].
- Chiyuan Zhang, 2017. UNDERSTANDING DEEP LEARNING REQUIRES RETHINKING GENERALIZATION [online]. *In:* Samy Bengio, Moritz Hard, Benjamin Recht, Oriol Vinyals, eds., *Computer Vision and Pattern Recognition,* 2017. Available from: https://arxiv.org/pdf/1611.03530.pdf [Accessed 05 August 2023].
- John Suler, 2004. The Online Disinhibition Effect [online]. eds., *Cyberpsychology and Behaviour,* 2004. Available from: https://www.researchgate.net/publication/8451443_The_Online_Disinhibition_Effect [Accessed 08 August 2023].
- Erin E. Buckles, 2014. Trolls just want to have fun [online]. *In:* Paul D Trapnell, Delroy L. Paulhus, Available from: https://scottbarrykaufman.com/wp-content/uploads/2014/02/trolls-just-want-to-have-fun.pdf [Accessed 09 August 2023].
- Michela Del Vicario, 2015. The spreading of misinformation online [online]. *In:* Alessandro Bessi, Fabiana Zollo, Fabio Petroni, Antonio Scala, Guido Caldarelli, H. Eugene Stanley, Walter Quattrociocchi. Available from: https://www.pnas.org/doi/10.1073/pnas.1517441113 [Accessed 10 August 2023].
- Savvas Zannettou, 2019. Who Let The Trolls Out? Towards Understanding State-Sponsored Trolls [online]. *In:* Tristan Caulfield, William Setzer, Michael Sirivianos, Gianluca Stringhini, Jeremy Blackburn, eds., *Computer Vision and Pattern Recognition,* 2019. Available from: https://arxiv.org/pdf/1811.03130.pdf [Accessed 11 August 2023].
- Christian Clausner, 2019. Efficient and effective OCR engine training [online]. *In:* Apostolos Antonacopoulos, Stefan Pletschacher, eds., *International Journal on Document Analysis and Recognition (IJDAR),* 2019. Available from: https://link.springer.com/article/10.1007/s10032-019-00347-8 [Accessed 12 August 2023].
- Douwe Kiela, 2021. The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes [online]. *In:* Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, Davide Testuggine, eds., *Computer Vision and Pattern Recognition,* 2021. Available from: https://arxiv.org/pdf/2005.04790.pdf [Accessed 12 August 2023].