Project report

EXPLAINABLE HATE SPEECH CLASSIFICATION:
Comparative Analysis

Welemhret Welay Baraki

Supervisor: Prof. Juhee Bae
Examiner: Prof. Tove Helldin

Project in Informatics
with a specialisation in Data Science
Spring term 2021

ABSTRACT

In this research, the researcher studies three pre-trained Bidirectional Encoder Representations from Transformers (BERT) models that are used to detect hate speech and offensive language on Twitter data sets. The data set used for the study is based on real Twitter data, where sentences have been manually classified as either hate, offensive, or neither by users.

The natural language processing data sets have similar patterns and characteristics which can be reused. Training machine learning Natural Language Processing (NLP) models from scratch is a repetitive process and computationally expensive which requires adapting the previous model's knowledge and reuse in other related tasks. So, pre-trained BERT transformers are fine-tuned using the collected data sets, with less computational costs. The three models were evaluated using accuracy and f1-score performance measurement metrics in chapter four.

From the eXplainable Artificial Intelligence (XAI) perspective, the prediction output of the BERTweet model is explained, to emphasize and show the most important words and terms in the hate speech classification. The explanations were presented using saliency plots, heat maps, and bar charts to visualize the words and their contribution scores.

The first chapter explores related works and problem domain-related concepts; the problem statement and motivation are presented in chapter two; the research approach, process model, and implementation methods are discussed in chapter three; a detailed discussion of the experimental setup, performance, and explanation results are given in chapter four. Finally, the results and conclusions are discussed in the fifth and sixth chapters respectively.

TABLE OF CONTENTS

CHAPTER
1 - INTRODUCTION

Hate speech has been defined as “when people are devalued, attacked or when hatred or violence is called against them" by [1]. Hate speech is described as a public speech that promotes violence towards an individual or group based on their ethnicity, faith, sexual identity, or other characteristics. Different countries implement laws against hate speech to enforce citizens to restrain such behavior. Facebook and Twitter usually react back to hate speech by blocking and suspending hate speech spreader accounts on their platforms. Most of the social media contents are filtered incorrectly and hate speech spreading is on the rise.
Social sites like Twitter, Facebook, and LinkedIn, are associated and used by millions of people of the world in their day-to-day life. Every day, a massive volume of data floods and streams on various social media sites, some of which could be offensive, racist, or hateful. Detecting these hate and offensive speech is thus a very important issue that requires close attention and technical support to remove and block to create a resilient and stable society. Hate speech receives more attention than usual and insightful tweets from social media users, which can affect a large number of users in a matter of seconds.

There are hate speech databases like hatebase[2] that stores hate speech words and phrases feed from registered users. The Hatebase [2] monitors hate speech incidents using a broad multilingual vocabulary based on nationality, ethnicity, religion, gender, sexual discrimination, disability, and class, as well as NLP engines are used to perform linguistic analysis of public conversations to determine the likelihood of a hateful context.
Natural Language Processing (NLP) is a subfield of artificial intelligence that studies how to analyze natural language data. Natural language can be either textual or speech. NLP is a hot research area that demands extensive research. Transformer [3]models are a deep learning model that adopts the mechanism of attention and weights the relative importance of each part of input data. They are widely used in the area of NLP.
Transformer-based BERT models are the most prominent deep neural network models to handle the contextual representations of words in different contexts. Transformer models [3] like BERT are the recent and dominant architectures in the area of Natural Language Processing.

Explainable Artificial Intelligence(XAI)[4][5] is the way of interpreting and explaining the machine learning models’ prediction process and results to be understood by the target users (end users). Additionally, Explainable AI [6] will help developers understand the model behavior, to easily debug and improve model performance.
Explainability[7] is crucial for organizations to provide a complete understanding of AI decision-making processes to foster confidence and transparency, as well as model suitability, for the effective use and deployment of machine learning models.

The three important components of XAI algorithms are their ability to offer transparency, interpretability, and explainability [8]. *Transparency*[8] concerns the machine learning model in which different aspects are considered like model structure, individual model components, learning algorithm, and how the solution is deduced by the algorithm. *Interpretability* presents the underlying processes of the machine learning model for decision-making in a way that is understandable to humans. E*xplainability* entails the inner mechanics of machine learning models that can be explained in understandable human terms. Generally, transparency focuses on machine learning models and their constituent components, interpretability deals with the model and data, and Explainability is concerned with the machine learning model, data, and human involvement (end users).

In [9], Bidirectional Encoder Representations from Transformers (BERT) was used in the identification and classification of misogyny (i.e. gendered or not gendered) and aggression (i.e. Not Aggressive, Covertly Aggressive, Overtly Aggressive) for English, Hindi, and Bengali. BERT was applied for twitter data with little pre-processing and high performance was scored. In [9] BERT was used as a feature extractor, and finally, the researchers suggested fine-tuning the model using the training data.

N-gram and TF-IDF were used in [1] for feature extraction and Naïve Bayes, SVM, and Logistic Regression for classification of hate speech (Hateful, Offensive, and Clean) in pre-processed Twitter data and scored high accuracy level on the Logistic Regression classifier. The main problem of the N-gram is the high feature space sparsity of unseen instances and TF-IDF has not extracted the semantic meaning of words in different contexts.

In the study presented in [10], the performance of LSTM was measured by changing different parameters of the machine learning model to determine the best parameters on the number of layers, number of neurons, regularization, and epochs. Bi-LSTM with three layers has scored the highest performance in [10] for the classification of hate speech.

In studies presented in [10], [11], and [12], LSTM was used to analyze sentiments on Twitter data achieved promising results. Even though, LSTM is suitable for text classification, as it doesn't consider the contextual meaning of a word in a sentence and it is also slow to train the LSTM model.

Recurrent models like Long Short-Term Memory (LSTM) consume a considerable amount of memory and process the data sequentially. The transformers [3] improves the limitation of recurrent models by introducing attention mechanisms that are suitable for longer sequences and parallelization.

The BERT[9] is the most suitable and dominant Deep Neural Network architecture in Natural Language Processing and outperforms other machine learning models. BERT[13] is the transformer-based machine learning technique for Natural Language Processing that achieves high results in different competitions and researches. Therefore, BERT and its variants are the machine learning methods selected for this study.

In [5] a survey of different explainability techniques was discussed. SHAP [14] is a novel unified approach of interpreting model predictions. SHAP is popular, mathematically well-grounded, and the prediction score is fairly distributed and be decomposed to individual contributions.

In this research project, saliency plots, force plots, heat maps [5], and bar charts were used from the SHAP (SHapley Additive exPlanations) explanation tool. Human interpretable explanations

will be produced by highlighting words based on their contribution score to the classes. Explanation methods were discussed in detail in chapter 3.

For the successful implementation and deployment of large-scale artificial intelligence in different sectors and industries [4] model fairness, transparency and explainability are vital to be able to have a clear understanding of AI systems. Systems that work on personal data (e.g. hate speech on Twitter data) is important to show the end-users how the model works, and minimize the gap of understanding how the texts are analyzed using the BERT transformer Models.

A detailed comparative analysis of BERT variants BERT[15], RoBERTa[16], and BERTweet[17] results will be reported and communicated by the researcher using the accuracy, and f1-score metrics. In most of the previous researches, the concepts and principles of explainable artificial intelligence were not introduced. The research project develops an explainable hate speech classification model in Twitter data with the selected BERT variants. Finally, a detailed analysis of the models and a discussion of results will be documented.

CHAPTER
2 - PROBLEM SPECIFICATION

Natural Language Processing (NLP) (Xie et al. 2018) is a branch of Artificial Intelligence that mainly focuses on the use and integration of natural language in the current development of technology to support people in their day-to-day life. Most of the time, human beings use natural language in the form of speech, text, and sign to communicate with others. In this current era of technological development (digital revolution), the communication media (method) has changed into digital connectivity rather than face-to-face or paper-based (letter) communication. As a result, many varieties and differences in methods of social media platforms like Twitter, Facebook, and many more were established and revolutionized the way we communicate and share information with friends, community, and the world as a whole.

In digital communication,(Xie et al. 2018) the main challenge is the precise natural language representation and handling the different forms (contexts) of natural language due to the lack of uniform rules in natural language use. Due to these issues, NLP research is a hot research area of Artificial Intelligence that needs further research by developers, researchers, and scientists to minimize the gap.

The main advantage of social media platforms is the easiness of connecting groups of people, building relationships, positively influencing others, share expertise, learn new skills, etc. Besides the advantage of using social media, there are also disadvantages in that it enables people to engage in disseminating hate speech, harass others, motivating ethnic clashes, wars, and other irresponsible sharing of thoughts that cause harm to individuals, groups, or governments.

Based on the survey of different kinds of literature, the hate speech classification in social media platforms is a trending research area. As the main focus of this research is on the classification of hate speech in Twitter data sets, which is part of the natural language processing research domain, this project will mainly focus on the analysis and development of explainable Twitter hate speech classification using BERT.

With the revolution of digital technology and social media platforms, sharing data, resources, exchanging opinions was simplified. Even though communication and sharing of rich (important) information were among different communities from different backgrounds was simplified, a malicious and irresponsible person can easily spread hate speech, fake news, etc. to the community and chaos, and instability will trouble individuals, organizations, and the community as a whole.

Hate speech and crimes(Ezeibe 2021; Perry et al. 2020; Williams et al. 2020) have increased in recent decades. Usually, states and officials support hate speech spreaders and hate speech targeting a particular society, political party, and groups of people that endanger the rule of law. Hate speech spreaders can easily spread hate speech on ethnicity, sexual orientation, harassment, religion, race, and so on. Therefore, the social media platforms (Twitter in our case) have both blessings and curses.

Governments (Perry et al. 2020) in different countries enforce common policies on fighting hate speech and hate crime. Albeit the policies and administrative laws against hate speech, hate speech proliferation on social media is on the rise. These policies and international laws don't make an impact in mitigating and avoiding hate-speech spread. The hate speeches distributed on different social media platforms are difficult to handle with traditional administrative law enforcement,

which requires an integrated and automated hate speech classification system to handle and ease the early detection and removal of such hate speech contents and spreaders.
Many hate speech classification systems have been developed in the past few years. Based on the law of the Twitter social media platform, accounts that motivate hateful conduct may be suspended. Due to the varieties of hate speech, explicit (direct), hate speech which is easily detected and implicit (indirect) hate speech are still difficult to flag as hate speech.
The main challenges are implicit hate speech expressions are difficult to detect, lack of optimized contextual model representation, and difficulty of handling image and video, and use of multiple languages in a single tweet.
Researches starting from conventional machine learning(Ibrohim and Budi 2019; Razia Sulthana, Jaithunbi, and Sai Ramesh 2018) methods to deep neural network architectures(Gambäck and Sikdar 2017; Kamble and Joshi 2018; Zhang and Luo 2019) conducted research and development most of them lack Explainability.
Even though different researchers developed hate speech classification systems with different methods and techniques, lack the explainability of the machine learning models, and their prediction results. So developing an explainable Artificial Intelligence on hate speech classification models is the most important to ease the understandability of the end-users.
Generally, the main problem that the researcher addressed by reviewing different kinds of literature is that most of the hate speech classification researches conducted so far are black-box models that require explainability of results to the end-users.
This research project will develop an explainable hate speech classification deep learning model using the prominent model of DNN architecture BERT. After developing the hate speech classification system using BERT variants a detailed comparative analysis of the results of the models will be reported to the scientific community, guiding future work within the area.

2.1. Research Questions

Based on the problem definition, the following research questions are created.
The research questions of this research project are:

RQ1: How to build a hate speech classification model for Twitter datasets?
RQ2: Comparative Analysis: Which of the BERT Variants /BERT, RoBERTa, BERTweet/ are effective in Hate Speech classification?
RQ3: How the classification results are explained to the users?

2.2. Objectives

The main objective of this research project is to develop an explainable hate speech classification system by doing the following activities:

- Detailed literature and related works review of the problem domain to choose the best research methods and techniques to develop the system.
- From the survey of explainable AI in Natural Language Processing, suitable methods were chosen and used to interpret and explain the deep learning model prediction results.
- Suitable data sets were selected for the model training and evaluation.
- Tune model parameters and review their performance.
- Results, contributions, challenges, limitations, and future research directions will be forwarded at the final stage of the research.

CHAPTER 3 - METHOD

3.1. RESEARCH APPROACH AND METHODOLOGY

The research approach used for the study was design science research methodology which is adopted from [1]. As stated in [1], the most common research approaches used in information systems discipline are behavioral science which tries to verify and develop theories that predict or explain human behavior and design science which tries to extend the boundaries of human and organizational capabilities by creating new artifacts. As Hevner et al [1] explained, the main purpose of design science research is achieving knowledge and understanding of a problem domain by building an application of a designed artifact. Behavioral science focuses on theories, whereas, design science focuses on producing innovative artifacts to solve problems. As in [1] explained, the design science research approach is the most suitable for most of the researches in the area of computing.

Design science [1] is essentially a problem-solving paradigm that is frequently used in engineering and science to push the limits of human and organizational capabilities through the creation of new innovative artifacts. Its goal is to develop new concepts, practices, technological skills, and products that can be used in information systems.

The environment [1] in the design science research frameworks is the problem space that resides the phenomenon of research interest and is composed of people, organizations, and technology. The researcher addresses the research project by building theories and artifacts to justify and evaluate the business need. The knowledge base, which is made up of foundations and methodologies, offers scientific raw materials from which the research is carried out. Figure 1 below shows the customized design science framework, the environment includes people (researcher), organizations (Twitter), and technology (Hate Speech classifier) in which the data set is collected and the developed model is going to be delivered.

Theories, frameworks, constructs, models, methods, and techniques from the scientific society are acquired by the researcher throughout the development of the explainable hate speech classification in which previous researches are reviewed, referenced, used and, new insights and methods and constructs are going to be produced.

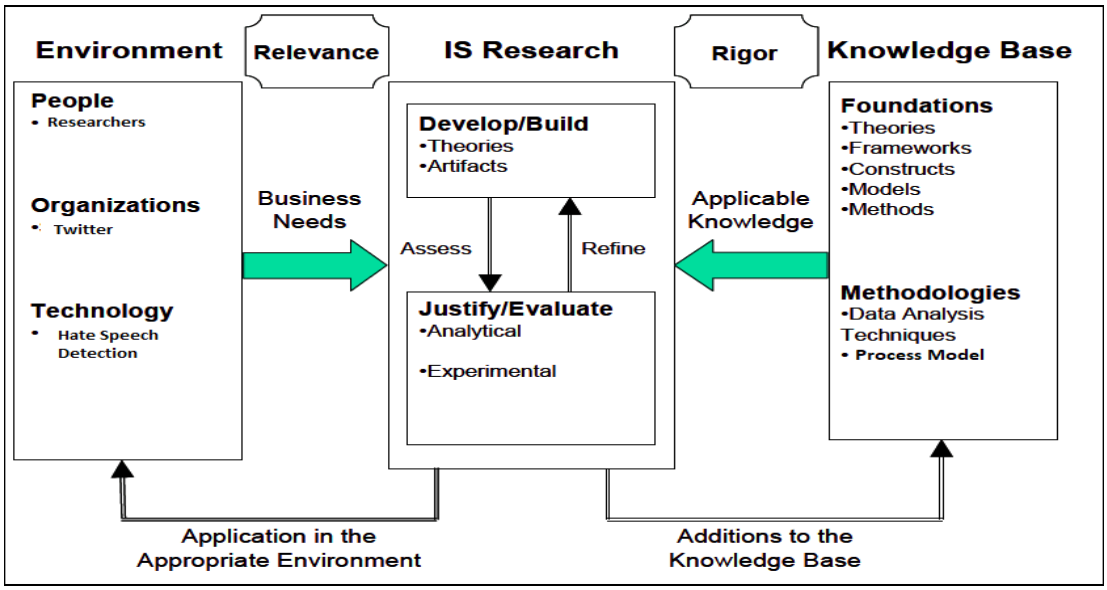


Figure 1 Design Science Framework for the Hate Speech Classification [Adopted from [1] *]*

Process model [2] is a way and method of executing and controlling the research process in a clear and identified path. There are four possible research entry points [2] in the design science research process model: problem centered in which the problem is know in advance, objective centered to satisfy certain known objectives, design and development centered, and Client /context/ initiated. From these entry points explained in [2], the problem centered research entry point was chosen since the research problems are known and stated in advance. The design science research methodology process model (DSRM) is employed during the research process, particularly the problem-centered approach is suitable for this type of research project in which the research problem is known in advance.

As shown in figure 2 below, the DSRM process model starts by clearly identifying the problem and motivate how the research problem will be solved using the document analysis typically reviewing related works and literatures. After identifying the research gaps, defining the objectives of developing an explainable hate speech classifier were used to guide the research process and pipeline by having clear conceptual representation in the researchers mind. In the design and development phase data was collected, and the data science project will be implemented with the three BERT models. The demonstration phase comes up by testing the implemented data science project and checks if it correctly works as expected using the testing data sets. Finally, the performance results and comparative analysis of the BERT models will be reported. After the evaluation results of the three BERT variant models, the best performing model will be selected for explanation. After selecting the BERTweet model, the iteration goes back to the design and development phase to include the SHAP explanation implementations. Additionally, if a new requirement and objective are identified the whole process will iterate through until sufficient and required performance and explanation result is obtained and the research output will be communicated through presentation (i.e. thesis defense).

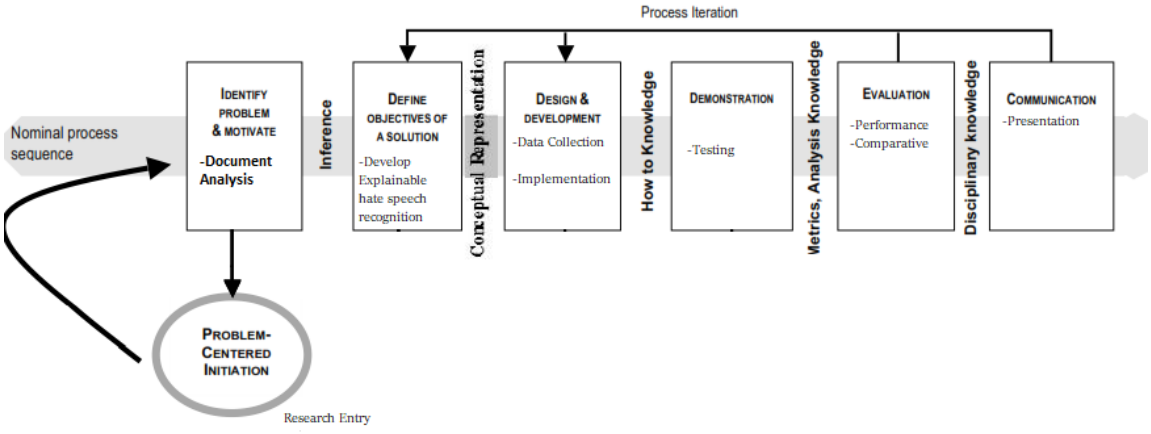


Figure 2 DSRM Process Model for Explainable Hate Speech Classification [Adopted from[2] *]*

3.2. Data Preparation

To start the project, a data preparation phase will be conducted, making the dataset ready for usage. The main activities conducted in the data preparation and acquisition process are stated as follows:

- **Data Collection**: The collected data set has three classes: hate speech, offensive, and neither, and was annotated by CrowdFlower users. The data set was available in[3]. Based on the highest number of votes of the users to each tweet, their label was assigned. All, the tweets are in the English language. Originally the data set have seven columns but the most important features class and tweet columns were selected for this study. The collected data have three labels: 0-Hate Speech, 1-Offensive Language, and 2-Neither. The total number of hate speech samples are 24783 tweets, of these 1430 are hate speech tweets, 19190 tweets are offensive and 4163 tweets are neither.

Table 1 Data Set Labels, and number of samples in each class

Label	Name	Number of Examples	
0	Hate	1430	**24783**
1	Offensive	19190	
2	Neither	4163	

	count	hate_speech	offensive_langu	neither	class	tweet
0	3	0	0	3	2	!!! RT @mayasolovely: As a wo
1	3	0	3	0	1	!!!!! RT @mleew17: boy dats co
2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Daw
3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @
4	6	0	6	0	1	!!!!!!!!!!!! RT @ShenikaRoberts:
5	3	1	2	0	1	!!!!!!!!!!!!!!"@T_Madison_x: Th
6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can no
7	3	0	3	0	1	!!!!“@selfiequeenbri: ca
8	3	0	3	0	1	" & you might not get ya bi

Figure 3 Hate and Offensive Language sample datasets

- **<u>Data Preprocessing</u>**: Data cleaning and preprocessing is the most important step in the data analysis project. The data preprocessing may include the removal of (non-ASCII characters, symbols, punctuation marks), and tokenizing the tweets will be conducted during the preprocessing step of the Twitter data.

After the Twitter data is ready, the cleaned data will be used as input to the two machine learning models, and their performance analysis results of the machine learning model will be reported in the next sections with the recommended design science research process model.

3.3. IMPLEMENTATION TECHNIQUES

3.3.1. Transformer based models

A transformer [4] is a deep learning model that uses the attention mechanism to weigh the importance of each element of the input data differently. The input sequences should have to encode, to their numerical representations based on the word and position embedding's of each token, to use and process in the transformer architecture. Embeddings are a numerical vector representation of words or tokens. The output sequence (vectors) are produced and accordingly, the prediction will be conducted.

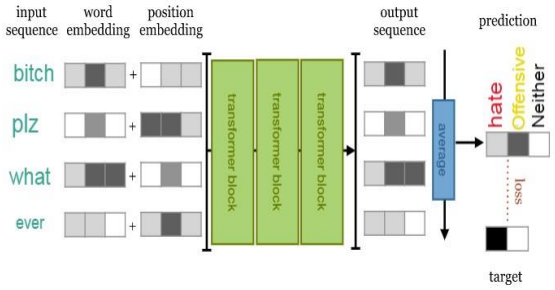


Figure 4 General Transformer Architecture

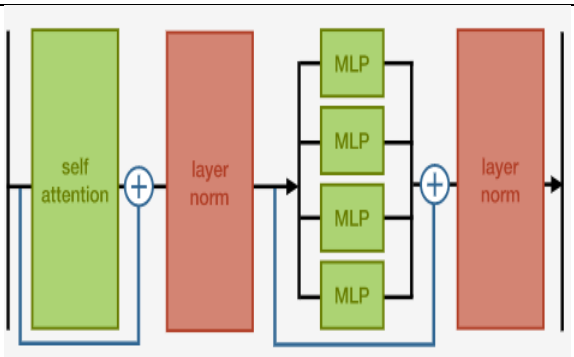


Figure 5 Transformer Block of the General Architecture

The transformer block contains Self-attention, layer normalization, and feed-forward layer (MLP). The self-attention layers are the fundamental sequence-to-sequence operation of any transformer architecture. It takes sequences of input vectors and produces a weighted average of all the input vectors. A softmax function applies to map the large values to [0,1]. Layer Normalization and residual connections are used to help deep neural networks train faster and accurately.

3.3.2. BERT Variants

Transformer-based [5] BERT models are the most prominent deep neural network models to handle the contextual representations of words in different contexts. Recurrent models like LSTM consumes a considerable amount of memory and the processing of the data is sequential. The transformers [4] improve the limitation of recurrent models by introducing attention mechanisms that are suitable for longer sequences and parallelization.

The BERTbase [6] was pre-trained with static masked language modeling and next sentence prediction (NSP). During the Masked Language Modeling (MLM) 15% of the words in the training corpus were replaced with the word [MASK] and tried to predict in an unsupervised method. The MLM [6] is used to learn the Bidirectional representation of a sentence. The NSP is used to predict if two sentences were following each other or not. The BERTbase [6] model uses 12 layers of transformer blocks, a hidden size of 768, and 12 self-attention heads.

The RoBERTa uses the BERTbase architecture and is pretrained with a large volume of data and different pretraining approaches. In RoBERTa pretraining, the model uses dynamic masking instead of static masking, and removes the next sentence prediction. BERTweet model is based on RoBERTa which includes Twitter datasets. According to the [7] the BERTweet and RoBERTa outperform the BERT model. Based on the literature review of different researches, BERT [5], [6], [8] is suitable for this research project. Currently, BERT was the most popular transformer-based machine learning technique for natural language processing which was successfully used in recent international competitions and researches[5], [9],[10]. BERT can handle different contexts based on the left and right sequences and can easily be adapted to different NLP tasks[11]. As result, the researcher selected BERT, RoBERTa, and BERTweet models for the implementation of this research project.

3.3.3. Explanation Methods

The most popular explanation tools used in the area of XAI are SHapley Additive exPlanations (SHAP) [12], [13] and Local Interpretable Model-agnostic Explanations (LIME) [14]. SHAP (SHapley Additive exPlanations) [13] is a game-theoretic technique to explain any machine learning model's output. It uses the classic Shapley values from game theory and their related extensions to correlate optimal credit allocation with local explanations.

LIME [14], [15] used to explain individual predictions of black box machine learning models using local surrogate models. Surrogate models are trained to approximate the predictions of black box models.

SHAP [15] can decompose the final prediction score/result/ into individual contributions in an additive way, which is not possible in LIME. LIME creates a surrogate model locally to identify the most important feature in a particular data point of interest. As in [15] stated, SHAP don't require parameter tuning, uses consistent principles and mathematically well-grounded and, involves on fairly distributing both gains and costs to several features working together. Whereas, LIME [15] is not grounded on consistent principles, doesn't guarantee the prediction is fairly distributed and it can require tricky parameter tuning. LIME [15] assume linear behavior of the machine learning model locally, but there is no theory as to why this should work. Practically machine learning models especially the neural networks have no linear behavior.

SHAP [15], [12], [13] is the most suited explanation technique to explain the prediction results of this particular scenario. Especially, the additive property of the Shapley values is the most important feature available in SHAP Explanation tool to find contributions of individual words based on prediction result. SHAP [15] method is the only method to deliver full explanation. SHAP is simple to use, provide an overall view of the prediction results, and how their values are impacting the model's predictions. To explain text input and prediction results of the models to the end-users, saliency plots, heatmaps, force plots and bar charts were used to explain the output results of the BERT model.

CHAPTER 4 - RESULTS

In this section, the major data science project development and implementation pipeline will be discussed in detail, specifically, on data preprocessing, model creation, training, evaluation, and visualization. The best BERT variant was selected based on the performance results of each of the models. The prediction results of the best BERT model will be explained using SHAP explanation tool. The general experimental setup of the data processing and model building process is depicted in Figure 6 below.

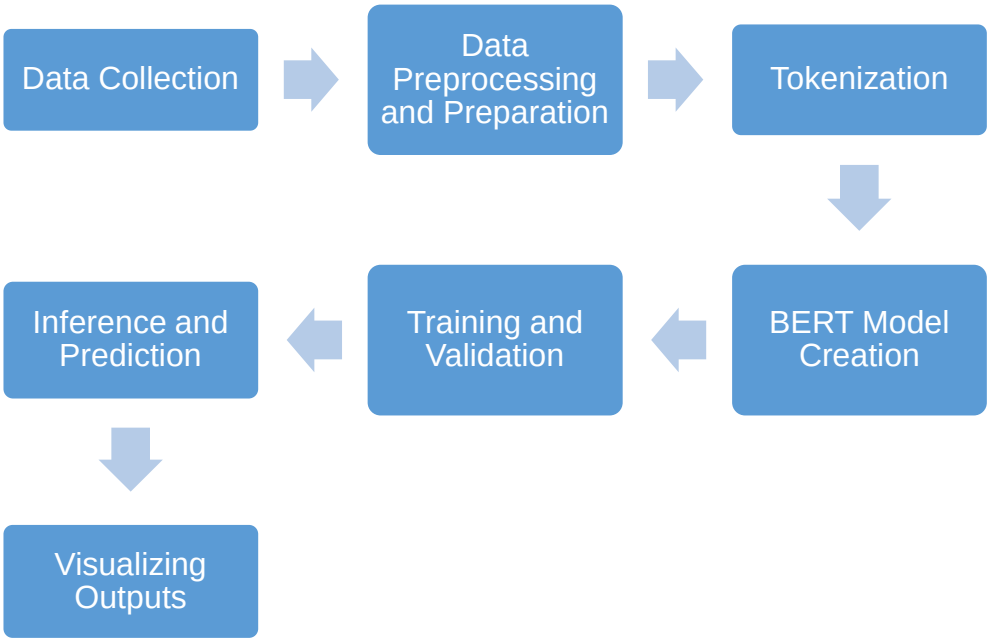


Figure 6 Experimental setup and development pipeline of the project

The general experimental setup includes data set collection, data preprocessing and preparation, input embedding (tokenization), BERT Model instantiation, training and validation to fine-tune the BERT Model, inference and prediction, finally explaining the prediction results of the selected model.

4.1. Data Preparation and Preprocessing

The collected Twitter data was cleaned and preprocessed to make it ready to fine-tune the BERT model. Irrelevant fields (columns) except the class (i.e. hate, offensive, and neither) and text (i.e. tweet) were all deleted. The class and text(tweet) columns were rearranged from the [label, tweet] to [tweet, label] form. Duplicates were checked and redundant records were removed.

From tweets newlines, and HTML tags were removed. Regular expressions were used to remove hyperlinks, hash characters, and user names. For the sake of confidentiality of personal data, usernames were removed from each tweet.

In the first experiment, 2000 tweets were used for training (50%) and evaluation (50%). The performance results were reported in section 4.4.1.

Finally, the whole cleaned data was split into three different datasets: 90 % of the datasets used for training, 10% of the training datasets were used for validation, and 10% of the whole data sets

were used for testing. Finally, the data set was formatted and changed to the DataSet dictionary. These data sets are tokenized and used for BERT variant Models. Finally, the training datasets have 20073 examples, 2231 examples were used for validation, and 2479 examples are used for testing.

4.2. Tokenization

Tokenization is the process of decomposing sentences and complicated words into their sub-components. We can use any other BERT tokenizer, but we will get the best results if we tokenize the tweets with the tokenizer that the BERT model was trained on. The pre-trained BERT library provides different tokenizers for the BERT models. The BERT tokenizer uses the maximum length of 128, padding to the maximum length, and truncate if there are sequences that are longer than the maximum length.

For the three BERT Variants in which 'bert-base-uncased' for BERT, 'roberta-base' cased for RoBERTa, and 'vinai/bertweet-base' cased were used for the tokenization of the prepared data sets. These tokenized data sets were used to train and evaluate the performance of the three BERT models. Since the Deep Neural Network architectures deal with numbers, so as the encoded data using the tokenizers are numbers. The additional field of "text" in the tokenized dataset was removed. Finally, the tokenized dataset contains these fields: 'attention_mask', 'input_ids', 'label', and 'token_type_ids'. The 'attention_mask' is used when batching sequences together to indicate the model which tokens should be attended to or not, mostly used in padding. The 'input_ids' are the numeric representation of the tokens. The label is the numeric representation of the three classes (i.e. 0 -- for Hate Speech, 1—for Offensive Language, and 2 – for Neither). The 'token_type_ids' is used to identify a token to which sequence belongs to. Token type IDs are also called segment IDs.

4.3. Model Building

The BERT Model accepts tokenized data that incorporates word embeddings, position embeddings and token type embeddings (Segment IDs). The tokenized data accepted to the BERT model is followed by layer normalization and dropout layers. The layer normalization and dropout layers are used in every layer of the BERT models to speed up the training(fine-tuning) process of the Model. The Multi-head Self-Attention and Multilayer Perceptron are used in the encoders in addition to the dropout and layer normalization. Finally, the classification head uses neural network classifier to produce the prediction result.

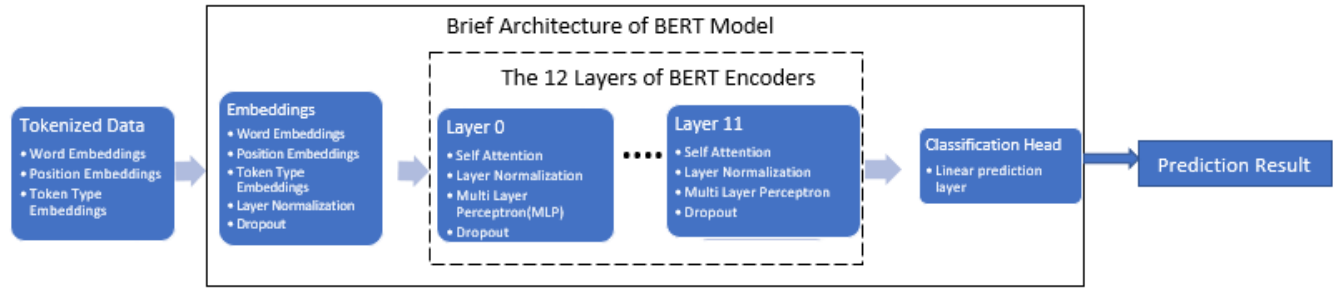


Figure 7 General Architecture of the BERT Model

BERTweet is a pre-trained model on twitter date set and adapted RoBERTa architecture pre-trained on 850M Tweets. RoBERTa improves the architecture of the BERT by pre-training on the large volume of text data and replaces the Next Sentence Prediction by dynamic masking. The BERTbase has 12 encoders with 12 bidirectional self-attention heads. The model is pretrained from books corpus with 800M words and Wikipedia with 2,500M words.

There are lots of pre-trained BERT libraries for model building. Of the many BERT Variants available 'bert-base-uncased' for BERT, 'roberta-base' cased for RoBERTa, and 'vinai/bertweet-base' cased pre-trained models were used in the model building and fine-tune process. The tokenized data sets were used to train and evaluate the performance of the three BERT models: BERT Base, RoBERTa, and BERTweet. The model's configuration was customized to map labels (Hate Speech, Offensive Language, Neither classes) and to ids (0, 1, and 2) and vise versa.

The common parameters used in the model building for the three models are epoch -3, the learning rate of 5e-05, optimizer AdamW, 500 steps, 32 batch size, and 'steps' evaluation strategy. As batch size increases, a high GPU memory is required to run a given architecture to perform the training operations. The researcher has chosen a small learning rate to allow the BERT model to learn an optimal or even globally optimal set of weights but consumes lots of time during the model training period. A linear prediction layer on top of the pooled output with softmax activation function was used to predict the respected class.

4.4. Model Evaluation and Performance Results

4.4.1. Fine-tuning and Evaluating with Small data sets

In the first experiment, two thousand tweets have been chosen to fine-tune and evaluate the three BERT models. One thousand tweets were used for training. During the training phase, the other 1000 tweets were used for evaluation using the steps evaluation strategy. Evaluation is logged after every 100 steps. In this experiment the difference is only data sets size and the number of steps used to evaluate the model. The other parameters are identical to the above models parameter specifications. The validation results of the BERTweet, BERT-base, and RoBERTa models were enlisted in Table 2 below.

Table 2 Performance results of the three models using small data sets

Model	Step	Validation Loss	Accuracy	F1
BERTweet cased	***100***	***0.315537***	***0.913***	***0.8866***
	200	0.405742	0.885	0.8848
	300	0.403091	0.896	0.8835
BERT base uncased	100	0.603496	0.873	0.8701
	200	***0.571902***	***0.886***	***0.8739***
	300	0.638352	0.887	0.8786
RoBERTa cased	100	0.468853	0.894	0.8646
	200	0.473342	0.889	0.8742
	300	***0.413236***	***0.881***	***0.8732***

In the BERTweet-cased BERT model, the best score was achieved in the first epoch of the training phase, whereas the others BERT-base and RoBERTa achieved their minimum validation loss after the second and third epochs of the training phase respectively. Comparatively, the BERTweet-cased model has minimum validation loss of the other two models even at their optimized and best step. The BERTweet model outperforms the other two models with small training datasets. The validation loss of the RoBERTa base is lower and is even faster than the BERT-base.

The summary of the test results of the three BERT variants using small dataset are listed in the following table 3.

Table 3 The Overall evaluation results of the three BERT Variants using 1000 examples

Model	Accuracy	Weighted F1 Score
BERTweet cased	91.3	88.7
BERT base uncased	88.6	87.4
RoBERTa cased	88.1	87.3

The BERTweet model achieves the best result of the three variants of BERT models with small training data. This entails that transfer learning is good practice in natural language processing. Especially, if the model is trained on related data sets, we can easily be fine-tuned with a much smaller amount of data. The BERTweet model can easily be fine-tuned with a small amount of data in this case 1000 tweets to achieve good performance results.

4.4.2. Fine-tuning and evaluating with large data sets

The three models were trained and evaluated with the full data sets as stated in section 4.1. The training datasets have 20073 examples, 2231 examples were used for validation, and 2479 examples were used for testing. The overall testing results of the three models were reported in table 4 below.

Table 4 The Overall testing results of the three BERT Variants after training with the whole training datasets

Model	Accuracy	Weighted F1 Score
BERTweet cased	91.6	89.1
BERT base uncased	91.8	90.6
RoBERTa cased	90.8	89.7

From table 4 above the evaluation results with large training data sets achieved almost similar performance scores. Based on evaluation results using different sizes of training data, the BERTweet model is taking advantage of the pretraining tweeter datasets that were explicitly revealed when training with small datasets. Because tweeter data sets are not similar to other text data like books and wiki data, they are short in length and have different characteristics. Models pretrained in similar data in different problem domains will give the advantage of easily fine-tune with a small amount of training data sets in this case using 1000 training tweets instead of 24,000 tweet samples.

4.5. Explaining the Model Results

The model outputs were explained using heat maps, bar charts, force plots and saliency plots which were available in the SHAP Model explanation tool. The pink color is used to show the positive impact of words and tokens on a specific class and the blue indicates a negative contribution to the class.

The output of the selected BERT Model was rendered using the SHAP (SHapley Additive exPlanations). The model's explanation is based on the score of the marginal contribution of each word related to that particular sentence. The users can easily understand the important words based on the explanation results of the chosen BERTweet Model. To compute the Shaply values partition algorithm was used.

To demonstrate the explanation results of the BERTweet model seven tweets (i.e. 3 hate speech, 2 offensive, and 2 tweets neither) were chosen from the hate and offensive language datasets, and their explanation results were rendered using three plotting methods: heat maps (i.e , figure 8 and table 6), bar charts (i.e. Figures 12, 13, 14, 15, and 16), saliency plots (i.e. Table 7) and Force plots (i.e. figures,9,10,and 11).
The chosen sample tweets and their corresponding class category are listed in Table 5 below.

Table 5 Sample tweets used for Visualization and Explanation

Sample Tweet	Class Category
My grandma used to call me a porch monkey all the time Come to think of it she did refer to a broken bottle as a nigger knife	Hate Speech
At least I am not a nigger	
California is full of white trash who moved from Oklahoma	
bitch plz whatever	Offensive Language
Stay beautiful you bitch	
Just follow the yellow brick twitter.	Neither
he is in sole heaven	

Generally, in all the visualizations the pink color represents the positive impact (Contribution) and the blue color represents the negative impact (contribution) of words towards a particular class in the BERTweet prediction results. Specifically, in the heatmap and saliency plots, dark pink shows the high positive impact, light pink shows the low positive impact, dark blue depicts the high negative impact, and light blue shows the low negative impact. In the heatmaps and saliency plots, gradient of colors were used. In the bar charts and force plots, two colors: pink for positive contribution and blue used for negative contribution.

4.5.1. Visualizing all the Samples on all of the three classes

A. Heatmap Results

Heatmap is a data visualization technique that shows the magnitude /marginal contribution/ of tokens /words/ in texts and image pixels in images with gradients of colors. The color variation may give obvious explanation visuals to the reader how a particular text is influencing and contributing to the three classes.
Visualization results based on the individual contribution of each token to the prediction results are visualized using heatmaps and saliency plots.
Based on Figure 8, the word **'a nigger knife'** and **'nigger'** are positively contributing to the "Hate Speech" class. Whereas the *"My grandma used to call me a porch monkey all the time Come to think of it she did refer to a broken bottle as"* and *"At least I am not a"* are negatively contributing towards the "Hate Speech" class. The detailed heat-map visuals of the sample tweets are listed in Table 6 based on their class categories.

From the heatmaps, we can figure it out and identify the terms and words that positively or negatively contributing to a particular class. Additionally, we can simply display their marginal contribution on the classes, first by selecting the particular class and clicking on each of the words, each word's contribution to a particular class can be displayed above each word. If the user wants to display the individual contribution of words to each class the saliency plots (see Table 7) are more easily and intuitive to navigate with the individual contribution in each of the three classes.

Figure 8 Visualization Heat map results that show the contribution of terms in 'Hate Speech' class

The heatmap results for each sample text based on the three classes are visualized in Table 6. The dark pink colors are those words that have strong positive contributions and the dark blue color shows/highlights/ the words that negatively contribute towards a specific class. The gradient of colors is used to highlight the words based on the contribution/score/ towards a specific class.

Table 6 Results of the heat-map visualizations of three classes on sample tweets

Hate Speech	Offensive Language	Neither
My grandma used to call me a porch monkey all the time Come to think of it she did refer to a broken bottle as a nigger knife	My grandma used to call me a porch monkey all the time Come to think of it she did refer to a broken bottle as a nigger knife	My grandma used to call me a porch monkey all the time Come to think of it she did refer to a broken bottle as a nigger knife
At least I am not a nigger	At least I am not a nigger	At least I am not a nigger
California is full of white trash who moved from Oklahoma	California is full of white trash who moved from Oklahoma	California is full of white trash who moved from Oklahoma
bitch plz whatever	bitch plz whatever	bitch plz whatever
Stay beautiful you bitch	Stay beautiful you bitch	Stay beautiful you bitch
Just follow the yellow brick twitter	Just follow the yellow brick twitter	Just follow the yellow brick twitter
he is in sole heaven	he is in sole heaven	he is in sole heaven

From the above Table 6, the user can easily understand the words that have low/high positive and negative impact. In Table 6 in the hate speech column, the words *'a nigger knife'*, *'nigger'* and *'white trash'* which are highlighted with dark pink have a high positive impact on the 'Hate Speech' class score, and their likelihood of categorizing these tweets to the 'Hate Speech' class is high. But the words that are shaded with light blue colors are negatively contributing to the hate speech class. The users can simply observe the category of a particular tweet, based on the heat map visualization.

B. Saliency Plot Results

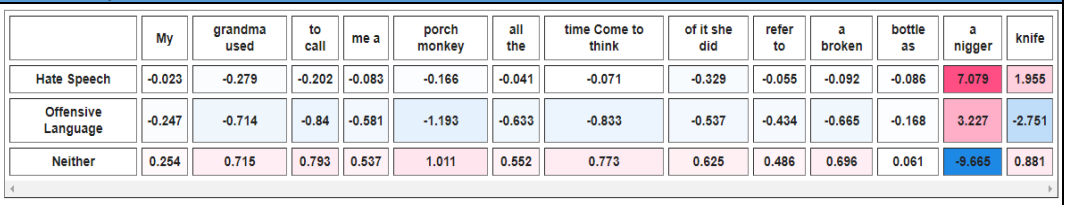
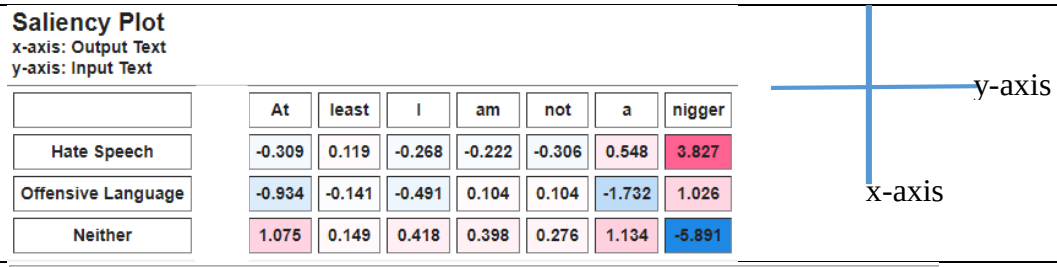
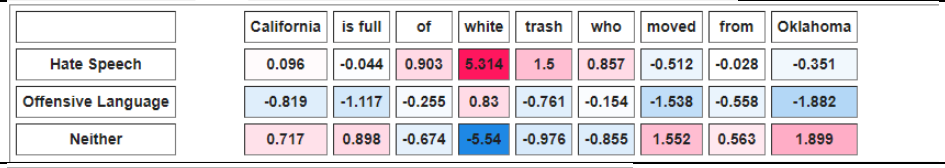
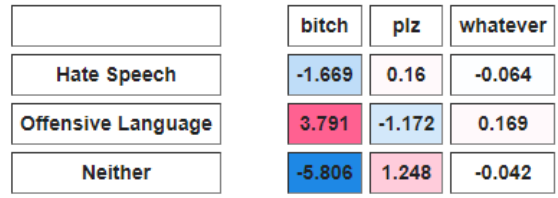
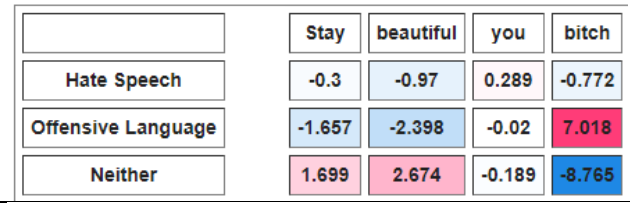
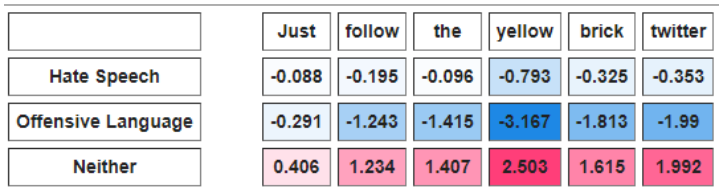
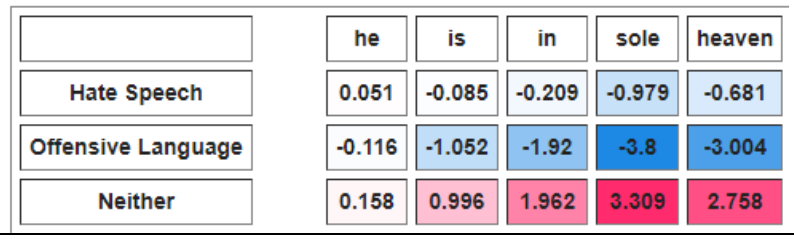
Similar to the heatmaps, the saliency plots are used to show the detailed individual contribution of tokens/terms/ towards each of the classes. The scores which are not directly displayed in the

heatmaps are easy to show in the saliency plots. Input text was used in the y-axis (i.e. as columns), whereas the output classes are in the x-axis (i.e. like row ids) and their corresponding scores/contributions/ of each word are listed in a tabular form, and their corresponding contribution to each of the three classes can easily be navigated. The x-axis and y-axis are not the same as that of the Cartesian coordinates used, they are reversed. Simply, the input-text (y-axis) is used as a column names (as used in table) and the output-text /class names/ are used in the x-axis (row-indexes as in data-frames and tables)

If the user is more interested in all the marginal contribution scores of each of the word tokens, the saliency plot is suitable. The impact score of each word is displayed as a value in each of the tabular cells to each of the classes. In addition to the numeric contribution score of the words, the gradient of colors used in the heat map is used to show the positive and negative impact of words in different classes. The saliency plots of the sample texts are depicted in Table 7.

From Table 7, let's consider the tweet *'he is in sole heaven'*. The five tokens of the sample tweet *'he is in sole heaven'* are *'he', 'is', 'in', 'sole' a*nd *'heaven'* are used as a column name and the three classes "Hate Speech", "Offensive Language" and "Neither" are used as a row index(identifier) and finally, the table cells are populated with their corresponding contribution scores. The contribution scores of all the tokens to the "Hate Speech" and "Offensive Language" classes are all negative (shaded with varieties blue colors), whereas contributions to the "Neither" class of all the tokens are positive (shaded with pink colors). From the saliency plots, the user can easily visualize the exact individual impact of each of the words to each of the classes with numerical scores and colors of gradients similar to the heat maps.

Table 7 Saliency Plots of seven Sample Tweets

Sample Tweet	Saliency Plots
My grandma used to call me a porch monkey all the time Come to think of it she did refer to a broken bottle as a nigger knife	
At least I am not a nigger	
California is full of white trash who moved from Oklahoma	
bitch plz whatever	
Stay beautiful you bitch	
Just follow the yellow brick twitter.	
he is in sole heaven	

	My	grandma used	to call	me a	porch monkey	all the	time Come to think	of it she did	refer to	a broken	bottle as	a nigger	knife
Hate Speech	-0.023	-0.279	-0.202	-0.083	-0.166	-0.041	-0.071	-0.329	-0.055	-0.092	-0.086	7.079	1.955
Offensive Language	-0.247	-0.714	-0.84	-0.581	-1.193	-0.633	-0.833	-0.537	-0.434	-0.665	-0.168	3.227	-2.751
Neither	0.254	0.715	0.793	0.537	1.011	0.552	0.773	0.625	0.486	0.696	0.061	-9.665	0.881

Saliency Plot
x-axis: Output Text
y-axis: Input Text

y-axis

x-axis

	At	least	I	am	not	a	nigger
Hate Speech	-0.309	0.119	-0.268	-0.222	-0.306	0.548	3.827
Offensive Language	-0.934	-0.141	-0.491	0.104	0.104	-1.732	1.026
Neither	1.075	0.149	0.418	0.398	0.276	1.134	-5.891

	California	is full	of	white	trash	who	moved	from	Oklahoma
Hate Speech	0.096	-0.044	0.903	5.314	1.5	0.857	-0.512	-0.028	-0.351
Offensive Language	-0.819	-1.117	-0.255	0.83	-0.761	-0.154	-1.538	-0.558	-1.882
Neither	0.717	0.898	-0.674	-5.54	-0.976	-0.855	1.552	0.563	1.899

	bitch	plz	whatever
Hate Speech	-1.669	0.16	-0.064
Offensive Language	3.791	-1.172	0.169
Neither	-5.806	1.248	-0.042

	Stay	beautiful	you	bitch
Hate Speech	-0.3	-0.97	0.289	-0.772
Offensive Language	-1.657	-2.398	-0.02	7.018
Neither	1.699	2.674	-0.189	-8.765

	Just	follow	the	yellow	brick	twitter
Hate Speech	-0.088	-0.195	-0.096	-0.793	-0.325	-0.353
Offensive Language	-0.291	-1.243	-1.415	-3.167	-1.813	-1.99
Neither	0.406	1.234	1.407	2.503	1.615	1.992

	he	is	in	sole	heaven
Hate Speech	0.051	-0.085	-0.209	-0.979	-0.681
Offensive Language	-0.116	-1.052	-1.92	-3.8	-3.004
Neither	0.158	0.996	1.962	3.309	2.758

4.5.2. Visualizing the contribution of words in a single class

In this section, the contribution of each of the tokens and their corresponding scores in each of the three classes were visualized.

A. Visualizing the contribution of words in each tweet on the "Hate Speech" class

From Figure 9, the words that increase the output chance of the model to the “Hate Speech” class when included are shaded with pink color, and the words that decrease (negatively impact) likelihood of becoming the “Hate Speech” class are shaded with blue color.

When the user hovers on the text, it shows the score on the plots, when the user hovers on the plots, the corresponding text will be underlined.

For the first instance in Figure 9 the overall tweet score (i.e. f(x) in the figure) is calculated is displayed in bold. The overall tweet score value is positive which shows that tweet is “Hate Speech”. Length of the colors of gradients is proportionate to the contribution score of the tokens. The word *‘a nigger’* has a positive impact and has the highest score of 5.236 and the other words have low negative impacts. If the whole sentence score is positive meaning it can be classified as hate speech. If the whole sentence score (i.e. f(x) in the figure) is negative meaning it is not “Hate Speech” class. The user can easily understand the overall score of the classification results of a given sentence and individual contributions of each of the words included in the tweet. From the figure below the “Offensive Language” and “Neither” classes of tweets overall sentence scores are negative meaning they don’t belong to the “Hate Speech” class.

Figure 9 Visualizing the contribution of words in 'Hate Speech' class

B. Visualizing the contribution of words in each tweet on the "Offensive Language" class

From Figure 10, the words that increase the output chance of the model to the “Offensive Language” class when included are shaded with pink color, and the words that decrease (negatively impact) likelihood of becoming the “Offensive language” class are shaded with blue color.

The word *‘bitch’* in the tweet sample “*bitch plz whatever*” has a positive impact and has the highest score of 4.247 which has the longest gradient of colors shaded with pink and the other words have low negative impacts. The whole sentence score is positive(3.808239) meaning the tweets class is “Offensive Language”. If the whole sentence score is negative meaning it is not “Offensive Language” class. As a result the “Hate Speech” and “Neither” classes of tweets overall sentence scores are negative meaning they don’t belong to the “Offensive Language” class. This is interactive and the user can easily understand the overall score of the classification results of a given sentence and individual contributions of each of the words included in the tweet.

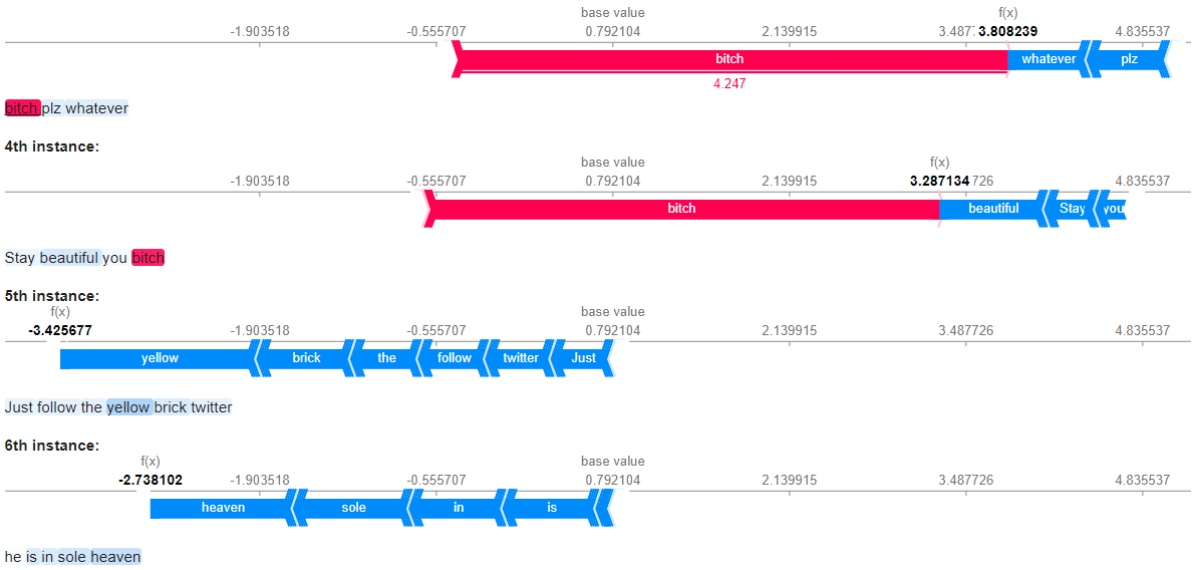


Figure 10 visualizing the contribution of words in 'Offensive Language' class

C. **Visualizing the contribution of words in each tweet on the "Neither" class**

From Figure 11 below, the words that increase the model output chance to the “Neither” class when included are shaded with pink color, and the words that decrease (negatively impact) likelihood of becoming the “Neither” class are shaded with blue color. When the user hovers the text, it shows the score on the plots, when the user hovers on the plots, the corresponding text will be underlined.

The tweet sample *“Just follow the yellow brick twitter”* have a positive impact on the “Neither” class and is shaded with pink color. The words are plotted along the line proportionate to their impact score, and the texts are highlighted with a gradient of colors (similar heatmap). The whole sentence score is positive (3.210611) meaning the tweet class is “Neither”. If the whole sentence score f(x) is negative meaning it is not “Neither” class. As a result, the “Hate Speech” and “Offensive Language” classes are negative meaning they don’t belong to the “Neither” class. This is interactive and the user can easily understand the overall score of the classification results of a given sentence and individual contributions of each of the words included in the tweet.

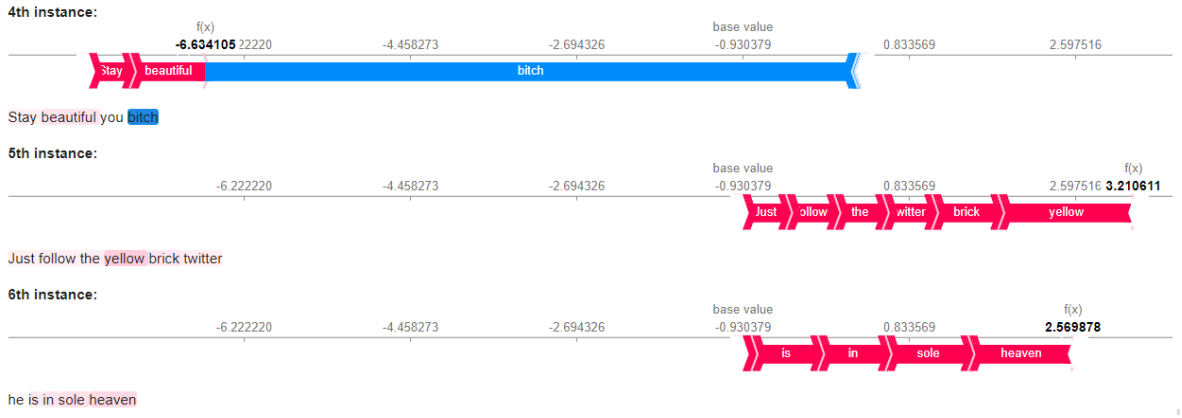


Figure 11 visualizing the contribution of words in the 'Neither' class

4.5.3. Visualizing top words impacting on a single using Bar charts

To visualize the top influencing words in each of the three classes (i.e. hate speech, offensive and neither) bar charts are plotted with the top impacting words and their contribution score. The bar graph is simple to understand and intuitive to identify the top impacting words in a particular class.

A. Top Words Impacting the "Hate Speech” Class

The researcher uses the average/mean/ of each word in the sample tweets that have a high impact /negatively or positively/ on the hate speech class. In Figure 12 below, *'nigger' 'white',* and *'trash'* have a high positive influence on hate speech class. Whereas, *"yellow"* and *'whatever'* are the words that negatively impact the hate speech class.

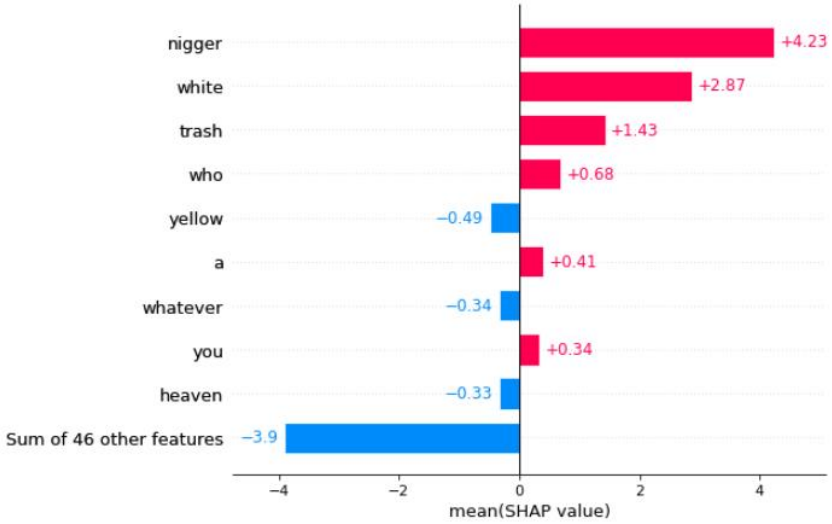


Figure 12 The top words impacting negatively and positively to the "Hate Speech" Class

To identify the topmost words that are negatively impacting the hate speech class we can use the mean and plot the bar chart in descending order so that the top negatively impacting words are displayed as in Figure 13 below. The *wo*rds *'yellow'*, 'whatever', 'heaven' and so on are the top negatively impacting words in the "Hate Speech" the class.

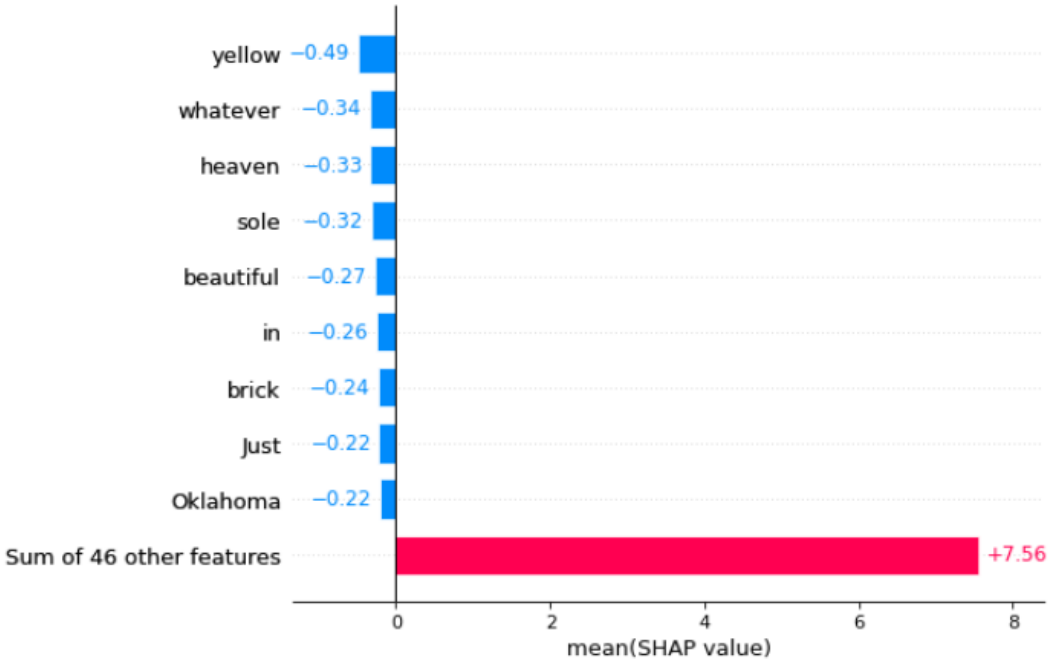


Figure 13 the top words impacting negatively to the "Hate Speech" Class

To identify the topmost words that are positively impacting the hate speech class we can use the mean and plot the bar chart in ascending order so that the top positively impacting words are displayed as in

Figure 14 below. The *words 'nigger*, 'white', 'trash' and so on are the top words positively impacting the hate speech class.

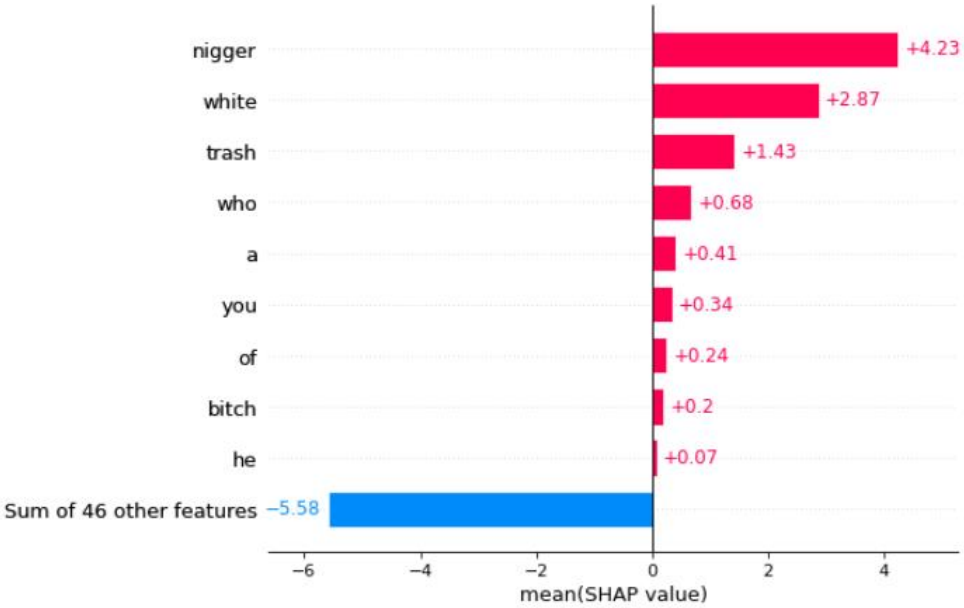


Figure 14 The top words impacting positively to the "Hate Speech" Class

B. Top Words Impacting the "Offensive Language" Class

The researcher uses the average/mean/ of each word in the sample tweets that have a high impact /negatively or positively/ on the Offensive Language class. In Figure 15 below, *'bitch'* and *'nigger'* are the words that have a high positive influence on the Offensive Language class. Whereas, *"yellow"* and *'heaven'* are the words that negatively impact the Offensive Language class.

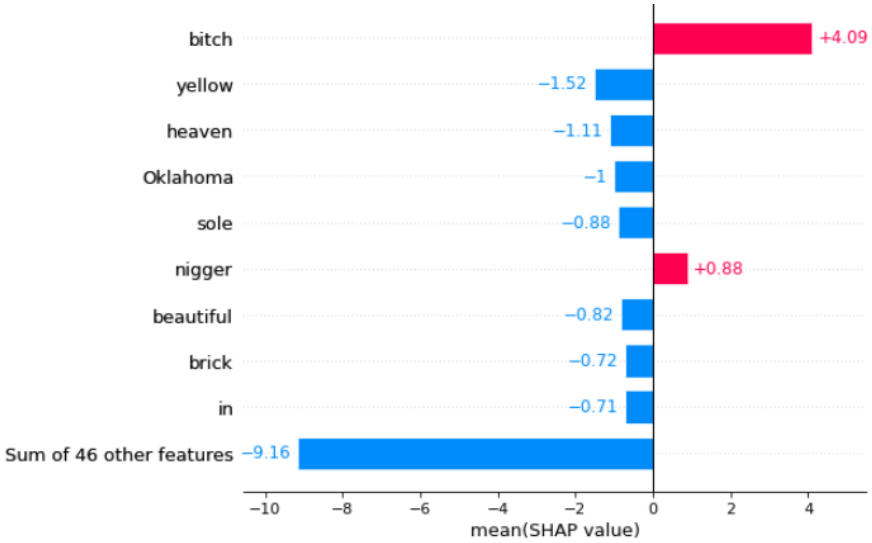


Figure 15 The top words impacting negatively [blue bars] and positively [Red bars] to the "Offensive Language" Class

C. Top Words Impacting the "Neither" Class

The researcher uses the average of each word in the sample tweets that have a high impact /negatively and positively/ on the "Neither" class. In Figure 16 below, *'bitch'* and *'nigger'* are the words that

have a high negative influence on the "Neither" class. Whereas, *"yellow"* and *'heaven'* are the words that are positively impacting the "Neither" class.

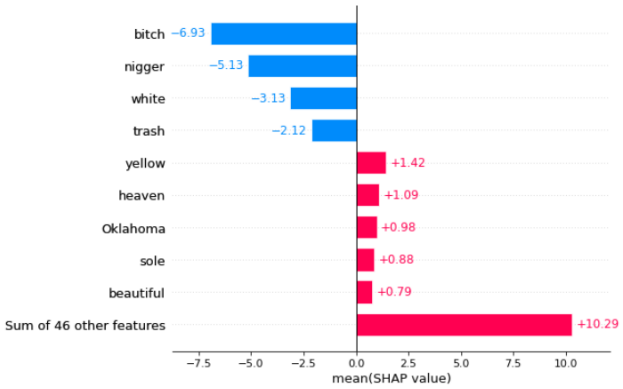


Figure 16 The top words impacting negatively [blue bars] and positively [Red bars] to the "Neither" Class

Therefore, using the bar charts we can easily identify and understand top words that are either negatively or positively impacting a specific class. Additionally, bar charts are easy to understand and interpret.

4.6. Reflections on the Research Questions

***RQ1**: How to build a hate speech classification model for Twitter datasets?*

The first task in the data science research project is reviewing related works, to identify the best choices of different research and development pipelines, methods, and choosing suitable data. The researcher reviewed different kinds of literature, in the NLP hate speech classification problem domain so that BERT is the most effective transformer-based deep neural architecture which handles contextual meaning of a word in different contexts and can easily be fine-tuned with small custom data sets using the principles of transfer learning. The BERT models are chosen for the development of the hate speech and offensive language classification problem. From the varieties of BERT models the RoBERTabase, BERTbase, and BERTweetbase were used. Although the BERT model was good, the models that come after BERT like RoBERTa(Liu et al. 2019) and BERTweet(Nguyen, Vu, and Tuan Nguyen 2020) outperforms in comparison BERTbase. The researcher have chosen the three BERT models to build, test and finally, explain(Danilevsky et al. 2020) for hate speech and offensive language classification model. The RoBERTa model was chosen according to the performance and pretraining method as described in (Liu et al. 2019).
Finally, the hate speech and offensive language classification models were developed and tested for the three BERT models (see the Experimental Setup Chapter 4). BERT, RoBERTa and BERTweet pretrained models were fine-tuned to classify the tweets as hate speech, offensive language, and neither categories and were proven in the experimental setup process.

RQ2: *Comparative Analysis: Which of the BERT Variants /BERT, RoBERTa, BERTweet/ are effective in Hate Speech classification?*
Comparing BERT models is difficult. The three BERT models are transformer(Vaswani et al. 2017) based NLP architectures, which combine sets of encoders with the self-attention mechanism. They have slight differences (Liu et al. 2019) in the pretraining procedure, data and training batch sizes, and masking methods that they use.
In fine-tuning with small data sets of section 4.4.1, the researcher has chosen 1000 tweets fine tune the three models, their evaluation result was promising. The evaluation results of the three models with

1000 training samples showed that BERTweet model can easily be fine-tuned and can be achieved a high-performance score even with small hundreds of twitter datasets. The other BERT and RoBERTa models can achieve a similar performance score to that of BERTweet after fine-tuning with the whole data sets (around 20k tweet samples used). This proves that fine-tuning (transfer learning) is the most convenient method of adapting knowledge of pre-trained models to perform a specific task (i.e. text classification).

Comparatively, the BERTweet model is effective in the classification of hate speech and offensive language scenarios. Especially, if the Twitter data set is too small, the BERTweet is the preferred one. The BERTweet is taking advantage of the unique characteristics of Twitter datasets from traditional texts like books and news, which reflect in the fine tuning of the models with small datasets as reported in section 4.4.1. Tweets are different from other texts like books, and wiki due to the length of Tweets, use informal grammar, irregular vocabulary, abbreviations, typographical errors, and hashtags. The BERTweet tokenizer uses an emoji package to translate the emotion icons into text strings. These unique characteristics of Twitter data are adapted in the BERTweet pretraining process and thus easily adapted and fine-tuned with small data sets.

RQ3: *How the classification results are explained to the users?*

The explanation tool used is SHAP. The explanations are local. The classification results input to output mappings is explained in detail in section 4.5 of this chapter. To explain the input (tweets) to output (classes) model prediction relationships and mappings, the researcher uses SHAP. With the SHAP explanation tool, bar charts were used to show top impacting (contributing) words in the prediction results, saliency plots were used to show the detailed contribution score of each token to each class, force plots were used to show the contributions of each of the words in each tweet with their contribution on each class, and heatmaps to highlight the words based on their impact to each class.

The explanations are easy to understand by the end-users. The prediction results of seven selected tweet samples were used for output visualization and explainability. Heat maps, saliency plots, force plots and bar charts were selected based on their intuitiveness and comprehensiveness in explaining texts. They are easy to interpret and understand by any user.

CHAPTER

5 - DISCUSSIONS

The best BERT variant for this particular research project is BERTweet. The BERTweet outperforms the other models and achieves a good result. To pre-train the BERT models hundreds of thousands of training steps were executed on the large volume of data for days using high processing GPUs. It is computationally expensive to train BERT transformers rather than fine-tuning pre-trained models will be simplified by using transfer-learning. Transfer learning is the process of storing previous experiences/knowledge/ which will be used to solve/predict/ other related problems. Transfer learning can be applied in fields like computer vision and natural language processing.

In this research project, we are using different variants of BERT models to classify tweets to a particular class /hate, offensive, or neither/.

RQ1*: How to build a hate speech classification model for the Twitter datasets?*

RQ2*: Comparative Analysis: Which of the BERT Variants /BERT, RoBERTa, BERTweet/ are effective in Hate Speech classification?*

RQ1 and RQ2:

The hate speech classification model building was explained in detail in Chapter 4 –Results (***RQ1)***. BERT (Devlin et al. 2019) models were chosen as the development method, due to the ease of fine-tuning the pre-trained models on large data sets, taking long sequences in parallel, and handling of contextual representation of words.

The BERTweet model achieves a high-performance score at the early stage of the training and fine-tuning phase with small datasets. This signifies that Twitter data have different characteristics from the traditional texts(like books) and uses informal grammar and symbols like emotion symbols. BERTweet uses 850M twitter datasets for pretraining. As a result, the BERTweet is effective for small training datasets in this particular scenario. Whereas the BERT and RoBERTa were pretrained in the common English texts taken from Wikipedia and BookCorpus which is different from the Twitter texts. So, the pre-trained knowledge of BERT and RoBERTa is slightly different and far from that of BERTweet due to the nature of the pretraining datasets that they use.

Generally, BERT Classification models are effective and transfer learning is the most important method to adapt the characteristics and features of an NLP Model. The BERT variants can be used for different problem scenarios in diverse contexts with small training datasets.

RQ3*: How the classification results are explained to the users?*

The words that are found in the text were highlighted with Saliency plots and heatmaps. Additionally, the most impacting words can easily be visualized with bar charts. The explanation visuals are discussed in detail in Chapter-4 Results. The main goal of explanation is to ease the interpretability of model outputs and introduce transparency and trust to the end-users. This research question is answered by providing intuitive visuals using bar charts, saliency plots, and heatmaps. The results of visuals reflect the ground truth and any user can simply understand the input (tweets) to output (classes) relationships and mapping in the model prediction. From the visuals, any person can understand the impact of each word on each prediction class, and identify top impacting words in different classes

either negatively or positively. SHAP is easy to use and can provide an overall overview of the prediction results such as visualizing impacting words to each of the classes. The limitation of SHAP is that computing shape values is slow. The Explanations are local.

The main contributions of this research project are:

- Explainability using Saliency plots, heatmaps, and bar charts to predict results, so as users can easily understand prediction results.
- Selecting a better model on hate speech classification
- The Fine-tuned models can be shared to[1] for further researches and developments on hate speech problem domain.

5.1. Challenges, Limitations, and Delimitations

The challenges faced during the implementation of the research project are:

- Lack of computational resources to test and run the project in PCs.
- Computing SHAP[2] values using the SHAP is slow
- The BERT and RoBERTa models consume a huge amount of memory than that of BERTweet.

The limitations of this research project are:

- These models will not run ordinary PCs that don't own high GPU.

The delimitations of this research project are:

- From the XAI perspective, the internal architecture of the model is not explained, the prediction results of the model were explained using saliency plots, bar charts, and heat maps.
- BERTweet has its normalization functionality if the data is not preprocessed (raw), which is not used in this research project in which future researchers can use.
- The hyper-parameters can be further tuned probably to get better results, the researcher hasn't tried exhaustively all the possible options.
- The Prediction results of RoBERTa and BERT were not explained.

Generally, the variants of BERT models can be fine-tuned easily to achieve promising performance results with small data by tuning the models' hyper-parameters. The explanation visuals provide ease of interpretability and trust prediction results of black-box models to users. XAI is a broad area that requires further research and development on the internal model operations and prediction results concerning the input of machine learning models.

[1]BERT Models can shared https://huggingface.co/models

[2] SHAP values interpret the impact of having a word for a given sentence in comparison to the prediction results.

CHAPTER

6- CONCLUSIONS

Transfer learning (Fine tuning) is a process of training transformer- models on new data by initializing the pre-trained model's weights, adapting for a new task somewhat related to the data. Lack of computational resources(GPU) was the most challenging part of this research project. Google's collab provides GPU and CPU resources for developers which solve the problem, and models still demand more.

Based on the nature of Twitter data sets, texts don't follow conventional English grammar. The tweets are short, uses irregular grammar, introduces new symbols and abbreviations. This can be challenging to handle the Twitter data for the models that are trained with conventional English language grammar texts. In the fine-tuning process, the BERTweet model achieves its maximum performance in the early stages of the training phases. In the case of BERT and RoBERTa a considerable amount of time and computational resource is required to achieve the performance score of BERTweet. This implies that the nature of data affects the performance of the model that is going to be fine-tuned. Accordingly, the BERTweet is better than the other two models.

BERTbase, RoBERTabase, and BERTweetbase pre-trained models were fine-tuned and evaluated by selecting suitable hyperparameters for each of the models.

Generally, the three BERT models achieved a good and similar performance result after finetuning the models with the large(whole) hate speech and offensive language datasets. For performance measurement of the models, accuracy and F1-score were selected.

From the perspective of XAI, the BERTweet model's predictions were explained using saliency plots, heat maps, and bar charts by highlighting words of different impacts in the prediction. The explanation methods are intuitive and comprehensive which can easily understand by any non-technical user. The explanation tool used is SHAP. The explanations are local. SHAP is easy to use and can provide an overall overview of the prediction results such as visualizing impacting words to each of the classes. The limitation of shap is that computing shape values is slow.

Generally, the research project will have practical importance in the area of Natural Language Processing particularly in the problem domain of hate speech recognition and mitigation.

REFERENCES

[1] A. Gaydhani, V. Doma, S. Kendre, and L. Bhagwat, "Detecting hate speech and offensive language on twitter using machine learning: An N-gram and TFIDF based approach," *arXiv*, 2018.

[2] "Hatebase." https://hatebase.org/ (accessed Sep. 15, 2021).

[3] A. Vaswani *et al.*, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 5999–6009, 2017.

[4] A. Barredo Arrieta *et al.*, "Explainable Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, no. October 2019, pp. 82–115, 2020, doi: 10.1016/j.inffus.2019.12.012.

[5] M. Danilevsky, K. Qian, R. Aharonov, Y. Katsis, B. Kawas, and P. Sen, "A survey of the state of explainable AI for natural language processing," *arXiv*, no. Section 5, 2020.

[6] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, "Explainable AI: A Brief Survey on History, Research Areas, Approaches and Challenges," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11839 LNAI, no. January 2020, pp. 563–574, 2019, doi: 10.1007/978-3-030-32236-6_51.

[7] A. M. P. B. Bras¸oveanu and R. Azvan Andonie, "Visualizing Transformers for NLP: A Brief Survey; Visualizing Transformers for NLP: A Brief Survey," *2020 24th Int. Conf. Inf. Vis.*, 2020, doi: 10.1109/IV51561.2020.00051.

[8] R. Roscher, B. Bohn, M. F. Duarte, and J. Garcke, "Explainable Machine Learning for Scientific Insights and Discoveries," *IEEE Access*, vol. 8, pp. 42200–42216, 2020, doi: 10.1109/ACCESS.2020.2976199.

[9] N. Safi Samghabadi, P. Patwa, S. PYKL, P. Mukherjee, A. Das, and T. Solorio, "Aggression and Misogyny Detection using {BERT}: A Multi-Task Approach," *Proc. Second Work. Trolling, Aggress. Cyberbullying*, no. May, pp. 126–131, 2020, [Online]. Available: https://www.aclweb.org/anthology/2020.trac-1.20.

[10] A. R. Isnain, A. Sihabuddin, and Y. Suyanto, "Bidirectional Long Short Term Memory Method and Word2vec Extraction Approach for Hate Speech Detection," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 14, no. 2, p. 169, 2020, doi: 10.22146/ijccs.51743.

[11] D. Li and J. Qian, "Text sentiment analysis based on long short-term memory," *2016 1st IEEE Int. Conf. Comput. Commun. Internet, ICCCI 2016*, pp. 471–475, 2016, doi: 10.1109/CCI.2016.7778967.

[12] A. Hassan and A. Mahmood, "Efficient deep learning model for text classification based on recurrent and convolutional layers," *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2017-Decem, pp. 1108–1113, 2017, doi: 10.1109/ICMLA.2017.00009.

[13] T. Caselli, V. Basile, J. Mitrović, and M. Granitzer, "HateBERT: Retraining BERT for abusive language detection in English," *arXiv*, 2020.

[14] S. Dowlagar and R. Mamidi, "HASOCOne@FIRE-HASOC2020: Using BERT and Multilingual BERT models for Hate Speech Detection," pp. 0–7, 2021, [Online]. Available: http://arxiv.org/abs/2101.09007.

[15] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," no. 1, 2019, [Online]. Available: http://arxiv.org/abs/1907.11692.

[16] D. Q. Nguyen, T. Vu, and A. Tuan Nguyen, "BERTweet: A pre-trained language model for English Tweets," pp. 9–14, 2020, doi: 10.18653/v1/2020.emnlp-demos.2.

[17] Y. Xie, L. Le, Y. Zhou, and V. V. Raghavan, "Deep Learning for Natural Language Processing," *Handb. Stat.*, vol. 38, pp. 317–328, 2018, doi: 10.1016/bs.host.2018.05.001.

[18] M. L. Williams, P. Burnap, A. Javed, H. Liu, and S. Ozalp, “Hate in the Machine: Anti-Black and Anti-Muslim Social Media Posts as Predictors of Offline Racially and Religiously Aggravated Crime,” *Br. J. Criminol.*, vol. 60, no. 1, pp. 93–117, 2020, doi: 10.1093/bjc/azz049.

[19] C. Ezeibe, “Hate Speech and Election Violence in Nigeria,” *J. Asian Afr. Stud.*, vol. 56, no. 4, pp. 919–935, 2021, doi: 10.1177/0021909620951208.

[20] B. Perry, D. Akca, F. Karakus, and M. F. Bastug, “Planting hate speech to harvest hatred: how does political hate speech fuel hate crimes in Turkey?,” *Int. J. Crime, Justice Soc. Democr.*, vol. 9, no. 2, pp. 195–211, 2020, doi: 10.5204/IJCJSD.V9I4.1514.

[21] A. Razia Sulthana, A. K. Jaithunbi, and L. Sai Ramesh, “Sentiment analysis in twitter data using data analytic techniques for predictive modelling,” *J. Phys. Conf. Ser.*, vol. 1000, no. 1, 2018, doi: 10.1088/1742-6596/1000/1/012130.

[22] M. O. Ibrohim and I. Budi, “Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter,” pp. 46–57, 2019, doi: 10.18653/v1/w19-3506.

[23] B. Gambäck and U. K. Sikdar, “Using Convolutional Neural Networks to Classify Hate-Speech,” no. 7491, pp. 85–90, 2017, doi: 10.18653/v1/w17-3013.

[24] Z. Zhang and L. Luo, “Hate speech detection: A solved problem? The challenging case of long tail on Twitter,” *Semant. Web*, vol. 10, no. 5, pp. 925–945, 2019, doi: 10.3233/SW-180338.

[25] S. Kamble and A. Joshi, “Hate speech detection from code-mixed Hindi-english tweets using deep learning models,” *arXiv*, 2018.

[26] A. R. Hevner, S. T. March, J. Park, and S. Ram, “Design Science in Information Systems Research,” *MIS Q.*, vol. 28, no. 1, pp. 75–105, Apr. 2004, doi: 10.2307/25148625.

[27] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, “A Design Science Research Methodology for Information Systems Research,” *J. Manag. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, 2007, doi: 10.2753/MIS0742-1222240302.

[28] I. Davidson, Thomas and Warmsley, Dana and Macy, Michael and Weber, “Automated Hate Speech Detection and the Problem of Offensive Language,” 2020, Accessed: Aug. 31, 2021. [Online]. Available: https://data.world/thomasrdavidson/hate-speech-and-offensive-language.

[29] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, no. Mlm, pp. 4171–4186, 2019.

[30] M. Mozafari, R. Farahbakhsh, and N. Crespi, “A BERT-Based Transfer Learning Approach for Hate Speech Detection in Online Social Media,” *Stud. Comput. Intell.*, vol. 881 SCI, pp. 928–940, 2020, doi: 10.1007/978-3-030-36687-2_77.

[31] A. Nikolov and V. Radivchev, “Offensive Tweet Classification with BERT and Ensembles,” pp. 691–695, 2019, doi: 10.18653/v1/s19-2123.

[32] J. A. Leite, D. F. Silva, K. Bontcheva, and C. Scarton, “Toxic language detection in social media for Brazilian Portuguese: New dataset and multilingual analysis,” *arXiv*, 2020.

[33] “SHAP documentation.” https://shap.readthedocs.io/en/latest/index.html (accessed Sep. 15, 2021).

APPENDIX A: Hate and Offensive Language sample data sets

	count	hate_speech	offensive_language	neither	class	tweet
0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't complain about cleaning up your house. & as a man you should always take the trash out...
1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn bad for cuffin dat hoe in the 1st place!!
2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby4life: You ever fuck a bitch and she start to cry? You be confused as shit
10	3	0	3	0	1	" Keeks is a bitch she curves everyone " lol I walked into a conversation like this. Smh
11	3	0	3	0	1	" Murda Gang bitch its Gang Land "
204	3	2	1	0	0	"@NoChillPaz: "At least I'm not a nigger" http://t.co/RGJa7CfoiT" Lmfao
206	3	2	1	0	0	"@NotoriousBM95: @_WhitePonyJr_ Ariza is a snake and a coward" but at least he isn't a cripple like your hero Roach lmaoo

APPENDIX B: BERT Models implementation

```
#Importing the necessary libraries
import shap
import pandas as pd
import numpy as np
import re
from sklearn.model_selection import train_test_split
from transformers import BertTokenizer
import torch
from transformers import AutoTokenizer
from transformers import TrainingArguments
from transformers import Trainer
import datasets
from transformers import AutoModelForSequenceClassification
import scipy as sp
from sklearn.metrics import accuracy_score,  f1_score

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')


TwitterData=pd.read_csv("Hate Speech and Offensive Language.csv")
# Data Preprocessing
#Droping the Uncessary columuns
FinalTwitterData=TwitterData.drop(['Unnamed: 0','hate_speech','offensive_lan-
guage','neither','count'],axis=1)

#Extracting the list of Columns in the dataframe
cols = FinalTwitterData.columns.tolist()
#Exchange the order of the columns [label,text] -->[text, label]
cols =cols[-1:] + cols[:-1]
FinalTwitterData=FinalTwitterData[cols]
FinalTwitterData.columns=['text','label']
#Checking for Duplicates
#
newData= FinalTwitterData.drop_duplicates(keep='first')
#Removing newlines and html tags
import html
for i in range(len(newData['text'])):
  x=newData.replace('\n','')
  FinalTwitterData = html.unescape(x)
#Cleaning Links, Hash Characters and Usernames
```

```
FinalTwitterData.replace(r"(@[A-Za-z0-9_]+)|[^\w\s]|#|http\S+","",re-
gex=True, inplace = True)
#Reseting the index of the cleaned dataset to the defualt numerical indexes
FinalTwitterData.reset_index(inplace = True, drop=True)

# Splitting Training and Testing data sets
# The XTrain,X_Train are the corresponding index of the samples
XTrain,XTest,YTrain,YTest= train_test_split( FinalTwitterData.index.values,
                                             FinalTwitterData['label'],
                                             test_size=0.10,
                                          stratify=FinalTwitterData['label'] )
# Splitting Training and Validation datasets from the whole training datasets
X=FinalTwitterData.loc[XTrain]

X_Train,X_Valid,Y_Train,Y_Valid= train_test_split( X.index.values,
                              X['label'], test_size=0.10, stratify=X['label'] )


#Filtering the training, validation and testing datasets with their corre-
sponding index locations
TrainingData=FinalTwitterData.loc[X_Train]
ValidationData=FinalTwitterData.loc[X_Valid]
TestingData=FinalTwitterData.loc[XTest]

TrainingData.reset_index(inplace = True, drop=True)
ValidationData.reset_index(inplace = True, drop=True)
TestingData.reset_index(inplace = True, drop=True)


#Converting the datasets to DatasetDict
import pyarrow as pa
TrainArrowTable=pa.Table.from_pandas(TrainingData)
ValidArrowTable=pa.Table.from_pandas(ValidationData)
TestingArrowTable=pa.Table.from_pandas(TestingData)


TrainDataset= datasets.Dataset(TrainArrowTable)
ValidDataset= datasets.Dataset(ValidArrowTable)
TestingDataset= datasets.Dataset(TestingArrowTable)

FullDataSetDict=datasets.DatasetDict({"train":TrainDataset,"valid":ValidDatase
t,'test':TestingDataset})
```

BERTweet Tokenizer

```
BERTweet = AutoTokenizer.from_pretrained("vinai/bertweet-base")
def tokenizeTweets(TwitterData):
```

```
            return BERTweet(TwitterData['text'], padding='max_length', trunca-
            tion=True, max_length=128 )
TokenizedData=FullDataSetDict.map(tokenizeTweets,batched=True)


TokenizedData = TokenizedData.remove_columns(['text'])
```

The tokenized datasets prepared to the Model building

```
Final_train_dataset = TokenizedData ["train"]
Final_eval_dataset = TokenizedData ["valid"]
Final_test_dataset= TokenizedData ["test"]


#Defining the BERT Model and its configuration
BERTweetModel=AutoModelForSequenceClassification.from_pretrained(
    'vinai/bertweet-base',
    num_labels=3,
    return_dict = False
    )
label2id = {
    "Hate Speech": 0,
    "Offensive Language": 1,
    "Neither":2
  }
config = BERTweetModel.config

id2label = {y:x for x,y in label2id.items()}
config.label2id = label2id
config.id2label = id2label
config._num_labels = len(label2id)
```

Measurement metrics

```
def compute_metrics(p):
    pred, labels = p
    pred = np.argmax(pred, axis=1)

    accuracy = accuracy_score(y_true=labels, y_pred=pred)
    f1 = f1_score(y_true=labels, y_pred=pred,average='weighted')
    return {"accuracy": accuracy,   "f1": f1}
#Defining the trainer API and feed tuning the hyperparameters
training_args = TrainingArguments(
    output_dir="ModelOutput",
    evaluation_strategy="steps",
    learning_rate=5e-05,
    eval_steps=500,
```

```
    per_device_train_batch_size=32,
    per_device_eval_batch_size=32,
    num_train_epochs=3,
    seed=0,
    load_best_model_at_end=True
)


#Instantiate the Trainer
trainer = Trainer(
    model=BERTweetModel,
    args=training_args,
    train_dataset=Final_train_dataset,
    eval_dataset=Final_eval_dataset,
    compute_metrics=compute_metrics
        )
# fine-tune our model, we just need to call
trainer.train()

trainer.evaluate(Final_test_dataset,metric_key_prefix='BERTweet' )
```

For the BERT and RoBERTa the process is similar you can simply replace the tokenizers and pretrained model in the above implementation.

For BERT model use 'bert-base-uncased'and
For the RoBERTa model use "roberta-base"

Explanation Implementations

```
Test Samples
# 0 "My grandma used to call me a porch mon-
key all the time Come to think of it she did refer to a broken bot-
tle as a nigger knife"
# 0 "At least I am not a nigger"
# 0 California is full of white trash who moved from Oklahoma
# 1 " bitch plz whatever "
# 1 Stay beautiful you bitch
# 2 Just follow the yellow brick twitter.
# 2 "he is in sole heaven"

TestSample=["My grandma used to call me a porch mon-
key all the time Come to think of it she did refer to a broken bot-
tle as a nigger knife","At least I am not a nigger", 'Califor-
nia is full of white trash who moved from Oklahoma',"bitch plz whatever",
            'Stay beautiful you bitch', 'Just follow the yellow brick twit-
ter', "he is in sole heaven"]
classifier = transformers.pipeline('text-classification', model= BERTweet-
Model.cpu(), tokenizer=BERTweet, return_all_scores=True)
```

```
explainer = shap.Explainer(classifier,algorithm='partition') #
shap_values = explainer(TestSample)
## Visualize the Sample Tweets on the three classes using Saliency plot and
heat maps
shap.plots.text(shap_values)
## Visualizing the impact of words in each tweet on Single class
shap.plots.text(shap_values[:,:,'Hate Speech'])
shap.plots.text(shap_values[:,:,'Offensive Language'])
shap.plots.text(shap_values[:,:,'Neither'])
## The Top Words Impacting on each classes using mean
shap.plots.bar(shap_values[:,:,"Hate Speech"].mean(0))
shap.plots.bar(shap_values[:,:,"Hate Speech"].mean(0), order=shap.Explana-
tion.argsort)
shap.plots.bar(shap_values[:,:,"Hate Speech"].mean(0), order=shap.Explana-
tion.argsort.flip)
shap.plots.bar(shap_values[:,:,"Offensive Language"].mean(0))
shap.plots.bar(shap_values[:,:,"Neither"].mean(0))
shap.plots.bar(shap_values[:,:,"Neither"].mean(0), order=shap.Explana-
tion.argsort)
shap.plots.bar(shap_values[:,:,"Neither"].mean(0), order=shap.Explana-
tion.argsort.flip)
```