School of Electronic Engineering
and Computer Science

Final Report

Programme of study:
Computer Science

Project Title:
Determine the semantic meaning shifting in toxic word

Supervisor:
Matthew Purver

Student Name:
Yujie Chen

Final Year
Undergraduate Project 2023/24

Date: 24/04/2024

Queen Mary
University of London

Acknowledgements

I would like to thank my supervisor Matthew Purver who guided and give me useful advice on this project, this project would not able to be finished without his help. I would also thanks everyone who helped me in this project mentally and physically .

Abstract

The proliferation of offensive and aggressive speech on online social platforms has raised significant concerns due to its adverse psychological impact on users. This project endeavours to address this issue by developing a toxic language classification model. Utilizing contemporary methodologies to the fullest extent of my capabilities, I have processed and analyzed datasets from two distinct online sources. Through the application of word embedding techniques—namely word2vec, GloVe, and PPMI—I examined the distribution of words within the vector space of embeddings. The results of these embeddings were then used to track shifts in semantic meaning across various timeframes, offering insights into the evolution of language deemed toxic within digital communication.

Contents

Table of Tables

Table of Figure

Chapter 1: **Introduction**

1.1 Background

We are currently in an era where the internet profoundly influences how information is accessed and how individuals communicate, particularly through social media. While social media platforms have become integral to modern societal interactions, they also present challenges, notably the prevalence of toxic, offensive, hated, and harmful language. This environment not only hinders constructive communication but also raises significant concerns about the psychological and social impacts of such interactions. The study of aggressive language patterns on these platforms, therefore, becomes imperative. Utilizing advanced language classifiers, this research aims to dissect and understand the nature and dynamics of aggressive language in online communication, offering insights into its evolution and potential mitigation strategies.

1.2 Problem Statement

The classification of aggressive and toxic language represents a dynamic research domain within natural language processing (NLP), utilizing an array of analytical tools and models from machine learning, deep learning, and large-scale language models (Oraşan, 2018). Despite technological advances, the discipline confronts notable challenges. A principal concern is the evolving nature of language; the meanings of words are not static, which can undermine the efficacy of models that do not account for temporal variations (Jatowt & Duh, 2014). The classification is further complicated by the inherent subjectivity of aggression and toxicity in language, which is contingent on individual perceptions and situational context (Balayn et al., 2021). Notwithstanding these complexities, there is currently a scarcity of research dedicated to the diachronic study of aggressive language evolution. For the purposes of this report, the terms 'aggressive language' and 'toxic language' will be used interchangeably to denote language that is potentially harmful or offensive.

1.3 Aim

This report aims to investigate whether the semantic meanings of words have shifted over time.

1.4 Objectives

Within the confines of limited funding, time, and personal expertise, the objectives for this project are delineated as follows:

1. Automated Classification Approach: Considering the budgetary and temporal restrictions in amassing a substantial corpus of toxic language, an automated classification approach has been adopted. This strategy involves evaluating and comparing the efficacy of various methodologies—spanning machine learning, deep learning, and other potential techniques—to discern the most effective.
2. Model Training and Fine-Tuning: The model selected based on the initial objective will undergo training or fine-tuning to acquaint itself with the characteristics of toxic language, which is a prerequisite for subsequent analysis.
3. Data Collection: Employ the trained model from the second objective to gather an extensive and varied dataset of language classified as toxic.
4. Analytical Assessment: Utilize models such as Word2Vec, GloVe, and PPMI to analyze the collected corpus, focusing on measuring cosine similarity as the primary metric for evaluation.

Each objective is crafted to methodically address the project's challenges, leveraging the available resources to maximize the insights gained from the toxic language data.

1.5 Research Questions

1. What methodologies in machine learning and deep learning currently yield the highest accuracy in classifying toxic language?
2. How do different models compare in their ability to discern nuanced expressions of toxicity across various datasets?
3. how can the semantic changing of word be systematically quantified and analyzed?
4. Which word became more toxic or less toxic?
5. Which word embedding techniques—Word2Vec, GloVe, PPMI, or others—demonstrate superior performance in capturing the semantic evolution of language?

Chapter 2: **Literature Review**

Owing to the complexities inherent in these tasks, this section will provide a concise overview of some prevalent models and methodologies in natural language processing.

2.1 automatic toxic language detection

2.1.1 Automatic toxic language detection

In the task of automatic toxic language detection, both machine learning and deep learning have been used in this task. For machine learning approaches, methods like SVM(support vector machine), CBOW(counties Bag of words), Naïve Bayes, and Random Forest (Balayn *et al.*, 2021) are been normally used. In deep learning, CNN (convolution neural network) and RNN (revolution neural network) like LSTM have also been used for classification tasks (Molero *et al.*, 2023). These models are appreciated for their simplicity and interpretability. The deep learning approach based on transformer is more advanced, the state-of-the-art transformer model BERT represents a significant leap forward, often outperforming its predecessors in aggressive language detection tasks due to its deeper contextual understanding (Von der Mosel, Trautsch & Herbold, 2023). This evolution highlights a trend towards more sophisticated models capable of capturing nuanced language patterns.

For task is acquired to produce a higher accuracy model. Researchers prefer to use the transformer-based model Bert and the modified version of Bert. Using Bert not only gets the benefit of increasing accuracy but also can be trained and modified for special language processing tasks. For instance, Bert in a multilingual version(m-bert) can be a good performance in multi-language tasks(Pelicon et al., 2021). Furthermore hate-Bert, a Bert pre-trained social media text, shows more robustness when dealing with aggressive language tasks, especially in imbalanced datasets (Caselli et al., 2021).

For some experiment results from others' work see Appendix [1].

2.1.2 Transformers

The Transformer architecture represents a significant milestone in the field of natural language processing (NLP) within deep learning. It is distinguished by its streamlined network design, which departs from traditional sequence transduction models by eschewing recurrent layers. Instead, it employs a mechanism based entirely on multi-headed self-attention, which underpins its encoder-decoder structure (Vaswani et al., 2017)

2.1.2.1 Tokenizer and Word Embedding

In the domain of natural language processing, tokens are considered the fundamental units of text (Zhang et al., 2023). The tokenizer process is thus conceptualized as a mapping mechanism, whereby a word or a sub-word is assigned a unique and consistent numerical identifier.

Figure 1 An example of Tokenizer , OpenAi. (2024)

In the Transformer architecture, as well as in its variants and any model based on the Transformer framework, word embedding occurs within a hidden layer. This layer transforms an input token into a learnable vector with a dimension 'd'. The dimensionality of this vector, 'd', can vary depending on the specific model configuration and the requirements of the task at hand.

2.1.2.2 Self-Attention and Multi-head Attention

The attention mechanism within the Transformer architecture is conceptualized as a function that computes the relevance of different words in a sequence. This is achieved through the interaction of queries, keys, and values, yielding a set of attention weights for each word (Vaswani et al., 2017). The mathematical representation of this mechanism is given by:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad \textit{Equation 1}$$

Where Q represents the query matrix, K is the key matrix, V is the value matrix and d_k is the dimension of the key vector. For the scaling factor, using $\sqrt{d_k}$ instead of d_k which can avoid the risk of extremely large gradients during the backpropagation process, which can hamper the training stability and efficiency.

The multi-head attention mechanism consists of several self-attention blocks operating in parallel. The output from these blocks is then concatenated and subsequently projected through a linear layer.

2.1.2.3 Encoder and decoder

The encoder begins with a multi-head self-attention mechanism, which is then followed by a position-wise fully connected feed-forward network. A residual connection is employed between each layer, and this is subsequently followed by layer normalization. The Encode Layer can be described as equation 2:

$$Encoder(X) = Norm(Norm(MulitHead(x) + X) + FFN(Norm(MulitHead(x)X)) \qquad \textit{Equation 2}$$

Where the Norm is layer Normalization, the FFN is a fully connected feed-forward network.

2.1.3 BERT

BERT, which stands for Bidirectional Encoder Representations from Transformers, is a groundbreaking pre-training model that has significantly advanced the state of natural language processing (NLP). Bert was introduced by Google's research team in the article BERT, which employs a multi-layer bidirectional Transformer encoder and operates through a two-stage framework of pre-training and fine-tuning.

2.1.3.1 unidirectional and Bidirectional language model

For a given sequence of input data comprising M tokens $(t_0, t_1, t_2, \dots, t_n, \dots, t_m)$, a unidirectional language model predicts each token based on the preceding tokens in the sequence:

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^{N} p(t_k | t_1, t_2, \dots, t_{k-1}) \qquad \textit{Equation 3.1}$$

In contrast, a bidirectional model incorporates context from both directions and predicts any token t_n using all the other tokens in the sequence:

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^{N} p(t_k | t_1, t_2, \dots, t_{n-1}, \dots, t_m) \; where \; m \geq n. \qquad \textit{Equation 4.2}$$

For intricate tasks that encompass question answering, semantic role labeling, and sentiment analysis, models with bidirectional architectures consistently surpass the performance of their unidirectional counterparts. The ELMo model (Embeddings from Language Models), as elucidated by Peters et al. (2018), exemplifies this advantage, showcasing a marked improvement over traditional recurrent neural network (RNN) models, such as Long Short-Term Memory (LSTM) networks. This performance benefit is further amplified when the BERT model is employed, leveraging its robust bidirectional architecture to achieve even greater advancements in these tasks.

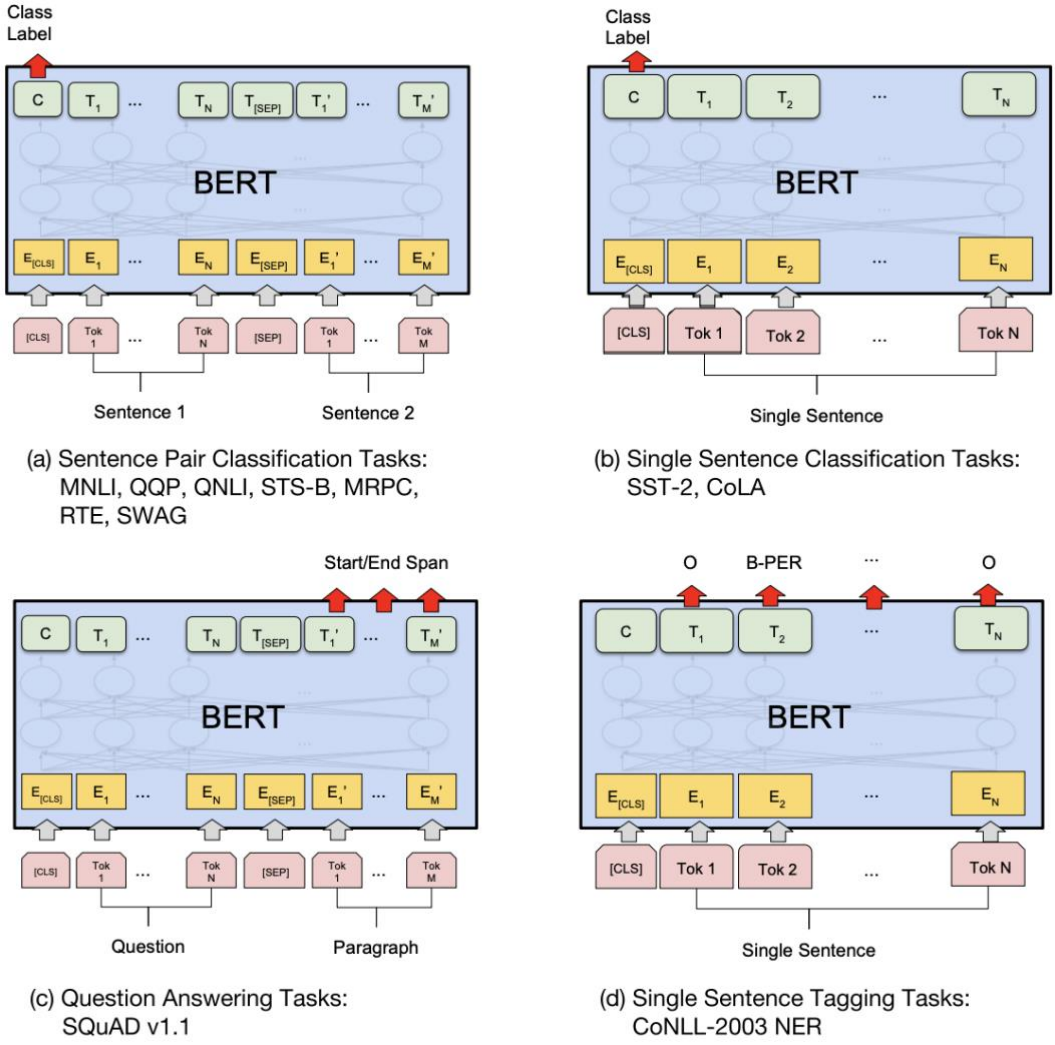
2.1.4 Model Architecture

The BERT model employs a multi-layered bidirectional Transformer architecture, with a modification in the self-attention heads. The $BERT_{BASE}$ configuration consists of 12 Transformer layers, with a hidden size of 768, and each layer containing 12 attention heads. In contrast, $BERT_{LARGE}$ model expands upon this, featuring 24 Transformer layers, increasing the hidden size to 1024, and utilizing 16 attention heads. Both $BERT_{BASE}$ and $BERT_{LARGE}$ have demonstrated significant advancements in performance when compared with prior models such as ELMo and LSTM-based architectures (Devlin et al., 2019).

2.1.5 Pre-training and fine-tuning

According to Devlin et al. (2019), BERT's training regimen consists of two primary stages: pre-training and fine-tuning. During pre-training, the model employs two strategies—Masked Language Model (MaskedLM) and Next Sentence Prediction—across the entirety of the training data. These methods enable BERT to develop a robust understanding of language structure and context. In the subsequent fine-tuning stage, BERT is specifically adapted to meet the nuances and requirements of individual tasks. This tailored approach ensures that the model performs optimally for a variety of specific applications. Additionally, the architecture of the BERT model can be modified as needed to better suit the specific characteristics of each task, further enhancing its adaptability and effectiveness.

Figure 2 Illustrations of Fine-tuning BERT on Different Tasks(Devlin et al., 2019)



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

2.1.6 Roberta

RoBERTa, an acronym for 'A Robustly Optimized BERT Pretraining Approach,' is an optimization of BERT, the pioneering language model. Developed by Liu et al. (2019), RoBERTa refines the pretraining process of BERT by incorporating several key modifications. The researchers found that extending the duration of the model's pretraining, utilizing larger batch sizes, and preprocessing an expanded corpus significantly enhance the model's performance. Moreover, RoBERTa introduces a dynamic masking pattern that alters the masked tokens each time the data is passed through the model, in contrast to BERT's static masking. These methodological improvements enable RoBERTa to outperform the original BERT model, setting new benchmarks on a variety of natural language processing tasks.

2.2 Word embedding and analysis method

2.2.1 Usage

The analysis of semantic distances between words can be effectively conducted using word embedding methodologies. The similarity between two words can then be quantitatively assessed by measuring the distance between their corresponding vectors in the embedded space.

2.2.2 Word embedding by machine learning approach

Conventional text representation techniques like the Bag of Words (BoW) and one-hot encoding models offer a simplistic approach by merely tracking the occurrence of words within a corpus. While these methods are computationally straightforward, they cannot inherently discern semantic nuances; each word is treated as an independent entity devoid of context. This binary representation strips words of their interrelations and the rich semantic tapestry they weave in language. As a result, BoW and one-hot encoding are considerably inadequate for complex natural language processing tasks that demand a deep understanding of word semantics and syntax. Such tasks include but are not limited to sentiment analysis, machine translation, and word sense disambiguation, where the grasp of polysemy and synonymy is critical. The advent of word embedding models marked a paradigm shift by embedding words in a continuous vector space, where semantic distances and relationships can be accurately captured, thereby dramatically enhancing the effectiveness of semantic analysis in computational linguistics.

2.2.2.1 Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA)

Both Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) are unsupervised machine-learning techniques that are employed to discover latent structures within text data.(Mikolov *et al.*, 2013). Latent Semantic Analysis, which leverages singular value decomposition (SVD) on term-document matrices, is primarily focused on uncovering semantic relationships between words by reducing the dimensionality of the text data (Mudadla, S., 2023). On the other hand, LDA is a generative probabilistic model that identifies topics by clustering documents, assuming that each document is a mixture of a small number of topics and that a topic is a distribution over words. While LSA can utilize methods such as TF-IDF (term frequency-inverse document frequency) for pre-processing, LDA models the probability distribution of words to infer the latent topic structures within documents.

2.2.3 Word2vec

The Word2Vec algorithm constitutes a pivotal development in word embedding techniques. Prior to Word2Vec, one-hot encoding was commonly utilized for the vector representation of words. Yet, this method was inherently limited as it did not encapsulate the semantic similarity between words—the cosine similarity between any two one-hot vectors is invariably zero, reflecting no semantic correlation (Zhang et al., 2023).

The Word2Vec algorithm encompasses two distinct components: the Continuous Bag of Words (CBOW) model and the Skip-gram model (Mikolov *et al.*, 2013).

2.2.3.1 The Skip-Gram model and the Continuous Bag of Word

Zhang et al., (2023) demonstrate that for a given sequence length *n* with tokens $(\boldsymbol{w_0, w_1, w_2, \dots, w_t})$, consider the centre word $\boldsymbol{w_c}$ and surrounding by context word $\boldsymbol{w_o}$. The Skip-Gram model predicts the centre word based on the context.

$$\boldsymbol{P(w|w_c) = \frac{exp(u_o^T v_c)}{\sum_{i \in V} exp(u_i^T v_c)}} \qquad \textit{Equation4.1}$$

$$P(w_c|w_o) = \frac{exp(u_c^T v_o)}{\sum_{i \in V} exp(u_i^T v_o)} \qquad \textit{Equation4.2}$$

where u_o is the vector representation of the context word w_o, and v_c is the vector representation of the centre word w_c. Equation [4.1] illustrates the mechanism by which the Skip-Gram model uses a target word to predict its context words, thus optimizing the model by maximizing this conditional probability across the training corpus. Conversely, equation [4.2] incorrectly references the CBOW model, which predicts the centre word based on context words, not as indicated here. The optimization goal for Skip-Gram is to adjust the word vectors v_c and u_i to enhance the likelihood of the true context words observed around a given centre word. The model is optimized when the predicted distribution of context words closely aligns with their actual distribution in the corpus

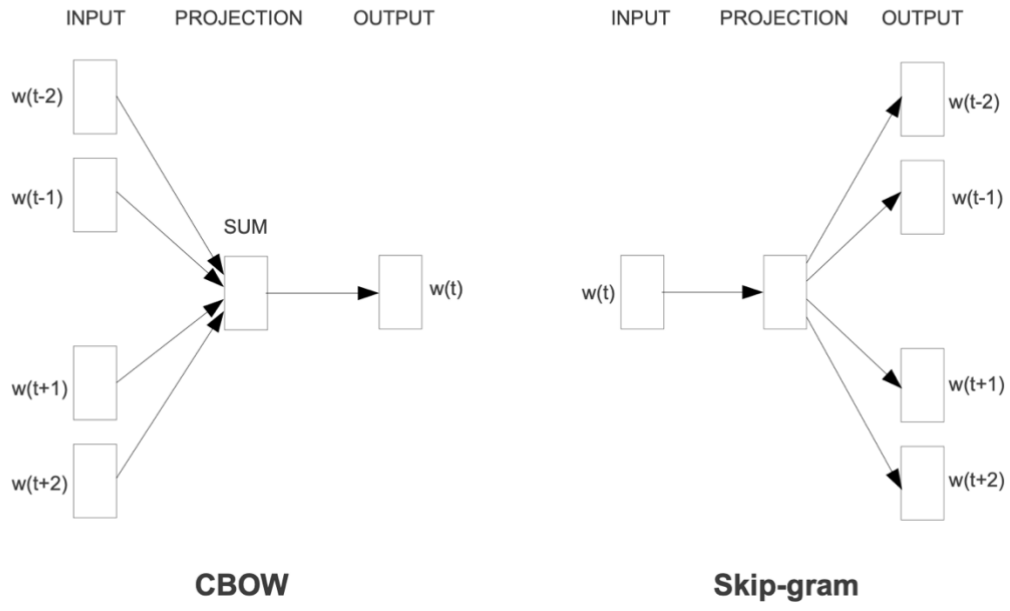


Figure 3 Figure of illustration of CBOW and Skip-gram, (Mikolov et al., 2013)

2.2.4 Capture the similarity of embedding word vector

For any two vectors A and B of the same dimensionality, their similarity can be quantified using the formula for cosine similarity:

$$\frac{A^T B}{|A||B|} \in [-1,1] \quad [5]$$

where $A^T B$ is the dot product of vectors A and B, and $\|A\|$ and $\|B\|$ are the magnitudes (or Euclidean norms) of vectors AA and BB, respectively. The result is a value between -1 and 1, where 1 indicates identical directionality, 0 indicates orthogonality, and -1 indicates opposite directionality. For any word embedding method, we should able to use the cosine similarity to capture the similarity level of 2 vectors.

2.2.4.1 Improvement of word2vec

The distributed representations of words generated by the Word2Vec algorithm, which leverages neural network models, have been demonstrated to outperform Latent Semantic Analysis (LSA) in preserving linear regularities among words. Additionally, Latent Dirichlet Allocation (LDA) tends to be computationally intensive, particularly with large datasets(Mikolov *et al.*, 2013).

2.2.5 GloVe

While the neural network-based model Word2Vec, as developed by Mikolov et al. (2013), excels at capturing semantic word relationships through vector representations, it exhibits limitations when dealing with polysemous words—words that have multiple meanings. Consider the word 'bank,' which, according to the Cambridge Dictionary (2024), can refer to both a financial institution and the land alongside a river. In a hypothetical corpus with a skewed distribution of sentences like 'Bank is a financial institute.' appearing four times, 'She deposited her paycheck at the bank yesterday.' also appearing four times, and 'He sat on the river bank and watched the sunset.' only occurring twice, the Word2Vec model might struggle to accurately represent the semantic nuances of 'bank' in the less frequent context of a river bank. Although such an extreme case may be unlikely in practical applications of Word2Vec, it illustrates a critical shortcoming: the model's efficiency can be compromised by its method of leveraging word-word co-occurrences within context windows. These co-occurrences often contain rich semantic information that Word2Vec may not fully exploit, especially in the case of words with multiple meanings.

The GloVe model (Global Vectors for Word Representation), developed by Pennington, Socher, and Manning (2014), demonstrates enhanced performance in capturing vector representations for polysemous words compared to Word2Vec. By leveraging co-occurrence probabilities within its matrix factorization approach, GloVe effectively distills semantic meanings from extensive textual data. Comparative evaluations reveal that GloVe exhibits superior results in various linguistic tasks, including word analogies, assessing word similarity, and recognizing named entities.

2.2.6 PPMI

Positive Pointwise Mutual Information (PMI) is a statistical measure that quantifies the degree of association between two variables, indicating the extent of information that is shared between them (Bouma, 2014).

$$\mathrm{pmi}(x;y) = \max\left(0, \log_2 \frac{p(x,y)}{p(x)p(y)}\right) \qquad Equation 5$$

where *x* and *y* represent two distinct events. For any two words, $word_A$ and $word_b$, a PMI score of 0 indicates statistical independence, suggesting that the occurrence of one is not influenced by the occurrence of the other.

2.3 Related word Toxic language classification

The classification of toxic language is an area of study that has garnered substantial attention from researchers seeking to mitigate the adverse effects of harmful online communication. A multitude of approaches have been explored:

- Machine Learning Techniques: Davidson et al. (2017) employed a range of machine learning techniques—including SVM, Random Forests, Logistic Regression, Naïve Bayes, and Linear SVMs—to distinguish hate speech and offensive language from normal discourse. McGillivray et al. (2022) also utilized a specialized SVM technique for identifying offensive language.

- Deep Learning Techniques: Advancements in deep learning have led to the adoption of BERT (Bidirectional Encoder Representations from Transformers) models for complex classification tasks. Camacho-Collados et al. (2022), Molero et al. (2023), Sreelakshmi et al. (2023), and {Citation} have all contributed BERT-based models to classify toxic language in English, demonstrating the model's effectiveness in this domain. Additionally, Pelicon et al. (2021) expanded BERT's applicability by employing it to classify offensive language in a multilingual context.

2.4 Related work on the word embedding method with diachronic word

The study of semantic change over time using word embeddings is a key area of interest in computational linguistics. Hamilton, Leskovec, and Jurafsky (2016) used Word2Vec and PPMI to track the semantic evolution of words across the 20th century, while Schlechtweg et al. (2019) explored similar shifts in earlier historical corpora with these methodologies. Li and Siew (2022) focused on PPMI to analyse English language changes among a consistent user group from the 1970s to 2000. Complementing this, Loureiro et al. (2022) examined sentiment changes in Twitter data over time, and Tredici and Fernandez (2018) combined Word2Vec and LSTM to investigate semantic shifts on Reddit. Balayn et al. (2021) gathered and classified toxic language across platforms to study its distribution and targeting, and Baele, Brace, and Ging (2023) aimed to discern the patterns of aggressive language on online platforms. Collectively, these studies illuminate the dynamics of language evolution and demonstrate the diverse applications of embedding technologies in capturing linguistic changes.

Chapter 3: **Classification Preparation**

3.1 Training

3.1.1 Training data

The definition of toxic language highlights the variance and subjectivity prevalent across different settings and cultural backgrounds (Balayn et al., 2021). Such ambiguity presents challenges in compiling well-structured and comprehensive training data. Moreover, apart from universally recognized aggressive words such as racist, there is a consensus on the specific format and phrasing that constitute toxic language (Balayn et al., 2021; Davidson et al., 2017). Given these complexities, a larger training dataset encompassing multiple sources appears to be a more effective solution for the generalisation ability of my toxic language classifier model. To enhance the breadth of potential training material, I have selected four distinct datasets for my classification model. The mixture of those four datasets may have a more concise, well-defined, complete set of aggressive language. Those four datasets are works from different periods and different resources.

The first training dataset utilized in this study is the 'Hate Speech and Offensive Language' dataset (Davidson et al., 2017) maintained until 2019. This dataset has been further extended by the research of Sreelakshmi et al. (2024) in India, which focuses on the behaviour of users who predominantly use languages other than English. Specifically, it amalgamates six sub-datasets encompassing user interactions in Kannada-English, Malayalam-English, and Tamil-English code-switched language environments. This comprehensive dataset serves as an extension to the collections from the HASOC datasets initially compiled by Chakravarthi et al. (2020). As a result, the datasets now can be considered as well-defined and structured datasets. Significantly, this dataset is curated by human evaluators and provides representation for a considerable segment of non-native English speakers, offering insights into the nuances of code-switched language use in digital communications.

The second training dataset is from Kaggle(2017). This dataset is the oldest, and closest to the dataset that I used to classify and analyse the semantic meaning changes. The third training dataset is from the work by Vidgen *et al.*, (2021). This dataset is collecting the user from Reddit with ~25,000 entries. All labelled data are labelled by trained annotators from different ethics and education backgrounds. The users of Reddit can certainly represent most of the mainstream of online platforms such as Twitter (now named as X) and YouTube. The fourth training dataset is from Hugging Face by Patel (2023). This dataset is still updating in 2023 and shows a balanced distribution in each class.

3.1.2 Data cleaning and annotations

After downloading all training datasets for my classification models, I performed data cleaning to the dataset. In a normal post on Reddit or Twitter, it may contain a mentioned symbol '@+username' or '&gt+username', 'RT+username'. Those mentioned contexts do not affect the offensive or hate speech's context but increase the input size of training data. I have removed them by Regular Expressions.

As I mentioned before, the definition of offensive and hate language has no common consensus. Each dataset has its definition and method to annotate and label toxic languages. Some of them may have a certain overlap of definitions. However, there is not a clear method to distinguish between those dataset's definitions. In a such complex circumstance, collapsing the training dataset to only two classes: toxic and non-toxic would be a better solution. All languages being labelled as offensive, slur or containing any format of toxic language mentioned and defined in the dataset need to be labelled as a toxic class. And vice versa.

After I collapse the classes of language in each dataset. The distribution of the toxic class and non-toxic class shows a certain level of imbalance expect the fourth dataset. The distributions of those datasets are 16.7% toxic and 83.3% non-toxic in the first dataset, 10.0% toxic and 90.0% non-toxic in the second dataset, 17.7% toxic and 82.3% non-toxic in the third dataset. To make more toxic language be recognised by my model, I randomly dropped 10%-20% of the non-toxic language in those imbalanced datasets.

3.2 Pre-requires and used library

3.2.1 CUDA and GPU

Given the considerable computational demands of advanced language models, hardware with CUDA (Compute Unified Device Architecture) capabilities is crucial. GPUs with significant memory and floating-point computation capacity are particularly critical during the training phase. In this endeavour, my personal RTX-3080 GPU, which contains 16 GB of RAM, was utilized, along with an A-100 GPU with 80 GB of RAM from the EECS department's server.

The intensive computational requirements posed by sophisticated language models underscore the importance of CUDA-enabled hardware. NVIDIA's CUDA is a powerful API that facilitates GPU-accelerated matrix operations, essential for swiftly processing vast amounts of data in neural networks, including diverse architectures such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and complex models like BERT and Transformers.

The value of CUDA lies in its ability to significantly shorten computation times by parallelizing tasks that CPUs would normally process sequentially. Throughout the training and application stages of neural network models, a multitude of matrix operations are carried out repeatedly and simultaneously. These operations, when performed by CPUs, can be computationally burdensome and inefficient due to their general-purpose design and sequential processing nature.

CUDA further augments GPUs' capabilities to execute numerous tasks in parallel across thousands of threads, making it exceptionally fit for the parallel processing demands of neural networks. Such parallelism is crucial for managing the extensive data and intricate computations integral to modern language models, encompassing areas from natural language processing to image recognition. With the increasing complexity of neural network models, the significance of CUDA in harnessing GPU capabilities to tackle these demanding computational tasks is ever-growing. CUDA not only expedites training but also improves the real-time processing essential for the practical deployment of AI models.

Hence, the adoption of CUDA technology is indispensable for research and development activities in artificial intelligence, especially in the realm of deep learning with sizable datasets. It empowers researchers and developers to extend the limits of

current capabilities, fostering the creation of more advanced and effective models that propel innovation in various fields.

3.2.2 PyTorch, Hugging Face

PyTorch is an open-source library that provides a suite of convenient APIs for neural network training. A key feature of PyTorch is its automatic differentiation engine, which facilitates the computation of gradients for each layer, thus enabling the seamless updating of model parameters via the chosen optimization algorithm. The absence of PyTorch would considerably complicate the training process, as it would require manual gradient computation and parameter updates, which are both error-prone and computationally intensive.

Hugging Face is a comprehensive open-source platform that hosts a multitude of transformer-based models. It allows users to access a wide array of pre-trained transformer models, which can be leveraged for various natural language processing tasks.

3.3 Training model

As previously outlined in Section 2, transformer-based models like BERT and RoBERTa are acclaimed for their superior accuracy in classification tasks, an attribute of their refined self-attention and multi-head mechanisms. To empirically substantiate this claim, a Logistic Regression model, facilitated by the Scikit-learn library, was employed for a preliminary analysis. This model was trained on the English language data from the TRAC dataset, known for its role in aggression identification and classified into three categories by Kumar et al. (2018). To prevent data overlap with my primary corpus, the TRAC dataset was excluded from the main model training.

The Logistic Regression model attained an accuracy of 0.5725, only slightly higher than the baseline accuracy of 0.4275 obtained by a naive classifier defaulting to the most frequent class prediction. This marginal improvement accentuates the shortcomings of traditional machine learning models, such as Logistic Regression, in processing the intricate nature of linguistic data. Additionally, considering the analysis of the diachronic evolution of word semantics, relying on a potentially biased dataset could yield distorted outcomes. Thus, a pivot to more advanced methodologies was deemed necessary.

Subsequently, a $BERT_{BASE}$ model was developed, drawing inspiration from Khang (2023). This BERT model leverages the 'tweet sentiment multilingual' dataset, created by Francesco, Luis, and Jose (2022), which contains data in eight languages: Arabic, English, French, German, Hindi, Italian, Portuguese, and Spanish. For the purpose of training an English-specific classifier, only the English subset of the data was utilized, with annotations categorized into negative (0), neutral (1), and positive (2) sentiments.

During the training phase, the AdamW optimizer was utilized, with an initial learning rate of 2e-5, paired with a linear schedule to manage the learning rate adjustments. This BERT classifier demonstrated reasonable gains over the Logistic Regression baseline, but an overall accuracy of 0.6793 signalled the potential for further improvement through the adoption of a pre-trained model.

```
Epoch 4/4
Validation Accuracy: 0.6793
              precision    recall  f1-score   support

           0       0.72      0.75      0.74       123
           1       0.52      0.57      0.55       114
           2       0.79      0.71      0.75       131

    accuracy                           0.68       368
   macro avg       0.68      0.68      0.68       368
weighted avg       0.69      0.68      0.68       368
```

Figure 4 An Image of the BERT training result

Accordingly, I commenced experiments with the RoBERTa model, drawing on the insights presented in Section 2. RoBERTa's sophisticated deep learning architecture, which has been validated across a range of linguistic tasks, offers a profound capability to discern subtle linguistic patterns. Particularly for classification tasks, the bidirectional nature of the model is poised to deliver enhanced performance due to its comprehensive contextual understanding. Informed by these considerations, I fine-tuned a pretrained RoBERTa model better to capture the specific linguistic features of my dataset. This approach is in line with the prevailing trend in computational linguistics, which involves adapting and refining state-of-the-art models to meet the demands of specialized research tasks.

Consequently, I selected the 'twitter Roberta base hate latest' model developed by Camacho-Collados et al. (2022) for my classification task. The primary rationale for choosing this model resides in the overlap between its training dataset, which comprises Twitter data, and the dataset targeted for my classification task. This Roberta model is pre-trained based on the tweet data.

	Emoji	Emotion	Hate	Irony	Offensive	Sentiment	Stance	Topic	NER
SVM	29.3	64.7	36.7	61.7	52.3	62.9	67.3	30.5	-
fastText	25.8	65.2	50.6	63.1	73.4	62.9	65.4	24.0	-
BLSTM	24.7	66.0	52.6	62.8	71.7	58.3	59.4	27.0	-
RoB-Base	30.9	76.1	46.6	59.7	79.5	71.3	68.0	50.1	58.0
RoB-Twitter	29.3	72.0	46.9	**65.4**	77.1	69.1	66.7	-	-
TweetEval	31.4	78.5	52.3	61.7	80.5	72.6	69.3	56.8	56.8
BERTweet	33.4	79.3	**56.4**	*82.1**	79.5	73.4	71.2	52.7	58.7
TweetNLP (TimeLMs-21)	**34.0**	**80.2**	55.1	64.5	**82.2**	**73.7**	**72.9**	**58.8**	**59.7**
Evaluation metric	M-F1	M-F1	M-F1	$F^{(i)}$	M-F1	M-Rec	AVG (F)	M-F1	M-F1

Figure 5 results of the nine TweetNLP-supported tasks (Camacho-Collados,et.al 2022)

The visual data presented in the previous section offers a compelling comparative analysis of various computational models applied to natural language processing tasks specific to Twitter data. The results unambiguously suggest that TweetNLP occupies a leading position in terms of performance. Its effectiveness transcends that of conventional machine learning approaches, exemplified by the Support Vector Machine, a staple in the field known for its robust classification capabilities. The superiority of TweetNLP is further exemplified when juxtaposed with advanced neural network-based architectures, such as the Bidirectional Long Short-Term Memory

(BLSTM), which is renowned for its ability to capture long-range dependencies in sequential data.

Notably, TweetNLP not only rivals but often surpasses the performance benchmarks set by BERTweet—a model specifically tailored to the idiosyncrasies of Twitter language through the BERT framework. The performance margin is particularly pronounced across an array of tasks that were part of the evaluation, underlining the efficacy of TweetNLP in understanding and processing the concise and often noisy linguistic environment of social media. This distinction underscores the potential of TweetNLP as a tool for researchers and practitioners seeking state-of-the-art solutions for parsing and interpreting Twitter feeds, with implications for enhanced sentiment analysis, more accurate semantic role labelling, and improved response generation in conversational systems.

Chapter 4: **Classification Experiment**

4.1 The fine-tuning Pipeline

To train a RoBERTa model, begin by reading the text data from files, ensuring that the data is cleanly pre-processed (convert data to correct data type) and structured appropriately for input. Once the data is loaded, tokenize the text using RoBERTa's tokenizer, which converts the raw text into a format suitable for the model, including adding special tokens and converting words into token IDs. Next, organize the tokenized data into batches using DataLoaders, which will be used to efficiently manage memory and speed up the training process. It is crucial to separate a portion of the data to form validation and test sets, allowing for the performance evaluation of the model during and after training.

Define the loss function for the training, typically cross-entropy loss, which is effective for classification tasks common with RoBERTa models. Set up the optimizer, such as Adam or AdamW, which will adjust the model's parameters (weights) based on the computed gradients during training to minimize the loss function. Proceed to train the model on the training data in mini-batches, continuously evaluating on the validation set after each epoch to monitor progress and adjust learning rates if necessary. This iterative process helps in fine-tuning the model parameters effectively. Thank to the highly pre-difined hugging face API. The most important training hyperparameters like learning rate and optimizer are been defined as the default value. This helps me who do not have sufficient knowledge in deep learning to train a model. The whole processing will do 3 epochs (A epoch is the model see the whole training dataset)

Finally, after training is complete, evaluate the model's performance on the unseen test dataset to assess its generalization capabilities. This evaluation will provide insights into how well the model has learned and its potential effectiveness in real-world scenarios, ensuring that the training process has successfully optimized the model's parameters for the tasks at hand.

4.2 Code

Here is a sample code of the training pipeline from the toxicBert.ipynb file.

```
training_args = TrainingArguments(
    output_dir="/content/drive/MyDrive/result",
    num_train_epochs=3,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=64,
    warmup_steps=500,
    weight_decay=0.01,
    logging_dir="/content/drive/MyDrive/logs",
    evaluation_strategy="steps",
    logging_steps=10,
)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_train_datasets,
```

```
    eval_dataset=tokenized_val_datasets
)


trainer.train()
```

This configuration demonstrates how the Hugging Face library facilitates efficient and effective training by abstracting much of the boilerplate code necessary for setting up a deep learning training loop.

4.3 Result

The pretrained RoBERTa model exhibited the highest accuracy among the models employed in my analysis, achieving an overall accuracy of 0.976. Additionally, it maintained this high level of performance across all three-evaluation metrics—recall, precision, and F1 score—each equal to or exceeding 0.976. These results indicate a comparable probability of false negatives and false positives. However, there is a potential concern regarding the model's generalization capabilities, which may be suboptimal and warrant further investigation to ensure robustness across diverse datasets.

Figure 6 The figure of test set accuracy

4.4 Bad generalization ability

After completing the fine-tuning process, I subjected the model to an unseen dataset to evaluate its generalization performance. For this purpose, I employed the 'TweetEval' dataset, curated by Basile et al. (2019), which categorizes hate speech into two classes. Despite the high accuracy achieved during training, the classifier's performance on this new dataset was suboptimal. The low precision scores particularly indicated a tendency of the model to erroneously label non-toxic content as toxic, resulting in an elevated rate of false positives. Although the issue remains unresolved within the scope of my current knowledge, the model as developed is the best iteration achievable with the available resources and methodologies.

```
F1 Score: 0.5113752122241088
Recall: 0.6014376996805112
Precision: 0.44477259303012406
Accuracy: 0.5154882154882154
```

Figure 7 Image of the test Accuracy

Chapter 5: **Semantic Shifting Experiment**

5.1 Collected data

Due to limitations in my capabilities, time, funding, and equipment, my data classification efforts were limited to specific segments of available corpora. These segments were derived from two publicly accessible repositories.

The first dataset consists of Reddit comments made available by the user Stuck_In_the_Matrix (2015), encapsulating platform activity from 2007 to 2015, chiefly in English. The periods selected for classification include: October to December 2007; the entire year of 2008; April 2009; January 2010; and February 2014. In retrospect, the labor-intensive nature of data classification was misjudged, resulting in an uneven distribution across the selected timeframes—an aspect that might impact the analysis. The distribution of the classified toxic corpus is summarized as follows: 24,163 instances in 2007; 201,332 in 2008; 2,736 in 2009; 62,653 in 2010; 125,450 in 2011; and 103,431 in 2014.

The second dataset is drawn from the Twitter Stream Grab (2012), which documents a diverse linguistic range of user-generated content on Twitter from 2012 through January 2024. For the purposes of this study, only English-language texts were classified. Classification was selectively conducted on the data from 2014 and, to a lesser extent, from 2022 and 2023. Due to the close temporal range and scant quantity of the data from 2023—limited to January, March, April, and May—and the partial data from January and February 2022, it was decided to consolidate these into one corpus, henceforth termed the '2023 corpus'. The classified toxic corpus encompasses 646,172 texts from 2014 and 240,563 texts from the aggregated 2023 corpus.

Upon reviewing the distribution of the classified both toxic corpuses, a discernible degree of imbalance is apparent across the datasets. Such disparity has the potential to introduce bias into the results, rendering them less reliable.

5.2 Pro-processing

After assembling my corpus, I conducted initial preprocessing, which entailed the application of lemmatization to the entire text corpus using SpaCy's tokenization tools. Lemmatization systematically transforms words to their base or dictionary forms through morphological analysis, which involves stripping inflectional endings using a detailed vocabulary. This process unifies the different inflected forms of a word to a single root form, effectively streamlining the dataset (Saravia, 2020). For example, the sentence 'She was dancing gracefully while he was running quickly' would be lemmatized to 'she be dance gracefully while he be runnig quickly.' This critical step in preprocessing not only broadens the usable vocabulary but also minimizes the distortion caused by multiple forms of the same word, such as ‘dance’, ‘dances’, and ‘danced’. This is because each term, despite its tense or plurality, should map to the same core meaning and corresponding embedding vector.

Additionally, I excised all URLs and user mentions from the corpus, typically denoted by '@..' for mentions and similar symbols for URLs. This purification step is vital to the integrity of the analysis, as it removes text that is irrelevant to the content, such as usernames, which generally lack semantic value in the context of toxic language analysis. Moreover, usernames often recur without contributing meaningful data, thus

their exclusion is imperative. URLs, in particular, are devoid of semantic significance and can distort the analysis if included. Their removal ensures that the focus remains solely on the linguistic elements pertinent to the task at hand

5.3 Shifting word

I utilized a Linear SVM (Support Vector Machine) classifier to analyze the coefficients of words within the 2023 Reddit corpus that I had previously collected and classified. Bakharia (2016) demonstrates that it is possible to access the coefficients corresponding to each feature, or word, from an SVM classifier. Additionally, Davidosn et al. (2017) note that machine learning classifiers such as the SVM are particularly sensitive to toxic words when tasked with distinguishing aggressive from non-aggressive language. These coefficients indicate the relative significance of each word in the model. Consequently, words with positive coefficients were identified as key candidates for further analysis to track their frequency across different time periods.

To establish 'unshifting' centre vectors for comparison with each candidate word vector, I initially selected four words universally recognized for their toxic and aggressive connotations: 'f*ggot', 'n*gger', 'a*s', and 'm*therf*cker' as four words with constant toxic words. These terms were designated as unshifted and inherently toxic for the purpose of this analysis. I calculated the mean of their word vectors to form a centre vector, which served as the basis for subsequent similarity comparisons. I then identified words most similar to these centre vectors, excluding the original four, across each methodology employed. This process revealed that the words similarly positioned to the centre vectors also exhibited toxic characteristics, supporting my initial hypothesis that words identified as similar to the designated toxic words also possess toxic connotations, thereby affirming the model’s accuracy.

For each time period analysed, I computed the centre word vector of the four words with constant toxic words. Subsequently, I measured the cosine similarity between this centre vector and each candidate word, which I refer to as the 'shifting' words, to assess their relative positioning in semantic space. Following this, I calculated the gradient of these similarity values across successive time periods to detect significant changes in word usage or meaning.

Lastly, I implemented a thresholding technique to exclude words that are either uncommon or exhibit minimal semantic shift across the different periods. The specifics of these threshold values and the rationale for their selection will be elaborated upon in subsequent sections of this study

5.4 Used Methodology

In my analysis of semantic shifts in word meanings, I utilized three distinct word-embedding methodologies: Word2Vec, PPMI (normalized Positive Pointwise Mutual Information), and GloVe. Each of these methods has been introduced and briefly described in Section 2.2. All three methodologies are accessible through open-source implementations. Word2Vec and normalized PPMI are both sourced from the Gensim library, which was developed by Rehurek in 2009. The original GloVe model, created by Pennington, Socher, and D. Manning in 2014, is implemented in C, which presented usability challenges for my purposes. Consequently, I opted for an alternative implementation of GloVe, specifically 'glovpy 0.2.0' by Karods (2023), which is available via PyPI.

All data in this report are rounded to 4 decimal places.

5.5 Word2vec Result

For each corpus derived from various sources and periods in the Reddit corpus, I constructed a distinct Word2Vec model. The parameters for all models were uniformly set with a window size of 10, a word vector size of 200, and a minimum word frequency of 30. This minimum frequency threshold effectively filters out noise, such as typographical errors and mixed usernames, which are not relevant to the semantic analysis. Additionally, I established a similarity range for the analysis: the uppermost similarity threshold was set at 0.5, representing the lowest boundary for non-shifting words relative to the centre word, and the lowermost threshold at -1. Furthermore, to identify words exhibiting significant semantic shifts, I applied a gradient threshold, the gradient should keep the sign as same at all times.

Table 1

Word	*2007*	*2008*	*2009*	*2010*	*2011*	*2014*
Laden	-0.1851	-0.0975	-0.2476	0.0226	0.0685	0.1119
Man	0.1851	0.1842	0.1672	0.0924	0.0369	0.0320

The table displays the most significant semantic shifts in the Reddit corpus, as determined by word2vec embedding.

Table 1 shows the similarity changing of the words ‘laden’ and ‘man’ in word2vec embedding in the Reddit corpus. The word ‘laden’ shows its increasingly toxic meaning and at the same time the word ‘man’ has become less toxic. The word ‘laden’ holds -0.1851 similarity with the mean of toxic words in 2007 and ends as 0.1119.

Table 2

Word	**2014**	**2023**	**Gradient**
Man	0.3206	-0.0895	-0.4101

Table 2 displays the most significant semantic shifts in the Twitter streaming corpus, as determined by word2vec embedding(the word ‘laden’ does not occur). The word ‘man’ shows the same direction of shifting.

5.6 Normalised PPMI Result

Due to the sparse distribution of words, the Pointwise Mutual Information (PMI) scores between most word pairs result in zero. To address this issue and enhance the utility of the data, I opted to use normalized Positive Pointwise Mutual Information (PPMI) suggested by Bouma (2009). This adjustment effectively mitigates the impact of sparsity by normalizing the scores, thereby providing more meaningful and usable insights into word relationships. For the PPMI result, I created a PPMI vector for each corpus. The parameters of all models are also set as window size 10, minimum count

of words 30. Then I apply the threshold as the uppermost similarity of 0.5 with the mean vector.

Table 3

Word	**2007**	**2008**	**2009**	**2010**	**2011**	**2014**
Start	0.2665	0.2932	0.3009	0.3115	0.3167	0.3291
grab	0.0953	0.1024	0.1181	0.1296	0.1450	0.1576
horrible	0.0883	0.1200	0.1222	0.1295	0.1407	0.1440
pant	0.0886	0.1462	0.1528	0.1627	0.1696	0.1880
bush	0.2954	0.2598	0.1888	0.1560	0.1301	0.1056
sheeple	0.1552	0.1363	0.0914	0.0630	0.0598	0.0416
fact	0.2486	0.2454	0.2438	0.2401	0.2336	0.2089

Table 3 illustrates the most significant semantic shifts within the Reddit corpus, as analysed using PPMI embeddings. It shows that the words 'start,' 'grab,' 'horrible,' 'park,' and 'pant' has exhibited a continuous decrease in toxic semantic associations, while the words 'bush,' 'sheeple,' and 'fact' have demonstrated a consistent increase in such associations

Table 4

Word	**2014**	**2023**	**Gradient**
start	0.1245	-0.0052	-0.1298
grab	0.0720	-0.0236	-0.0956
horrible	0.2735	0.0828	-0.1907
pant	-0.0258	0.0031	0.0290
bush	-0.2751	0.0150	0.2901
fact	0.0435	-0.1471	-0.1906

Table 4 illustrates the most significant semantic shifts within the Twitter streaming corpus, highlighting divergent trends compared to the Reddit corpus. Specifically, the words 'start,' 'garb,' and 'bush' exhibit semantic shifts in opposite directions when

compared to their trends in the Reddit data. Conversely, the words 'pant' and 'fact' demonstrate similar directions of semantic change across both corpora.

5.7 Glove Result

The parameter settings for the GloVe model were aligned with those of the Word2Vec model: a vector size of 200, a window size of 10, and a minimum count of 30. This minimum count threshold serves to filter out uncommon words, enhancing the reliability of the embeddings generated. The maximin similarity threshold is also 0.5 in this chapter.

Table 5

word	**2007**	**2008**	**2009**	**2010**	**2011**	**2014**
bush	-0.0261	0.2753	0.2018	0.1201	0.1206	-0.1980
throat	0.1681	0.2242	0.1946	0.1613	0.1335	0.1531

Table 6

word	**2014**	**2023**	**gradient**
bush	-0.2750	0.0151	0.2901
throat	-0.1461	0.0002	0.1462

Table 5 and 6 highlights notable semantic shifts: 'bush' and 'throat' decrease in toxicity in the Reddit corpus but increase on Twitter, illustrating differences across platforms

Table 7

	2006	**2007**	**2008**	**2009**	**2010**	**2014**
Word 'gt' in PPMI embedding	0.2939	0.3228	0.3378	0.3478	0.3210	0.3304
Word 'gt' in GloVe embedding	0.0513	0.3786	0.3841	0.2697	0.2813	0.2407
Word 'gt' in word2vec embedding	0.0306	0.0991	0.0651	0.0326	-0.0009	0.0279

Word 'cuz' in word2vec embedding	0.6279	0.4456	0.4745	0.5331	0.3404	0.4598
Word 'cuz' in GloVe embedding	0.1224	0.3095	-0.1149	0.2000	0.2384	0.4283
Word 'cuz' in PPMI embedding	0.0761	0.1133	0.1265	0.0992	0.1004	0.1529

Table 7 illustrates two noise words "cuz" and 'gt''s semantic meanings changing in the Reddit corpus

The all word in other embedding are in appendix B

Chapter 6: **Evaluation**

6.1 Observed result

In Section 5, we observe notable changes in the semantic meanings of specific words across different embedding methodologies.

The word 'sheeple' has shown a decrease in toxic semantic meaning across all word embedding methods within the Reddit corpus. The similarity with the mean of toxic words decreased from 0.155 to 0.042 in PPMI embedding, from 0.196 to 0.081 in GloVe embedding, and from 0.444 to 0.210 in Word2Vec embedding, between 2008 and 2014. Initially, in 2009, 'sheeple' was used in contexts suggesting gullibility, such as 'Me neither, I thought Reddit was just filled with sheeple. WAKE UP, DON'T ELECT A NI----9/11 WAS AN INSIDE JOB, WAKE UP SHEEPLE!!!!!!!!!!!!.' By 2014, the term evolved to describe manipulation by higher powers, evident in uses like, 'She's not a liar, just another brainwashed sheeple doing what the control system tells them to. They want us all secretly addicted to internet porn and lower testosterone so they can take over the world.'

Conversely, the word 'throat,' which initially appeared in less toxic contexts in 2006, such as in the argumentative statement, 'Fucking moron. Slit your throat already,' associated with political discourse, has seen an increase in toxic semantic meaning. By 2014, its usage had predominantly shifted to offensive sexual contexts, as evident in comments like 'Even those fucking shocker gross as fuck ones like with a lady who's missing her throat and makes noises like the bitch from the grudge? Fuck that stuff.' This change illustrates how 'throat' transitioned from a term used in political rhetoric to one with derogatory and sexually explicit connotations within the toxic language landscape on Reddit.

The analysis conducted using Word2Vec, normalized PPMI, and GloVe indicates that words associated with toxic language exhibit semantic shifts to varying degrees. These shifts are quantified using cosine similarity metrics against the mean vector of toxic words. Nonetheless, the direction and magnitude of these shifts display notable inconsistencies across different online platforms, and the results vary among the embedding techniques, suggesting that each method captures aspects of semantic change differently.

6.2 Noise word

The words 'cuz' and 'gt' serve as examples of noise words that are not typically removed by standard stop word filters. In the datasets analyzed, 'gt' appears approximately 100 times per corpus, while 'cuz' occurs around 1,600 times, making it more prevalent than most other words. Analysis of semantic meaning shifts reveals that Word2Vec generally performs better with less frequently occurring noise words compared to PPMI and GloVe. However, when dealing with noise words that frequently appear in the corpus, PPMI demonstrates more stable meanings, which is considered optimal for maintaining consistency in semantic interpretation. In contrast, Word2Vec shows less accuracy in these instances, indicating a variation in performance based on the frequency of noise words within the corpus.

Chapter 7: **Future work**

The results derived from the word embedding methodologies employed in this study have demonstrated a degree of unreliability and imprecision. Although the outcomes of these implementations are thoroughly documented, it is crucial to recognize that attempts to replicate this analysis could yield varying results. This variability has been consistently observed across multiple iterations of the analysis, highlighting the inherent instability associated with these embedding techniques. In response to these findings, I will conduct a thorough review and analyse potential improvements that could enhance the robustness and accuracy of this project.

7.1 Classification model

In Section 4.4, I addressed the limited generalization capability of the pre-trained RoBERTa model I developed. This limitation is evident as an underperformance when the model is applied to data distributions that diverge significantly from those seen during training. Despite utilizing a pre-trained RoBERTa model, which represents the best version achievable within my current capabilities, challenges in adapting to varied data contexts continue to persist. Extensive efforts were made to diversify the training set to enhance its representativeness and address this issue. However, given the complexity of the task and my own limitations, the model may still encounter difficulties in effectively managing such sophisticated challenges.

For future experiments, a crucial objective will be to develop a more accurate model. A promising approach involves upgrading from $\mathrm{RoBERTa}_{\mathrm{base}}$ to $\mathrm{RoBERTa}_{\mathrm{large}}$. The increased depth of RoBERTa_{large}. might lead to better performance in classification tasks. Furthermore, switching to a more state-of-the-art model, might be another considerable option.

Additionally, the subjective nature of defining toxic language suggests that a more robust approach would involve incorporating perspectives from a diverse range of demographic groups, including different ethnicities and genders. Should additional funding become available, such inclusivity would likely enhance the model's understanding of nuanced linguistic elements, thereby improving its accuracy and applicability across various social contexts.

7.2 Word Embedding Method and Corpus

Both Word2Vec and GloVe word embedding methods generally perform well in semantic analysis compare with the PPMI. GloVe operates on a matrix-based method, while Word2Vec utilizes a shallow neural network for its training process. Considering embeddings generated by larger language models could be beneficial due to their advanced contextual understanding capabilities. However, employing such models necessitates a corpus with richer contextual data.

Given the limitations of this experiment, including the scale and diversity of the current corpus, it may not be sufficient to effectively train large language models. Therefore, expanding the corpus to include a broader range of contexts would likely enhance the quality of the embeddings produced.

7.3 Needed knowledge

The expertise of linguists plays a pivotal role in the analysis of semantic shifts in language. Insufficient linguistic knowledge can lead experiments astray, resulting in misguided directions and the derivation of erroneous or irrelevant conclusions. Therefore, incorporating robust linguistic insights is essential to ensure the validity and utility of findings in studies of language evolution.

Chapter 8: **Conclusion**

In this study, I developed a relatively high-accuracy toxic word classifier using the RoBERTa model published by Camacho-Collados et al. (2022). This model was applied to corpora from two distinct datasets. Additionally, I employed three word embedding methodologies—Word2Vec, GloVe, and PPMI—to analyze and verify the semantic shifts of toxic words. The analysis aimed to understand how the meanings of these words have evolved over time within different linguistic contexts.

The code for this project is publicly available and can be accessed at https://github.com/abathu/Final-Year-Project.

References

Baele, S., Brace, L., Ging, D., (2023) A Diachronic Cross-Platforms Analysis of Violent Extremist Language in the Incel Online Ecosystem. Talyor & Francis. Available from: https://www.tandfonline.com/doi/full/10.1080/09546553.2022.2161373

Bakharia, A.(2016) 'Visualising Top Features in Linear SVM with Scikit Learn and Matplotlib'. Available from: https://aneesha.medium.com/visualising-top-features-in-linear-svm-with-scikit-learn-and-matplotlib-3454ab18a14d.

Balayn, A., Yang, J., Szlavik, Z. & Bozzn, A. (2021) 'Automatic Identification of Harmful, Aggressive, Abusive, and Offensive Language on the Web: A Survey of Technical Biases Informed by Psychology Literature', *ACM Transactions on Social Computing*, 4(3), pp. 1–56. Available from: https://doi.org/10.1145/3479158.

Basile,V., Bosco,C., Fersini,E., Nozza,D., Patti,V., Rangel,P., Francisco,M., Rosso,P. & Sanguinetti,M. (2019)' SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter' Available from: https://aclanthology.org/S19-2007/

Barbieri, F., Espinosa-Anke, L. & Camacho-Collados, J. (2022)MXLM-T: Multilingual Language Models in Twitter for Sentiment Analysis and Beyond. *European Language Resources Association*. Available from: https://huggingface.co/datasets/cardiffnlp/tweet_sentiment_multilingual

Bouma, G. (2014) 'Normalized (Pointwise) Mutual Information in Collocation Extraction'. Available from: https://svn.spraakdata.gu.se/repos/gerlof/pub/www/Docs/npmi-pfd.pdf

Camacho-Collados, J., Rezaee,K., Riahi,T., Ushio,A., Loureiro,D., Antypes,D., Boission,J., Espinosa-Anke,L., Liu,F., Martinez-Camara,E., Medina,G., Buhrmann,T., Neves,T. & Barbieri,F.(2022) 'TweetNLP: Cutting-Edge Natural Language Processing for Social Media', Association for Computational Linguistics. Available from: https://arxiv.org/pdf/2206.14774.pdf

Cambridge Dictionary. (2024) Available from: https://dictionary.cambridge.org

Caselli, T., Basile, V., Mitrovic, J. & Granitzer, M. (2021) 'HateBERT: Retraining BERT for Abusive Language Detection in English'. *Association for Computational Linguistics.* Available from: https://arxiv.org/abs/2010.12472

Chakravarthi, B.R., Kumaresan, P.K., Sakuntharajc, R., Madasamyd, A.K., Thavareesanc, S., Premjithe, B., Sreelakshmie, K., Navaneethakrishnanf, S.C., McCraea, J,P. & Mandlg, T. (2020) Overview of the HASOC-DravidianCodeMix Shared Task on Offensive Language Detection in Tamil and Malayalam. Available from : https://ceur-ws.org/Vol-3159/T3-1.pdf

Davidosn, T., Warmsely,D., Macy, M. & Weber, I. (2017) Automated Hate Speech Detection and the Problem of Offensive Language. *AAAI.* Available from: https://ojs.aaai.org/index.php/ICWSM/article/view/14955

Devlin, J., Chang, W., Lee, K. & Toutanova, K. (2019) 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding'. *arXiv*. Available from: http://arxiv.org/abs/1810.04805 (Accessed: 25 November 2023).

Gerlof, B., (2009) ' Normalized (Pointwise) Mutual Information in Collocation Extraction'. Available at: https://svn.spraakdata.gu.se/repos/gerlof/pub/www/Docs/npmi-pfd.pdf

Hamilton,W,L., Leskovec, J. & Jurafsky, D. (2016) 'Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change', in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany: Association for Computational Linguistics, pp. 1489–1501. Available at: https://doi.org/10.18653/v1/P16-1141.

Jatowt, A., & Duh, K.,(2014) ' A Framework for Analyzing Semantic Change of Words across Time'. IEEE. Available from :https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6970173

Kaggle (2017) Toxic comment classification chanllenge. Avaiable from: https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/overview

Kardos,M. (2023) 'glovpy 0.2.0'. pypi. Available from: https://pypi.org/project/glovpy/

Kumar, R., Reganti, A.N., Bhatia, A. & Maheshwari, T. (2018) Aggression-annotated Corpus of Hindi-English Code-mixed Data**.** *European Language Resources Association (ELRA).* Available from: https://sites.google.com/view/trac1/shared-task

Li, Y., Siew, C.S.Q., (2022) Diachronic semantic change in language is constrained by how people use and learn language. Springer. Available from: https://link.springer.com/article/10.3758/s13421-022-01331-0

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer , L. & Stoyanov, V., (2019) 'RoBERTa: A Robustly Optimized BERT Pretraining Approach'. *Arxiv*. Available from: https://arxiv.org/abs/1907.11692

Loureiro, D., Barbieri, F., Neves, L., Anke, L,E. & Camacho-Collados, J., (2022) ' TimeLMs: Diachronic Language Models from Twitter '.arXiv:2202.03829v2

McGillivray, B., Alahapperuma, M., Cook, J., Bonaventura, C.D., Meroño-Peñuela, A., Tyson, G. & Wilson, S.R. (2022) Leveraging time-dependent lexical features for offensive language detection. Association for Computational Linguistics. Available from: https://aclanthology.org/2022.evonlp-1.7.pdf

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013) 'Efficient Estimation of Word Representations in Vector Space'. arXiv. Available at: http://arxiv.org/abs/1301.3781 (Accessed: 16 November 2023).

Molero, J.M., Perez-Martin, J., Rodrigo,A. & Penas, A. (2023) 'Offensive Language Detection in Spanish Social Media: Testing From Bag-of-Words to Transformers Models', *IEEE Access*, 11, pp. 95639–95652. Available from: https://doi.org/10.1109/ACCESS.2023.3310244.

Mudadla, S.(2023) 'Difference between Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) in Natural Language Processing'. *Medium*. Available from: https://medium.com/@sujathamudadla1213/difference-between-lda-and-lsa-f7fefa6b4bfd

Nguyen, D.Q., Vu, T. and Tuan Nguyen, A. (2020) 'BERTweet: A pre-trained language model for English Tweets', in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online: Association for Computational Linguistics, pp. 9–14. Available at: https://doi.org/10.18653/v1/2020.emnlp-demos.2.

OpenAi. (2024) 'Tokenizer'. Available at : https://platform.openai.com/tokenizer

Orasan, C.,(2018) 'Aggressive language identification using word embeddings and sentiment features'. Available from: https://aclanthology.org/W18-4414.pdf

Patel, P.,(2023) badmatr11x/hate-offensive-speech. *HuggingFace* Available from: https://huggingface.co/datasets/badmatr11x/hate-offensive-speech

Pelicon, A., Shekhar, R., Skrlj,B., Purver, M. & Pollak, S. (2021) 'Investigating cross-lingual training for offensive language detection', *PeerJ Computer Science*, 7, p. e559. Available at: https://doi.org/10.7717/peerj-cs.559.

Pennington, J., Socher,R. & D.Maning,C., (2014) 'GloVe: Global Vectors for Word Representaition' Available From: https://nlp.stanford.edu/projects/glove/

Peter, M.E, Neumann, M., Iyyer, M. & Gardener,M. (2018) 'Deep contextualized word representations'. *Arxiv.* Available from: https://arxiv.org/abs/1802.05365

Pham, K., (2023) 'Text Classification with BERT'. Medium, Available at: https://medium.com/@khang.pham.exxact/text-classification-with-bert-7afaacc5e49b

Rehurek, R., (2014) 'Topic modeling for humans'. Gensim. Available at https://radimrehurek.com/gensim/index.html

Schlechtweg, D., Hatty, A., Tredici, M.D. & Walde, S.S.I. (2019) A Wind of Change: Detecting and Evaluating Lexical Semantic Change across Times and Domains. *Arxiv*. Available from: https://arxiv.org/abs/1906.02979v1

Saravis, E.,(2023) Fundamentals of NLP - Chapter 1 - Tokenization, Lemmatization, Stemming, and Sentence Segmentation. Available from: https://dair.ai/notebooks/nlp/2020/03/19/nlp_basics_tokenization_segmentation.html

Stuck_In_the_Matrix (2015) 'I have every publicly available Reddit comment for research. ~ 1.7 billion comments @ 250 GB compressed. Any interest in this?' Available from:https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/?rdt=59291

Sreelakshmi, K., Premjith, B., Chakravarathi, B,R. & Soman, K,P. (2024) Detection of Hate Speech and Offensive Language CodeMix Text in Dravidian Languages Using Cost-Sensitive Learning Approach. *IEEE Access*. Available from: https://ieeexplore.ieee.org/abstract/document/10419328

Tredici, M,D., Fernandez, R.,(2018) 'Semantic Variation in Online Communities of Practice'. arXiv:1806.05847v1

Twitter Stream Grab (2012) ' Archive Team Twitter Grabs'. Internet Archive. Available From: https://archive.org/details/twitterstream?tab=collection

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A, N., Kaiser, L. & Polosukhin,I., (2017) 'Attention Is All You Need'. arXiv:1706.03762

Vidgen, B., Nguyen, D., Margetts, H., Rossini, P., & Tromble, R.(2021) 'Introducing CAD: the Contextual Abuse Dataset', in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, pp. 2289–2303. Available from: https://doi.org/10.18653/v1/2021.naacl-main.182.

Von der Mosel, J, Trautsch, A. & Herbold, S., (2023) ' On the Validity of Pre-Trained Transformers for Natural Language Processing in the Software Engineering Domain'. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, VOL. 49, NO. 4. Available from: https://ieeexplore.ieee.org/document/9785808

Zhang, A., Lipton, Z,C., Li, M. & Smola, A,J., (2023) 'Dive into Deep Learning', *Cambridge University Press*, Available From: https://D2L.ai

Appendix A

The comparison with machine learning, and deep leering transformer-based model

Appendix 1

Figure 8 The Results in machine learning, deep learning, and fine-tuned BERT in the task classification (Molero et al., 2023)

Model	Precision	Recall	F1
RoBERTuito	0.9011	0.9011	0.9011
BETO	0.8855	0.8855	0.8855
Best at MeOffendES	0.8815	0.8815	0.8815
CNN	0.8403	0.8403	0.8403
Bi-LSTM	0.8374	0.8374	0.8374
SGD oversampling	0.8385	0.8325	0.8346
baseline MeOffendES	0.8285	0.8285	0.8285
Bi-LSTM 2 layers	0.8227	0.8227	0.8227
CNN 3 layers	0.8235	0.8235	0.8235
SGD	0.8356	0.8216	0.8233
SVM	0.8399	0.8060	0.8203
SVM oversampling	0.8330	0.8028	0.8158
GradientBoosting	0.8135	0.8325	0.8147
RandomForest	0.8106	0.8304	0.8058
AdaBoost	0.7873	0.7149	0.7401

Appendix B

Table of words in different word embedding in Reddit corpus

	2007	2008	2009	2010	2011	2014
bush	-0.0261	0.2753	0.2018	0.1201	0.1206	-0.1980
throat	0.1681	0.2242	0.1946	0.1613	0.1335	0.1531
start	-0.1292	0.2816	0.1786	0.2226	0.3016	0.2937
grab	0.2786	0.1896	-0.0685	0.0605	0.2257	0.2421
horrible	0.0751	0.1875	0.2255	0.2228	0.2055	0.2837
pant	0.4115	0.2930	0.2220	0.2285	0.2440	0.2941
sheeple	0.2147	0.2300	0.0736	0.0266	0.0007	0.0871
fact	0.0248	0.2254	0.1600	0.1805	0.1375	0.1708
laden	-0.1071	-0.0201	-0.2276	-0.0982	-0.1791	-0.3359
man	0.1461	0.3019	0.3086	0.2093	0.2228	0.2276

Table 8 the similarity of All words with the mean toxic vector in the Twitter corpus by GloVe embedding

	2007	2008	2009	2010	2011	2014
bush	0.0880	0.0336	0.0362	0.1793	0.1556	0.2011
throat	0.5997	0.3024	0.6462	0.4023	0.3165	0.3280
start	0.0382	-0.0316	0.1855	0.0109	-0.0080	0.0097
grab	0.4867	0.1823	0.4822	0.0103	0.0961	0.1140
horrible	0.1867	-0.1833	0.0770	-0.0615	-0.1995	-0.2441
pant	0.7040	0.3639	0.5384	0.2767	0.2417	0.1997
sheeple	0.4145	0.2493	0.4928	0.0981	0.2022	0.2101
fact	-0.0278	0.0305	-0.0959	-0.0034	-0.0315	0.0073
laden	-0.1851	-0.0975	-0.0248	0.0226	0.0685	0.1120
man	0.1851	0.1843	0.1673	0.0924	0.0369	0.0320

Table 9 the similarity of All words with the mean toxic vector in the Twitter corpus by Word2vec embedding

	2007	2008	2009	2010	2011	2014
bush	0.2954	0.2598	0.1888	0.1560	0.1302	0.1056
throat	0.1342	0.1112	0.1626	0.1061	0.1319	0.1039
start	0.2665	0.2931	0.3009	0.3115	0.3167	0.3281
grab	0.0953	0.1024	0.1181	0.1296	0.1450	0.1576
horrible	0.0884	0.1200	0.1222	0.1295	0.1407	0.1440
pant	0.0886	0.1462	0.1528	0.1627	0.1696	0.1880
sheeple	0.1552	0.1363	0.0914	0.0630	0.0598	0.0419

fact	0.2486	0.2455	0.2438	0.2402	0.2336	0.2089
laden	-0.1851	-0.0975	-0.0248	0.0226	0.0685	0.1120
man	0.1851	0.1843	0.1673	0.0924	0.0369	0.0320

Table 10 the similarity of All words with the mean toxic vector in the Twitter corpus by PPMI embedding