Sentiment Analysis on Movie Reviews from Rotten Tomatoes

Deenadhayalan Ravi
ID: 220053953
MSc. Data Science
deenadhayalan.ravi@city.ac.uk

1 Problem statement and Motivation

In today's fast-paced world, people highly value their time and want a quick overview before deciding to watch a movie. With numerous movie reviews available online, it becomes difficult to read every review to make a decision. To address this issue, organizations such as Rotten Tomatoes and IMDb rate movies based on people's reviews from various social media platforms like Facebook, Twitter, and Instagram, and interviews. Sentiment analysis can be used to evaluate the ratings given to movies based on these reviews.

The traditional approach of evaluating movies based on expert opinions has its limitations, as it is subjective and does not consider the audience's preferences. However, with the rise of social media platforms, people can now express their opinions and rate movies based on their personal experiences. This vast pool of user-generated data provides an opportunity to obtain a more comprehensive and unbiased evaluation of movies. By applying sentiment analysis techniques on this data, we can gain insights into how people perceive movies, which can be valuable for movie producers and marketers in making informed decisions.

2 Research hypothesis

The research hypothesis is that incorporating contextual information using advanced neural network architectures such as DistilBERT and Bi-LSTM can improve the accuracy of sentiment analysis on movie reviews from Rotten Tomatoes compared to traditional machine learning models such as SVM and LR. Additionally, we believe that the choice of vectorizer/encoder significantly impacts the performance of sentiment analysis models. Pre-trained language models such as DistilBERT can outperform traditional vectorizers like TF-IDF vectorizer or one-hot encoding in accurately predicting the sentiment of movie reviews. Our goal is to identify the best model that can accurately predict the sentiment of movie reviews and demonstrate the effectiveness of using advanced machine learning techniques for sentiment analysis.

3 Related work and background

In a comparative study of various machine learning approaches for sentiment analysis on the Rotten Tomatoes movie review corpus, Jain et al. (2016) [1] found that Support Vector Machines (SVMs) achieved the highest accuracy of 80.68%, followed by Naïve Bayes with 79.97% accuracy. They also experimented with Deep Learning and Neural Networks (MLP), but found that these models did not perform as well as traditional machine learning models. Overall, their study demonstrated the effectiveness of SVMs and Naïve Bayes for sentiment analysis on movie reviews.

In a related study [2], the authors used the Word2Vec method in combination with a Naïve Bayes model to perform sentiment analysis on a movie review dataset. The study investigated the effect of stemming on accuracy and found that the model without stemming achieved an accuracy of 72%, whereas the model with stemming achieved an accuracy of 64%. The authors also compared the accuracy of Word2Vec models constructed with 100 and 300 dimensions, and found that the 300-dimensional model achieved an accuracy of 72%, while the 100-dimensional model achieved an accuracy of 71%.

The study conducted by the author in paper [3] involved analysing sentiment on the IMDB dataset and comparing accuracy differences with various pre-processing techniques such as unigrams,

bigrams, and trigrams using LR, NB, and SVM models. The results showed that LR and NB with trigrams achieved higher accuracy levels of 97% and 94%, respectively, which were superior to the unigram and bigram combinations. In the aforementioned paper, the author experimented with count vectorizers and TF-IDF on Twitter dataset, using combinations of n-grams with LR and NB. Among the combinations, LR with count vectorizer and bigrams yielded the highest accuracy of 82%.

The study presented in reference [4] focuses on sentiment analysis of movie reviews using Long Short-Term Memory (LSTM) and Word2Vec. By employing the LSTM method aided by Back Propagation Through Time (BPTT), the research achieved an accuracy of 86.75%. The LSTM approach has the advantage of being capable of storing previous data patterns in its network structure.

The study described in [5] conducted sentiment analysis on the IMDB dataset utilizing BERT and various other machine learning models. The findings demonstrated that BERT outperformed other models, such as LR and NB, with an accuracy rate of 93%.

4 Accomplishments

This section gives an overview of the accomplishments in this project,

- Task 1: Preprocess dataset – Completed
- Task 2: A baseline model has been created to evaluate the accuracy of different feature selection combinations using count vectorizer and TF-IDF, as well as unigram, bigram, and trigram in both Logistic Regression model (Model 1) and SVM model (Model 2). Performed a hyperparameter optimization for SVM model – Completed
- Task 3: A fancy model has been created to evaluate the accuracy of feature extraction using one-hot encoding and pre-trained Word2Vec in a Bi-LSTM Model (Model 3). – Completed.
- Task 4: A fancy model has been created to evaluate the accuracy of DistilBERT Model (Model 4) - Completed
- Task 5: collected the incorrect predictions from all three models and performed in-depth analysis - Completed.

5 Approach and Methodology

This section explains about the general methodology, detailed approach of my baseline models and the fancy model, and names of the supporting files related to this project.

5.1 Methodology:

The general methodology for all the models used here is load dataset → split train, validation and test datasets → pre-processing → feature extraction → modelling on train dataset → evaluating on test dataset.

The performance of a machine learning model is evaluated using various metrics, including accuracy, precision, recall, and f1-score.

- Accuracy measures the proportion of correctly predicted instances out of the total instances in the dataset. It provides an overall measure of the model's performance.
- Precision measures the proportion of correctly predicted positive instances out of all the instances predicted as positive by the model. It represents the model's ability to correctly identify positive instances.
- Recall measures the proportion of correctly predicted positive instances out of all the actual positive instances in the dataset. It represents the model's ability to identify all positive instances.
- F1-score is the harmonic mean of precision and recall, and provides a balance between these two metrics. It is a good metric for evaluating the model's overall performance when the classes are imbalanced.

5.2 Approach

Baseline Models:

The baseline models are as under,
Model 1 - Logistic Regression and
Model 2 - Support Vector Machine.

LR and SVM are linear models that can efficiently learn the linear boundaries between positive and negative sentiments in data. These models are suitable for binary classification tasks, which are commonly used in sentiment analysis for

distinguishing between positive and negative sentiment.

Feature Extraction of baseline:

In this, I have used two feature extraction methods,

- Bag Of Words (BOW), this is a simple method that involves counting the frequency of each word in a document and then representing each document as a vector of word counts.
- Term Frequency-Inverse Document Frequency (TF-IDF) - It is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The TF-IDF score is calculated by multiplying the number of times a term (word) appears in a document by the inverse frequency of the term in the corpus. The higher the TF-IDF score of a word in a document, the more important that word is in that particular document.
- Why to go with TF-IDF, TF-IDF considers the importance of a word in a document and in the entire corpus, whereas count vectorizer only counts the number of occurrences of each word in a document. TF-IDF gives more weight to unique words in a document and less weight to common words across all documents.
- Ngrams - It refer to contiguous sequences of n words in a given text. They capture the local context and sequential relationships between words in a text. For example, in a sentence "The movie is not that bad", unigram plots this as negative sentence based on the word 'bad' in it, whereas trigram takes 'not that bad' into account.

So, I have tried a combination of both BOW, TF-IDF and unigram, bigram and trigram and ran all these combination in Model 1 – LR and Model 2 - SVM

Limitations of baseline:

The above baselines work only on the frequency of the tokens, this methos cannot understand the context of the words/sentence.

Packages used:

In the baselines we used the following packages,

- nltk for pre-processing,
- sklearn for feature extraction, modelling and performance metrics.
- Some basic packages like pandas and NumPy

Fancy Model – Bi-LSTM:

Model 3: Bidirectional LSTM (Long Short-Term Memory) Model

Bi-LSTM is suitable for sentiment analysis over other neural network models because it has the ability to capture contextual dependencies in the input text. Unlike traditional LSTM models, Bi-LSTM reads the input text in both forward and backward directions, allowing it to capture the meaning of a word in the context of both its preceding and succeeding words. This is especially useful for sentiment analysis, as the sentiment of a sentence can be heavily influenced by the words that come before or after a particular word.

Feature Extraction for Bi-LSTM:

The words will be converted into vectors and given as input to the Bi-LSTM. Converting the tokens into vectors is called word embedding. It is basically an Artificial Neural Network which runs in the background to convert the indexed numbers into vectors. These vectors are optimized to capture the semantic relationships between words in the corpus, such that words with similar meanings are located closer to each other in the vector space. In this there are two types,

- Simple One-hot encoding: In this, the text is indexed, padded and a word embedding matrix is created to send as an input to the Bi-LSTM model, where the weights of the vectors continuously change and, in the word embedding ANN, as per the training corpus.s
- Pre-trained Word2Vec: In this, a pre-trained word embedding model will be pull to the script, which is trained on google news, the vectors of the tokens or the weights of the embedding matrix is from the google news with 300 dimensions and will pass this to the dataset and create embedding matrix and pass it to word embedding layer with a note of trainable option in the embedding layer as False. So, the weights of the word will not get changed during the training.

Since the pre-trained Word2Vec is trained on a large corpus of google news it is expected to give better results than the one hot encoding model.

Limitations of Word2Vec:

One limitation of Word2Vec is that it may not capture the contextual meaning of words. For example, the word "bank" could refer to a financial

institution or a riverbank, but Word2Vec may not differentiate between these meanings.

Packages used:

In the baselines we used the following packages,

- tensorflow keras for both feature extraction and modelling.
- nltk for pre-processing,
- Some basic packages like pandas and NumPy

Fancy Model - DistilBERT:

Model 4: DistilBERT

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained language model developed by Google that uses deep neural networks to generate contextualized word embeddings. It is bidirectional, which means that it considers the entire input sequence when generating word embeddings, taking into account the context in which each word appears. DistilBERT is a Smaller, faster, cheaper, lighter version of BERT base.

Feature Extraction in DistilBERT:

DistilBERT, uses a technique called "self-attention" to extract features from text. Self-attention allows the model to analyze each word in the context of the entire sentence, taking into account both the words that come before and after it. This approach overcomes the limitations of previous models, such as Model 1, 2 & 3, by capturing the context of each word.

Packages used:

In the baselines we used the following packages,

- Pytorch and transformers for both feature extraction and modelling.
- nltk for pre-processing,
- Some basic packages like pandas and NumPy

Incorrect predictions for the hyperparmeter optimized SVM model, Bi-LSTM with Word2Vec model and DistilBERT model were separately stored in an excel file for error analysis.

5.3 File Names

NLP-Coursework Report-DEENA – A detailed coursework report (Self).

NLP_CW_Baseline_Models_LR_&_SVM_DEENA – python codes of baseline model 1 and 2.

NLP_CW_Fancy_Model_Bi_LSTM_DEENA – python codes of fancy model 3.

NLP_CW_Fancy_Model_DistilBERT_DEENA – python codes of fancy model 4

df_incorrect_predictions_SVM – incorrect predictions of SVM model.

incorrect_predictions_BILSTM – incorrect predictions of Bi-LSTM model.

df_incorrect_predictions_DistilBERT – incorrect predictions of DistilBERT model.

6 Dataset

The dataset is Rotten Tomatoes movie reviews, sourced from hugging face[6] which contains movie reviews labeled as either positive or negative based on their overall sentiment. For example, a positive review say "This movie was fantastic, with great acting and a gripping plot," while a negative review say "I was really disappointed with this movie, the plot was weak and the acting was wooden."

The dataset presents several challenges for sentiment analysis, including the presence of sarcasm, irony, and other forms of figurative language that may not be captured by simple word-based approaches. In addition, the dataset is relatively small compared to other popular NLP datasets, which may limit the generalizability of models trained on this data.

The dataset contains 8530 train data, 1066 validation data and 1066 test data. The dataset is well balanced with positive and negative reviews equally in all the datasets - train, validation and test.

6.1 Dataset preprocessing

This section outlines the preprocessing steps taken in the project and provides justification for why they were necessary,

- Text cleaning: removing HTML tags, punctuation marks, and special characters from the text.
- Text normalization: converting all letters to lowercase, handling contractions, and removing stop words.
- Tokenization: splitting the text into individual words or tokens.
- Lemmatization: converting each word to its base form to reduce the dimensionality of the data.

- Have not done stemming: because stemming is creating spelling mistakes on the tokens which may create problem in the contextual representation.
  For example in our dataset, I have taken 3 sentences,
  Before stemming:–
  Review 1: ['rock destined 21st centurys new conan hes going make splash even greater arnold schwarzenegger jeanclaud van damme steven segal',
  Review 2: 'gorgeously elaborate continuation lord rings trilogy huge column words adequately describe cowriterdirector peter jacksons expanded vision j r r tolkiens middleearth',
  Review 3: 'sometimes like go movies fun wasabi good place start']
  After stemming – I have highlighted the words in orange which are with spelling mistake and some good conversions are highlighted in green.
  Review 1: ['rock destin 21st centuri new conan he go make splash even greater arnold schwarzenegg jeanclaud van damm steven segal',
  Review 2: 'gorgeous elabor continu lord ring trilog huge column word adequ describ cowriterdirector peter jackson expand vision j r r tolkien middleearth',
  Review 3: 'sometim like go movi fun wasabi good place start']

7 Baselines and its Results

Logistic Regression and Support Vector Machine models with BOW & TF-IDF feature extraction with different ngrams are used as my base line models.

MODEL 1: Logistic Regression

Here is the performance metrics table for all the combination of Logistic Regression model.

Sl. No.	Model	Vectorizer & N-gram range	Accuracy	Weighted Avg Precision	Weighted Avg Recall	Weighted Avg F1-score
1	LR	CountVectorizer (1, 1)	74.7%	0.75	0.75	0.75
2	LR	CountVectorizer (1, 2)	75.4%	0.76	0.75	0.75
3	LR	CountVectorizer (1, 3)	75.3%	0.75	0.75	0.75
4	LR	TfidfVectorizer (1, 1)	76.0%	0.76	0.76	0.76
5	LR	TfidfVectorizer (1, 2)	74.8%	0.75	0.75	0.75
6	LR	TfidfVectorizer (1, 3)	74.0%	0.74	0.74	0.74

Table 1: LR performance metrics table

Inference:

The best performing model is LR with TF-IDF Vectorizer and unigram with an accuracy of 76% and weighted average precision, recall, and F1-score of 0.76.

Even though the difference in accuracy between BOW and TF-IDF is small (1%), it is not unexpected. This is because TF-IDF takes into account the importance of each word in the entire corpus, whereas BOW simply counts the occurrences of each word in the document.

In count vectorizer, increasing the n-gram range can help capture more contextual information, leading to better accuracy. This is because it considers a sequence of n words as a single feature and thus captures more information about the order of words in a sentence.

However, in TF-IDF, increasing the n-gram range can lead to sparsity of the feature matrix. This is because TF-IDF weights the n-grams by their frequency of occurrence in the corpus, and higher-order n-grams occur less frequently in the corpus. As a result, the feature matrix becomes sparser, leading to a drop in accuracy. Additionally, higher-order n-grams may capture more noise in the data and hence may not improve the model's performance.

MODEL 2: Support Vector Machine

Here is the performance metrics table for all the combination of Support Vector Machine model.

Sl. No.	Model	Vectorizer & N-gram range	Accuracy	Weighted Avg Precision	Weighted Avg Recall	Weighted Avg F1-score
1	SVM	CountVectorizer (1, 1)	70.7%	0.707	0.707	0.707
2	SVM	CountVectorizer (1, 2)	72.6%	0.726	0.726	0.726
3	SVM	CountVectorizer (1, 3)	73.2%	0.732	0.732	0.732
4	SVM	TfidfVectorizer (1, 1)	75.2%	0.752	0.752	0.752
5	SVM	TfidfVectorizer (1, 2)	75.9%	0.759	0.759	0.759
6	SVM	TfidfVectorizer (1, 3)	75.8%	0.758	0.758	0.758

Table 2: SVM performance metrics table

Inference:

The SVM model with TF-IDF Vectorizer and n-gram range (1, 2) achieved the highest accuracy of 75.9%, while the model with Count Vectorizer and n-gram range (1, 3) achieved the highest weighted average precision, recall, and f1-score of 0.732. Generally, using TF-IDF Vectorizer resulted in higher performance compared to Count Vectorizer, and increasing the n-gram range improved the performance of the SVM models.

Based on the results of the both the models, it appears that the LR model performs slightly better than the SVM model in terms of accuracy. For the LR model, the best accuracy achieved was 76% using the TF-IDF vectorizer with a unigram approach. For the SVM model, the best accuracy achieved was 75.9% using the TF-IDF vectorizer with a bigram approach.
The difference in accuracy between the best-performing models for each approach is relatively small, at around 0.1-0.2%. Additionally, the results suggest that increasing the n-gram range generally improves the accuracy of the models, at least up to a certain point, which is expected. Finally, both models achieved comparable results using either the BOW or TF-IDF feature extraction approaches.

Even the results of the hyperparameter optimized SVM model is 76% with hyperparameters 'C': 1, 'kernel': 'rbf', 'TFIDF__max_features': 10000.

8 Results, error analysis

In this section let's see the results of our fancy models,

MODEL 3: Bi-LSTM
Here is the performance metrics table for Bi-LSTM model with one hot encoding and Wor2Vec vectorization.

Sl. No.	Model & feature selection	Accuracy	Weighted Avg Precision	Weighted Avg Recall	Weighted Avg F1-score
1	Bi-LSTM with one-hot encoding	67.0%	0.67	0.67	0.67
2	Bi-LSTM with Word2Vec	78.0%	0.78	0.78	0.78

Table 3: Bi-LSTM performance metrics table

Inference:
From the its clear that the accuracy of the Word2Vec is significantly higher than the one-hot encoding vectorization with 78%, but that not very high when compared with the baseline SVM model which is 76%.
The observed improvement in accuracy can be attributed to the ability of Bi-LSTMs to capture the contextual meaning of a word by considering both preceding and succeeding words, as well as the use of Word2Vec feature extraction which helps establish semantic relationships between words in the corpus, this supports our hypothesis statement even.

MODEL 4: DistilBERT
Finally, we experimented with DistilBERT and achieved a test accuracy of 83%, which outperformed the other models we tried. This success can be attributed to DistilBERT's ability to learn contextual representations of words using self-attention mechanisms and the architecture of the BERT model. The pre-trained DistilBERT model was able to understand the meaning of words in the context of the entire sentence, which was crucial for accurate sentiment analysis.

Error Analysis 1: SVM vs BI-LSTM
Three reviews are picked from the incorrect predictions of SVM model and it is here under,

Review 4: 'compassionately explores seemingly irreconcilable situation conservative christian parent estranged gay lesbian child' true label - 1
Review 5: 'soundtrack alone worth price admission' true label - 1
Review 6: 'importance earnest thick wit play like reading bartlett familiar quotation' true label – 1

After manually checking the incorrect predictions of the Bi-LSTM model, the three reviews that the SVM model failed to predict correctly were not found. This indicates that the Bi-LSTM model's performance in predicting complex sentence structures and nuanced language was better than the SVM model. The Bi-LSTM model's ability to understand the semantic and syntactic structure of the reviews was found to be the reason for its improved performance, while the SVM model relied mainly on word frequency and occurrence. Therefore, it is important to consider the semantic and syntactic structure of text in sentiment analysis tasks.

Error Analysis 2: Bi-LSTM vs DistilBERT
Simillarly, three incorrect predictions were selected from the Bi-LSTM model, and they were analyzed to compare the model's performance with that of DistilBERT. Reviews 7, 8, and 9 were identified as incorrect predictions by the Bi-LSTM model, and it was checked if DistilBERT could correctly predict them. It was found that reviews 8 and 9 were predicted correctly by DistilBERT, but review 7 was still incorrect. Further analysis of the part of speech (POS) in the review may be required to

understand why DistilBERT failed to predict it correctly.
Review 7: 'importance earnest thick wit play like reading bartlett familiar quotation'
true label – 1
Review 8: 'moviegoer would automatically bypass hiphop documentary give scratch second look'
true label – 1
Review 9: 'babyfaced renner eerily convincing bland blank man unimaginable demon within'
true label - 1

9 Lessons learned and conclusions

In this report, we explored the application of various machine learning models and techniques for sentiment analysis on movie reviews from Rotten Tomatoes.
We started with data preprocessing, which involved cleaning and tokenizing the text data. During this stage, stemming has given me some spelling errors like 'movie' to 'movi' most of the last words with e, i, y, were removed which will have an impact of the dimensionality of the data. So, didn't used it.
We then experimented with different feature extraction/word embedding techniques, including frequency based - BOW, TF-IDF and context based - one-hot encoding, and pre-trained language models like Word2Vec and DistilBERT. During this stage learnt about the working principles of each and every vectorizers like BOW only counts the frequency of the tokens, whereas TF-IDF looks one step further and sees the importance of the word in that particular document. Then learnt about the one-hot encoding vectorizer and Word2Vec where the token are converted into vector and given as an input to the main model, for which they are using an ANN. And understood that these can only understand the context of the token with the context of the front and back words in the dimensions window. Whereas the self-attention mechanism used in the DistilBERT really understands the context of the words using the query, key, values.

Sl. No.	Model	Vectorizer/Encoder	Accuracy
1	DistilBERT	Pre-trained DistilBERT	83%
2	Bi-LSTM	Pre-trained Word2Vec	78%
3	LR	TfidfVectorizer (1, 12)	76%
4	SVM	TfidfVectorizer (1, 2)	76%
5	Bi-LSTM	One-hot encoding	67%

Table 4: Accuracy of all best models in this paper.

We then compared the performance of various classification algorithms, including logistic regression, support vector machines, and neural networks like Bi-LSTM and DistilBERT and we understood that DistilBERT with its self-attention mechanism, architecture and pre-trained feature extraction technique to understand the context of each and every token has made it to stand out with 83% of accuracy from the other traditional machine learning models and Bi-LSTMs.
We then performed a manual error analysis is done as explained above which showed us which are the reviews correctly predicted and by which model. A deeper analysis on the POS level can be performed to understand more.

Furthermore, as our hypothesis statement our findings indicate that the choice of vectorizer/encoder has a significant impact on the performance of sentiment analysis models. Pre-trained language models like DistilBERT can outperform traditional vectorizers like TfidfVectorizer or one-hot encoding.

In conclusion, this report demonstrates the effectiveness of using advanced machine learning techniques and pre-processing methods for sentiment analysis. It also highlights the importance of selecting appropriate feature extraction techniques and classification algorithms for accurate prediction.

References

[1] J. Wu and Y. Pao, "Predicting Sentiment from Rotten Tomatoes Movie Reviews." Accessed: May 05, 2023. [Online]. Available: https://nlp.stanford.edu/courses/cs224n/2012/reports/WuJean_PaoYuanyuan_224nReport.pdf
[2] S. Rizal, Adiwijaya, and M. D. Purbolaksono, "Sentiment Analysis on Movie Review from Rotten Tomatoes Using Word2Vec and Naive Bayes," *IEEE Xplore*, Nov. 01, 2022. https://ieeexplore.ieee.org/document/10030009 (accessed May 05, 2023).
[3] S. Bandyopadhyay, D. Sharma, and R. Sangal, "Sentiment Analysis: An Empirical Comparative Study of Various Machine Learning Approaches," NLPAI, 2017. Accessed: Apr. 26, 2023. [Online]. Available: https://aclanthology.org/W17-7515.pdf
[4] R. S. Murti, *Analisis Sentimen pada Ulasan Film Menggunakan Word2Vec dan Long Short-Term Mermory (LSTM)*. Universitas Telkom, 2020.

Accessed: May 05, 2023. [Online]. Available: https://openlibrary.telkomuniversity.ac.id/pustaka/164276/analisis-sentimen-pada-ulasan-film-menggunakan-word2vec-dan-long-short-term-mermory-lstm-.html
[5] S. González-Carvajal and E. Garrido-Merchán, "Comparing BERT against traditional machine learning text classification," Jan. 2021. Available: https://arxiv.org/pdf/2005.13012.pdf
[6] Rotten Tomatoes movie reviews dataset https://huggingface.co/datasets/rotten_tomatoes
[7] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10. Association for Computational Linguistics, pages 79–86
[8] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phraselevel sentiment analysis. In Proceedings of the conference on human language technology and empirical methods in natural language processing. Association for Computational Linguistics, pages 347–354
[9] Tony Mullen and Nigel Collier. 2004. Sentiment analysis using support vector machines with diverse information sources. In EMNLP. volume 4, pages 412– 418
Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Proceedings of the 43rd annual meeting on association for computational linguistics. Association for Computational Linguistics, pages 115–124

Total Word Count – 3827 (excluding reference)