Natural Language Processing

NLP Module Assessment

submitted towards differentiation for the degree of

MSc Artificial Intelligence

by

Joseph McInerney

School of EEECS

Queen's University Belfast

Lecturer
Dr Barry Devereux

Word Count: 2967

Student ID: 40460459

April 2025

Contents

1 | Text Processing and Linguistic Insights

1.1 Writing Style and Dependency Parsing

1.1.1 Introduction

Dependency parsing structures sentences into head-dependent pairs. Dependency length serves as a proxy for syntactic complexity, where greater distances indicate higher cognitive effort. The spaCy dependency parser [9] provides an efficient and precise syntactic dependency parser and functionality to analys the tree. As each dependent has a respective head, the maximum dependency length is the head and dependant relation with the greatest distance.

1.1.2 Analysis

SpaCy provides the index of each token in the sentence. So in traversing the dependency tree, we can calculate the distance and the difference between the head and dependent index. A comparison is made between a short-dependency sentence from the 'concise' style and a long-dependency sentence from the 'descriptive' style.

Concise: "He built worlds with his mind."

The following figure 1.1, shows the visualisation of SpaCy's dependency parser representation of the sentence. The maximum dependency here is 2, with "built" as the head and "with" as the dependent and also "mind" as a dependent of "with".

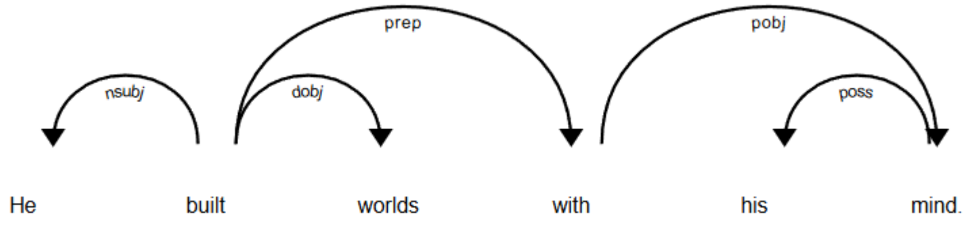


Figure 1.1: "He built worlds with his mind." SpaCy dependency parse visualisation

To understand the calculation of the algorithm. The heads and dependants are outputted along with their respective indices. The difference in these indices (dependence length is also outputted). The ouput for ""He built worlds with his mind" is shown in listing 1.1. The output is of the form head (idx) → dependent (idx).

```
    built (1) -> He (0) | DISTANCE: 1
    built (1) -> built (1) | DISTANCE: 0
    built (1) -> worlds (2) | DISTANCE: 1
    built (1) -> with (3) | DISTANCE: 2
    mind (5) -> his (4) | DISTANCE: 1
    with (3) -> mind (5) | DISTANCE: 2
```

Listing 1.1: IPYNB output from sentence max dependency function for demonstration purposes for concise sentence.

The descriptive sentence is the following:

> Descriptive: ”In a reality where pillars of fire had danced from the heavens and sweaters of ash blanketed the ground, a solitary figure stood, steadfast in his quest for a semblance of the crumbled past.”

The visualisation of this dependency parsing is too large to display as an image in this report but is available in the accompanying IPYNB notebook. However, the IPYNB output is quite interpretable, showing the longest dependency to be 16 between ”blanketed” and ”In”.

```
blanketed (16) -> In (0) | DISTANCE: 16
```

Listing 1.2: IPYNB output from sentence max dependency function for demonstration purposes for descriptive sentence.

It was noticed that some sentences were connected with the string sequence ”\n\n”. The spaCy sentence tokenizer was unable to handle this. Therefore the text was preprocessed to remove ”\n\n”.

Rainbow plots were generated using Petit Prince [8], with a long-form DataFrame to handle varying distribution lengths. The following figure 1.2, shows these rainbow plots.

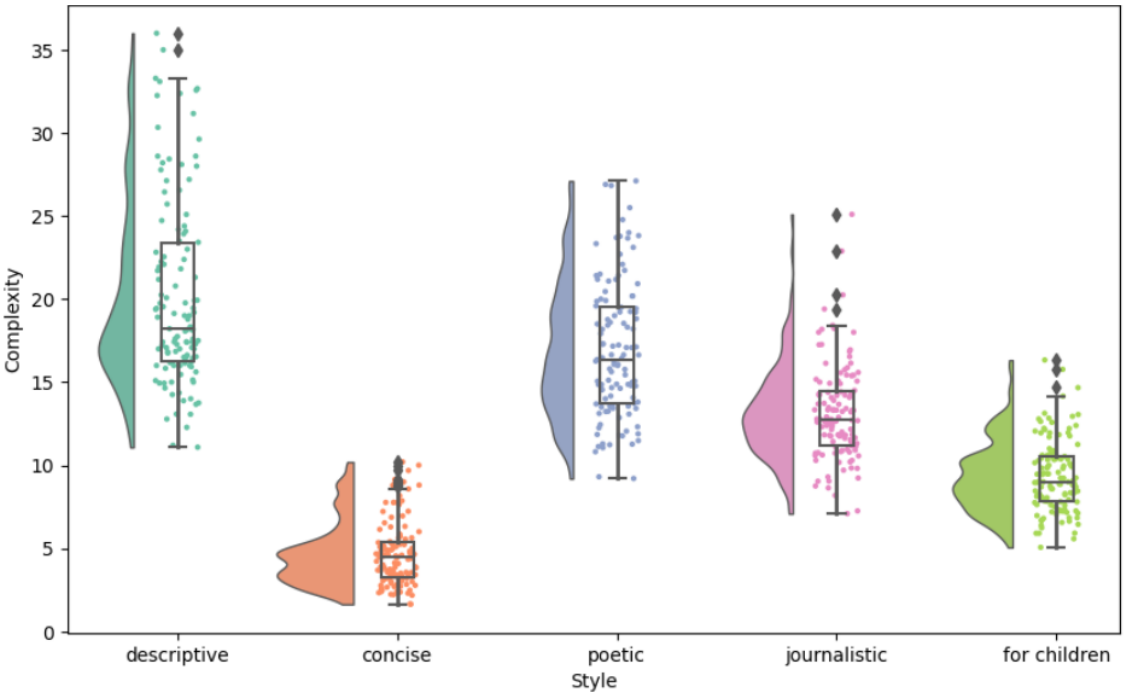


Figure 1.2: Rainbow Plot Displaying the Complexity (Average Maximum Sentence Dependency) Distribution by Writing Style.

The distributions seem to vary but in order to be sure a two tailed Mann Whitney U test was applied comparing each distribution. The following 1.1 is a table comparing the styles. The Mann Whitney U test was selected as it is non-parametric. So, it does not make the statistical assumption of normality in order to compare the central tendency of two distributions. This is crucial as, as the rainbow plot indicates that distributions are not clearly normal.It is also robust when there is a smaller number of samples. For the data set there are roughly 100 samples per distribution which would be reasonable if the distributions were normal and did not have so many outliers. It does assume independence of distributions which fits our data as distributions are generated independently of each other.

Table 1.1: Distribution Comparison from Mann–Whitney U Tests between Styles

”style”	descriptive	concise	poetic	journalistic	for children
descriptive	1	0.000	0.000	0.000	0.000
concise	0.000	1	0.000	0.000	0.000
poetic	0.000	0.000	1	0.000	0.000
journalistic	0.000	0.000	0.000	1	0.000
for children	0.000	0.000	0.000	0.000	1

As we can see all styles vary significantly from each other. This is useful for the interpretation of results as conclusions can be drawn with relative statistical confidence.

1.1.3 Interpretation

Unsurprisingly, children’s stories exhibit lower syntactic complexity, aligning with language acquisition theory. The low linguistic complexity of concise sentences highlights the crudeness of this measure as seemingly it correlates here with sentence length motivating further research where sentence length is controlled for. This is because children can tend to produce long but not syntactically complex sentences[6] (favouring coordination over subordination). Descriptive stories have the highest average maximum dependency length and also the highest variance by story. The journalistic stories have low variance and seem to follow a normal distribution which could be a result of the relatively consistent formatting of articles.

1.2 POS tagging

1.2.1 Introduction

POS tagging assigns syntactic categories to words. This section examines whether POS distributions differ between ’victory’ and ’defeat’ stories. It is hypothesized that they do not, as sentiment influences outcomes more than syntax. However, conceivably, a relationship between the sentiment of a story and certain POS tags could be present. As such, I will hypothesise there being a difference in distributions for the following 5 tags:

NN : Singular common nouns.
IN : Subordinating conjunctions and prepositions.
VBD : Verbs in the past tense.
RB : Adverbs.
PRP : personal pronouns.

1.2.2 Analysis

POS Tag by Style

POS were tagged using the NLTK tagger with Penn Treebank labels. Variations in POS frequency across styles are displayed below in figure 1.3.

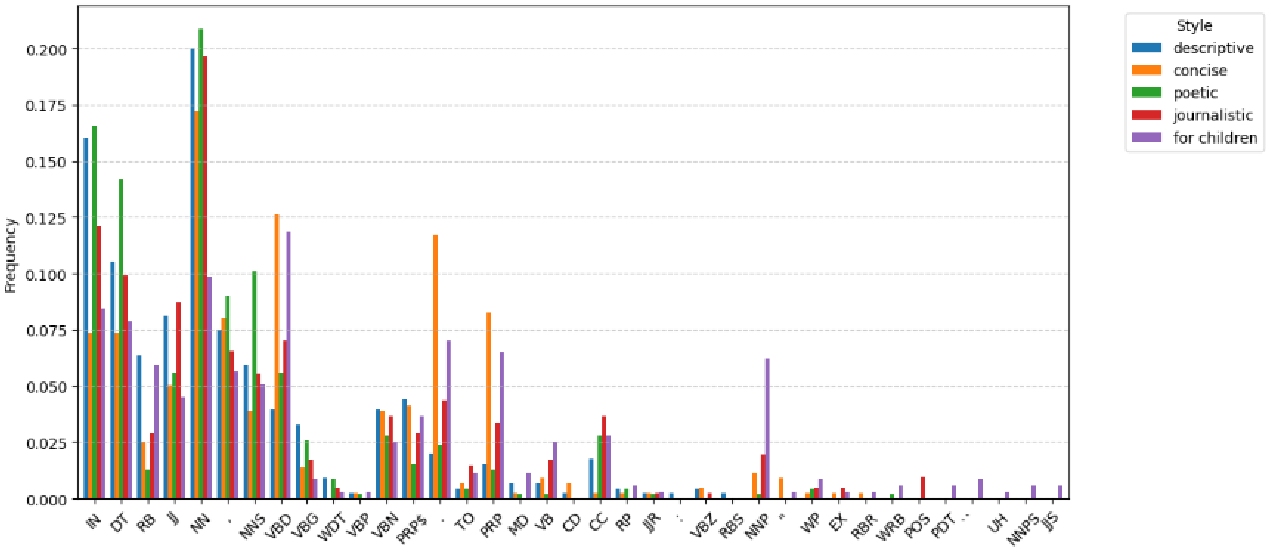


Figure 1.3: Frequency of POS Tag by Writing Style

Some POS frequencies stand out as being vary different depending on style.

POS Tag Distributions by Outcome

Before hypothesis testing, POS tag distributions are visualized. The graphs suggest non-normality and no obvious differences between 'victory' and 'defeat' stories.

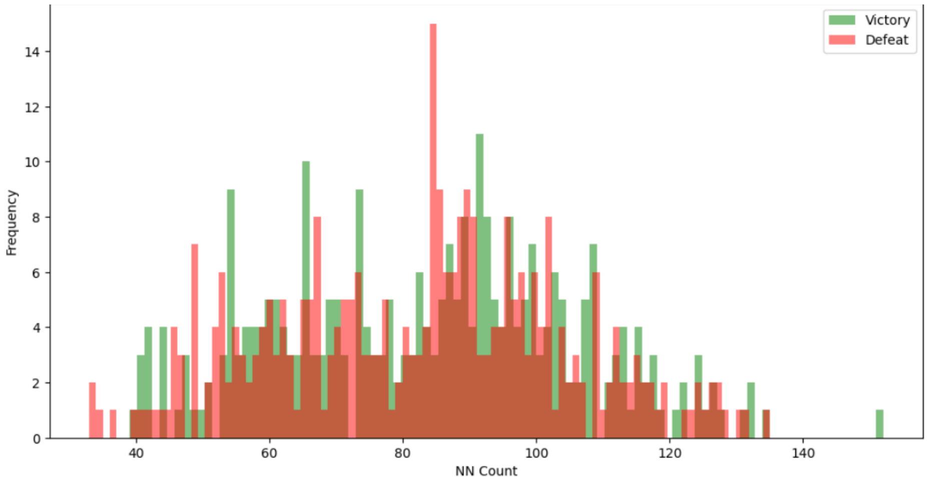


Figure 1.4: NN Count Distribution Comparing Stories by Outcome: "victory"/"defeat"

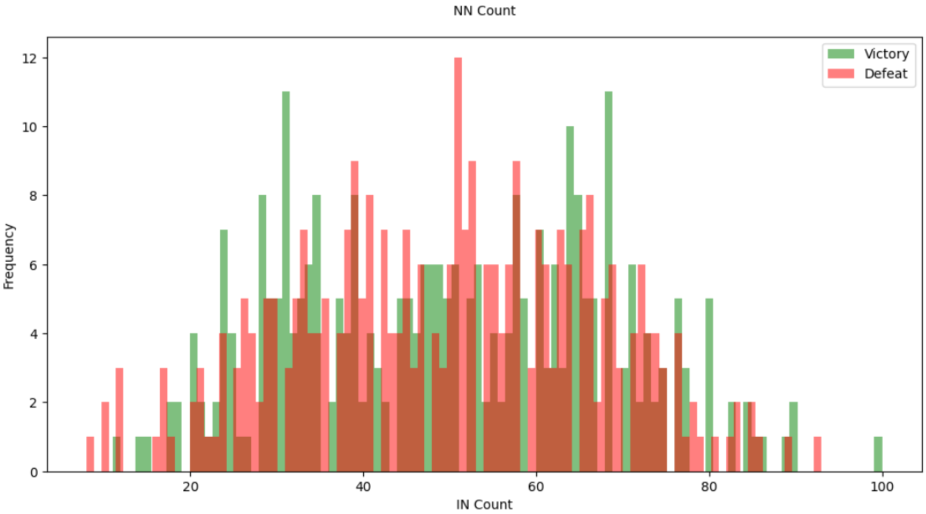


Figure 1.5: IN Count Distribution Comparing Stories by Outcome: ”victory”/ ”defeat”.

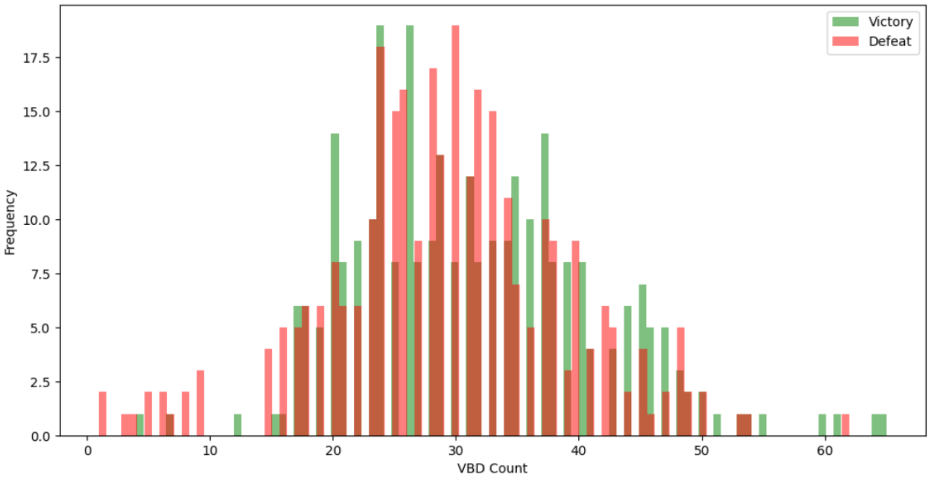


Figure 1.6: VBD Count Distribution Comparing Stories by Outcome: ”victory”/ ”defeat”.

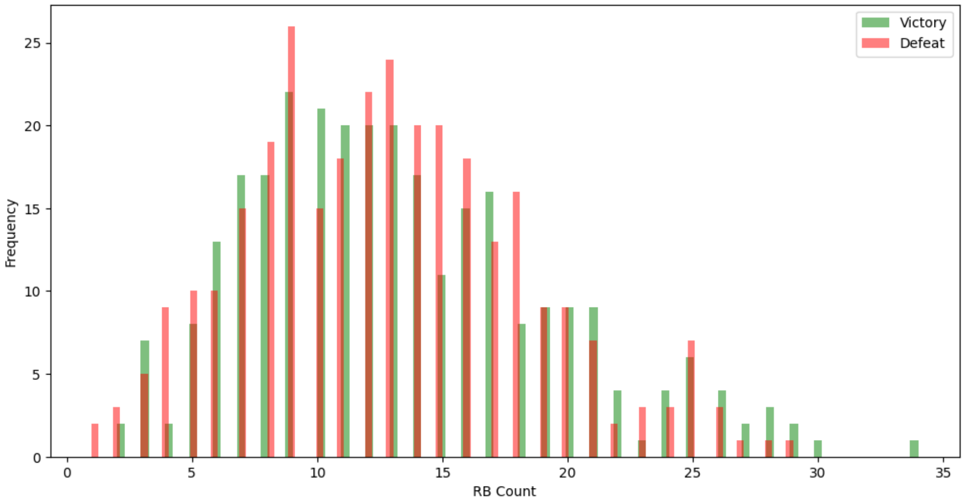


Figure 1.7: RB Count Distribution Comparing Stories by Outcome: ”victory”/ ”defeat”.

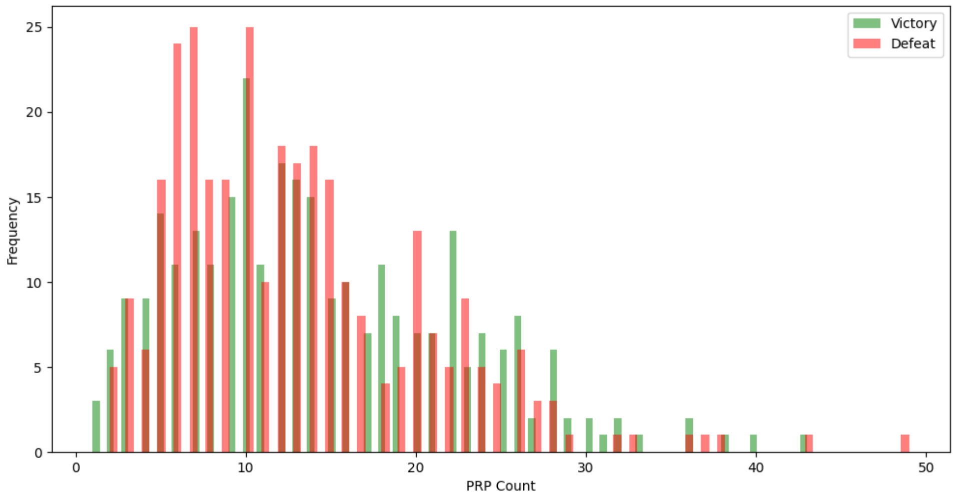


Figure 1.8: PRP Count Distribution Comparing Stories by Outcome: ”victory”/ ”defeat”.

A Mann-Whitney U test assessed differences in central tendency. As no directional hypothesis was set, two-sided tests were used, with significance at $p < 0.05$. This increase in the number of hypothesis tests made a statistically significant result more likely. As such, results are to be interpreted with this in mind. The results of the t-tests were that their was no difference in distributions based on story outcome and POS tag except for VBD and PRP as show below in table 1.2.

Table 1.2: P Values from Distribution Comparison from Mann–Whitney U Tests between POS Tags Based on Story Outcome.

	less	greater	two-sided
VB	0.190	0.810	0.381
IN	0.503	0.497	0.994
VBD	0.966	0.034*	0.067
RB	0.666	0.334	0.668
PRP	0.968	0.032*	0.064

So, ”victory” stories have more VBD (past tense verb forms) and PRP (personal pronouns) than ”defeat” stories according to these t-tests.

1.2.3 Interpretation

POS Tag By Style

Children’s stories show higher NNP frequency, potentially due to a preference for named entities over abstraction.

Concise stories have more full stops, this makes sense and can be seen as a proxy for sentence length. Stories ”for children” also seems to have a slightly higher full stop ratio and therefore shorter average sentence. This could indicate that the maximum dependency calculation is unsurprisingly not independent of sentence length.

”Poetic” and ”Descriptive” and to a lesser extent ”Journalistic” stories have more IN (subordinating preposition/conjugation) tags. This could relate to the proportion of clauses to sentences as outlined by Hunt to be indicative of language complexity [6]. These 3 styles being more complex is for this measure is also consistent with average maximum dependency findings.

”Concise” and ”for children” texts also have a higher frequency of VBD (Past tense verb forms) than the other styles.”Descriptive” and ”for children” stories have elevated adverb frequency when compared to other styles. This shows that tense may also be related to language complexity.

Interestingly ”for children” stories have a much lower proportion of NN (comoon noun) tags. This seems surprising but is potentially due to the increased number of proper nouns in the ”for children” stories.

POS Tag Distributions by Outcome

POS tag distribution does not significantly differ by sentiment. While VBD (p=0.034) and PRP (p=0.032) showed significance, however, after Bonferroni correction $\alpha = 0.0033$ ($\alpha_{corrected} = \frac{\alpha}{N}$), where N is the number of tests. Therefore, this warrents further investigation rather than firm conclusions

The fact that VBD distribution differs significantly but not VB is interesting and could support the hypothesis that stories reporting the past have a bias towards ”victory”, owing to this idea that it is the victors that write history. The increase of PRP in ”victory” stories, as mentioned previously, could indicate fewer characters in a story which could correlate with complications or difficulties for the *hero* in the story.

2 | Text Classification

The goal of this section was to classify the stories' "theme" based on the text of the story. In order to process automatically the natural language of the text, the words are embedded. This embedding maps words from text to vector space. Two methods of embedding were tested, Word2Vec and BERT.

2.1 Text classification with Word2Vec

2.1.1 Introduction

Word2Vec [7] used Skip-gram and CBOW models for word embeddings. CBOW predicts a word from context, while Skip-gram predicts context from a word. Negative sampling replaced soft-max for efficiency.

2.1.2 Analysis

The Word2Vec model was sourced from Hugging Face [5] and optimized with Gensim's Keyed Vectors [10] for faster look-up, as no retraining was required. Firstly to verify that Word2Vec was encoding semantic information as outlined by Mikolov et Al. [7], the following vectors cosine similarty was calculated:

$$\begin{aligned} x &= \textit{vector("Queen")} \\ v &= \textit{vector("King")} - \textit{vector("Man")} + \textit{vector("Woman")} \\ \cos\theta &= \frac{x \cdot v}{\|x\|\|v\|} \\ \cos\theta &= 0.73 \end{aligned}$$

Whereas, when *vector("queen")* is compared with *vector("car")*, the cosine similarity is 0.077. This gives an indication that the Word2Vec model is embedding as expected.

SpaCy was specified soley for tokenization which improved efficiency by disabling redundant preprocessing. Stop words were removed, this reduced noise in the data. Before getting the word embedding from the Word2Vec model, it is verified that the word is in the model, this is because Word2Vec does not handle unseen words. Each story is represented as the mean of its 300-dimensional word embeddings. So, the design matrix X has shape N x D, where N is the number of stories (602) and D is the number of dimensions (300).

The class balance of the settings was visualised and can be seen in figure 2.1, which shows minor class imbalance. This is not unbalanced enough to be a cause for concern in relation to inductive bias.

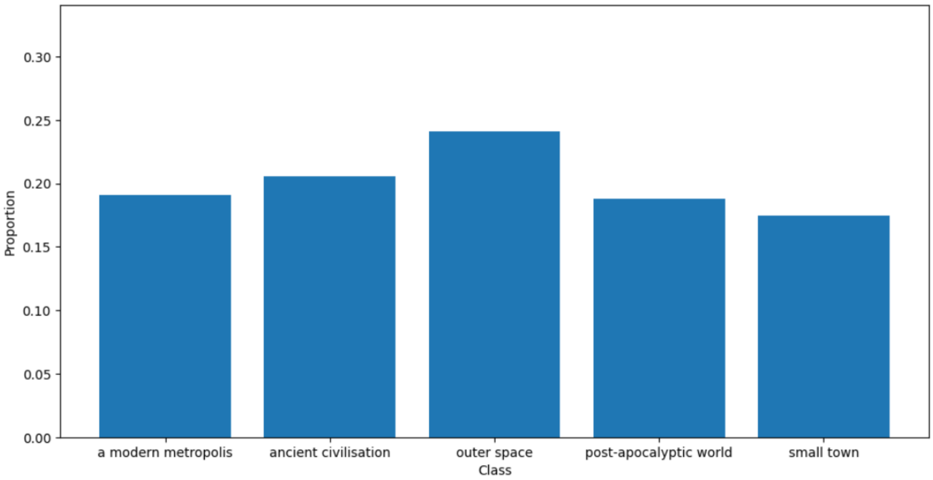


Figure 2.1: Proportion of Total Corpus By Class

Multiple classifiers were tested to gauge a baseline classification accuracy for each. The highest performing model was then selected to be fine-tuned. This is an iterative approach that can quickly an effective model without extensive hyper-parameter tuning of multiple models.

The following classifiers were tested - Random Forest (RF), Support Vector (SV), K-Nearest Neighbours (KNN), Linear Discriminate Analysis (LDA), Quadratic Discriminate Analysis (QDA) and Gaussian Naive Bayes (GNB). The models were initially all tested with their automatic parameters (out-of-the-box). K for KNN was set to 5, the number of settings. SV had the highest initial accuracy at 79%. It was then selected to be fine-tuned, the kernel was changed from linear to radial basis function (RBF). This increased accuracy from 79% to 83%. Finally, the SV regularisation parameter C was fine-tuned. It was increased from 1 to 10 giving a final model accuracy of 86%. The accuracy metric was acceptable in this case as there was no preference for false positives/ false negatives and the classes were relatively balanced.

Table 2.1: Classification Accuracy of Models

RF	SV	KNN	LDA	QDA	GNB
0.744	0.860	0.711	0.719	0.264	0.702

Figure 2.2, below, is the confusion matrix of the SV model with class labels shown.

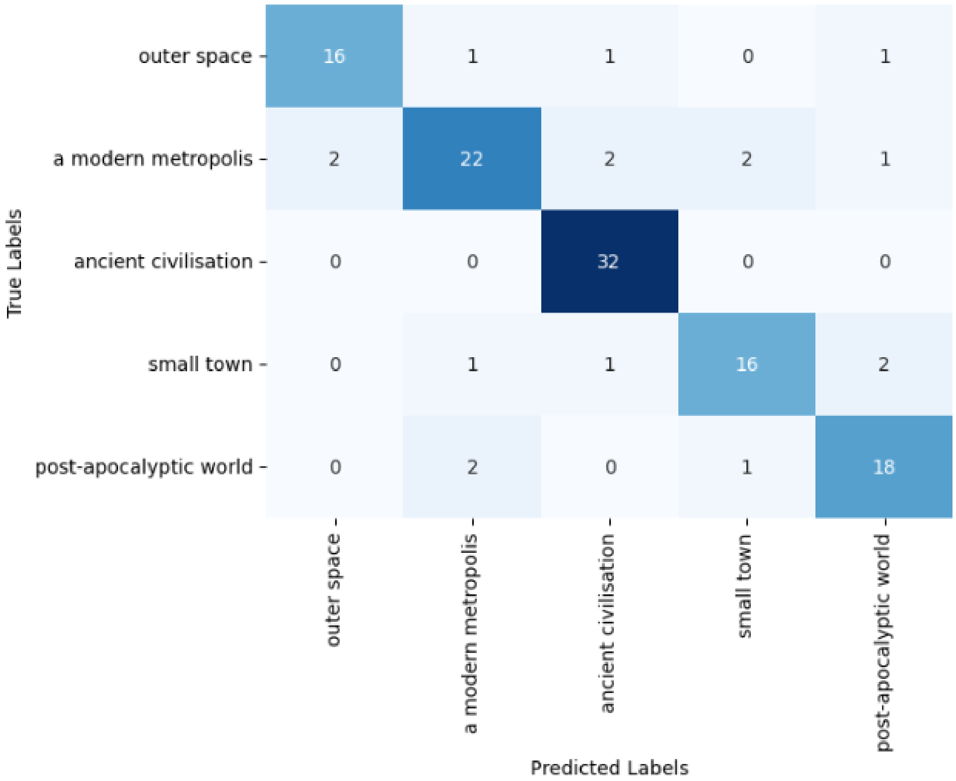


Figure 2.2: Confusion Matrix of SV Evaluation in Classifying Story Settings

2.1.3 Interpretation

QDA failed due to singular covariance matrices, exacerbated by high-dimensional embeddings and limited samples (N=602).

While RF, GNB, LDA and KNN do achieve reasonable accuracy, they still lag behind SV out-of-the-box. This highlights SV's strong out-of-the-box performance as a classifier and effectiveness in high dimensions with few samples proportionally. This is a problem with using Word2Vec, while the high dimensional vectors provide a lot of information, this added complexity makes training difficult (knowing which features are important, collinearity, noise).

SV initially achieved 79% accuracy. Switching to an RBF kernel raised it to 83%, and tuning C (1→10) further improved it to 86%.

2.2 Text classification with BERT

2.2.1 Introduction

The BERT [1] model dynamically embeds words based on context, distinguishing meanings (e.g., 'bank' in finance vs. nature), improving on Word2Vec's static embeddings.

2.2.2 Analysis

Using the same train-test split as Word2Vec, BERT embeddings were applied. Stories exceeding 512 tokens, displayed in figure 2.3 were truncated to fit the model's constraints.

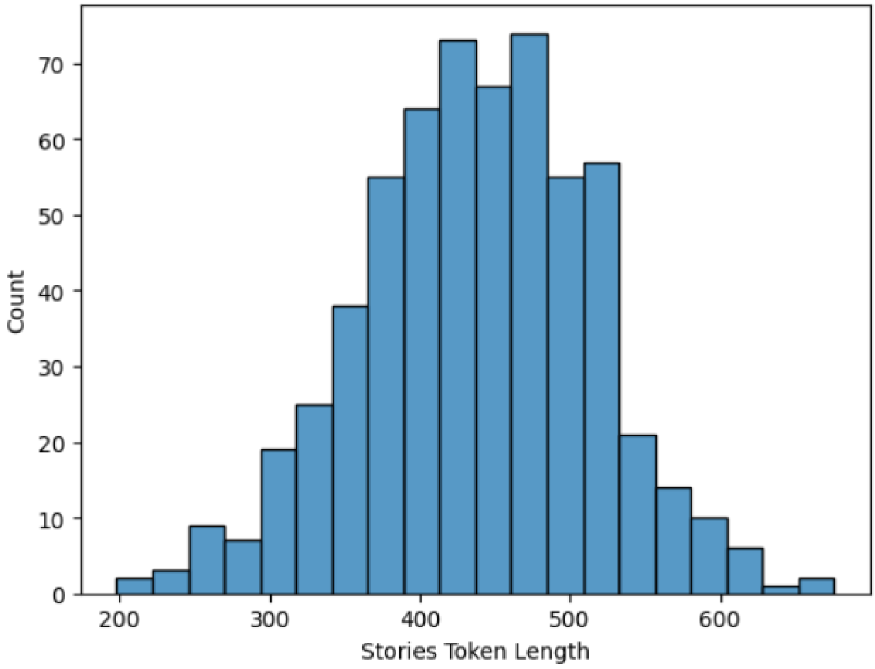


Figure 2.3: Stories Token Length Distribution

The data was first split 80/20 train/test and then from that test set, the data was split 80/20, train/validation. A validation set was used as this method involves the deep learning which of higher risk to over-fitting.

A linear classifier and dropout (p=0.3) were added to BERT. Fine-tuning with various learning rates (5e-5 to 1e-5) showed that 3e-5 provided meaningful classification. Code was adapted from geeks-forgeeks to provide early stopping functionality [3]. Figure 2.4 shows how the training stops at epoch 7, when the validation accuracy converges at 1. As BERT requires few epochs for transfer learning, usually between 2 and 4, the maximum number of epochs was set to 10, with δ set to 0.1 and patience set to 1.

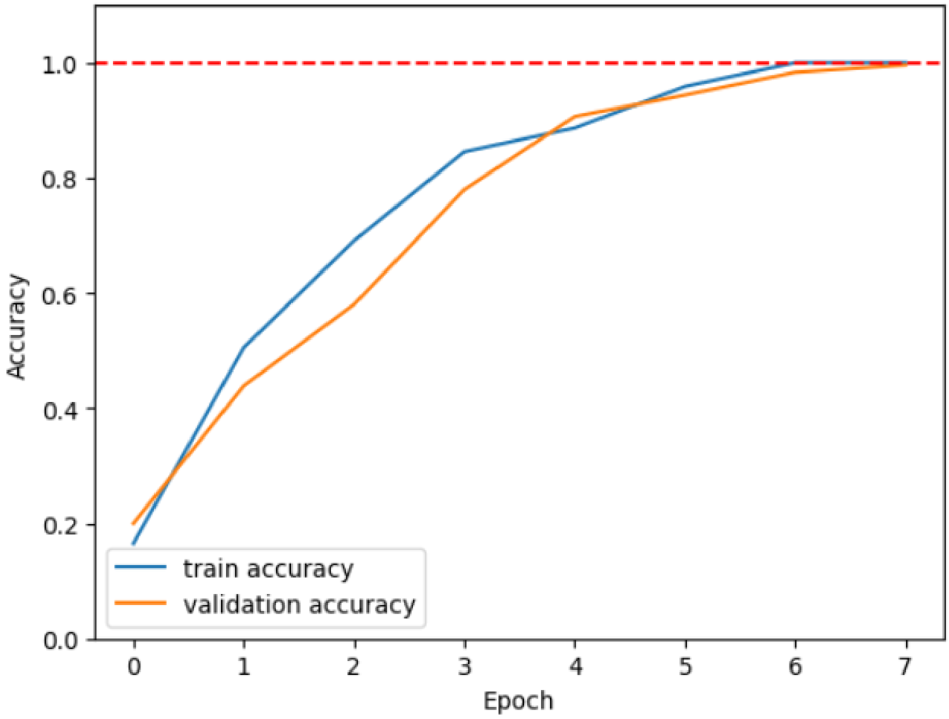
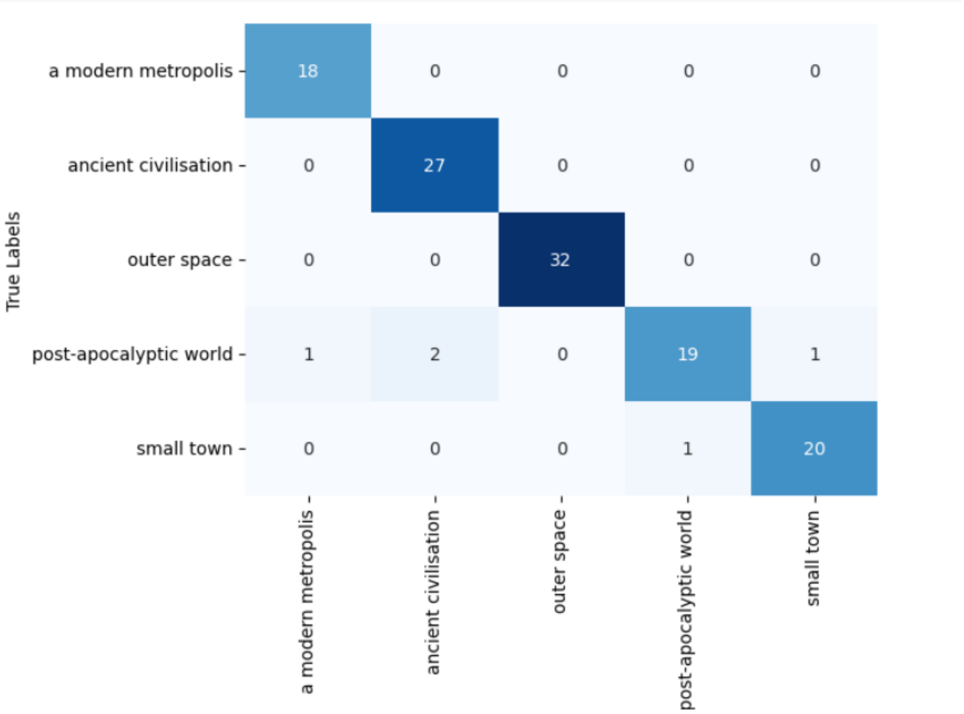


Figure 2.4: Training and Validation Accuracy per Epoch for BERT Classifier.

The test set was then evaluated. It achieved 96% accuracy, slightly lower than the validation accuracy (100%) but still very high. Below, in figure 2.5, the BERT embedded classifier confusion matrix is displayed.



True Labels	a modern metropolis	ancient civilisation	outer space	post-apocalyptic world	small town
a modern metropolis	18	0	0	0	0
ancient civilisation	0	27	0	0	0
outer space	0	0	32	0	0
post-apocalyptic world	1	2	0	19	1
small town	0	0	0	1	20

Figure 2.5: Confusion Matrix Displaying Test Set Accuracy of BERT Classifier.

2.2.3 Interpretation

Slightly lower test accuracy than validation suggests mild over-fitting, mitigated by early stopping. Misclassification was most common in the 'post-apocalyptic world' setting, though errors were minimal.

It can be concluded that the classifier trained with the BERT embedding was more accurate than the classifier trained with the Word2Vec embeddings. This is unsurprising, as BERT embeddings encode context. This is of course important when classifying the setting of a story, as noise is reduced and the model has a deeper language understanding when making predictions. However, Word2Vec is computatinoally less expensive, so could be applied to different NLP tasks with larger data sets and a less sophisticated language task.

3 | Sentiment and Emotion

3.1 Sentiment analysis with BERT

3.1.1 Introduction

Sentiment analysis quantifies subjective framing, categorizing text as positive or negative. Here, ’victory’ and ’defeat’ outcomes were classified based on the last sentence.

3.1.2 Analysis

The design matrix was built using the last sentences of each story, with outcomes extracted from metadata. SpaCy was used for sentence tokenization, accessing the last sentence at index -1. Sample last sentences were reviewed for correct extraction and sentiment relevance. Example of ’defeat’:

> ”The taste of innocent defeat, unnatural, yet strangely compelling.”

Now, here is one clearly indicating ”victory”:

> ”Their shared smiles, the thrill they embraced, declared a potent sentiment invention can’t encapsulate - a triumph of spirit over ruin.”

However some of them are definitely more ambiguous such as:

> ”His final gambit, a symphony of silence permeated by the relentless hum of victorious mechanical locusts.”

It even has the word ”victorious”, but it is in fact a defeat indicating that the locusts are the villains.

BERT transfer learning follows Section 2.2’s approach. Class balance was assessed to prevent inductive bias. The following figure 3.1, displays the proportion of ”defeat” and ”victory” stories.

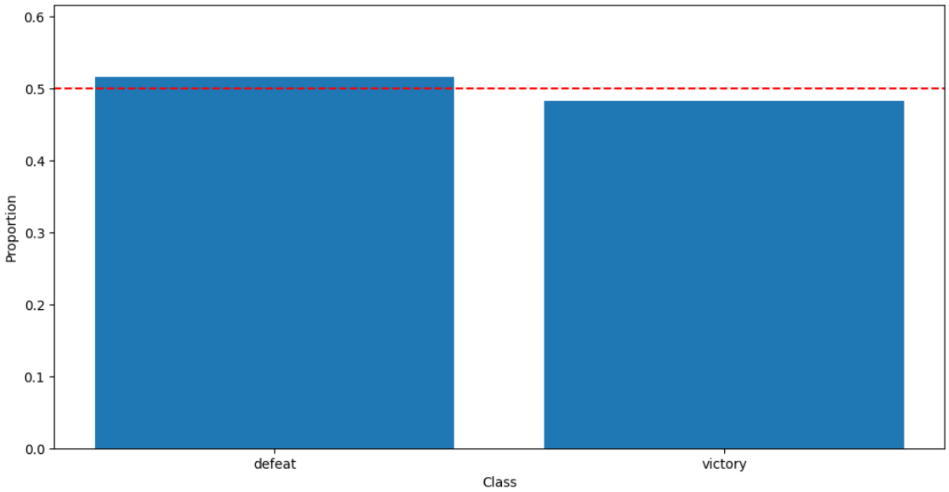


Figure 3.1: Class Balance Per Outcome

Here, the classes are almost equal so no measure were taken to balance their distribution. Validation accuracy converges at 1 after 8 epochs, displayed in figure 3.2

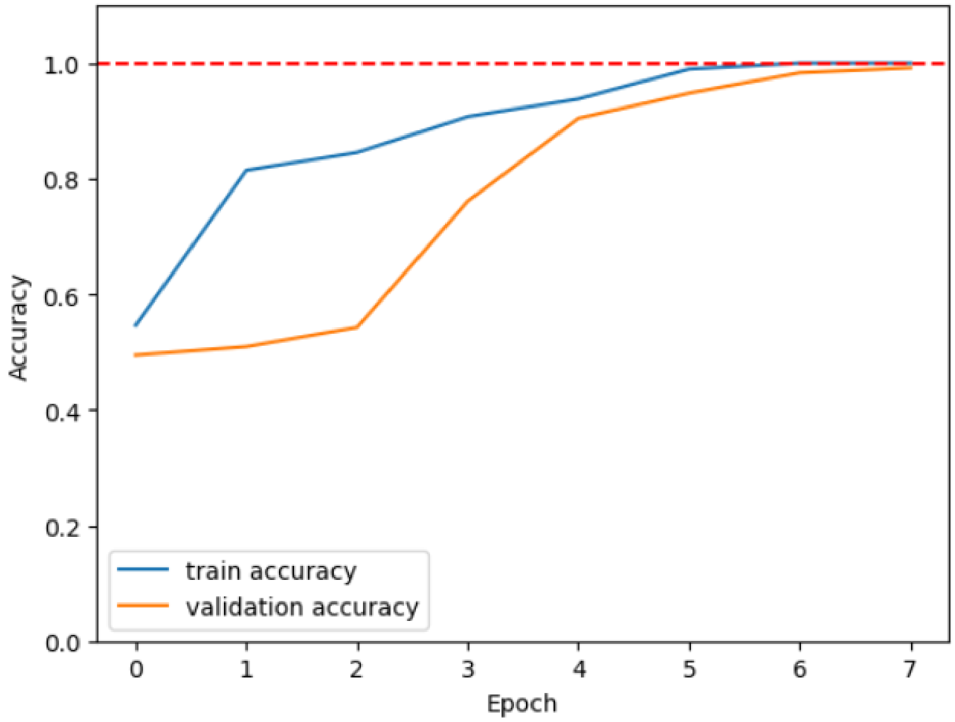


Figure 3.2: Training and Validation Accuracy of Bert Fine-tuned To Predict Victory or Defeat Based on Last Sentence.

The test set accuracy however is 80%, initially I thought that this was over-fitting. As such a more aggressive regularization method was implemented, increasing dropout from p=0.3 to p=0.5 and p=0.9. However, accuracy remained 80%. Learning rate scheduling and early stopping were already employed as well to prevent over-fitting. Then L2 regularisation was added to reduce variance as well but did not affect the test accuracy score. This could indicate a difference in distribution between the validation set and the test set, as such shuffling was introduced when the sets are split but this only increased accuracy by 2%. Next, early stopping patience was reduced from 2 to 1 but still accuracy remained 80%, so it was left at 2. Then, I thought that there must be a difference between the sampling of the distribution between the test set and the validation set, in relation to style. That maybe the stories in the validation set were easier to predict and also that the training set had not seen a balanced proportion of the stories in the test set. Therefore, I stratified by style which produced a training curve that can be seen in fig 3.3. Validation accuracy for the final epoch was at 82% which was similar to test accuracy at 83%. Below, in figure 3.4 the confusion matrix is displayed, showing the classed balanced error distribution.

3.1.3 Interpretation

This task highlights the over-fitting risks of BERT with small corpora, necessitating best practices for robust training and testing.

Considering story style improved generalization, aligning test and validation accuracy, highlighting the importance of data distribution..

The class imbalanced confusion matrix shows that the model biased when predicting class and is more likely to mischaracterise defeat as victory. This is not a case of induced bias, as there are actually slightly more cases of defeat in the data set. This could indicate that the last sentence in

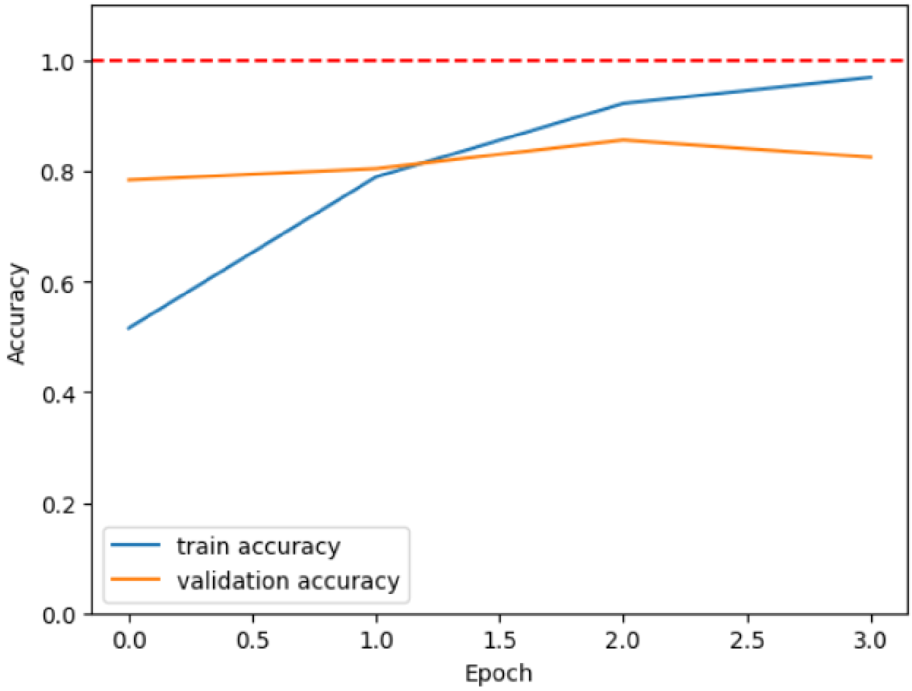


Figure 3.3: Enter Caption

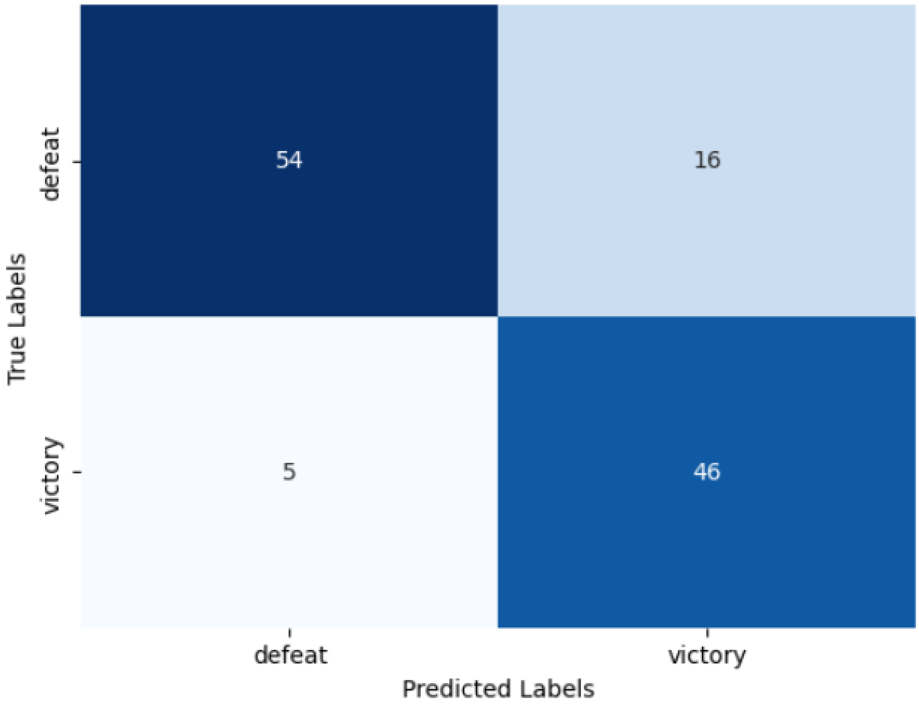


Figure 3.4: Confusion Matrix of Test Set for BERT Sentiment Classification.

victorious stories tends to be more ambiguous.

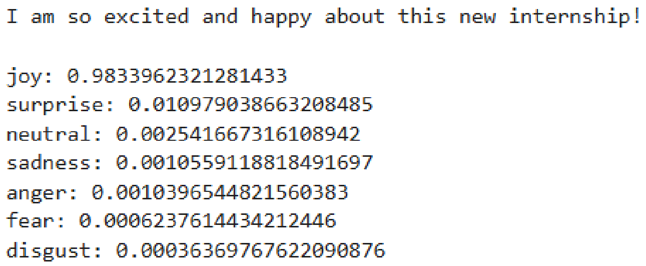
3.2 Emotional Content of Stories

3.2.1 Introduction

The next task was to compare the emotion distribution across story themes. The story themes are: “redemption”, “betrayal”, “discovery”, “rebellion” and “love”. A pre-trained model from hugging face is used to measure emotion. To get an emotion distribution for each story, the emotion distribution of all sentences in that story was aggregated.

3.2.2 Analysis

A distilled RoBERTa model, fine-tuned for English emotion detection, was used. Distillation retains model performance while reducing computational cost. The model is called DistilRoBERTa-base [4], it achieves 66% classification accuracy on test above a baseline of 14%. The emotions detected are those of Ekman’s taxonomy [2]: ”anger”, ”disgust” (with contempt), ”fear”, ”joy” (happiness), ”sadness”, ”surprise”. These are the ”basic emotions” and are universal across cultures, supported by empirical evidence relating to facial expressions. There is also a ”neutral” category is also important, as not all text contains conveys emotion. This model is therefore suitable for the task as it is not too computationally expensive and has respectable classification accuracy underpinned by real psycho-linguistic concepts. For each sentence, the emotion classifier provides an emotion distribution. We can see some example output in figure 3.5 below.



```
I am so excited and happy about this new internship!

joy: 0.9833962321281433
surprise: 0.010979038663208485
neutral: 0.002541667316108942
sadness: 0.0010559118818491697
anger: 0.0010396544821560383
fear: 0.0006237614434212446
disgust: 0.00036369767622090876
```

Figure 3.5: Emotion Distribution from DistilRoBERTa-base for Example Sentence: ”I am so excited and happy about this new internship!”

Sentence-level emotion distributions were computed, aggregated per story, then averaged per theme for comparison. The following figure 3.6 shows emotion distribution across themes. Custom colours for each emotion were set manually to allow for easier interpretation.

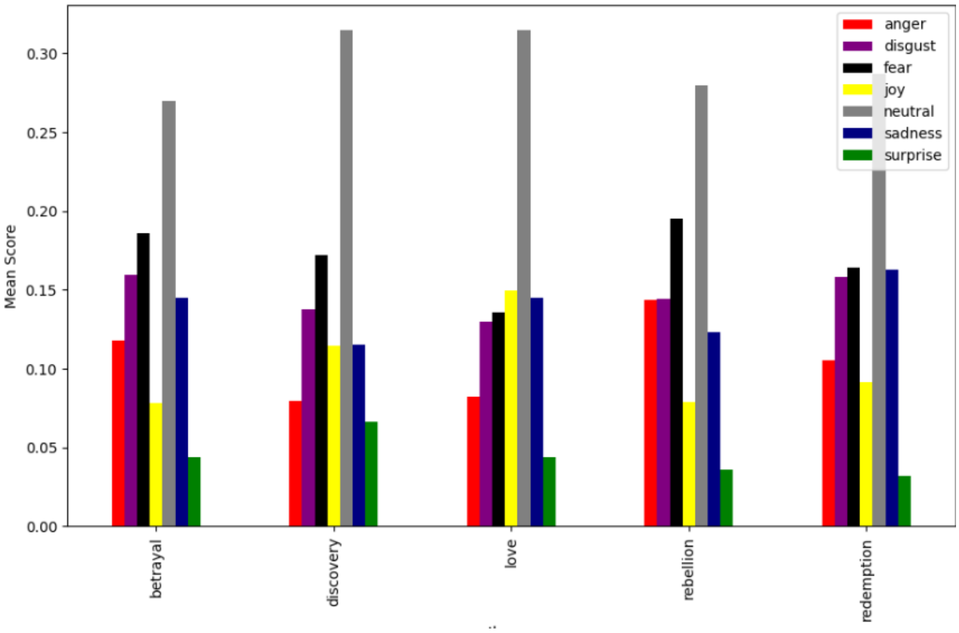


Figure 3.6: The Distribution of Emotions for each Theme (Including Neutral).

The difference between emotions across themes is not immediately apparent. A high proportion of the distribution is consists of ”neutral”. Therefore, In order to compare the distribution of the other emotions, figure

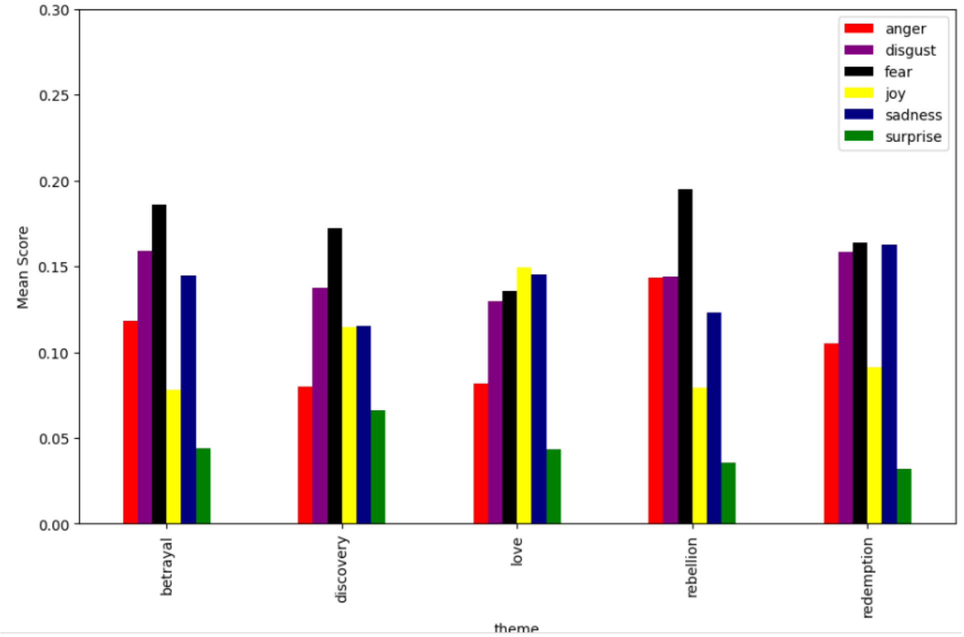


Figure 3.7: The Distribution of Emotions for each Theme (Excluding Neutral).

3.2.3 Interpretation

Neutral’ emotion was highest in ’love’ and ’discovery’ and lowest in ’rebellion’ and ’redemption’. Its correlation with ’joy’ suggests simpler, less emotionally complex narratives.

’Discovery’ had the most ’surprise’, aligning with its unexpected nature.

”Betrayal” and ”Rebellion” show elevated levels of anger potentially indicating the confrontation associated with these themes as opposed to ”love” and ”discovery”.

Sadness varies, being higher in "love" stories than "discovery" or "rebellion" stories. One may expect love stories to have more joy than sadness, however, half of these stories end in "defeat" which in the case of "love" is of course sad.

Fear is the highest in the *confrontational* stories ("betrayal" and "rebellion"). This could be due to the hero fearing a confrontation.

Disgust does not vary significantly, but is highest in "redemption" and "betrayal" stories. This makes sense for "betrayal" stories, as the act of being betrayed can lead to disgust (and contempt). "Redemption" on the other hand could reflect the hero's disgust at themselves before they are redeemed.

Emotions align with story themes, showcasing BERT's contextual modelling and the accessibility of transformer-based AI models.

Bibliography

[1] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. 2019, cs.CL. https://arxiv.org/abs/1810.04805

[2] Ekman, P. 1992, Psychological Review, 99, 550, doi: 10.1037/0033-295X.99.3.550

[3] GeeksforGeeks. 2024, How to Handle Overfitting in PyTorch Models Using Early Stopping. https://www.geeksforgeeks.org/how-to-handle-overfitting-in-pytorch-models-using-early-stopping/#step-6-train-the-model-with-early-stopping

[4] Hartmann, J. 2022, Emotion English DistilRoBERTa-base, https://huggingface.co/j-hartmann/emotion-english-distilroberta-base/

[5] Hugging Face. n.d., fse/word2vec-google-news-300. https://huggingface.co/fse/word2vec-google-news-300

[6] Hunt, K. W. 1965, Grammatical Structures Written at Three Grade Levels. NCTE Research Report No. 3, Tech. Rep. ED 113 735, National Council of Teachers of English, Champaign, IL. https://files.eric.ed.gov/fulltext/ED113735.pdf

[7] Mikolov, T., Chen, K., Corrado, G., & Dean, J. 2013, cs.CL. https://arxiv.org/abs/1301.3781

[8] pog87. n.d., PtitPrince, https://github.com/pog87/PtitPrince

[9] spaCy. n.d., Linguistic Features – Dependency Parse, https://spacy.io/usage/linguistic-features#dependency-parse, Explosion AI

[10] Řehůřek, R. n.d., Gensim: KeyedVectors. https://radimrehurek.com/gensim/models/keyedvectors.html