Text classification: Comparative Analysis of Unigram and Trigram Naive Bayes Classifiers

Livhuwani Mutshafa, (1717376) and Gloria Pucoe, (1477437)

***Abstract*—This study explores the application of text classification algorithms to determine the origin of pages from the Harry Potter book series. Utilizing a Naive Bayes classifier, we implemented both Unigram and Trigram models to classify text from the seven books. The dataset was preprocessed through tokenization, punctuation removal, and N-gram extraction before being split into training, validation, and test sets. Our results indicate that the unigram model outperforms the trigram model, demonstrating higher accuracy and better generalization across the test data. The findings suggest that a simpler model is more effective for this particular classification task.**

***Index Terms*—Text classification, Naive Bayes classifier, N-grams.**

I. Introduction

TEXT classification is an essential task in natural language processing, with applications in sentiment analysis and spam detection, is also valuable in literary analysis. This study aims to determine which book of Harry Potter the page comes from using text classification algorithms.
The Naive Bayes Classifier is a generative probabilistic model that leverages Bayes' theorem for text classification. Its simplicity and ability to handle high-dimensional data make it well-suited for such tasks [1]. Our implementation utilizes N-grams to capture the contextual relationships between words, enhancing classification accuracy.

The main goal of this research is to develop and evaluate a Naive Bayes classifier that can correctly identify the Harry Potter book a page from just the text on it.

II. Methodology

A. Data

Our dataset consists of seven texts from Harry Potter books. Each book is stored as individual text documents with file names ranging from *HP1.txt* to *HP7.txt* corresponding to each of Harry Potter books. The text document contains raw unprocessed content including all original formatting and punctuation.

B. Text Preprocessing

We preprocess the text to make it suitable for the Naive Bayes classifier:

- **Tokenization:** We used the NLTK `word_tokenize` function. The tokenizer breaks the text into the most fundamental relevant units. The text is first divided into a list of words and punctuation marks depending on the whitespace. It further applies a set of rules to separate punctuations from words.
- **Punctuation removal:** To standardize our tokens involved changing all text to lowercase and eliminating all punctuation. By taking these methods, we lowered the dimensionality of our feature space and guaranteed that words were handled uniformly regardless of capitalization or sentence structure.
- **N-gram extraction:** `process_pages()` processes the pages by extracting n-grams and counting their occurrences.

C. Data Splitting

`split_page_data()` splits the processed data into training, validation, and test sets based on defined ratios.The dataset was divided into 70% for training data, 15% for validation data, and 15% for test data.

D. Naive Bayes Classifier Implementation

The NaiveBayesClassifier class contains methods for training and predicting. `train()` calculates the class probabilities (prior probabilities of each book) and the feature probabilities (likelihood of each word given the book). `predict()` calculates the probability of each book given a set of n-grams and returns the book with the highest probability.

E. Training and Evaluation

We trained separate models using both 1-gram and 3-gram sizes. The models were evaluated using accuracy as the primary metric, calculated as the number of correctly classified pages divided by the total number of pages in the test set. Additionally, we used precision, recall, and F1-score to provide a more comprehensive assessment of the Naive Bayes classifier's performance across different books.

III. Results and Discussion

Table I shows that the Unigram model achieves high accuracy on both validation and test sets. The close values between validation and test accuracy suggest that the model generalizes well and performs consistently on unseen data.

The 3-gram model, on the other hand, demonstrates significantly lower accuracy compared to the Unigram model. As shown in Figure 1, both validation and test accuracies are around 50%, indicating poor performance and a lack of generalization. Based on these metrics, the Unigram model is preferable for this classification problem as it offers better performance and generalization.

As shown in Table II and Figure 2, certain books, such

as HP4 and HP7, yield better results compared to others like HP1 and HP2. For instance, HP1's F1-score is lower due to its relatively poor recall of 0.2687, which indicates that the model missed a significant number of pages from this book. However, HP4's high precision, recall, and F1-score indicate that the model is accurate and effective in identifying pages from this book.

The high training accuracy combined with poor validation and test accuracy suggests potential overfitting. While the model performs exceptionally well on training data, it struggles with new, unseen data. The low performance across most classes implies that the increased complexity of trigrams may not be advantageous in this case.

Table IIIprovides a clear snapshot of the model's performance across each book and its overall performance across all classes. A low score in precision, recall, or F1-score for a specific book suggests that the model struggles to distinguish that book from others. For example, the model fails to correctly identify any pages from HP1, as indicated by the precision, recall, and F1-score all being 0.0000. This means that none of the pages predicted to be from HP1 were correct (precision of 0.0000), and the model failed to identify any of the actual HP1 pages in the test set (recall of 0.0000).

Overall, the Unigram model performs better than the 3-gram model, with higher accuracy and better performance metrics across most classes. The increased training accuracy and better validation/test accuracies suggest that the Unigram model is more effective and generalizes better compared to the 3-gram model.

Classifier Type	Validation Accuracy	Test Accuracy
Unigram (Baseline)	70%	70%
3-gram	51%	50%

TABLE I
ACCURACY OF DIFFERENT CLASSIFIER TYPES

TABLE II
CLASSIFICATION REPORT OF N-GRAM =3

Class (Book)	Precision	Recall	F1-score	Support
HP1	0.0000	0.0000	0.0000	67
HP2	1.0000	0.0533	0.1013	75
HP3	0.8750	0.1474	0.2523	95
HP4	0.6000	0.7278	0.6578	169
HP5	0.3955	0.8616	0.5421	224
HP6	0.6512	0.3784	0.4786	148
HP7	0.6887	0.6047	0.6440	172
Macro Avg	0.6015	0.3962	0.3823	-
Weighted Avg	0.5926	0.5200	0.4692	-

IV. CONTRIBUTION

We split the labour equally between the two of us. Gloria was responsible for gathering the findings of the research into a document that could be used for creating reports, while Livhuwani concentrated mostly on code compilations.

V. CONCLUSION

The Naive Bayes classifier classifier provides a straightforward and efficient approach for text classification, its assumption of feature independence can limit its effectiveness in capturing the nuanced dependencies that exist in natural language. This limitation was evident in the reduced accuracy of the Trigram model compared to the simpler Unigram model. Future work should explore advanced feature engineering techniques, such as refining the selection of features, including or excluding stopwords, and experimenting with different N-gram values, like Bigrams. These efforts may enhance the model's ability to account for linguistic dependencies and improve overall classification accuracy.

TABLE III
CLASSIFICATION REPORT OF THE UNIGRAM

Class (Book)	Precision	Recall	F1-score	Support
HP1	0.7826	0.2687	0.4000	67
HP2	1.0000	0.2933	0.4536	75
HP3	0.7407	0.4211	0.5369	95
HP4	0.8214	0.8166	0.8190	169
HP5	0.5349	0.8884	0.6678	224
HP6	0.7820	0.7027	0.7402	148
HP7	0.8034	0.8314	0.8171	172
Macro Avg	0.7807	0.6032	0.6335	-
Weighted Avg	0.7478	0.6989	0.6841	-

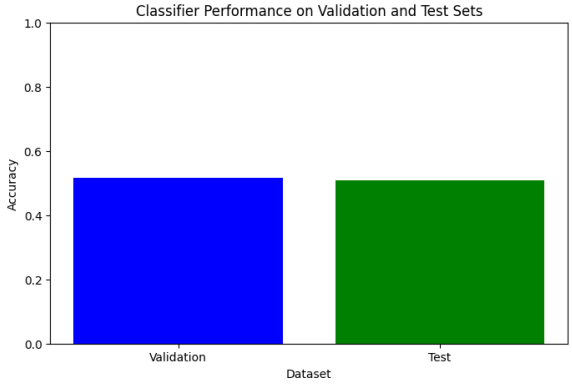

Fig. 1. Distribution of validation and test set for 3-gram.

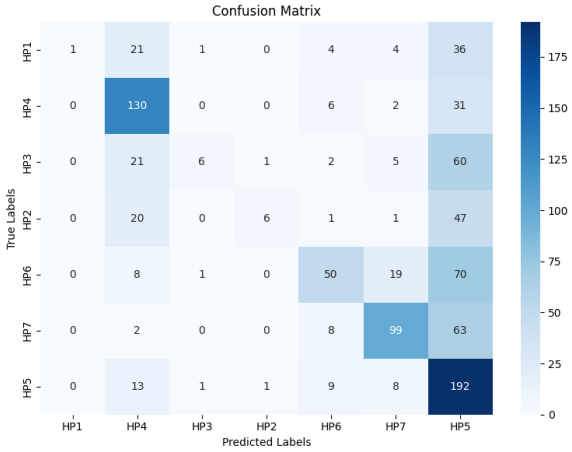

Fig. 2. Confusion matrix of books.

REFERENCES

[1] *Evaluation of Different Machine Learning, Deep Learning and Text Processing Techniques for Hate Speech Detection*, Shawkat, Nabil, 2023.