Group 45

Link to the source code/experiments (zip files): https://drive.google.com/drive/folders/13sWRg7b2ykfuL_vxw8b3Z5Nlo0GTkr8i?usp=sharing

NLP: SENTIMENT, TOPICS, AND NERC INSIGHTS

This project focuses on processing a given test set to evaluate sentiment, topic, and perform Named Entity Recognition and Classification (NERC), showcasing our ability to extract insights from text using NLP techniques.

Work Division

O.J. Closs (2720653): 1) coding: NERC, 2) analysis: NERC, 3) poster preparation: overall & NERC
D.G.J.K. Linger (2741629): 1) coding: topic, 2) analysis: topic, 3) poster preparation: overall & topic
J.D.K. Linger (2533051): 1) coding: topic, 2) analysis: topic, 3) poster preparation: overall & topic
R.R. Autar (2622234): 1) coding: sentiment, 2) analysis: sentiment, 3) poster preparation: overall & sentiment

NERC

Data

Selection

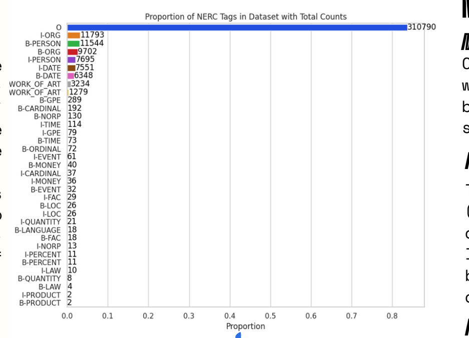
To construct an effective training set for this task, the unique NERC tags were identified in the test set: "B-PER", "I-PER", "B-PERSON", "I-PERSON", "B-ORG", "I-ORG", "B-WORK_OF_ART", "I-WORK_OF_ART", "B-DATE", "I-DATE". "B-PER" and "I-PER" referred to the same objects as "B-PERSON" and "I-PERSON" respectively and therefore these tags were unified in the test set. Through searching Kaggle and Hugging Face open source datasets, identified was the Hugging Face hosted conll2-12_ontonotesv5 [1] dataset, which encompassed all the target NERC tags required for the study. From this dataset the word, pos and nerc tag were all extracted. However the dataset also contained words and sentences that were not in the scope of the test set such as "B-Product" and "B-Percent"; therefore further preprocessing of the 115,812 sentences of training data was necessary.

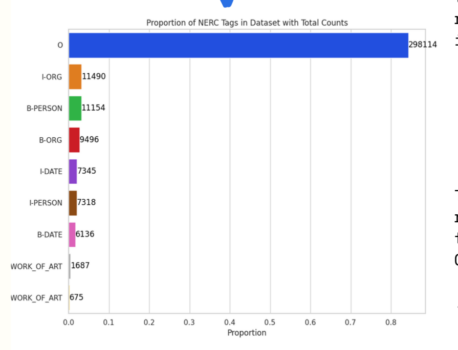
Preprocessing

In this preprocessing phase, sentences containing words with non-target NERC tags were removed. This approach was based on the understanding that irrelevant tags could disrupt the contextual learning process of the model. By excluding these sentences, the dataset was focused solely on sentences with the test-set target NERC tags.

Description

The dataset comprises 36,370 sentences and 353,415 tokens, with 84% of the tokens tagged as 'O'. The 'I-ORG' and 'B-PERSON' tags each constitute 3% of the dataset, while the remaining tags each account for 2% or less. This distribution mirrors the test set closely, which features 82% of tokens with the 'O' tag and approximately 3% for both 'I-ORG' and 'B-PERSON' tags. A notable distinction is observed in the test set, where 'I-WORK_OF_ART' tags represent 4.6% of the data.





Methods

Motivation

Conditional Random Fields (CRF) [2][3] was chosen as the main method. Unlike SVM, which treats each data point independently, CRF leverages the entire sequence context by selecting and including features. Transformers model 'bert-base-NER' was also selected for comparative insights in its ability to understand sequence context.

NERC-CRF 1

To prepare for NERC-CRF training, each word was split into features: part of speech (POS) tag, as well as the preceding and succeeding words, its constituent characters, and additional characteristics such as the presence of digits (source: notebook NERC lab4a.2 [2]). In this system, POS tags were not utilized as features for the test set but were included in the training set. This approach is permissible within CRF [3][4], as CRF models allow for flexibility in feature inclusion across different datasets.

NERC-CRF 1 (Tuned Parameters)

The previous CRF was observed to remember words, particularly names and persons, and therefore overfit the training data. Therefore this system contains tuned c1 (L1 regularisation) and c2 (L2 regularisation)[2][3] parameters. c1 was particularly increased to combat this overfitting.

NERC-CRF 2 (POS Tag)

POS tags were included in the test set and therefore as features by method of NLTK [5] which follows the same syntax as conLL2012 format *Penn Treebank* [6].

Transformers (bert-base-NER)

This method uses transformer model 'bert-base-NER' [2] from the Hugging face repository, which is fine-tuned for Named Entity recognition. This method does not use the conLL2012 dataset as before; it offers an intriguing basis for comparison with the CRF systems.

.

Results

NERC-CRF 1

```
               precision    recall  f1-score   support

       B-DATE       1.00      1.00      1.00         1
        B-ORG       0.00      0.00      0.00         3
     B-PERSON       0.00      0.00      0.00         6
B-WORK_OF_ART       0.25      0.25      0.25         4
       I-DATE       0.25      1.00      0.40         1
        I-ORG       0.00      0.00      0.00         6
     I-PERSON       0.00      0.00      0.00         3
I-WORK_OF_ART       0.19      0.67      0.30         9
            O       0.90      0.86      0.88       160

     accuracy                           0.76       193
    macro avg       0.29      0.42      0.31       193
 weighted avg       0.77      0.76      0.76       193
```

NERC-CRF 1 (Tuned)

```
               precision    recall  f1-score   support

       B-DATE       1.00      1.00      1.00         1
        B-ORG       0.00      0.00      0.00         3
     B-PERSON       0.00      0.00      0.00         6
B-WORK_OF_ART       0.00      0.00      0.00         4
       I-DATE       0.20      1.00      0.33         1
        I-ORG       0.00      0.00      0.00         6
     I-PERSON       0.00      0.00      0.00         3
I-WORK_OF_ART       0.44      0.44      0.44         9
            O       0.86      0.96      0.91       160

     accuracy                           0.82       193
    macro avg       0.28      0.38      0.30       193
 weighted avg       0.74      0.82      0.78       193
```

NERC-CRF 2

```
               precision    recall  f1-score   support

       B-DATE       1.00      1.00      1.00         1
        B-ORG       0.33      0.33      0.33         3
     B-PERSON       0.40      0.33      0.36         6
B-WORK_OF_ART       0.00      0.00      0.00         4
       I-DATE       0.25      1.00      0.40         1
        I-ORG       0.00      0.00      0.00         6
     I-PERSON       0.00      0.00      0.00         3
I-WORK_OF_ART       0.00      0.00      0.00         9
            O       0.88      0.93      0.90       160

     accuracy                           0.79       193
    macro avg       0.32      0.40      0.33       193
 weighted avg       0.75      0.79      0.77       193
```

bert-base-NER

```
               precision    recall  f1-score   support

       B-DATE       0.00      0.00      0.00         1
       B-MISC       0.00      0.00      0.00         0
        B-ORG       1.00      0.67      0.80         3
     B-PERSON       1.00      1.00      1.00         6
B-WORK_OF_ART       0.00      0.00      0.00         4
       I-DATE       0.00      0.00      0.00         1
       I-MISC       0.00      0.00      0.00         0
        I-ORG       1.00      0.33      0.50         6
     I-PERSON       1.00      1.00      1.00         3
I-WORK_OF_ART       0.00      0.00      0.00         9
            O       0.97      1.00      0.98       160

     accuracy                           0.90       193
    macro avg       0.45      0.36      0.39       193
 weighted avg       0.90      0.90      0.89       193
```

NERC-CRF 1 (Overfitting)

y=B-PERSON top features		y=I-PERSON top features	
Weight?	Feature	Weight?	Feature
+5.868	-1:word.lower():ms.	+4.099	-1:word.lower():ludcke
+5.136	-1:word.lower():mr.	+3.492	-1:word.lower():sharfman
+4.746	word.lower():kate	+3.459	-1:word.lower():dempsey
+4.303	word.lower():babysitter	+3.457	-1:word.lower():zarett
+4.209	word.lower():colonists	+3.232	+1:word.lower():baltimore
+4.191	-1:word.lower():jimco	+3.160	-1:word.lower():know
+4.123	word.lower():shakespeare	+2.928	-1:word.lower():sr
+3.976	word.lower():uday	+2.928	+1:word.lower():maui
+3.940	word.lower():miers	+2.903	-1:word.lower():--
+3.888	word.lower():qingqing	+2.852	word.lower():raised
+3.783	word.lower():kramers	+2.840	+1:word.lower():close

Discussion & Analysis

In analyzing the performance of NERC-CRF 1, it was noted for its high precision and recall on 'B-DATE' and 'O' tags, and it successfully identified true positives for 'B-WORK_OF_ART', 'I-WORK_OF_ART', and 'I-DATE'. However, it failed to identify any true positives for the remaining tags, resulting in zero values for precision, recall, and f1-scores. This issue may be attributed to the model's inability to generalize across a broader range of tags, possibly due to overfitting on specific names within the 'PERSON' category, such as 'sharfman', 'kate', and 'zarett'. To address potential overfitting, adjustments were made to the regularization parameters in the tuned version of NERC-CRF 1. Despite these adjustments, there was no improvement in the 'B-PERSON' or 'I-PERSON' categories, and its accuracy (0.82) was carried by its effectiveness in classifying the 'O' tag, which represents a significant portion of the test set (support=160). The incorporation of POS tags from NLTK into the test set for NERC-CRF 2 resulted in improved metrics for 'B-ORG' and 'B-PERSON' tags. This improvement indicates the value of POS information in enhancing the model's ability to accurately classify entities. However, this adjustment led to a decline in the model's ability to classify 'B-WORK_OF_ART' and 'I-WORK_OF_ART' tags, suggesting a trade-off between incorporating POS tags and maintaining performance across a diverse set of entity categories. The transformers model, 'bert-base-NER', outperformed the CRF models in classifying 'B-PERSON', 'I-PERSON', 'O', and 'B-ORG' tags, achieving the highest overall accuracy. However, similar to the CRF models, the transformers model struggled to classify other target labels, indicating a potential limitation in the training data and the need for fine-tuning specific to those labels.

Limitations

The evaluation of NERC-CRF and transformer models reflects the critical roles of feature selection, model architecture, and regularization in NERC task performance optimization. Despite the systems demonstrating high accuracy due to labelling of 'O' tag, achieving consistent performance across diverse entity categories proves to be a challenge. To address this, future efforts could involve implementing cross-validation techniques and increasing the dataset for underrepresented tags to improve model generalization. Additionally, specific categories such as 'B-WORK_OF_ART', may benefit from integrating a Knowledge Base, providing a reference framework that allows the model to recognise nuanced entity types.

Topic Classification

Data

Selection and Description

Preprocessing

To cover the topics in the test set—sports, books, and movies—three different datasets representing each topic were merged into a unified training set. The following sets were used for this purpose: 3982 unique Amazon book reviews , 49582 unique IMDB movie reviews, and 22360 FIFA World Cup tweets, all sourced from Kaggle [7][8][9][17].

Initially, the datasets underwent a cleaning process, which involved the removal of newline characters, URLs, hashtags, reference- and HTML symbols, along with non-Western alphabet characters. Following this, each text entry was segmented into individual sentences, with only sentences containing 8 to 25 tokens retained in the dataset. This focus on sentences of moderate length aims to ensure that each sentence provides adequate information for meaningful analysis, striking a balance between brevity and verbosity to maintain dataset quality. Additionally, for each dataset, the corresponding topic was assigned.

This cleaning process aimed to streamline the datasets and focus solely on the text content, making them easier to work with for subsequent processing steps. Afterward, 1700 instances were randomly sampled from each dataset, resulting in a total of 5100 instances. This ensures that each topic category contributes equally to the training process, preventing any single topic from dominating the model's learning. This combined training set featured evenly distributed topic sentences.

The test set comprises only 10 sentences, with 3 labeled as sports, 4 as movies, and 3 as books. Finally, both the training and test sets underwent punctuation removal and sentence transformation to lowercase for consistency.

Methods

Motivation

To perform the task of topic classification RoBERTa and Naive Bayes have been chosen [10][11]. Each model offers distinct advantages that cater to different aspects of the task. RoBERTa is a pre-trained language model and is an extension of BERT with which it shares the same foundational architecture [12]. However, RoBERTa was trained longer and with more data than BERT. Additionally, RoBERTa utilizes dynamic masking during pretraining, meaning that it masks different tokens at each epoch. These enhancements lead to a better performance on topic classification compared to BERT [11][12].

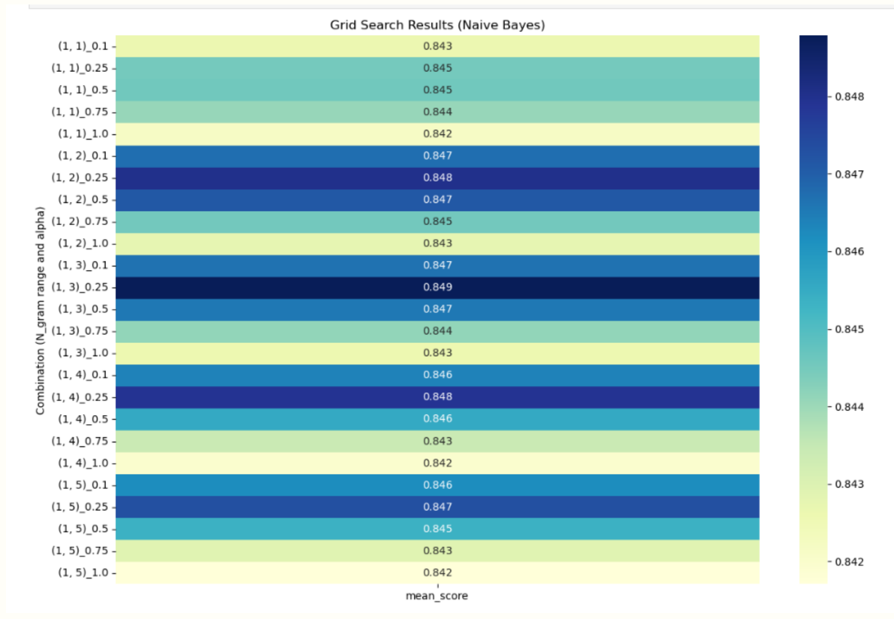
Moreover, as a state of the art language model based on transformer architecture, RoBERTa excels in capturing intricate patterns within text. While it is not inherently a supervised machine learning model, RoBERTa can be trained on a specific topic classification task during the fine-tuning process with labeled examples [11][13]. This allows RoBERTa to adapt the model's parameters to the specific characteristics of the task and dataset.

Naive Bayes on the other hand is a completely supervised approach [10]. While simpler, With sufficient data supervised models often achieve high performance and are easy to interpret [13], topped off with relatively fast training times. Naive Bayes is based on a probabilistic framework that naturally extends to handle multiple classes. It calculates the probability of each class given the input features and selects the class with the highest probability. It is known for its computational efficiency and reliable performance, especially when working with smaller datasets.

RoBERTa (Parameter Tuning)

The initial step of tuning the RoBERTa model involves dividing the training set into two subsets: a training set and a validation set. Specifically, 10% of the data is allocated to the validation set, while the remainder stays in the training set. Additionally, the samples extracted from the training set are stratified based on topics to ensure class balance among both subsets. Evaluation is enabled during the training phase and the model is trained for 10 epochs with a batch size of 46, resulting in 100 steps per epoch.

To mitigate the catastrophic forgetting problem, the learning rate was set to 2e-5 [14]. The catastrophic forgetting problem refers to a tendency in transformer learning to forget previously learned information when trained on new tasks, to counter this a common practice is to set the learning rate low. The maximum sequence length is set to 64 tokens. While increasing this parameter could potentially improve performance, it may also lead to longer training times. However, given that each training sentence has a maximum length of 25 words and considering the additional tokenization incorporated into the RoBERTa model, sequences exceeding 64 tokens appear unlikely, rendering such an increase unnecessary. To prevent overfitting, early stopping was implemented using the specified

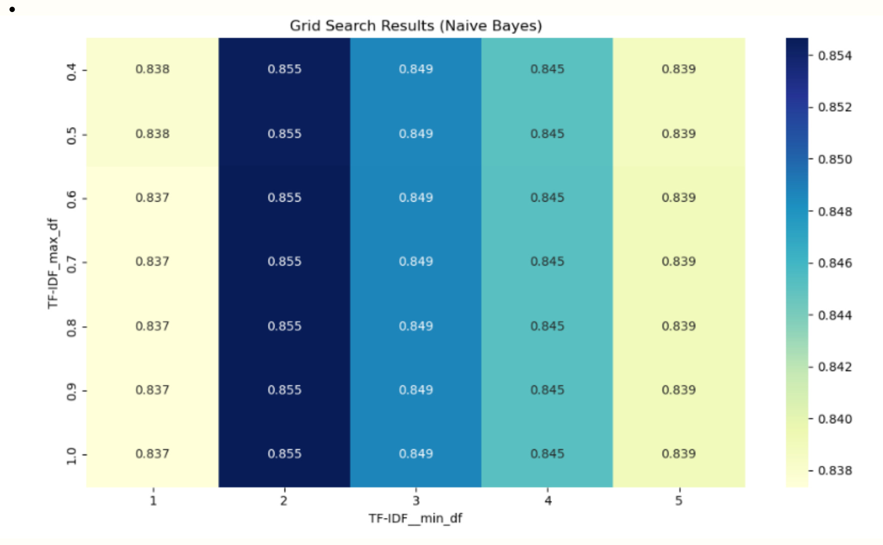


criteria, including an early stopping delta, metric, and patience [15]. Additionally, the frequency of validation runs during training steps was set to every 32 steps, even though this does not align with the batch size of 46, this measure was taken to ensure stability during training, furthermore aligning these two values is not necessary for effective optimization. Furthermore, the frequency of validation runs during training steps was set to every 32 steps. Although the frequency of validation runs does not precisely match the batch size (46), this discrepancy does not compromise the stability of training. Moreover, strict alignment between these values is not essential for achieving effective optimization. Finally, the 'roberta-base' model is specifically employed as a classification model [12].

Naive Bayes (Parameter Tuning)

To optimize for the best hyperparameters, a grid search was implemented. This process explores several hyperparameter settings for the TF-IDF vectorizer and Multinomial Naive Bayes classifier [10][16]. For the TF-IDF vectorizer the following three hyperparameters were taken into consideration:

- *Maximum Document Frequency (max_df)*: This parameter controls how common a word can be before being taken out of consideration. This measure helps filtering out stop words while also maintaining enough context. The higher the max_df is set the more stop words will be present. A lower max_df focuses on words that appear in a smaller subset of documents, which could highlight more specific themes. Setting the max_df too low however could also filter out frequent words that do provide significant context.
- *Minimum Document Frequency (min_df)*: This parameter determines a threshold for how rare a word can be and still be considered a feature. Setting the min_df higher secludes words that only appear in a few instances. Lowering the min_df allows rarer words which could reveal unique characteristics of a text. This could also lead to more noise however.
-



- *N-gram Range*: This parameter determines how many consecutive words will be considered as a single feature. Using only single words will focus on the individual word importance. Allowing bigrams or higher N's enables the model to consider multi- word phrases as a unit, which could potentially improve classification for terms that depend on each other to provide the correct context.

Finally, concerning the Multinomial Naive Bayes classifier tuning, the *alpha smoothing parameter* was adjusted. A higher alpha reduces the influence of unseen words on classification while a lower alpha value gives more weight to unseen features, which could improve the performance when dealing with unseen words in new text. This comes at the cost of risking overfitting however. Finding the right balance was key to stabilizing performance.

The grid search tested different values for max_df (0.4 ,0.5, 0.6, 0.7, 0.8, 0.9, 1.0), min_df 1, 2, 3 ,4 ,5), N-gram ranges ((1, 1), (1, 2), (1,3), (1,4), (1,5)), and smoothing parameters for the Naive Bayes classifier (0.1, 0.25, 0.5, 0.75, 1.0). This setup aimed to configure the best hyperparameters for text feature extraction and classification within this model. After applying these to a 10-fold cross validation, the resulting recommended hyperparameters were the following: {'nb__alpha': 0.25, 'tfidf__max_df': 0.6, 'tfidf__min_df': 2, 'tfidf__ngram_range': (1, 3)}.

Results

```
RoBERTa       precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       0.67      1.00      0.80         4
           2       1.00      0.33      0.50         3

    accuracy                           0.80        10
   macro avg       0.89      0.78      0.77        10
weighted avg       0.87      0.80      0.77        10
```

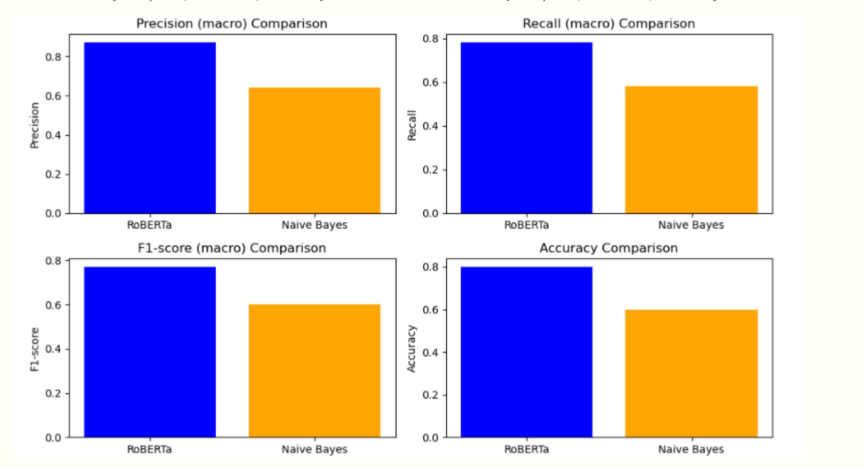
Classification Report RoBERTa (0 = Sports, 1 = Movie, 2 = Book)

```
Naive Bayes   precision    recall  f1-score   support

           0       1.00      0.67      0.80         3
           1       0.60      0.75      0.67         4
           2       0.33      0.33      0.33         3

    accuracy                           0.60        10
   macro avg       0.64      0.58      0.60        10
weighted avg       0.64      0.60      0.61        10
```

Classification Report Naive Bayes (0 = Sports, 1 = Movie, 2 = Book)



Discussion & Analysis

The comparison between the RoBERTa and Naive Bayes models in the topic classification task reveals interesting insights into their performance and characteristics. Somewhat expected, the RoBERTa model outperforms the Naive Bayes model on precision, recall and F1-score for each class.

For the sports topic (class 0) RoBERTa manages to attain a perfect score for precision and recall. Naive Bayes also does well on this class albeit not as good as its competitor, with a precision of 1.00 and a recall of 0.67. This shows it had a slightly harder time capturing all instances of this class. Overall, both models scored very well on this particular class. This could be explained by the fact that the sports class is probably more distinct from the other two classes than movies and books are from each other. These latter two classes, being closely related, will probably share more characteristics with each other, making it harder to differentiate between them.

Continuing to the movie topic (class 1), RoBERTa seems to struggle slightly in correctly identifying all instances, but at the same time does manage to find all instances belonging to this class. This is reflected in a precision of 0.67 and a recall of 1.00. Naive Bayes on the other hand, has a harder time with correctly identifying and capturing all instances of this class, reflected in a precision of 0.60 and a recall of 0.75.

Lastly, the most challenging topic for both models to classify correctly appeared to be the book topic (class 2). While RoBERTa maintains strong performance in correctly identifying the class, it falls short in capturing all instances, as reflected in a precision of 1.00 and a recall of only 0.33. Naive Bayes performs even worse, scoring only 0.33 for both precision and recall.

In conclusion, the comparison highlights the impact of RoBERTa's superior architecture and sophistication. While Naive Bayes offers a straightforward and easy to interpret solution to topic classification it cannot compare to RoBERTa's extensive and nuanced training method. This seems to underscore the importance of selecting appropriate machine learning architectures based on the level of complexity of the task.

Limitations

While the comparison between RoBERTa and Naive Bayes provides valuable insights, there are certain aspects that could be improved for future research. Firstly, the evaluation was conducted on a small dataset. Expanding the size of the test data could yield a more precise evaluation of both models' performance. With a larger dataset, it becomes easier to discern overarching trends, thereby enhancing the generalizability of the results. Therefore, future studies could benefit from leveraging larger datasets to ensure more robust and comprehensive evaluations. Additionally, fine-tuning and experimenting with different hyperparameter configurations could yield more valuable insights into the models' sensitivities and optimal configurations. Lastly, exploring additional feature engineering or representations of the text data could further enhance the performance of the models. By incorporating these improvements into future studies, further advancements can be made in the field of topic classification.

Sentiment

Data

Selection and Description

Test Data

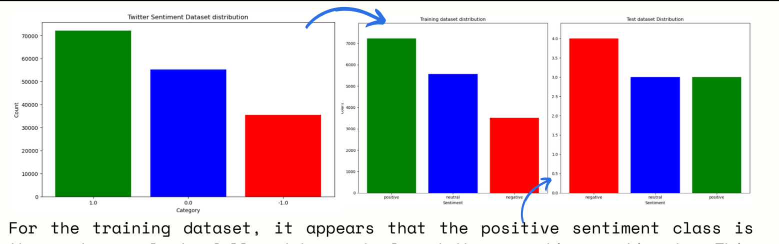
The language observed in the sentences of the test data seemed to be informal, resembling social media posts(or similar platforms), highlighted by the use of words such as 'HOOKED', 'stronggg', or phrases like 'I was about to throw hands', along with various spelling errors. For effective machine learning model training, it is important to use a dataset that is not only sufficiently large for meaningful results but also shares similar characteristics with the test data [18]. This approach is meant to provide the basis for achieving more accurate and generalizable models.

Training Data

The selected dataset for model training is the 'Twitter Sentiment Dataset', which captures posts from users on Twitter regarding a variety of topics. It consists of approximately 160,000 tweets, categorized into two distinct fields: the polarity of the tweet (negative (-1), neutral (0), and positive (+1)) and the textual content of the tweet [19].

Preprocessing (Initial)

To ensure consistency between the training and test data, the column names in the Twitter Sentiment Dataset were updated to match those of the test set. During the inspection of the Twitter dataset, it was observed that the positive sentiment class (1.0) is the most prevalent, followed by neutral (0), and then negative (-1) sentiments. Accordingly, the numeric values in the sentiment column were mapped to their corresponding string values. Additionally, NaN values were detected in the dataset, 4 in the 'clean_text' column and 7 in the 'category' column. These were addressed by replacing the NaN values with empty strings, to ensure no data point was excluded due to missing values. To streamline the training process and manage resources effectively, only a portion of the data set was selected for training(a random 10% of the rows from the original dataset, with a specified seed for reproducibility). Regarding the test dataset, minimal preprocessing was required at this stage; the 'topic' column was removed as it was deemed irrelevant to this sentiment analysis.



For the training dataset, it appears that the positive sentiment class is the most prevalent, followed by neutral and then negative sentiments. This indicates a relatively imbalanced training dataset, with a higher proportion of positive instances. This imbalance could potentially influence the performance of the models, as they might become more prone to predict positive sentiments due to their prevalence in the training data. For the test dataset, given the small size of the test dataset depicted here, each class is almost equally represented. However, with the small test set, the value of the performance metrics may be limited, making it hard to generalize the model's performance.

Methods

The sentiment analysis is conducted at the sentence level using SVM. The model is assessed using various approaches to this task, including different preprocessing steps and model parameters, Finally, an analysis of the results is performed to identify potential areas for improvement and to better understand the model's performance characteristics.

Motivation

The application of Support Vector Machines (SVMs) in sentiment analysis has been well-documented across various datasets, including those derived from Twitter. The ability of SVMs to classify sentiments from such data with seemingly consistent high precision and recall makes them a suitable choice for this analysis [20]. Although the paper 'Application of Support Vector Machine (SVM) in the Sentiment Analysis of Twitter DataSet' is referenced here, it is important to clarify that the experiments conducted for this sentiment analysis, utilizing a different dataset, will not replicate those, particularly regarding experiments with different kernel functions. On the contrary, the kernel function for the experiments conducted in this sentiment analysis will be kept as a linear kernel, which is reported to be effective for text classification tasks [21]. This allows efforts to be directed towards fine-tuning the model and preprocessing steps for optimizing performance on the specific dataset.

Experiment outline

Preprocessing Steps(applied to both the training and test data):
To enable a machine to understand and derive meaning from texts, it is important to convert the text into a form that the machine can interpret [22]. the following are used to achieve the properties to do this:

- The text is split into individual words using a tokenizer.
- TF-IDF vectorizer is chosen, because it not only counts the word frequency but also adjusts for the word's importance across all the texts by penalizing common words, providing a more nuanced representation of texts[22].
- Stop word removal can help control the noise in the text data by eliminating common words that might not be contributing much [23].
- Lemmatization can be used to identify the lemmas (dictionary forms) of words in context [24].

Model Parameters:

- Kernal: As previously mentioned, the kernel is kept as a 'linear' kernel.
- The regularization parameter (C) can be used to control the trade-off between increasing the margin of the decision boundary from the classes (or support vectors) and optimizing for the highest number of accurately classified points within the training dataset [25]. It can be explored using grid search with cross-validation to try and avoid overfitting and to help the model generalize well to unseen data [26].

Analysis:

- In total four final configurations of SVMs will be compared to each other based on precision, recall and f1-score.
  - Without stopwords and without lemmatization (with the best C value)
  - Without stop words but with lemmatization (with the best C value)
  - With stop words but without lemmatization (with the best C value)
  - with stop words and with lemmatization (with the best C value)

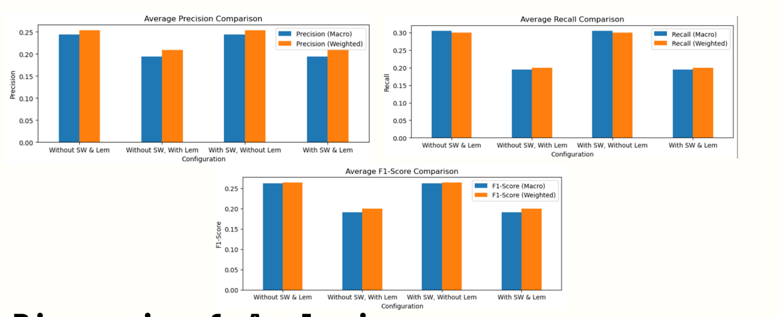
Results

```
Without stopwords and without lemmatization,with best C: 5
              precision  recall  f1-score  support
negative           0.33    0.25      0.29      4.0
neutral            0.40    0.67      0.50      3.0
positive           0.00    0.00      0.00      3.0
accuracy           0.30    0.30      0.30      0.3
macro avg          0.24    0.31      0.26     10.0
weighted avg       0.25    0.30      0.26     10.0
```

```
With stop words but without lemmatization, with best C: 5
              precision  recall  f1-score  support
negative           0.33    0.25      0.29      4.0
neutral            0.40    0.67      0.50      3.0
positive           0.00    0.00      0.00      3.0
accuracy           0.30    0.30      0.30      0.3
macro avg          0.24    0.31      0.26     10.0
weighted avg       0.25    0.30      0.26     10.0
```

```
Without stop words but with lemmatization, with best C: 5
              precision  recall  f1-score  support
negative           0.33    0.25      0.29      4.0
neutral            0.25    0.33      0.29      3.0
positive           0.00    0.00      0.00      3.0
accuracy           0.20    0.20      0.20      0.2
macro avg          0.19    0.19      0.19     10.0
weighted avg       0.21    0.20      0.20     10.0
```

```
With stop words and with lemmatization, with best C: 5
              precision  recall  f1-score  support
negative           0.33    0.25      0.29      4.0
neutral            0.25    0.33      0.29      3.0
positive           0.00    0.00      0.00      3.0
accuracy           0.20    0.20      0.20      0.2
macro avg          0.19    0.19      0.19     10.0
weighted avg       0.21    0.20      0.20     10.0
```



Discussion & Analysis

-With the configuration '**Without Stopwords and Without Lemmatization (Best C: 5)**' the model performed best for the neutral class with an f1 score of 0.50 and it performed the worst for the positive class(f1:0.00).
-With the configuration '**Without Stop Words but With Lemmatization (Best C: 5)**' the model performed (tied) best for the negative and neutral class with an f1 score of 0.29 and it performed the worst for the positive class(f1:0.00).
-With the configuration '**With Stop Words but Without Lemmatization (Best C: 5)**' the model performed (tied) best for the neutral class with an f1 score of 0.50 and it performed the worst for the positive class(f1:0.00).
-With the configuration '**With Stop Words and With Lemmatization (Best C: 5)**' the model performed (tied) best for the negative and neutral class with an f1 score of 0.29 and it performed the worst for the positive class(f1:0.00).

The lemmatization and the inclusion of stopwords do not appear to improve the model's overall performance. In fact, configurations without lemmatization regardless of stopwords removal seem to perform slightly better. Across all classes, the models struggle to achieve notable success, with the most effective performance observed in the recall for the neutral class without lemmatization. This suggests that models are better at classifying actual 'neutral' sentiments when lemmatization is omitted, regardless of stopwords removal. Notably, none of the models were able to correctly classify positive sentiments, as indicated by all the scores for positive sentiments being 0.00 across all configurations. This could be due to a combination of factors, including overfitting on the training data leading to poor generalization; the way sentiments are expressed might differ between the training and test datasets, despite the apparent similarity in data sources; or limitations in the feature representation (TF-IDF) that might not capture the nuances necessary to distinguish positive sentiments effectively. Furthermore, all precision scores being below 0.50, indicates that the models frequently misclassify non-relevant sentiments as relevant across all categories.
Across all configurations, the macro and weighted averages are notably low, ranging from 0.19 to 0.31. This indicates that the models uniformly underperform across all sentiment classes, demonstrating no particular bias towards the more frequently represented classes in the dataset.

Limitations

The training set could benefit from being larger and more balanced. A larger dataset might help in capturing the array of sentiment expressions better, and a more balanced distribution of classes could improve model fairness. Additionally, The lack of clarity regarding the annotation process and criteria raises concerns about the consistency and reliability of the labels in the training dataset, due to the subjective nature of sentiment analysis [27]. Experimenting with different or additional feature representations e.g word embeddings could enhance a model's ability to grasp nuanced semantic relationships. Furthermore, a more extensive exploration of the 'C' parameter and the use of cross-validation was limited due to computational constraints. Lastly, Investigating specific instances at the token or word level where the model failed to correctly classify positive sentiments could reveal insights into which words or phrases were overlooked or misinterpreted.

References

[1] "conll2012_ontonotesv5 at main." https://huggingface.co/datasets/conll2012_ontonotesv5/tree/main
[2] I. Markov, 'Lab4' GitHub Repository, cltl/ba-text-mining, Lab Session lab4, Available at:https://github.com/cltl/ba-text-mining/tree/master/lab_sessions/lab4
[3] VU Text Mining for AI, "tm-ba-lecture-5a-nerc.pdf", Ilia Markov
[4] "sklearn-crfsuite – sklearn-crfsuite 0.3 documentation." https://sklearn-crfsuite.readthedocs.io/en/latest/
[5] I. Markov, 'Lab1' GitHub Repository, cltl/ba-text-mining, Lab Session lab1, Available at:https://github.com/cltl/ba-text-mining/tree/master/lab_sessions/lab1
[6] "Treebank-3 - Linguistic Data Consortium." https://catalog.ldc.upenn.edu/LDC99T42
[7] Tirendaz Academy, "FIFA World Cup 2022 Tweets," in Kaggle, 2022. [Online]. Available: https://www.kaggle.com/datasets/tirendazacademy/fifa-world-cup-2022-tweets.
[8] Lakshmi N, "Sentiment Analysis of IMDb Movie Reviews," in Kaggle, 2022. [Online]. Available: https://www.kaggle.com/code/lakshmi25npathi/sentiment-analysis-of-imdb-movie-reviews/input?select=IMDB+Dataset.csv.
[9] Shruti Mehta, "Amazon Book Reviews (Webscraped)," in Kaggle, 2022. [Online]. Available: https://www.kaggle.com/datasets/shrutimehta/amazon-book-reviews-webscraped?resource=download&select=Reviews.csv.
[10] scikit-learn. "Naive Bayes," in scikit-learn: Machine Learning in Python, [Online]. Available: https://scikit-learn.org/stable/modules/naive_bayes.html.
[11] Hugging Face. "RoBERTa: A Robustly Optimized BERT Approach," in Hugging Face: Transformers, [Online]. Available: https://huggingface.co/transformers/model_doc/roberta.html.
[12] Hugging Face. "BERT: Bidirectional Encoder Representations from Transformers," in Hugging Face: Transformers, [Online]. Available: https://huggingface.co/transformers/model_doc/bert.html.
[13] I. Markov, "Week 6: Text Categorisations and Topic Modeling," in Vrije Universiteit Amsterdam Canvas, [Online]. Available: https://canvas.vu.nl/courses/75225/pages/week-6-text-categorisations-and-topic-modeling?module_item_id=1209086.
[14] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to Fine-Tune BERT for Text Classification?" Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, School of Computer Science, Fudan University, 825 Zhangheng Road, Shanghai, China.
[15] I. Markov, "Topic Classification with BERT," GitHub Repository, cltl/ba-text-mining, Lab Session 6.4, Available at: https://github.com/cltl/ba-text-mining/blob/master/lab_sessions/lab6/Lab6.4-Topic-classification-BERT.ipynb.
[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
[17] Kaggle. (n.d.). Kaggle. Retrieved Jan. 20, 2024, from https://www.kaggle.com/
[18] H. Smolic, "Training Data vs. Test Data in Machine Learning – Essential Guide," Medium, Sep. 22, 2022. [Online]. Available: https://pub.towardsai.net/training-data-vs-test-data-in-machine-learning-essential-guide-c58404849cea
[19] "Twitter Sentiment Dataset," Kaggle, May 14, 2021. https://www.kaggle.com/datasets/saurabhshahane/twitter-sentiment-dataset/data
[20] K. Han, W. Chien, C. Chiu, and Y. Cheng, "Application of Support Vector Machine (SVM) in the sentiment analysis of Twitter DataSet," Applied Sciences, vol. 10, no. 3, p. 1125, Feb. 2020, doi: 10.3390/app10031125.
[21] A. Kowalczyk, "Linear Kernel: Why is it recommended for text classification ?," SVM Tutorial, Jul. 26, 2020. https://www.svm-tutorial.com/2014/10/svm-linear-kernel-good-text-classification/
[22] I. Markov, "Machine learning basics" GitHub Repository, cltl/ba-text-mining, Lab Session 2.1, Available at:https://github.com/cltl/ba-text-mining/blob/master/lab_sessions/lab2/Lab2.1.ml.basics.ipynb
[23] VU Text Mining for AI, "tm-ba-lecture-2-linguistics-nlp.pdf", Ilia Markov
[24] I. Markov, "Introduction to Natural Language Processing tools " GitHub Repository, cltl/ba-text-mining, Lab Session 1.1, Available at:https://github.com/cltl/ba-text-mining/blob/master/lab_sessions/lab1/Lab1.1-introduction.ipynb
[25] S. Yıldırım, "Hyperparameter tuning for support vector machines — C and gamma parameters," Medium, Dec. 14, 2021. [Online]. Available: https://towardsdatascience.com/hyperparameter-tuning-for-support-vector-machines-c-and-gamma-parameters-6a5097416167
[26] N. Beheshti, "Cross validation and grid search - towards data science," Medium, Feb. 12, 2022. [Online]. Available: https://towardsdatascience.com/cross-validation-and-grid-search-efa64b127c1b
[27] VU Text Mining for AI, "tm-ba-lecture-4-sentiment.pdf", Ilia Markov