Btnlp Final Project Report

Yu-Ci Chen
yuch00003@uni-saarland.de

Ruveyda Betül Bahceci
ruba00002@uni-saarland.de

1 Introduction

Sentiment analysis extracts opinions and emotions from textual data, categorizing them into predefined classifications such as positive, negative, or neutral (Haddi et al., 2013; Bhoir and Kolte, 2015; Neri et al., 2012), while also assessing polarity (Prabowo and Thelwall, 2013) and subjectivity (Caffi and Janney, 1994).

The accuracy of sentiment analysis relies heavily on the quality of the underlying data, highlighting the critical importance of preprocessing text data (Krouska et al., 2016; Singh and Kumari, 2016; Nhlabano and Lutu, 2018; Uysal and Gunal, 2014). Preprocessing techniques such as lowercasing, stemming/lemmatization, and stop-word removal are crucial steps that refine raw text, extract meaningful features, improving the data quality, and ultimately enhancing the performance of sentiment analysis tasks.

In this paper, a comparative study on sentiment analysis tasks executed using NLTK/VADER (Elbagir and Yang; Pai et al., 2022; Budianto et al., 2022), SpaCy/Textblob (Susrama et al., 2021; Khan et al., 2021), and HuggingFace (Devi and Nayyar, 2024). We analyzed the results obtained from two versions of code. The impact of preprocessing steps, and sentiment analysis methods on the performance was assessed through confusion matrices generated by all three models.

2 Dataset

The sentiment analysis dataset consists of Twitter texts categorized into four labels: positive, neutral, negative, and irrelevant.

3 Methodology

In this section, we outline the methodology employed to preprocess the Twitter dataset for the sentiment analysis. Given that data collected from social media often have informal and ambiguous language, abbreviations, and misspellings, preprocessing is crucial to enhance the quality of the data and improve the performance of the sentiment analysis task.

3.1 Preprocessing

The preprocessing steps consist of two phases: preliminary data cleaning and advanced data cleaning. Preliminary data cleaning involves essential steps to prepare the data for further processing; advanced data cleaning involves more sophisticated techniques to further refine the text data.

3.1.1 Preliminary Data Cleaning

- Address Missing Values

  Missing values in the Text Column are replaced with "unknown" to maintain data integrity and prevent biases that may arise from incomplete data.

- Lowercasing

  Texts are lowercased to ensure consistency. Since in most cases the case of words do not carry semantic meaning, lowercasing helps to focus on the content rather than the formatting (Pahwa et al., 2018).

- Remove Non-ASCII Characters

  Non-ASCII characters are removed and replaced with whitespace to standardize text representation and ensure compatibility with standard text processing tools.

- Remove Additional Whitespace (Stripping)

  Stripping whitespace standardizes text representation and aids in tokenization by ensuring the correct identification of tokens, as whitespaces are commonly treated as a delimiter for splitting text into words (Pahwa et al., 2018).

3.1.2 Advanced Data Cleaning

- Remove "\n" and Other Symbols Followed by Letters

  Symbols like "\n" and others followed by letters are removed as they do not contribute semantic meaning and may disrupt tokenization.

- Remove Emojis and Emoticons

  Emojis and emoticons are excluded from the analysis as they do not contribute textual content, which is the focus of sentiment analysis. Removing them ensures that only relevant textual information is considered.

- Fix Spelling Errors

  Spelling mistakes are corrected using TextBlob to enhance the accuracy of sentiment analysis results. This step helps in improving the quality of the text data.

- Remove Stopwords

  Common stopwords like "is" and "the", are eliminated using NLTK, SpaCy, TextBlob, and Scikit-learn to reduce noise in the text data, focusing the analysis on meaningful content (Pahwa et al., 2018).

- Stemming and Lemmatization

  NLTK's Porter stemmer and WordNet lemmatizer are employed to normalize words by reducing them to their base or root form, decreasing vocabulary size, and improving NLP task performance.

- Remove Numbers and Non-Alphabetic Words

  Numeric data, such as user IDs, timestamps, and email addresses; and non-alphabetic words, such as punctuation marks do not contribute much to semantic meanings. Therefore, they are removed to further normalize the text data and avoid noise.

Overall, the text preprocessing steps focus on normalizing the text data, reducing noise, and removing information that does not have contextual meanings. The preprocessing enhances the quality of information extracted for the sentiment analysis task, resulting in better sentiment analysis performance.

4 Results

The sentiment analysis results with Version 1 and Version 2 are displayed in Figure 1 and Figure 2, respectively.

4.1 NLTK/VADER

Version 1

- Negative (class 0): Correctly classified: 11,939/22,542. Misclassifications: 5,071 (as Class 1), 5,532 (as Class 2).
- Neutral (class 1): Correctly classified: 5,062/18,318. Misclassifications: 5,849 (as Class 0), 7,407 (as Class 2).
- Positive (class 2): Correctly classified: 11,911/20,831. Misclassifications: 3,858 (as Class 0), 5,062 (as Class 1).

Version 2

- Negative (class 0): Correctly classified: 12,367/22,542. Misclassifications: 4,087 (as Class 1), 6,088 (as Class 2).
- Neutral (class 1): Correctly classified: 3,995/18,318. Misclassifications: 6,215 (as Class 0), 8,108 (as Class 2).
- Positive (class 2): Correctly classified: 13,839/20,831. Misclassifications: 3,500 (as Class 0), 3,492 (as Class 1).

Version 1 shows relatively balanced performance across classes. However, misclassifications are notable, particularly in neutral and positive classes. Version 2 struggles with neutral sentiment, indicating potential overfitting or bias towards extreme sentiments.

4.2 SpaCy/Textblob

Version 1

- Negative (class 0): Correctly classified: 9,738/22,542. Misclassifications: 7,875 (as Class 1), 4,929 (as Class 2).
- Neutral (class 1): Correctly classified: 7,175/18,318. Misclassifications: 4,516 (as Class 0), 6,627 (as Class 2).
- Positive (class 2): Correctly classified: 10,131/20,831. Misclassifications: 3,854 (as Class 0), 6,846 (as Class 1).

Version 2

- Negative (class 0): Correctly classified: 9,904/22,542. Misclassifications: 7,665 (as Class 1), 4,973 (as Class 2).
- Neutral (class 1): Correctly classified: 6,812/18,318. Misclassifications: 4,069 (as Class 0), 7,437 (as Class 2).
- Positive (class 2): Correctly classified: 12,494/20,813. Misclassifications: 2,825 (as Class 0), 5,512 (as Class 1).

Version 1 displays similar patterns to NLTK/VADER, with noticeable misclassifications across all classes. However, the performance is relatively consistent across sentiments. Version 2 suffers from significant misclassifications in neutral sentiments, similar to NLTK/VADER Version 2.

4.3 HuggingFace

Version 1

- Negative (class 0): Correctly classified: 19,809/22,542. Misclassifications: 2,733 (as Class 2), none as Class 1.
- Neutral (class 1): None correctly classified out of 18,318. Misclassifications: 13,739 (as Class 0), 4,579 (as Class 2).
- Positive (class 2): Correctly classified: 10,299/20,831. Misclassifications: 10,532 (as Class 0), none as Class 1.

Version 2

- Negative (class 0): Correctly classified: 19,782/22,542. Misclassifications: 2,760 (as Class 2), none as Class 1.
- Neutral (class 1): None correctly classified out of 18,318. Misclassifications: 12,268 (as Class 0), 6,050(as Class 2).
- Positive (class 2): Correctly classified: 12,639/20,831. Misclassifications: 8,192 (as Class 0), none as Class 1.

The HuggingFace model implemented in Version 1 and Version 2 shows advantages in classifying negative and positive sentiments, with minimal misclassifications. However, they fail to correctly classify neutral sentiments, with none of the instances correctly identified.

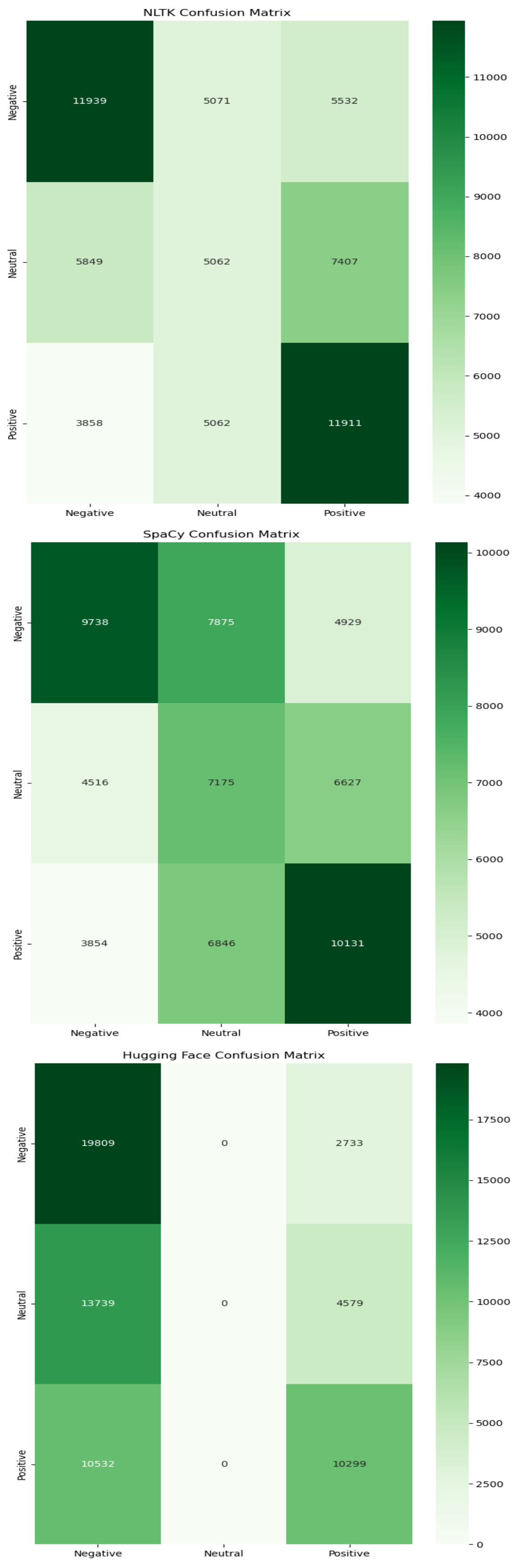

Figure 1: Confusion Matrices with Version 1

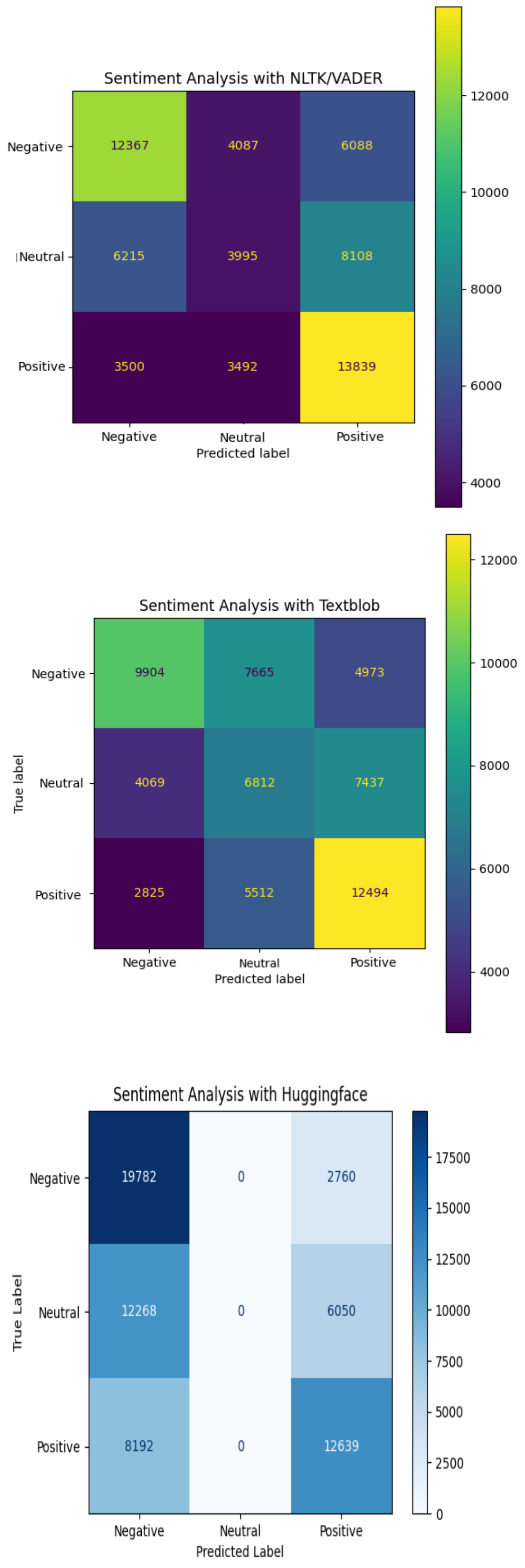

Figure 2: Confusion Matrices with Version 2

5 Discussion and Conclusion

5.1 Comparison of the Three Techniques for Sentiment Analysis

Version 1 highlights the superior performance of HuggingFace models in accurately classifying positive and negative instances, surpassing NLTK/VADER and SpaCy/TextBlob. While NLTK/VADER and SpaCy/TextBlob exhibit comparable trends in both Version 1 and Version 2, nuances exist in their misclassification patterns. NLTK/VADER rely on lexicon-based methods, assigning sentiment scores to individual words using predefined dictionaries. Conversely, SpaCy/TextBlob, while capable of leveraging machine learning models, typically employ simpler rule-based or statistical techniques. These distinctions likely account for the differences in misclassification patterns. However, both NLTK/VADER and SpaCy/TextBlob may yield similar results as they consider word polarity. The superior performance of HuggingFace models in sentiment analysis can be attributed to their utilization of attention mechanisms inherent in Transformer-based architectures, along with directionality, allowing them to capture contextual information more effectively (Wolf et al., 2020).

5.2 Comparison of the Two Preprocessing Versions

While we conducted similar preprocessing steps in both Version 1 and Version 2, we implemented them in different orders. In Version 1, stripping and removal of non-ASCII characters occurred early in the process, whereas Version 2 implemented these steps later on. Notably, in Version 1, tokenization of the text was postponed until the sentiment analysis stage. Conversely, Version 2 employed tokenization post-stemming and lemmatization. Furthermore, Version 1 utilized stemmed text for subsequent preprocessing steps, whereas Version 2 employed lemmatized text. These differences in preprocessing order and techniques may explain the observed discrepancies in sentiment analysis results. Additionally, in Version 2, instances labeled as "irrelevant" were excluded, thereby reducing dataset noise and potentially contributing to its slightly superior performance.

5.3 The Classification of Neutral Sentiments

During the sentiment analysis, we observed that all versions and sentiment analysis techniques performed better at classifying negative and positive sentiments compared to neutral sentiments. This could be interpreted as a challenge in identifying neutrality. The models may have been trained on heavily polarized datasets, resulting in an imbalance when analyzing neutral sentiments. The subjectivity and context dependency of neutral sentiments could make it harder for them to be classified correctly. As neutral sentiments exhibit greater ambiguity and context-dependency, analyzing them along with the context could improve performance. Otherwise, slightly positive or slightly negative sentiments could be misclassified, as seen in Figure 1 and Figure 2.

References

Purtata Bhoir and Shilpa Kolte. 2015. Sentiment analysis of movie reviews using lexicon approach. In *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pages 1–6.

A. G. Budianto, B. Wirjodirdjo, I. Maflahah, and D. Kurnianingtyas. 2022. Sentiment analysis model for klikindomaret android app during pandemic using vader and transformers nltk library. In *2022 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 0423–0427.

Claudia Caffi and Richard W. Janney. 1994. Toward a pragmatics of emotive communication. *Journal of Pragmatics*, 22:325–373.

Ajantha Devi and Anand Nayyar. 2024. *Decoding product sentiments: Unraveling reviews with explainable analysis using Hugging-Face transformer*, pages 173–199.

Shihab Elbagir and Jing Yang. Analysis using natural language toolkit and vader sentiment.

Emma Haddi, Xiaohui Liu, and Yong Shi. 2013. The role of text pre-processing in sentiment analysis. In *International Conference on Information Technology and Quantitative Management*.

Rashid Khan, Furqan Rustam, Khadija Kanwal, Arif Mehmood, and Gyu Sang Choi. 2021. Us based covid-19 tweets sentiment analysis using textblob and supervised machine learning algorithms. In *2021 International Conference on Artificial Intelligence (ICAI)*, pages 1–8.

Akrivi Krouska, Christos Troussas, and Maria Virvou. 2016. The effect of preprocessing techniques on twitter sentiment analysis. In *2016 7th International Conference on Information, Intelligence, Systems Applications (IISA)*, pages 1–5.

Federico Neri, Carlo Aliprandi, Federico Capeci, Montserrat Cuadros, and Tomas By. 2012. Sentiment analysis on social media. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 919–926.

V.V. Nhlabano and P.E.N. Lutu. 2018. Impact of text pre-processing on the performance of sentiment analysis models for social media data. In *2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD)*, pages 1–6.

Bhumika Pahwa, Taruna S., and Neeti Kasliwal. 2018. Sentiment analysis- strategy for text pre-processing. *International Journal of Computer Applications*, 180:15–18.

Aiswarya R Pai, Maria Prince, and C V Prasannakumar. 2022. Real-time twitter sentiment analytics and visualization using vader. In *2022 2nd International Conference on Intelligent Technologies (CONIT)*, pages 1–4.

Rudy Prabowo and Mike Thelwall. 2013. Sentiment analysis: A combined approach. *Journal of Informetrics*, pages 143–157.

Tajinder Singh and Madhu Kumari. 2016. Role of text pre-processing in twitter sentiment analysis. *Procedia Computer Science*, 89:549–554.

Gede Susrama, Ni Mandenni, Mohammad Fachrurrozi, Sunu Ilham Pradika, Kholilul Manab, and Nyoman Sasmita. 2021. Twitter sentiment analysis as an evaluation and service base on python textblob. *IOP Conference Series: Materials Science and Engineering*, 1125:012034.

Alper Kürşat Uysal and Serkan Gunal. 2014. The impact of preprocessing on text classification. *Information Processing Management*, 50:104 – 112.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.