NLP Final Project

Introduction:

For the final project, I would like to create a text summarization model that also accurately captures the sentiment of the text. Which means that, ideally, the sentiment analysis of the summarized text should have a similar or even better performance than the sentiment analysis performed on the entire text. I chose this topic because both sentiment and the context of the sentiment are useful for consumers to determine if they want to purchase the product based on the review, and not all text summarization does a good job capturing the sentiment of the text.

Literature Review:

Text summarization primarily employs two strategies: abstractive and extractive (Tomer and Kumar 2022). Abstractive summarization involves rewriting the text in one's own words, drawing on their understanding of the content. This approach aims to capture the essence of the original text, though it expresses ideas differently, making it more complex for machines to generate summaries that maintain linguistic coherence. Additionally, abstractive methods may lead to misleading or biased results. As a result, many organizations are favoring extractive summarization techniques, which select and compile key sentences from the original document to form a summary. Traditional extractive methods include TF-IDF (Term Frequency-Inverse Document Frequency) (Luhn 1957; Sparck Jones 1972) and TextRank (Mihalcea and Tarau 2004), which use statistical and graph-based approaches. Neural network-based techniques such as BERT (Devlin et al. 2018) have emerged as leading innovations in this area.

Sentiment analysis of natural language texts is challenging due to inherent ambiguities and the need to process large volumes of data. It can be approached using either lexical databases, such as SentiWordNet, which provides sentiment scores for words based on their parts of speech, or machine learning techniques that learn sentiment patterns from tagged data. SentiWordNet assigns positive and negative scores to words, aiding in document sentiment determination. Applications of sentiment analysis include social media monitoring, financial trading, and political analysis. Recent research has applied SentiWordNet to various domains, such as movie reviews and political opinion mining, while others have explored sentiment-focused text summarization and extraction, highlighting the complexity of generating accurate summaries from single or multiple documents (Guerini and Gatti et al, 2013).

Data:

The dataset I am using for this project is the Amazon Kindle Book Review Data. It is a small subset of the dataset of Book reviews from the Amazon Kindle Store category. 5-core dataset of product reviews from Amazon Kindle Store category from May 1996 - July 2014. The small version of the data contains a total of 12000 entries. Each reviewer has at least 5 reviews and each product has at least 5 reviews in this dataset. For simplicity, I created the sentiment label for the reviews by labeling review score 3 and above as positive, and below 3 as negative. The positive sentiment is represented by 1 and the negative sentiment is represented by 0, there are 8000 positive comments and 4000 negative comments after the transformation. Then I divided the data into 80% training data and 20% testing data for training the sentiment analysis model.

Methodology:

The initial step of sentiment analysis cleans up raw reviews by removing complex and unwanted content. It starts with tokenization and ends with removing meaningless words, digits, and words with fewer than three characters. Tokenization divides review sentences into tokens, which are then reduced to their singular forms through stemming. Stop words, special characters, dates, trivial words, and URLs are also removed. This process significantly reduces the dimensional space of the raw reviews. After the initial data cleaning, I trained a TF-IDF sentiment analysis model on the training data and got the accuracy of the model from the original cleaned test data. This model would be the one I used throughout the entire project to ensure the consistency of the sentiment analysis. I applied different summarization methods on the test data to evaluate how much sentiment related information was lost through summarization.

For the first step of text summarization, I picked one extractive model and one abstractive model to compare their performance. I chose the LexRank algorithm from the Sumy library for the extractive model and only kept 2 sentences per review. Sumy works by analyzing the input text and extracting the most important sentences based on the selected algorithm, helping to create concise summaries of larger documents. Then I used the pre-trained T5-small (Text-To-Text Transfer Transformer) model for abstractive summarization and kept 10 to 100 words from each review. T5-small is a compact version of Google's T5 model designed for a variety of natural language processing tasks, including abstractive text summarization. Unlike extractive summarization, abstractive summarization generates new sentences that capture the essence of the input text, potentially using different words and phrases.

After comparing the performance of the extractive and abstractive summarization, the abstractive summarization was very computationally expensive and did not improve the

sentiment analysis score. So I used the extractive summarization to explore a model with better performance.

To improve the performance of the sentiment analysis accuracy from extractive summarization of the reviews, I began by splitting each review into individual sentences. Next, I use NLTK sentiment analyzer to calculate the polarity scores for each sentence, identifying those that are clearly positive or negative. I filtered out neutral sentences to focus only on the most sentimentally charged content. The remaining sentences are then summarized using the Sumy library to produce a concise version of the reviews. Finally, I preprocessed the summarized text and evaluated the sentiment analysis performance. The intuition behind the method is that Sumy chooses sentences that are not significant for sentiment analysis sometimes, and by eliminating the neutral sentences we can get a better representation of the sentiment before the summarization.

Results:

Original:

```
TF-IDF Model Accuracy: 0.8491666666666666
TF-IDF Model Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.67      0.75       799
           1       0.85      0.94      0.89      1601

    accuracy                           0.85      2400
   macro avg       0.85      0.81      0.82      2400
weighted avg       0.85      0.85      0.84      2400
```

The TF-IDF sentiment analysis model has a 84.92% accuracy for the original testing data. The recall score indicates that the model is less good at identifying negative sentiment

compared to positive sentiment. The classification metrics below show the result of the sentiment analysis on summarized text.

Sumy (Extractive Summarization):

```
TF-IDF Model Accuracy: 0.79375
TF-IDF Model Classification Report:
               precision    recall  f1-score   support

           0       0.78      0.53      0.63       799
           1       0.80      0.92      0.86      1601

    accuracy                           0.79      2400
   macro avg       0.79      0.73      0.74      2400
weighted avg       0.79      0.79      0.78      2400
```

T5-small (Abstractive Summarization):

```
TF-IDF Model Accuracy: 0.7825
TF-IDF Model Classification Report:
               precision    recall  f1-score   support

           0       0.75      0.52      0.61       799
           1       0.79      0.91      0.85      1601

    accuracy                           0.78      2400
   macro avg       0.77      0.72      0.73      2400
weighted avg       0.78      0.78      0.77      2400
```

Improved (Extractive Summarization):

```
TF-IDF Model Accuracy: 0.7958333333333333
TF-IDF Model Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.53      0.63       799
           1       0.80      0.93      0.86      1601

    accuracy                           0.80      2400
   macro avg       0.79      0.73      0.75      2400
weighted avg       0.79      0.80      0.78      2400
```

The metrics reveal that Sumy performs slightly better than T-5, and the improved version performs slightly better than Sumy. However, it should be noted that all these models have bad performance at identifying negative sentiment. This could be caused by the imbalance of data in the negative and positive sentiment classes. The result is not the most ideal since it only improved the Sumy model by 0.2%, and it is still far from our goal – to have the same accuracy as the unsummarized review or even better. Beside the imbalanced data, this is most likely caused by the lack of contextual understanding in both the TF-IDF sentiment analyzer and the Sumy extractive summarizer, it is the cost of using simplistic methods over transformer based models.

Conclusion:

In this project, I explored various methods for combining text summarization with sentiment analysis to determine how well sentiment is preserved after summarization. By using both extractive and abstractive summarization techniques, I aimed to create concise reviews that retained the original sentiment of the text. The results indicated that extractive summarization using Sumy, particularly when neutral sentences were filtered out, performed slightly better in

maintaining sentiment accuracy compared to abstractive summarization with T5-small. However, all methods showed a limitation in accurately identifying negative sentiments, likely due to data imbalance and the lack of contextual understanding in the models used. While the improvement was modest, this project highlights the challenges and potential trade-offs in using simpler models for sentiment-aware summarization, suggesting that future work could benefit from more advanced, context-aware models to achieve better performance.

References:

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. “BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding.” ArXiv.org. October 11, 2018. https://arxiv.org/abs/1810.04805.

Mihalcea, Rada, and Paul Tarau. 2004. “TextRank: Bringing Order into Text.” ACLWeb. Barcelona, Spain: Association for Computational Linguistics. July 1, 2004. https://aclanthology.org/W04-3252/.

Tomer, Minakshi, and Manoj Kumar. "Multi-Document Extractive Text Summarization Based on Firefly Algorithm." Journal of King Saud University - Computer and Information Sciences 34, no. 8, Part B (2022): 6057-6065. https://doi.org/10.1016/j.jksuci.2021.04.004.

Sarowar Jahan Saurav. 2023. “6 Useful Text Summarization Algorithm in Python - Sarowar ahan Saurav - Medium.” Medium. Medium. October 17, 2023. https://medium.com/@sarowar.saurav10/6-useful-text-summarization-algorithm-in-pythn -dfc8a9d33074.

Sparck Jones, K. (1972) A Statistical Interpretation of Term Specificity and Its Application in Retrieval. Journal of Documentation, 28, 11-21.http://dx.doi.org/10.1108/eb026526

Yadav, N., and N. Chatterjee. "Text Summarization Using Sentiment Analysis for DUC Data." In Proceedings of the 2016 International Conference on Information Technology (ICIT), Bhubaneswar, India, 229-234. 2016. https://doi.org/10.1109/ICIT.2016.054.