Text Summarization

Chiriac Ella-Ana-Maria
ella-ana.chiriac@s.unibuc.ro

Semen Valentin-Ion
valentin-ion.semen@s.unibuc.ro

Stoica Elias-Valeriu
elias-valeriu.stoica@s.unibuc.ro

Abstract

The project focuses on exploring both extractive and abstractive methods to generate summaries from Romanian texts while preserving key information. The paper documents the implementation and evaluation of text summarization techniques involving pre-trained language models and traditional algorithms.

1 Introduction

We are aiming to create a summarization model for the Romanian language since it is an unpopular topic that needs more attention because there is a lack of models focusing on this specific language. Due to the diminishing attention span in today's generation, it is important to quickly and correctly extract relevant information from large volumes of data, which are filled with lots of redundant details. Text summarization is a tough challenge which needs extensive knowledge in Natural Language Processing. Computing a good accuracy could involve creating ampler models that can consume a larger amount of resources and could also be more time-consuming. This is the reason why we have focused more on exploring different ways of obtaining relevant results than on the accuracy itself.

2 Related Work

The field of Natural Language processing has seen significant advancements in the last years, with a variety of approaches explored during the research. In this section, we review some of the contributions on this subject and its usage and contribution in handling Romanian news articles.

Adhikari et al. (2020) [Adhikari et al., 2020] investigate various machine learning techniques for text summarization, presenting a detailed analysis of different algorithms. Their research, shared at the 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), highlights how these methods can be tailored for Romanian, emphasizing their potential for languages with limited research resources.

Tas and Kiyani (2007) [Tas and Kiyani, 2007] survey the development of text summarization methods, from basic extractive techniques to advanced abstractive models. Although their study is older, the foundational concepts they discuss are essential for understanding current technologies and can be particularly valuable in developing a summarization model for Romanian.

Lastly, *Haque, Pervin, and Begum (2013)* [Haque et al., 2013] review single document text summarization techniques using NLP, discussing various challenges and solutions found in previous research. Their insights are crucial for pinpointing research opportunities in summarization for specific languages, such as Romanian.

By having these benchmarks, from our research, that offered us an idea about the diverse methodologies, we managed to set a stage for the development of two summarization models for Romanian news articles by following 2 methods: extractive and abstractive.

3 Dataset

The datasets used in this project are the *RO Text Summarization dataset* [1], provided by ReaderBench and Alexandru Petrachi and hosted on Hugging Face and *AlephNews dataset* [2]. These datasets are specifically created for the Romanian language, representing valuable resource for our goal of developing a Romanian text summarization model.

3.1 Data Description

The datasets contain a variety of text sources, including news articles and informational content, which creates a diverse environment for training

[1] https://huggingface.co/datasets/readerbench/ro-text-summarization

[2] https://huggingface.co/datasets/readerbench/AlephNews

a model. Each text entry in the dataset is paired with a corresponding summary, which helps both training the model and validating the algorithm's accuracy. The latter dataset structures its content into a list of paragraphs as opposed to the former which has it structured in a single text block.

3.2 Data Acquisition and Annotation

The data were collected from multiple Romanian news websites, ensuring a wide coverage of topics and styles, which can be observed in the next subsection. The summaries seem to be generated both automatically and manually and they represent a strong resource for evaluating our models.

3.3 Data Distribution

The datasets cover a broad range of topics including general news and specialized reports. It is important to prevent model bias and grant more general results across various news types by providing this variation in the data.

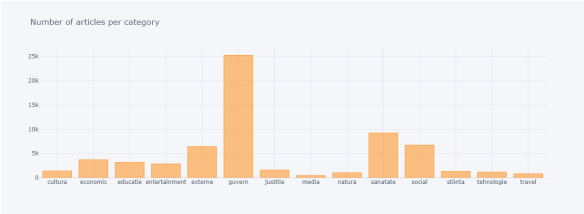

Figure 1: News Distribution

3.4 Relevance to the Model

Having a diverse set of data is beneficial for developing a summarization model that provides good result across all topics. The generation of decent summaries is also aided by the variety in text length and style.

4 Methods

In the following section we will describe, analyze and compare both methods of summarize texts using NLP, extractive and abstractive.

5 Extractive Method

We started developing the first model by using the extractive method, because it is a good start point in our research, since it is easier to understand than the abstractive one. It involves identifying and extracting the most important sentences from the original text. Extractive summarization can be done quickly using an unsupervised approach that does not require prior training.

5.1 Preprocessing

Preprocessing is one of the most important parts when developing any Natural Language Processing algorithm and you must pay attention to lots of details, since the quality of input data could directly affect the performance. For this project, we used several preprocessing methods that are well optimized for Romanian texts. We achieved this by removing unnecessary elements such as URLs, addresses, HTML tags, whitespaces and notes. In addition, we modified the corpus into a cleaner format by getting rid of special characters/punctuation using regular expressions and by lowering its case.

5.1.1 Text Cleaning

The first step involved cleaning the text, an essential part for reducing noise and normalizing the data. Using the Natural Language Toolkit (NLTK) we implemented sentence tokenization and split the text into an array of individual sentences. In this way, we prepare the data for the Graph Based method for summarization, where any information between each two sentences will be of utmost importance.

We also applied regular expressions to remove non-alphanumeric characters from each sentence. This standardization removes irrelevant elements like punctuation and special characters that might confuse the model:

```
sentence.replace("[^a-zA-Z0-9]", " ")
```

5.1.2 Tokenization and Correction

After cleaning, we tokenized the sentences using SpaCy's Romanian language model (`ro_core_news_sm`). This model not only tokenizes the text but also provides lemmatization and part-of-speech tagging, which are beneficial for understanding the grammatical structure of sentences and reducing words to their base or root form. Correct tokenization is fundamental for ensuring that the model interprets the text as intended:

```
doc = nlp(tok)
corrected_sent = ' '.join([token.text
    for token in doc])
```

5.1.3 Impact on the Model

The preprocessing steps have a great impact on any model's performance. Firstly, it was an important requirement for the strategy of the algorithm, where we needed individual sentences for the Graph Based method. Also, we reduced the complexity of the algorithm and its ability to rank the sentences and their impact, by getting rid of unnecessary content they had. In this way, we granted an efficient and accurate processing of the information.

Each step was chosen based on its potential to improve data quality and relevancy for the text summarization task. Even though it did not create a radical difference on the accuracy, it assured us that it takes only the important content of the dataset.

5.2 Model - Graph Based Method

In this method, every sentence of the news text is considered as a vertex of a graph. These sentences are connected with an edge if there exists a common semantic relation between them, and based on this score each edge has a weight. A graph based ranking algorithm is used to decide the importance of each vertex within the graph. The ones with high cardinality (high number of edges connected to it) are considered important sentences and are included in the summary.

For computing the similarities among all the sentences and returning the ones with maximum scores, we use Cosine Similarity as the similarity matrix and TextRank algorithm to rank them based on the importance.

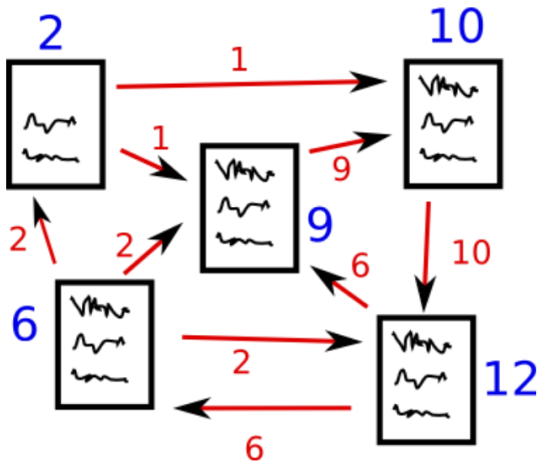

Figure 2: Graph-based Model Visualization. Source: Adapted from[3]

5.2.1 PageRank algorithm

The PageRank algorithm is the algorithm behind TextRank, that is why we will describe it first. This popular graph based algorithm, which is also used by Google to rank the web pages based on the search result, creates a graph with all the pages as vertices and all the links between them as edges. Then, it calculates the score for each page, like a probability of it being visited by the user. In this way, it manages to display the most relevant links and pages first, which is a good practice in keeping people connected and focused on this activity. [Roberts, 2016]

5.2.2 Our ranking

We use a similar approach but on the similarity of sentences instead of the probability of visiting a page. The most important point of any graph based algorithm is the VOTE. Each vertex that has a link to another one casts a vote for that vertex. Higher the number of votes, higher the importance of that vertex. In order to calculate the score of a vertex, we take into consideration two factors: the number of votes cast for it, and the importance of the vertices that cast votes for it. After extracting and creating vectors for all sentences, we need to calculate the Cosine Similarity between each pair of sentences. Given a document with n sentences, we construct an $n \times n$ similarity matrix, $\mathbf{S}$, where each element S_{ij} represents the similarity between sentence i and sentence j. This matrix is symmetric, with diagonal elements representing the self-similarity of each sentence, typically set to 1. With this matrix, we can rank the sentences based on the similarity score and return the first N number of sentences that are relevant and must be included in the summary.

5.2.3 Computing Sentence Similarity

To compute the similarity between sentences, we employed two distinct approaches: the Universal Sentence Encoder (USE) and Sentence Transformers. These models were chosen due to their robustness and effectiveness in capturing semantic meanings of sentences across multiple languages.

Universal Sentence Encoder The Universal Sentence Encoder (USE), developed by Google, is designed to convert text into high-dimensional vectors. These vectors are then used to calculate sentence similarities. The similarity between two sen-

[3]Available at https://www.kaggle.com/code/vshantam/text-summarization-extractive-bleu

tences, s_1 and s_2, is computed as follows:

$$\text{similarity}_{\text{USE}}(s_1, s_2) = 1 - \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}, \quad (1)$$

where $\mathbf{x}$ and $\mathbf{y}$ are the vector embeddings of s_1 and s_2, respectively, obtained using USE.

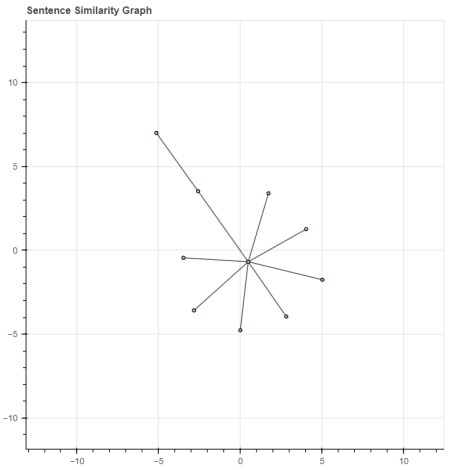

Figure 3: Sentence Similarity Graph showing the interconnection of sentences based on their semantic relationships, using USE

Sentence Transformers Alternatively, we use the Sentence Transformers framework, specifically the 'paraphrase-xlm-r-multilingual-v1' model. This framework enhances the BERT architectures to produce more accurate embeddings for sentence similarity tasks. The cosine similarity between two sentence embeddings is calculated as:

$$\text{similarity}_{\text{ST}}(s_1, s_2) = 1 - \text{cosine\_distance}(\mathbf{e}_1, \mathbf{e}_2), \quad (2)$$

where $\mathbf{e}_1$ and $\mathbf{e}_2$ are embeddings of s_1 and s_2.

Both methods output a similarity score that ranges from -1 (completely dissimilar) to 1 (identical), which we then utilize to populate the similarity matrix $\mathbf{S}$.

5.2.4 Interpreting Results

BLUE Scores Comparison Overview and Variability:

- The BLEU (Bilingual Evaluation Understudy) Scores Comparison chart plots the performance of each summarization model. The range is from 0 to 1 and higher scores indicate better matches with human manually translation.

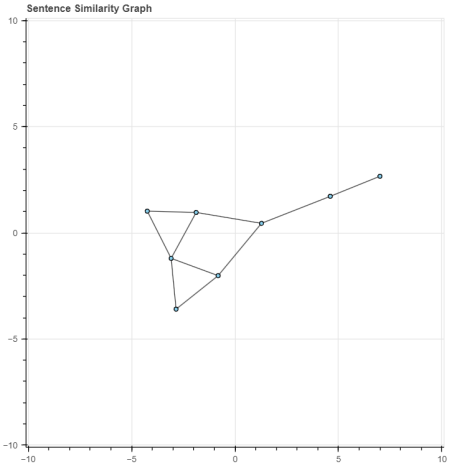

Figure 4: Sentence Similarity Graph showing the interconnection of sentences based on their semantic relationships, using SE

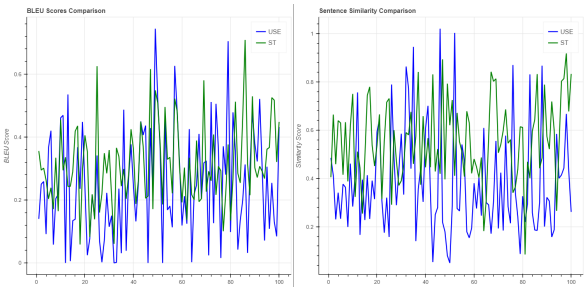

Figure 5: BLEU Scores and Sentence Similarity Scores comparison

- The chart shows significant variability in the performance of both models across different news texts. This suggests that the effectiveness of each model may depend heavily on the specific content or context of the text being processed.

Analysis:

- Neither model consistently outperforms the other across all instances, which could suggest that each model has strengths in different types of text or summarization tasks.
- The high variability in BLEU scores for both models across test cases suggests that both models might struggle with certain text types or that their performance could be highly context-dependent.
- Since the summaries from the dataset are made in an abstractive manner, there is a small chance of having a good similarity with an extractive summary, but we tried to get as close as we can to a good result.

Sentence Similarity Comparison Overview and Variability:

- This chart measures how well each model assesses the similarity between sentences, an important task for applications like document summarization and information retrieval.
- Similar to the BLEU Scores chart, this graph also shows a high degree of variability in the performance of both models.

Analysis:

- The consistency in performance of the ST model could be beneficial for applications requiring a stable output, such as ongoing content analysis systems or interactive NLP applications.
- The varying performance of the USE model might make it suitable for specialized tasks where it can be fine-tuned to leverage its high performance peaks.

6 Abstractive Method

We have also tried implementing an abstractive method for this project. We have tried three different models, including pre-trained models and a custom created project.

6.1 Preprocessing

For preprocessing, we have used several known preprocessing methods, well optimized for Romanian texts, in the same manner as for the extractive method.

6.2 Models

In our exploration of text summarization techniques, we employed a variety of models, both pre-trained and custom-developed, to evaluate their effectiveness across different text types and summarization needs.

6.2.1 MarianMT

We tried using the pre-trained models MarianMT, known for their robustness in machine translation tasks. These models are pre-trained using the Marian framework, optimized for accurate translation, which we have adapted for the task of summarization. A translation tool is useful in the context of text summarization, as it extracts and understands the context, before translating it, which is exactly what we need in order to summarize a text.

Below are our conclusions regarding the usage of MarianMT model for summarization purposes.

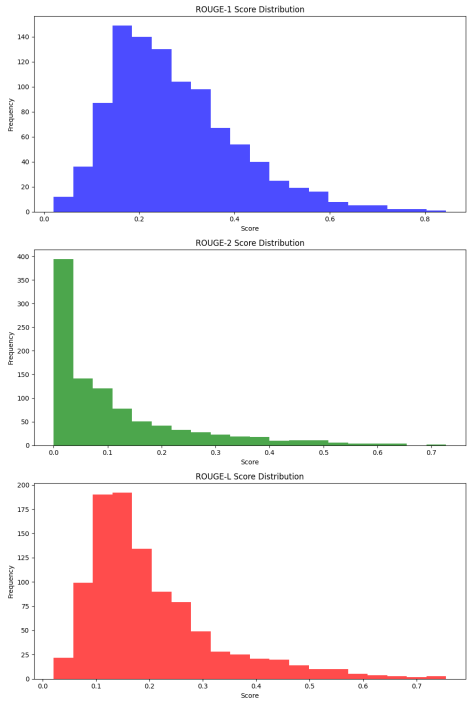

Figure 6: ROGUE score distributions for MarianMT model; first image presents ROGUE-1, second presents ROGUE-2 and the last one presents ROGUE-L

- **ROUGE-1 Score:** Achieving an average score of 0.275, this indicates that approxi-

mately 27.5% of the generated summaries matched the original summaries. The score distribution suggests moderate effectiveness with most scores distributed from 0.1 to 0.3.

- **ROUGE-2 Score:** With an average score of 0.111, this low score reflects challenges in maintaining coherent word sequences and the logical flow of ideas, with most scores under 0.2.
- **ROUGE-L Score:** The average score of 0.203 indicates the model's limited ability to preserve longer words, with scores ranging from 0.1 to 0.3.
- **Overall BLEU Score:** The score of 0.042 points to substantial challenges in achieving summaries that closely mirror the original texts, indicating issues with fluency and grammatical accuracy.

These findings suggest that while MarianMT can extract key words and phrases, overall coherence and grammatical accuracy of the summaries are sub-optimal. Even though we tried to experiment changing the parameters for improvements, the training time of the model would only go to even more extensive lengths, but the results would not improve significantly.

6.2.2 Custom LSTM Model

In addition to the pre-trained models, we tried to develop a custom model based on Long Short-Term Memory networks. LSTMs are suited for sequence prediction problems due to their ability to remember information for long periods. Our LSTM model was designed to gradually refine its understanding and generation of summaries through a series of recurrent layers.

The results for this model are presented in Figure 7:

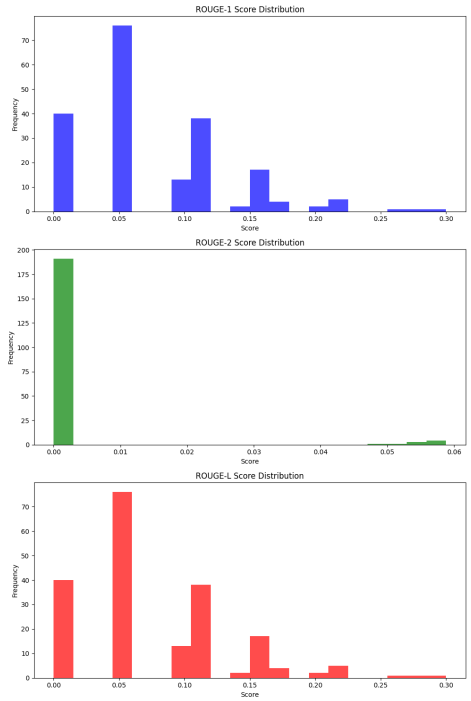

Figure 7: ROGUE score distributions for LSTM model; first image presents ROGUE-1, second presents ROGUE-2 and the last one presents ROGUE-L

- **ROUGE-1 Score:** The average ROUGE-1 score of 0.077 indicates that only about 7.7% of the generated summaries matched the original summaries, suggesting poor capture of essential content.
- **ROUGE-2 Score:** An extremely low average score of 0.002 reveals major difficulties in maintaining coherent word sequences.
- **ROUGE-L Score:** The average score, also at 0.077, underscores similar challenges in preserving the order and structure of words in the summaries, aligning poorly with the original summaries.
- **Overall BLEU Score:** A BLEU score of 0.111, while relatively better than the ROUGE scores, remains low, highlighting the generated texts' lack of fluency and grammatical accuracy compared to the reference texts.

These results indicate significant limitations in producing a coherent and relevant summary. The low scores demonstrate the model's inability to effectively capture the essence of the original texts and reconstruct the information in a logical manner.

However, we have found that this implementation would be the optimal adjustment for this model. We have tried adjusting the parameters in order to obtain a better accuracy, however in most cases, the training time would increase significantly, displaying an unreal amount of time until the model was trained.

6.2.3 Pre-trained Models from Transformers

T5 Small : The Text-to-Text Transfer Transformer (T5) is an encoder-decoder model that frames all NLP tasks as a text generation problem. We used the 'small' variant of T5, which is a

lighter version but still robust enough for generating accurate summaries.

Our statistics for the T5 Small model are shown below:

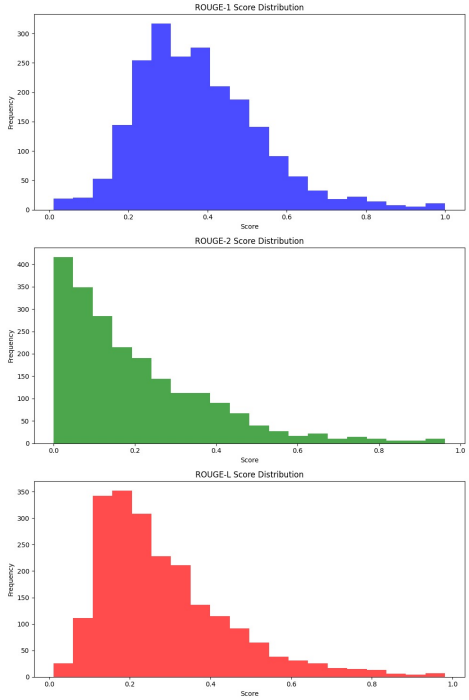

Figure 8: ROGUE score distributions for T5 Small model; first image presents ROGUE-1, second presents ROGUE-2 and the last one presents ROGUE-L

The T5 Small model was evaluated using various metrics, including ROUGE and BLEU scores. Here we discuss the results obtained:

- **ROUGE-1 Score:** The average ROUGE-1 score was 0.378, indicating that approximately 37.8% of the generated summaries matched the original summaries. The distribution primarily spans from 0.2 to 0.6, peaking between 0.3 and 0.4, suggesting moderate effectiveness in capturing key information.
- **ROUGE-2 Score:** The average ROUGE-2 score was 0.201, demonstrating that 20.1% of diagram from the generated summaries matched the original ones. This score, which is lower compared to ROUGE-1 reflects the model's difficulty in maintaining coherent word sequences. The majority of scores were below 0.2, indicating struggles in preserving sentence coherence.
- **ROUGE-L Score:** With an average score of 0.283, the ROUGE-L metric shows that about 28.3% of the longest common subsequences were correctly captured, highlighting challenges in maintaining longer word structures consistently. Scores mostly ranged between 0.1 and 0.3, underscoring issues with retaining structured language over longer stretches.
- **Overall BLEU Score:** The BLEU score of 0.151 points to substantial challenges in achieving translations or summarizations closely mirroring the reference texts.

These results suggest that while the T5 Small model is capable of generating summaries with reasonable coverage of keywords, it faces difficulties in maintaining coherence and complex sentence structures. These limitations could potentially be addressed by further model tuning on more diverse datasets or by exploring combinations of extractive and abstractive techniques to improve the coherence and accuracy of the summaries.

BART Large : BART (Bidirectional and Auto-Regressive Transformers) is a powerful model that combines both bidirectional and auto-regressive transformers. We employed the 'large' variant of BART for its superior performance in generating coherent and contextually rich summaries.

Our statistics for the BART Large model are shown below:

The BART model was evaluated on its ability to generate text summaries using ROUGE and BLEU metrics, yielding the following results:

- **ROUGE-1 Score:** The average ROUGE-1 score was 0.389, indicating that about 38.9% of the generated summaries matched the original summaries. This demonstrates a relatively good capability of the model to capture essential keywords from the source texts, with most scores clustering around 0.3 to 0.4.
- **ROUGE-2 Score:** Achieving an average score of 0.208, this metric shows the model's moderate effectiveness in maintaining coherent sequences, with most scores falling between 0.1 and 0.3.
- **ROUGE-L Score:** The average ROUGE-L score of 0.285 signifies the model's capacity to align about 28.5% of the longest common subsequences with the reference summaries, indicating a moderate ability to maintain text structure over longer stretches.

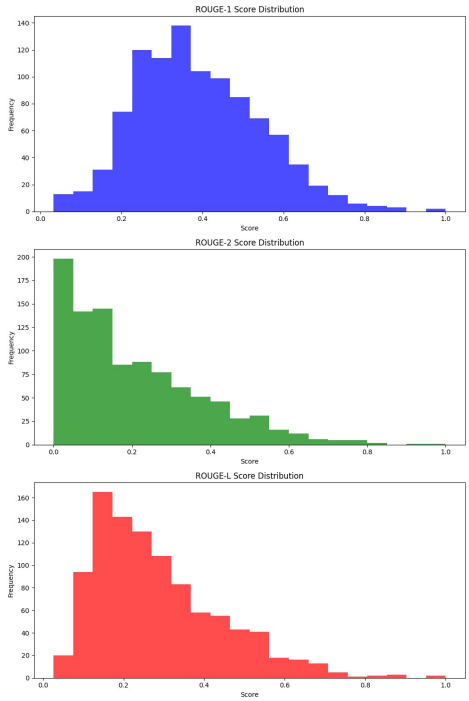

Figure 9: ROGUE score distributions for BART Large model; first image presents ROGUE-1, second presents ROGUE-2 and the last one presents ROGUE-L

- **Overall BLEU Score:** A BLEU score of 0.160, though modest, indicates an improvement in the fluency and grammatical accuracy of the generated texts compared to the original texts, compared to the previous models.

These results suggest that the BART model demonstrates enhanced efficiency in text summarization compared to other models, with a good capability for keyword capture and structure coherence. However, it still faces challenges in forming fully fluent and coherent sentences, suggesting room for further optimizations and training on more diverse datasets.

Each model brought its strengths and limitations to the project, contributing uniquely to our understanding of effective summarization techniques.

6.3 Results

To evaluate the effectiveness of our text summarization models, we utilized several widely recognized metrics. These include ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metrics and the BLEU (Bilingual Evaluation Understudy) score, each serving distinct purposes in measuring the quality of the generated summaries.

- **ROUGE-1**: It measures the overlap between the generated summaries and original summaries. This metric assesses the ability of the summarization model to capture the essential content.
- **ROUGE-2**: It provides insight into the coherence of the generated summary.
- **ROUGE-L**: This utilizes the longest common subsequence to evaluate the degree to which the generated summary preserves the original summary.

Alongside ROUGE metrics, we also employed the BLEU score.

These metrics collectively help determine how well the summaries generated by our models emulate human-like precision and fluency, covering both the accuracy of specific content and the overall coherence of the generated text.

7 Conclusion

Throughout this project we have gained a deeper understanding of the capabilities and limitations of the current NLP technologies for text summarization. We have learned about the importance of data preprocessing and the nuances of model performances.

One of the most enjoyable aspects of this project was experimenting with different models, especially the state-of-the-art transformer-based architectures. Seeing these models generate coherent and sometimes insightful summaries was particularly rewarding.

A significant challenge we have encountered was the extensive computational time required to train and fine-tune the models, even when using high-performance processors. This not only slowed down a lot of the project's process of development and testing, but also restricted the scope of our experiments.

8 Future Work

One interesting approach we started exploring, but did not manage to get good results on, was creating a custom abstractive model for the summarization tool. The model we created did not obtain high accuracy, but it would be very interesting to work more on it, in order to produce a model that excels at summarization.

The text summarization models can be valuable tools for individuals looking to streamline their

research process, or even manage large volumes of information. Using summarization tools, they can get the essential details without the need to go through the entire extensive documents. This tool can save substantial time and effort.

This project can be used as a standalone tool, as it offers practical utility for the users needing to digest large texts or conduct extensive research.

Ethical Statement

The text summarization tool, as most AI generative tools, can be misused. Unethical applications of this tool might include misleading summaries to manipulate public opinions, or even omitting crucial information of legal or financial documents.

Our project could have included biases present in the dataset, as the dataset could have been sourced from specific domains or demographics, which can lead to biased representations and generalizations. This issue is particularly acute in language models, which may inadvertently reinforce stereotypes or exclude minority viewpoints.

To address these concerns, we have made sure to use an extensive dataset, with data from multiple sources, which we knew that are not biased. However, since the dataset is large, including over 50.000 entries, we could not go through each of them, so we cannot exclude the fact that all of them might not be biased.

Limitations

While our project has achieved increasingly good results for the text summarization tool, several limitations warrant further investigation:

1. **Language and Morphological Complexity:** Our project primarily presents effectiveness in Romanian language. Other languages with different morphology or different alphabets, such as Arabic, may not be adequately served by our models.

2. **Scalability to Long Texts:**The models we created tend to decrease in performance as the length of the input text increases. This limitation is caused by the fact that the model may include more irelevant information, due to the extensive information in the original text.

3. **Computational Resources:** The substantial GPU and CPU resources required for training the models highlight a significant barrier to accessibility and scalability. This limitation present a great challenge to developing and testing models.

4. **Model Generalization:** The models we created are trained on specific datasets that include texts based on news and they may not be able to generalize well across different types of texts. This limits the use of our models in broader and vaster usage.

These limitations present our need for continued research into more robust and resource-efficient summarization techniques. Further, they highlight the importance of developing more general methods that can include summarization for different languages and lengths.

Acknowledgements

We would like to extend our thanks to each other for the dedication and teamwork throughout this project. Each member has played a crucial role and this collaborative effort proves our commitment to advance in the text summarization field.

References

Surabhi Adhikari et al. Nlp based machine learning approaches for text summarization. In *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pages 535–538. IEEE, 2020.

Md Majharul Haque, Suraiya Pervin, and Zerina Begum. Literature review of automatic single document text summarization using nlp. *International Journal of Innovation and Applied Studies*, 3(3):857–865, 2013.

Eric Roberts. The google pagerank algorithm. Technical report, Standford University, 2016.

Oguzhan Tas and Farzad Kiyani. A survey automatic text summarization. *PressAcademia Procedia*, 5(1): 205–213, 2007.