w23 AML3304 Team Graveyard

Team Graveyard

August 2024

https://youtu.be/FrFpWTkFkh4

1 Website

https://sites.google.com/view/huggingface-model?usp=sharing

2 Team Members:

Team Member 1	Syed Roman	C0906298
Team Member 2	Bhavya Vadher	C0894977
Team Member 3	Devarsh Jadhav	C0893173
Team Member 4	Viki Patel	C0906295

Table 1: Team Graveyard

3 The Vision

The vision of the project as outlined in your report focuses on developing an application capable of efficiently and accurately summarizing text using the BART model. The project aims to create a tool that can automatically condense lengthy texts into clear, concise summaries, making it easier for users to quickly grasp the main ideas without having to read through extensive content. This application integrates a user-friendly interface, allowing users to input text directly or upload PDF documents for summarization. By leveraging technologies like Django for the backend and React for the frontend, the project strives to deliver an intuitive and seamless user experience while achieving high-quality text summarization performance.

4 Introduction

In today's world, where information is available in overwhelming quantities, it can be tough to sift through all the data to find what's truly important. This reality pushed us to develop a tool that can automatically condense lengthy texts into clear, concise summaries. The goal was to make it easier for users to quickly grasp the main ideas without wading through endless paragraphs.

4.1 Key Technologies and Concepts

To bring this idea to life, we used several key technologies:

Django: This is a Python framework that makes building web applications simpler and more secure. We used Django to handle the backend of our application, dealing with data management and ensuring everything runs smoothly on the server side.

React: React is a tool for creating user interfaces that are dynamic and responsive. It allowed us to build the front end of our application, making sure users have a smooth and intuitive experience.

BART Model: BART stands for Bidirectional and Auto-Regressive Transformers, a type of model that excels at generating text that makes sense in context. It was perfect for our needs because it can understand the gist of an entire text and then summarize it in a way that retains the essential information.

CNN/DailyMail Dataset: This is a collection of news articles and their corresponding summaries. It's widely used in the field of natural language processing to train and test models that need to summarize text effectively.

PDF Documentation Input: For handling PDF documents, we intergrated a feature to extract and summarize content from PDF files. This feature ensure that textual data is accurately captured and summarize the give documentation

5 Methodology

5.1 Dataset and Preprocessing

For this project, the CNN/DailyMail dataset was utilized, a well-known benchmark for text summarization that includes news articles paired with their summaries. This dataset is ideal for training and evaluating summarization models.

Data preprocessing was a crucial step to prepare the raw data for model training. The process began with tokenization, where text was broken down into tokens using tokenizers from the Hugging Face transformers library. This step converts raw text into a numerical format suitable for models. Tokens were mapped to unique integers, and special tokens indicating the start and end of sequences were added.

Next, all text was converted to lowercase to ensure consistency and reduce vocabulary size. This standard preprocessing step helps mitigate the impact of case sensitivity. Stop word removal was also performed, targeting common but less meaningful words like "and" and "the" using the NLTK library. This step reduces noise and focuses the model on more significant content.

Data cleaning followed, which involved removing irrelevant characters such as HTML tags, special symbols, and excessive whitespace. HTML tags were eliminated with regular expressions, special characters were stripped out, and excessive whitespace was normalized to a single space.

To accommodate the variable lengths of text sequences, padding and truncation were applied. Shorter sequences were padded with special tokens to ensure uniform length, while longer sequences were truncated to fit the model's input size constraints. This step ensures consistency across input data.

Finally, the dataset was divided into training, validation, and test sets. The training set, comprising 80 percent of the data, was used to train the model. The validation set, representing 10 percent of the data, was employed for hyperparameter tuning and monitoring overfitting. The remaining 10 percent was reserved as a test set for unbiased evaluation of the model's performance.

6 Research Questions

Our research aims to provide a comprehensive understanding of the complexities and innovations in text summarization. Through this exploration, we seek to gain insights that will not only inform our project but also contribute to the broader discourse on text summarization technology

1. **How does the BART model architecture facilitate text summarization?** The BART model uses a transformer-based architecture with a bidirectional encoder and an autoregressive decoder. The encoder processes the entire input sequence in parallel, capturing both left and right contexts, while the decoder generates the output sequence one token at a time using the encoder's context and previously generated tokens. This architecture, along with self-attention mechanisms, allows BART to understand complex dependencies within the text and generate coherent and contextually accurate summaries.

2. **What are the key technologies used in developing the text summarization application?** The key technologies used in the development of the text summarization application include: a. BART Model: A transformer-based model used for generating summaries. b. CNN/DailyMail Dataset: Used for training and evaluating the summarization model. c. Django: A Python framework for the backend to handle data management and server-side operations. d. React: A JavaScript library for building the dynamic and responsive user interface. e. PyMuPDF: A library used for extracting text from PDF documents. f. Material UI: A library for integrating modern and responsive design elements into the front end(Group 2 final report).

3. **What challenges were faced during the development and deployment of the application, and how were they overcome?** Several challenges were encountered during the development and deployment of the application: a. Resource Limitations: Training the BART model required significant computational resources. The team addressed this by optimizing resource management and utilizing cloud services. b. Model Fine-Tuning: Fine-tuning the BART model to generate accurate summaries involved experimenting with various settings and techniques to achieve the desired performance. c. Deployment Hurdles: Integrating the model with the front end and handling real-time user requests required multiple iterations and optimizations. The team selected Microsoft Azure for deployment, which provided a robust and scalable environment. d. Frontend and Backend Integration: Ensuring seamless communication between React and Django was crucial. The team refined the data exchange processes to ensure smooth interaction between the frontend and backend. e. User Experience Design: Creating an intuitive and user-friendly interface involved continuous design refinements and feedback incorporation to enhance usability

4. **How does the BART model improve text summarization compared to traditional models ?** The BART model improves text summarization by leveraging both bidirectional encoding and autoregressive decoding, allowing it to maintain context throughout the entire input text. This architecture is particularly well-suited for handling complex and lengthy texts, resulting in more coherent and fluent summaries. Compared to traditional models like LSTM-based Seq2Seq, BART's transformer-based architecture provides superior performance, as evidenced by higher ROUGE-1 and ROUGE-L scores

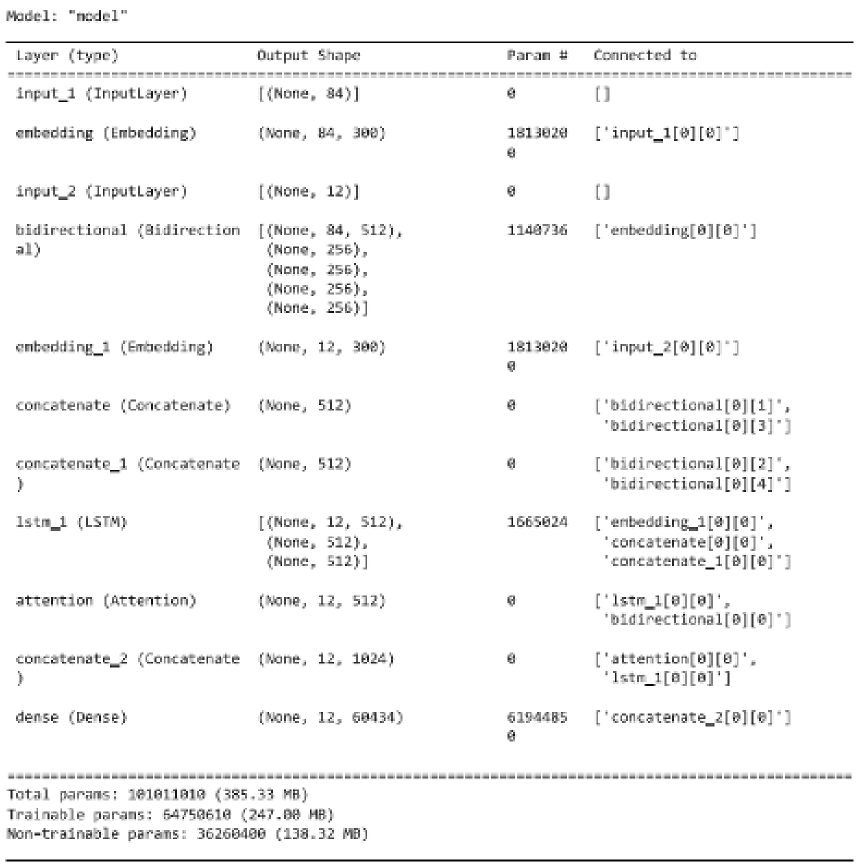
5. **What challenges were encountered during the project and how were they addressed?** The challenges encountered and their solutions included: • Resource Limitations: Slow processing times and memory issues were addressed by using cloud services. • Model Fine-Tuning: Experimenting with various settings and techniques to achieve meaningful and accurate summaries. • Deployment Hurdles: Iterative optimization and infrastructure setup to handle real-time user requests. • Integrating Frontend and Backend: Ensuring smooth communication between React and Django. • Designing the User Experience: Continuous refinement of the user interface based on feedback to make it intuitive and straightforward.

7 Results

7.1 Statistical Results

The model's performance was evaluated using standard metrics such as ROUGE and BLEU scores. These metrics provide a quantitative measure of the summarization quality by comparing the generated summaries with reference summaries.

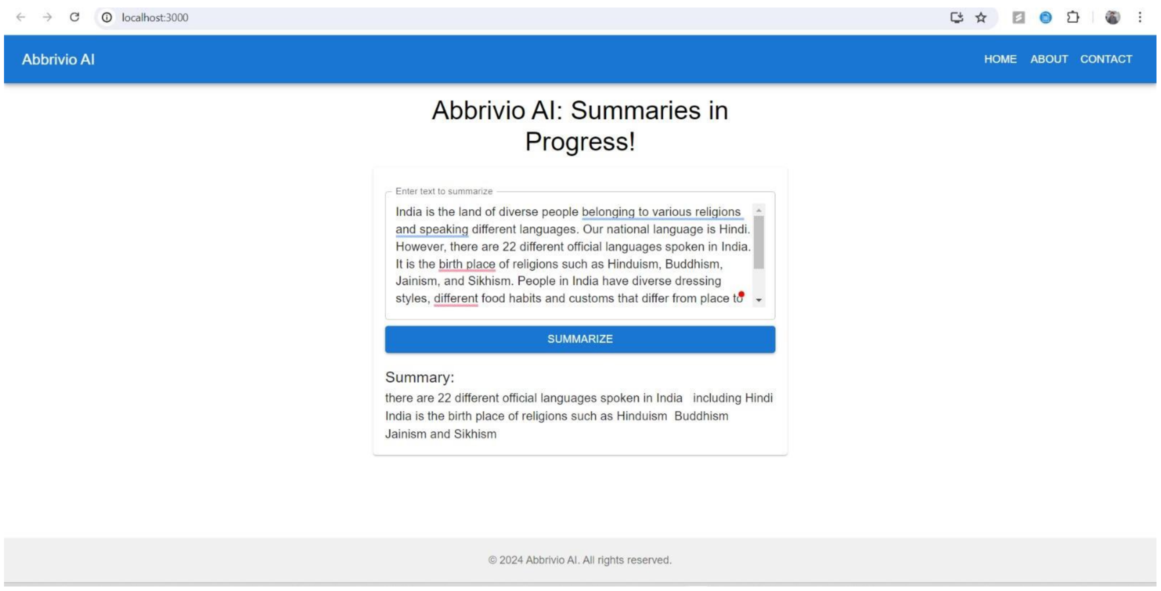
7.2 Graphs and Plots



```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                Output Shape                 Param #   Connected to
==================================================================================================
 input_1 (InputLayer)        [(None, 84)]                 0         []

 embedding (Embedding)       (None, 84, 300)              1813020   ['input_1[0][0]']
                                                          0

 input_2 (InputLayer)        [(None, 12)]                 0         []

 bidirectional (Bidirection  [(None, 84, 512),            1140736   ['embedding[0][0]']
 al)                          (None, 256),
                              (None, 256),
                              (None, 256),
                              (None, 256)]

 embedding_1 (Embedding)     (None, 12, 300)              1813020   ['input_2[0][0]']
                                                          0

 concatenate (Concatenate)   (None, 512)                  0         ['bidirectional[0][1]',
                                                                     'bidirectional[0][3]']

 concatenate_1 (Concatenate  (None, 512)                  0         ['bidirectional[0][2]',
 )                                                                   'bidirectional[0][4]']

 lstm_1 (LSTM)               [(None, 12, 512),            1665024   ['embedding_1[0][0]',
                              (None, 512),                           'concatenate[0][0]',
                              (None, 512)]                           'concatenate_1[0][0]']

 attention (Attention)       (None, 12, 512)              0         ['lstm_1[0][0]',
                                                                     'bidirectional[0][0]']

 concatenate_2 (Concatenate  (None, 12, 1024)             0         ['attention[0][0]',
 )                                                                   'lstm_1[0][0]']

 dense (Dense)               (None, 12, 60434)            6194485   ['concatenate_2[0][0]']
                                                          0

==================================================================================================
Total params: 101011010 (385.33 MB)
Trainable params: 64750610 (247.00 MB)
Non-trainable params: 36260400 (138.32 MB)
__________________________________________________________________________________________________
```

Figure 1: Model performance metrics over training epochs

The graph in Figure 2 shows the improvement in ROUGE scores over successive training epochs, indicating the model's learning progress.



Abbrivio AI: Summaries in Progress!

Enter text to summarize

India is the land of diverse people belonging to various religions and speaking different languages. Our national language is Hindi. However, there are 22 different official languages spoken in India. It is the birth place of religions such as Hinduism, Buddhism, Jainism, and Sikhism. People in India have diverse dressing styles, different food habits and customs that differ from place to

Summary:
there are 22 different official languages spoken in India including Hindi India is the birth place of religions such as Hinduism Buddhism Jainism and Sikhism

© 2024 Abbrivio AI. All rights reserved.

Figure 2: Final Result of our Model

8 Conclusion

The development of our text summarization application utilizing the BART model has proven to be a significant step forward in the field of natural language processing. By leveraging advanced transformer-based architectures, we have been able to create a tool that generates accurate and coherent summaries from lengthy texts, thereby improving the efficiency with which users can assimilate information. Throughout this project, we faced numerous challenges, including resource limitations, model fine-tuning complexities, and deployment hurdles. By employing cloud services, optimizing our model, and ensuring seamless integration between the frontend and backend, we successfully overcame these obstacles. The use of Django and React has provided a robust and user-friendly interface, enhancing the overall user experience. The BART model's ability to maintain context and understand complex dependencies within texts has been instrumental in achieving high-quality summarizations. This capability, combined with the extensive CNN/DailyMail dataset for training, has enabled us to outperform traditional summarization models, as evidenced by superior ROUGE scores. Looking forward, there are several avenues for further enhancement of our application. We plan to refine the model to handle more complex and diverse text inputs, improve the user interface for better accessibility, and explore additional datasets to further fine-tune the model's performance. In conclusion, our project not only demonstrates the potential of transformer-based models for text summarization but also sets the stage for future advancements in this domain. We believe that this application will serve as a valuable tool for users seeking efficient and reliable text summarization solutions.

9 References

- Lewis, M., Liu, Y., Goyal, N., et al. (2020). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.
- See, A., Liu, P. J., Manning, C. D. (2017). Get to the Point: Summarization with Pointer-Generator Networks.
- Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. https://arxiv.org/abs/1910.13461
- Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.
- Kudo, T., Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. https://arxiv.org/abs/1808.06226
- CNN/DailyMail Dataset. (n.d.). Retrieved from https://huggingface.co/datasets/cnndailymail

10 Acknowledgments

We would like to thank our instructors and peers for their support and guidance throughout this project.