Comparing Deep Learning Techniques for Abstractive Text Summarization

Nitisha Shetty[1], Navya Kriti[1], Cody Wang[1]
[1]New York University
ns6108@nyu.edu, nk3696@nyu.edu, cw3232@nyu.edu

Abstract

In the digital age, the exponential growth of data demands efficient techniques for digesting large volumes of text. Abstractive text summarization presents a sophisticated approach by generating novel phrases that encapsulate the essence of extensive texts, rather than merely extracting parts of the text. This project seeks to explore and compare various deep learning models to identify the most effective method for producing accurate and coherent text summaries. Utilizing the CNN/Dailymail Dataset [12], our research involves empirical testing of multiple models discussed in our deep learning coursework. Preliminary results indicate that with reasonable hardware configurations, these models achieve satisfactory accuracy. However, enhancements in computational resources, such as more advanced GPUs, show promise for significantly improving the precision and quality of the generated summaries.

Introduction

Text summarization is a vital challenge in the field of natural language processing, where the goal is to distill extensive texts into concise, informative summaries. Current state-of-the-art models offer promising results but often come at the cost of high computational demands. In this project, we explore efficient yet effective deep learning models for abstractive text summarization.

We begin with an overview of the landscape of abstractive summarization, introducing our innovative approach that integrates multiple deep learning strategies. Our study compares various models, including a traditional RNN, an RNN enhanced with Reinforcement Learning [7], BART (Bidirectional and Auto-Regressive Transformers) [4], T5 (Text-to-Text Transfer Transformer) [9], and a custom transformer model uniquely tailored to the challenges of summarization. Each model is rigorously evaluated based on its ability to generate accurate and concise summaries, as measured by ROUGE [5] scores.

While operating within hardware constraints, our models manage to strike a balance between computational complexity and effectiveness, yielding commendable results in text summarization.

Our Github repository:
https://github.com/Navya0203/Abstractive-Text-Summarization-Using-RNN-and-Transformers

Methodology

Transformers

T5 Architecture:
The T5 (Text-to-Text Transfer Transformer) [9] model utilizes a unified framework that converts all natural language processing tasks into text-to-text transformations. This model is structured around an encoder-decoder architecture based on the transformer model, which features a self-attention mechanism that assesses the significance of each word relative to others in the text. Initially pre-trained on a broad range of tasks, both unsupervised and supervised, T5 [9] is subsequently fine-tuned for specific applications such as text summarization on the CNN/Daily Mail dataset [12]. The model's training efficiency is optimized through the use of Automatic Mixed Precision (AMP), which significantly reduces computational load and memory requirements while maintaining robust performance, thus enabling effective handling of large datasets and complex model architectures.

BART Architecture:
BART (Bidirectional and Auto-Regressive Transformer) [4] functions as a denoising autoencoder, uniquely designed for pretraining sequence-to-sequence models. It combines a bidirectional encoder, like BERT (Bidirectional Encoder Representations from Transformers), which processes input text by fully exposing all words in the sequence to each other. This setup is complemented by a left-to-right auto-regressive decoder, reminiscent of GPT (Generative Pre-trained Transformer), that sequentially predicts the next word based on the previous context, applying a causal attention mechanism [11]. BART's [4] training involves initially distorting the text with a noising function and then learning to reconstruct the original content, enhancing its ability to interpret and generate contextually rich text.

Optimizer and Learning Rate:
The AdamW optimizer is selected for its effectiveness in managing sparse gradients and adaptive learning rates, which are vital for optimizing large pre-trained models like T5 [9] and Bart [4]. A learning rate of 0.0001 is used to fine-tune the model on the summarization task. This learning rate

helps in achieving a balance between convergence speed and training stability, reducing the risk of overfitting by allowing gradual updates to the model's weights.

Data Preprocessing:
The dataset was rigorously prepared by removing non-essential columns and ensuring the integrity of the data by systematically checking for and eliminating any missing values. To adapt the textual content for the transformer models, the data underwent a detailed preprocessing phase where it was tokenized using the T5 and BART [4] tokenizer. This process involved specifying maximum token lengths to maintain uniformity across data inputs and integrating special tokens required by the two-transformer architecture.

RNN

Architecture:
The model architecture comprises an encoder and a decoder, enhanced with an attention mechanism[11]. The encoder transforms tokenized article text into embeddings, processed by a bidirectional GRU to capture text sequence dependencies and generate encoded features and a final hidden state. The decoder, receiving tokenized ground truth summaries initiated with a special "starttoken," the encoder's final hidden state, and encoded features, utilizes an attention mechanism [11] to focus selectively on relevant parts of the input. It computes attention scores to form a context vector, which, combined with GRU outputs is used to predict the next token's probability distribution. The model employs teacher enforcing [3] to alternate inputs between its own predictions and the actual next token during training, enhancing learning efficiency in the initial stages.

Another hybrid RNN implementation combines a sequence-to-sequence framework with bidirectional LSTM encoders and a unidirectional LSTM decoder, enhanced by an attention mechanism [11] for effective text summarization. We incorporate Reinforcement Learning to optimize Rouge scores. [5], directly influencing the model’s performance improvements [7].

Data Preprocessing:
We have here as well utilized GloVe's [8] pre-trained word vectors from Wikipedia and Gigaword5 to represent our dataset's vocabulary as 300-dimensional embeddings, serving as initial weights for our models' embedding layers.

The dataset processing involves filtering articles and summaries to maximum lengths of 300 and 40 tokens respectively, after preprocessing. The articles are fed into an encoder that transforms the entire textual content into a set of encoded features. In the decoding stage, summaries are generated token-by-token from the encoded representations. This begins with a special initial token that signals the start of the summary. The decoder then sequentially predicts each subsequent token based on the previous tokens and the encoded article, producing the final summary.

The hybrid RNN model uses a subset of the first 200,000 articles and summaries from the CNN/DailyMail dataset [12]. Pre-processing includes cleaning text, expanding contractions, removing non-alphabetic characters, and filtering out stopwords and short words. The cleaned data is tokenized and padded to create uniform sequences for training, testing, and validation sets.

Optimizer and Learning Rate:
The model employs cross-entropy loss to measure the discrepancy between the predicted probability distribution of tokens and the actual distribution, where the correct token index is marked with a probability of 1. For optimization, we utilize the Adam [2] optimizer, chosen for its simplicity and effectiveness in handling sparse and noisy gradient estimations.

Custom Transformer

Architecture:
The Custom Transformer model has an encoder and decoder architecture, both consisting of multiple layers that include multi-head attention mechanisms [11] and position-wise feed-forward networks.

The encoder utilizes multi-head attention [11] to compute parallel attention scores for each token, focusing on different segments of the input, and integrates position embeddings to maintain the sequence order information. Attention scores represent the relevance of each token within its context, calculated using a softmax function on the scaled dot product of queries and keys, with masking to prevent future-looking information. The decoder mirrors the encoder's structure but includes an additional encoder-decoder attention layer that helps focus on relevant parts of the encoder output. Outputs are transformed into logits and then into a probability distribution using softmax with label smoothing, optimizing prediction confidence and reducing overfitting.

Optimizer and Learning Rate:
The ScheduledOptim class [10] is a learning rate scheduler designed to adjust the learning rate of an optimizer dynamically, based on the model dimensions and training step. It uses a specific formula that scales the learning rate inversely with the square root of the model dimension (d_model) and uses a warm-up period (n_warmup_steps), where the learning rate initially increases for a set number of warm-up steps and then decreases it, which helps in stabilizing the training early on and improving convergence.

Data Preprocessing:
We utilized GloVe's [8] pre-trained word vectors from Wikipedia and Gigaword5 to represent our dataset's vocabulary as 300-dimensional embeddings, serving as initial weights for our models' embedding layers.

We standardized the dataset by setting token thresholds, truncating article bodies to 300 tokens and summaries to 40 tokens, refining our data for uniformity. This data then undergoes embedding and positional encoding in the encoder and is fed into the decoder with preprocessed summaries and initial tokens to generate concise summary outputs.

Discussion

In this section, we discuss the performance and limitations of five deep learning models implemented to address natural language processing tasks, using a dataset of over 300,000 records, under varied hardware constraints.

The base RNN model gave us a ROUGE [5] score 0.16 and hence we implemented the hybrid model, integrating RNN's sequence processing with RL's decision-making capabilities and is trained on 10,000 records, its potential accuracy remains underexplored. Addressing complexity through chunking could enhance future performance.

Both T5 [9] and BART [4] demonstrated effective performance with Rouge [5] scores of 0.39 and 0.38, respectively, facilitated by Automatic Mixed Precision (AMP) at 16-bit precision. These models could achieve even better results on more capable hardware with 32-bit precision training. Similarly, our custom transformer, achieving a Rouge [5] score of 0.35, shows potential for improvement.

Transformers excel over RNNs in environments with hardware constraints thanks to their ability to process data in parallel, not sequentially like RNNs. This design allows them to utilize contemporary hardware more efficiently, speeding up training without succumbing to the issues like vanishing gradients that often hinder RNNs. This makes transformers particularly adept at handling complex tasks more swiftly and reliably, even when hardware resources are limited.

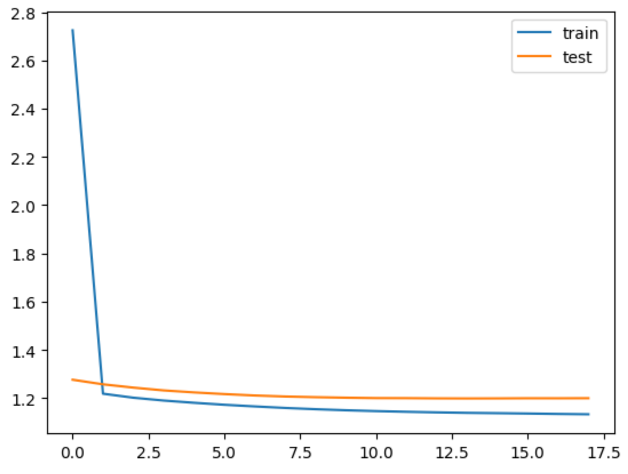

Figure 1: Train Loss vs Test Loss Curve for Hybrid RNN Model with RL

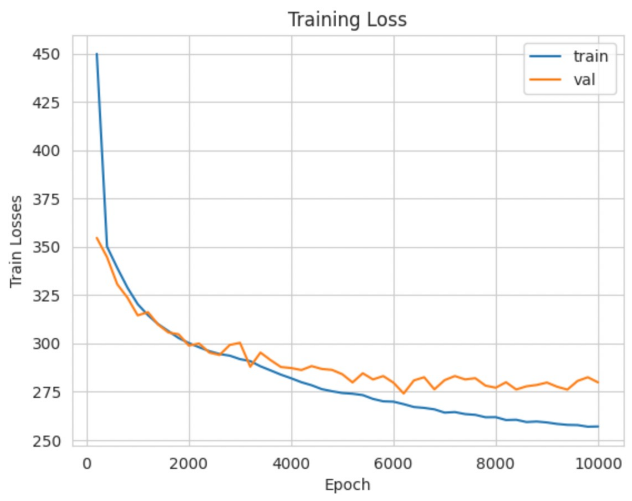

Figure 2: Train Loss vs Test Loss Curve for Custom Transformer

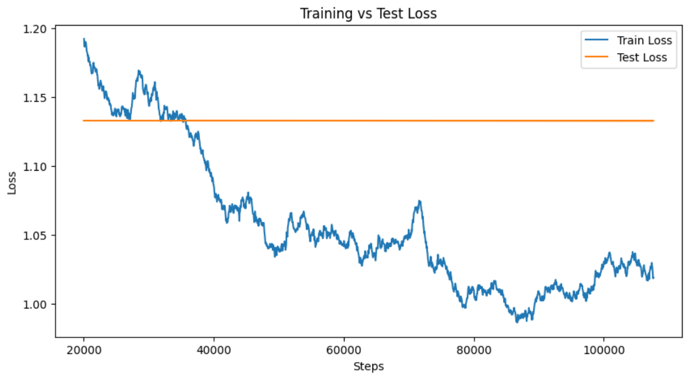

Figure 3: Train Loss vs Test Loss for BART Transformer [4]

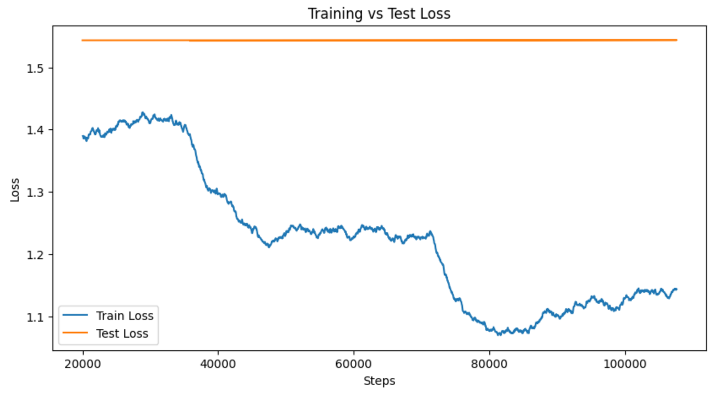

Figure 4: Train Loss vs Test Loss for T5 Transformer [9]

Results

The comparative analysis clearly shows that transformer models, with their parallel processing capabilities and sophisticated attention mechanisms [11], are better suited for complex tasks like abstractive text summarization under the constraints tested. The T5 [9] Transformer's top performance underscores its advanced pre-training and fine-tuning strategies, making it the most effective model among those we evaluated.

The decent performance of our Custom Transformer, achieving a Rouge-1 score [5] of 0.35, is particularly noteworthy. It validates our design choices and offers a promising foundation for further refinement and experimentation. Given more resources and perhaps even more tailored training approaches, it has the potential to match established models like T5 [9] and BART [4].

Model	**Rouge-1**	**Rouge-2**	**Rouge-l**
T5 Transformer	0.39	0.17	0.37
BART Transformer	0.38	0.18	0.35
Custom Transformer	0.35	0.17	0.28
RNN	0.16	0.01	0.12
RNN with RL	0.11	0.01	0.10

Figure 5: Rouge Score Comparison for all models implemented [6].

References

[1] Chen, Y.-C.; Bansal, M. 2018. Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. arXiv preprint arXiv:1805.11080. Available on: https://arxiv.org/pdf/1805.11080

[2]Kingma, D. P.; Ba, J. 2017. Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980. Available on: https://arxiv.org/abs/1412.6980

[3]Lamb, A. et al., 2016. Professor Forcing: A New Algorithm for Training Recurrent Networks. arXiv preprint arXiv:1610.09038. Available on: https://arxiv.org/abs/1610.09038

[4]Lewis, M. et. al. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Processing. arXiv preprint arXiv:1910.13461. https://arxiv.org/pdf/1910.13461

[5]Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. Barcelona, Spain: Association for Computational Linguistics.

[6] Pan, X., & Liu, P. 2017. Sequence-to-Sequence with Attention Model for Text Summarization. Available on: https://towardsdatascience.com/text-summarization-with-amazon-reviews-41801c2210b

[7]Paulus, R.; Xiong, C.; Socher, R. 2017. A deep reinforced model for abstractive summarization. arXiv preprint arXiv:1705.04304. Available on: https://arxiv.org/abs/1705.04304

[8]Pennington, J.; Socher, R.; Manning, C. D. 2014. GloVe: Global Vectors for Word Representation. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)

[9] Raffel, C. et. al. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv preprint arXiv: 1910.10683. Available on: https://arxiv.org/pdf/1910.10683

[10] Rehman, T.; Das, S.; Sanyal, D. K.; Chattopadhyay, S. 2023. Abstractive text summarization using attentive GRU based encoder-decoder. arXiv preprint arXiv:2302.13171. Retrieved from https://arxiv.org/abs/2302.13171

[11]Vaswani, A. et al., 2017. Attention Is All You Need. In Proceedings of the 31st Conference on Neural Information Processing Systems. arXiv preprint:1706.03762. Long Beach, CA, USA. Available on: https://arxiv.org/pdf/1706.03762 Available on: https://github.com/jadore801120/attention-is-all-you-need-pytorch/

[12] CNN/Daily Mail Dataset for Text Summarization. Available on: https://www.kaggle.com/datasets/gowrishankarp/newspaper-text-summarization-cnn-dailymail.

Acknowledgement

We extend our gratitude to Vaswani et al. [11], Lewis et al. [4], and Raffel et al. [9] for their foundational contributions to transformer models, and to Pan and Liu [6] for their insights on hybrid RNNs, which have greatly influenced our project. Additionally, we acknowledge Rehman et al. [10] for their work on abstractive text summarization using attentive GRU.

We are also thankful for the guidance and resources provided by Professors Chinmay Hegde, the teaching assistants, and the Google Colab team.