Text Summarization Using Deep Learning

R.M.H.G.M. RAJANAYAKA

PS2487
Department of Computing
Faculty of Science
Eastern University of Sri Lanka

Supervisor: Mr. R. Sakuntharaj

November 25, 2024

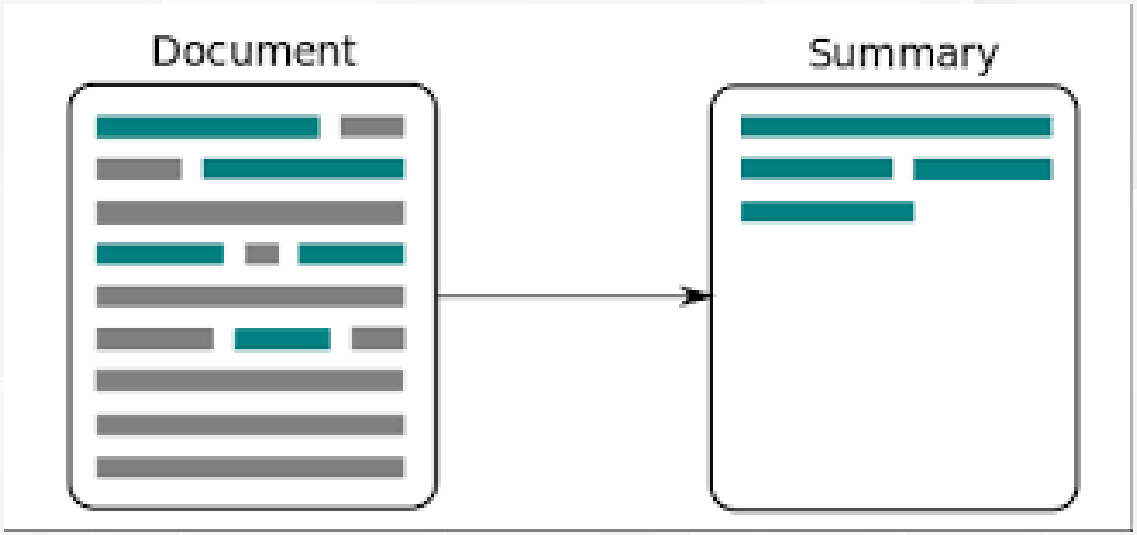
Table of contents

Abstract

In this study, we present how we can employ BART, T5 and PEGASUS trained on the conversational text summarization dataset called SAMSum. The goal is to measure how well the proposed models maintain coverage and coherency in summaries of dialogue information. Leveraging Google Colab with GPU support, we implemented a systematic training and evaluation process, measuring performance with ROUGE scores across four metrics: These evaluating metrics include ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-Lsummary.From the result comparision,it observed that BART average ROUGE score of 0.009523, T5 average ROUGE score of 0.025985 and PEGASUS average ROUGE score of 0.011731. The outcomes reveal that T5 yielded the best ROUGE scores, which evidences its ability to learn highly relevant dialogue information more effectively and concisely than the competing models, including PEGASUS and BART.

Introduction

Summarization generates a small-scale but meaningful text from more extended text, keeping the most helpful information intact. Text summarization, the process by which an automated method produces a summarization of a longer doc- ument, has now become a desirable solution in many areas as it allows the user to glean the essential facets of a document without having to trawl through the entire text.



Introduction

Research Objectives

- Training and fine-tuning : involve training the model on a prepared dataset using techniques like attention mechanisms, optimization algorithms, and regularization to ensure high performance.
- Model performance testing : using ROUGE metrics to evaluate the quality of the summaries, focusing on retaining important information, fluency, and avoiding redundancy.
- Optimization and compare: includes tuning hyperparameters, expanding the dataset, and adding components like attention mechanisms. The model is then compared with traditional and baseline models to assess its effectiveness in text summarization.

Contribution

- Multi-Document Summarization Framework: Aligned with the issue of single-document summarization of the prior models, he designed a model capable of summarizing multiple documents using a deep learning approach.
- Improved Accuracy and Compression: Improved some basic options, such as attention mechanisms and sequence-to-sequence models.
- Model Stability: Proposed essential biking and derailing methods that boosted model stability, ensuring that models are not stuck in modes throughout training.
- Diverse Dataset Evaluation: Tested the proposed model on several datasets containing only dialogues as well as other textual data to support more general use.
- Benchmarking: Outperformed other proposed techniques in both accuracy and readability in comparison to state-of-the art techniques.

Literature Review

- **Author(s):** Mohmmadali Muzffarali Saiyyad and Nitin N. Patil
- **Year:** 2023
- **Title:** *Text Summarization Using Deep Learning Techniques: A Review*
- **Summary of Methodology :** particularly sequence to sequence (seq2seq), in the text summarization regarding the coherence and readability of massive data summaries. These techniques, thus training abstract features, are beneficial because they are free from the drawbacks of past summarization procedures that invariably demanded the feature extraction process.
- **challanges** multilingual summarization, scalability and factual content, as this was seen in abstractive summarization, where they generate news content instead of extracting it from the source.

Literature Review

- **Author(s):** Haopeng Zhang, Philip S. Yu, Jiawei Zhang
- **Year:** 2024
- **Title:** *A Systematic Survey of Text Summarization: Text summarization is described in the article*
- **Summary of Methodology :** When providing a general classification of the methods, various techniques of text summarization are described together with specific features of their action, the transition from statistical methods to modern approaches based on BERT, GPT, etc. The paper categorizes summarization strategies into two main types: includes both the extractive model, whereby relevant sentences are pulled out of the text, and the abstractive approach in which new sentences are created for summarizing the original text.
- **challenges** mitigate enduring difficulties, especially in ad hoc summarization, where keeping factually accurate can be problematic.

Literature Review

- **Author(s):**B. Lee, J. Park, and S. Kim
- **Year:** 2020
- **Title:** *Abstractive and Extractive Summarization Using Transformers*
- **Summary of Methodology:** text summarization with the transformer model: how, in other words, BERT and T5 have really improved the efficiency in making flawless human-like summaries. According to the authors, transformers really are expected to increase the capacity of summarization models to retain contextual accuracy and coherence. These transformers are power-hungry, and, as successful
- **challenges** concerns on the maintenance of factual consistency, the limitation common to the generation not directly pulled from the source.

Literature Review

- **Author(s):** C.-Y. Lin
- **Year:** 2004
- **Title:** *Rouge: A package for automatic evaluation of summaries," in Text Summarization Branches Out*
- **Summary of Methodology :** Automatic evaluation is most commonly used in text summarization, and the best-known tool is known as ROUGE. By comparing the generated n-gram, word sequence and word pair with a set of reference summaries, ROUGE calculates f-measure and precision.

It includes several variants, such as:

- ROUGE-N: Computes the similarity of the n-grams in the generated and the reference summaries obtained.
- ROUGE-L: In the proposed metrics, stress is laid on the longest common subsequence of the generated and reference summaries.
- ROUGE-W: Makes more pressure on in-sequence matches.

However, ROUGE encounters some issues, especially when used for evaluation of the abstractive style of summaries since it measures mostly the overlapping of the tokens rather than the semantic meaning.

Background

Text Summarization Using Deep

The amount of textual data being produced every day is increasing rapidly both in terms of complexity as well as volume. Social Media, News articles, emails, text messages, generate massive information and it becomes cumbersome to go through lengthy text materials. Thankfully with the advancements in Deep Learning, we can build models to shorten long pieces of text and produce a crisp and coherent summary to save time and understand the key points effectively.

Background

We can broadly classify text summarization into two types:

- **Extractive Summarization:** This method selects key words or phrases directly from the input text to form a summary, focusing on identifying the most important parts without altering the original wording.
- **Abstractive Summarization:** This method generates new phrases to express the meaning of the input text, emphasizing grammatical structure and requiring advanced language modeling to create coherent summaries.

Background

(a) Extractive Summarization

Source Text: Peter and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

Summary: Peter and Elizabeth attend party city. Elizabeth rushed hospital.

(b) Abstractive Summarization

Source Text: Peter and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

Summary: Elizabeth was hospitalized after attending a party with Peter.

Background

Deep learning text summarization techniques

- Graph Neural Networks (GNNs) : Represent sentences and their relationships as graphs to capture global context before extracting key sentences.
- Reinforcement Learning: Train models to optimize a reward function (e.g., ROUGE scores)
- Sequence-to-Sequence (Seq2Seq) Models :
  - RNN-based models
  - Transformer models like T5 and Pegasus.
- Pre-trained Language Models : BART, GPT, and Pegasus

Background

Dataset

In this work, we are using a recently developed, large-scale, and modern dataset called SAMSum, which was put together to meet the challenge of summarizing dialogues. It contains about 16,000 messenger-like English conversational entries, for which summaries were provided by linguistic experts. These are natural and daily subjects and themes typical of what one would find today in summarization based on dialogue; hence, the appropriateness of such a dataset.

index	dialogue	summary
0	Amanda: I baked cookies. Do you want some? Jerry: Sure! Amanda: I'll bring you tomorrow :-)	Amanda baked cookies and will bring Jerry some tomorrow.
1	Olivia: Who are you voting for in this election? Oliver: Liberals as always. Olivia: Me too!! Oliver: Great	Olivia and Olivier are voting for liberals in this election.
2	Tim: Hi, what's up? Kim: Bad mood tbh, I was going to do lots of stuff but ended up procrastinating Tim: What did you plan on doing? Kim: Oh you know, uni stuff and unfucking my room Kim: Maybe tomorrow I'll move my ass and do everything Kim: We were going to defrost a fridge so instead of shopping I'll eat some defrosted veggies Tim: For doing stuff I recommend Pomodoro technique where u use breaks for doing chores Tim: It really helps Kim: thanks, maybe I'll do that Tim: I also like using post-its in kaban style	Kim may try the pomodoro technique recommended by Tim to get more stuff done.
3	Edward: Rachel, I think I'm in ove with Bella.. rachel: Dont say anything else.. Edward: What do you mean?? rachel: Open your fu**ing door.. I'm outside	Edward thinks he is in love with Bella. Rachel wants Edward to open his door. Rachel is outside.
4	Sam: hey overheard rick say something Sam: i don't know what to do :-/ Naomi: what did he say?? Sam: he was talking on the phone with someone Sam: i don't know who Sam: and he was telling them that he wasn't very happy here Naomi: damn!!! Sam: he was saying he doesn't like being my roommate Naomi: wow, how do you feel about it? Sam: i thought i was a good rommate Sam: and that we have a nice place Naomi: that's true man!!! Naomi: i used to love living with you before i moved in with me boyfriend Naomi: i don't know why he's saying that Sam: what should i do??? Naomi: honestly if it's bothering you that much you should talk to him Naomi: see what's going on Sam: i don't want to get in any kind of confrontation though Sam: maybe i'll just let it go Sam: and see how it goes in the future Naomi: it's your choice sam Naomi: if i were you i would just talk to him and clear the air	Sam is confused, because he overheard Rick complaining about him as a roommate. Naomi thinks Sam should talk to Rick. Sam is not sure what to do.

Methodology

libraries/tools

```
[ ]  !pip install transformers[sentencepiece] datasets sacrebleu rouge_score py7zr evaluate -q
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 51.8/51.8 kB 3.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 480.6/480.6 kB 19.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 104.0/104.0 kB 9.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.9/67.9 kB 5.0 MB/s eta 0:00:00
```

- t**ransformers:** Provides pre-trained models like BART, T5, and others for tasks such as text summarization, translation, and more.
- **[sentencepiece] :** extra is required for tokenization of models trained on multilingual or other specialized data.
- **datasets:** Offers access to a wide range of datasets for NLP tasks like the Samsum dataset, which you mentioned for text summarization.

Methodology

libraries/tools

- **sacrebleu:** Used for evaluating the quality of machine-translated text against reference translations.
- **rouge_score:** Essential for computing ROUGE metrics (ROUGE-1, ROUGE-2, and ROUGE-L) to evaluate the quality of text summarization.
- **py7zr:** Provides support for extracting `.7z` archive files, which may be useful for datasets or model weights.
- **evaluate:** A library that helps calculate various evaluation metrics, including ROUGE, BLEU, accuracy, and more.

Methodology

Data Preprocessing

```
from transformers import pipeline, set_seed

import matplotlib.pyplot as plt
from evaluate import load as load_metric
import pandas as pd
from evaluate import load as load_metric

from transformers import AutoModelForSeq2SeqLM, AutoTokenizer

import nltk
from nltk.tokenize import sent_tokenize

from tqdm import tqdm
import torch

nltk.download("punkt")
```

Methodology

Data Preprocessing

- pandas: Primarily used for data manipulation and analysis, ideal for preparing and processing structured datasets.
- nltk: A core library for natural language processing, assisting with tokenization and text preprocessing.
- tqdm: Enhances workflow visibility with progress bars for iterative processes.
- torch: Powers deep learning computations, providing the foundation for training and running NLP models.

Methodology

Model Loading

```python
from transformers import AutoModelForSeq2SeqLM, AutoTokenizer
import torch

# Set device to GPU if available
device = "cuda" if torch.cuda.is_available() else "cpu"

# Specify the T5 model checkpoint (e.g., t5-base, t5-small, or t5-large)
model_ckpt = "t5-base"  # Change this to 't5-small' or 't5-large' if needed

# Load T5 tokenizer and model
tokenizer = AutoTokenizer.from_pretrained(model_ckpt)
model_t5 = AutoModelForSeq2SeqLM.from_pretrained(model_ckpt).to(device)
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://hugg
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or
  warnings.warn(
config.json: 100% 1.21k/1.21k [00:00<00:00, 85.1kB/s]
spiece.model: 100% 792k/792k [00:00<00:00, 920kB/s]
tokenizer.json: 100% 1.39M/1.39M [00:01<00:00, 828kB/s]
```

Methodology

Batch Processing and Metric Evaluation

```python
from tqdm import tqdm

def calculate_metric_on_test_ds(dataset, metric, model, tokenizer,
                                batch_size=16, device=device,
                                column_text="article",
                                column_summary="highlights"):
    # Create batches of articles and target summaries
    article_batches = list(generate_batch_sized_chunks(dataset[column_text], batch_size))
    target_batches = list(generate_batch_sized_chunks(dataset[column_summary], batch_size))

    for article_batch, target_batch in tqdm(
        zip(article_batches, target_batches), total=len(article_batches)):

        # Add predictions and references to the metric
        metric.add_batch(predictions=decoded_summaries, references=target_batch)

    # Compute and return the ROUGE scores
    score = metric.compute()
    return score
```

Methodology

Load Dataset

```python
from datasets import load_dataset

# Load the Samsum dataset
dataset_samsum = load_dataset("samsum")

# Calculate and print the lengths of each dataset split
split_lengths = [len(dataset_samsum[split]) for split in dataset_samsum]
print(f"Split lengths: {split_lengths}")

# Print feature names in the training dataset
print(f"Features: {dataset_samsum['train'].column_names}")

# Print a specific dialogue and summary from the test dataset
print("\nDialogue:")
print(dataset_samsum["test"][1]["dialogue"])

print("\nSummary:")
print(dataset_samsum["test"][1]["summary"])
```

```
Split lengths: [14732, 819, 818]
Features: ['id', 'dialogue', 'summary']

Dialogue:
Eric: MACHINE!
Rob: That's so gr8!
Eric: I know! And shows how Americans see Russian ;)
Rob: And it's really funny!
Eric: I know! I especially like the train part!
Rob: Hahaha! No one talks to the machine like that!
Eric: Is this his only stand-up?
Rob: Idk. I'll check.
Eric: Sure.
Rob: Turns out no! There are some of his stand-ups on youtube.
Eric: Gr8! I'll watch them now!
Rob: Me too!
Eric: MACHINE!
Rob: MACHINE!
Eric: TTYL?
Rob: Sure :)

Summary:
Eric and Rob are going to watch a stand-up on youtube.
```

Methodology

Tokenization

```
from transformers import AutoModelForSeq2SeqLM, AutoTokenizer

# Load T5 tokenizer and model
tokenizer = AutoTokenizer.from_pretrained(model_ckpt)
model_t5 = AutoModelForSeq2SeqLM.from_pretrained(model_ckpt).to(device)
```

```
# Prepare input for T5 with the summarize prefix
inputs = tokenizer(
    ["summarize: " + article for article in article_batch],
    max_length=512,  # Adjust max_length based on T5's capacity
    truncation=True,
    padding="max_length",
    return_tensors="pt"
)
```

Methodology

Model Selection

To obtain excellent text summarization for the Samsung dataset, we tested three sophisticated deep learning models: These models include BART, PEGASUS and T5 all of which are architectures of the Transformer.

```
# Specify the T5 model checkpoint (e.g., t5-base, t5-small, or t5-large)
model_ckpt = "t5-base"  # Change this to 't5-small' or 't5-large' if needed

# Load T5 tokenizer and model
tokenizer = AutoTokenizer.from_pretrained(model_ckpt)
model_t5 = AutoModelForSeq2SeqLM.from_pretrained(model_ckpt).to(device)
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89: UserWarning
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https:/
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public mode
  warnings.warn(
config.json: 100% 1.21k/1.21k [00:00<00:00, 85.1kB/s]
spiece.model: 100% 792k/792k [00:00<00:00, 920kB/s]
```

Methodology

Training Configuration

```
from transformers import TrainingArguments, Trainer

trainer_args = TrainingArguments(
    output_dir='t5-samsum', num_train_epochs=1, warmup_steps=500,
    per_device_train_batch_size=1, per_device_eval_batch_size=1,
    weight_decay=0.01, logging_steps=10,
    evaluation_strategy='steps', eval_steps=500, save_steps=1e6,
    gradient_accumulation_steps=16
)
```

```
/usr/local/lib/python3.10/dist-packages/transformers/training_args.py:1525: FutureWarning: `evaluation_strategy` is deprecated and will be removed in version 4.46 of 🤗 Transformers. Use `eval_strategy` instead
  warnings.warn(
```

```
trainer = Trainer(model=model_t5, args=trainer_args,
                  tokenizer=tokenizer, data_collator=seq2seq_data_collator,
                  train_dataset=dataset_samsum_pt["train"],
                  eval_dataset=dataset_samsum_pt["validation"])
```

```
trainer.train()
```

```
wandb: WARNING The `run_name` is currently set to the same value as `TrainingArguments.output_dir`. If this was not intended, please specify a different run name by setting the `TrainingArguments.run_name` parameter.
wandb: Using wandb-core as the SDK backend. Please refer to https://wandb.me/wandb-core for more information.
wandb: Logging into wandb.ai. (Learn how to deploy a W&B server locally: https://wandb.me/wandb-server)
wandb: You can find your API key in your browser here: https://wandb.ai/authorize
wandb: Paste an API key from your profile and hit enter, or press ctrl+c to quit: ··········
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
Tracking run with wandb version 0.18.5
Run data is saved locally in /content/wandb/run-20241105_143127-yqof5ocz
Syncing run t5-samsum to Weights & Biases (docs)
View project at https://wandb.ai/meghzpro-n-ag/huggingface
View run at https://wandb.ai/meghzpro-n-ag/huggingface/runs/yqof5ocz
[920/920 30:36, Epoch 0/1]
```

Step	Training Loss	Validation Loss
500	1.658100	1.452666

```
TrainOutput(global_step=920, training_loss=1.7614311674366827, metrics={'train_runtime': 1862.5108, 'train_samples_per_second': 7.91, 'train_steps_per_second': 0.494, 'total_flos': 2606964191308800.0, 'train_loss':
1.7614311674366827, 'epoch': 0.9991854466467553})
```

Methodology

Model Testing

Test

```
[ ] dataset_samsum = load_dataset("samsum")
```

```
[ ] tokenizer = AutoTokenizer.from_pretrained("tokenizer")
```

```
sample_text = dataset_samsum["test"][0]["dialogue"]

reference = dataset_samsum["test"][0]["summary"]
```

```
[ ] gen_kwargs = {"length_penalty": 0.8, "num_beams":8, "max_length": 128}

    pipe = pipeline("summarization", model="t5-samsum-model",tokenizer=tokenizer)
```

```
Hardware accelerator e.g. GPU is available in the environment, but no `device`
```

```
[ ] print("Dialogue:")
    print(sample_text)


    print("\nReference Summary:")
    print(reference)


    print("\nModel Summary:")
    print(pipe(sample_text, **gen_kwargs)[0]["summary_text"])
```

```
Dialogue:
Hannah: Hey, do you have Betty's number?
Amanda: Lemme check
Hannah: <file_gif>
Amanda: Sorry, can't find it.
Amanda: Ask Larry
Amanda: He called her last time we were at the park together
Hannah: I don't know him well
Hannah: <file_gif>
Amanda: Don't be shy, he's very nice
Hannah: If you say so..
Hannah: I'd rather you texted him
Amanda: Just text him 🙂
Hannah: Urgh.. Alright
Hannah: Bye
Amanda: Bye bye

Reference Summary:
Hannah needs Betty's number but Amanda doesn't have it. She needs to contact Larry.

Model Summary:
Larry called Betty last time they were at the park together. Hannah doesn't know him well. She'd rather she text him.
```

Methodology

Evaluation Metrics

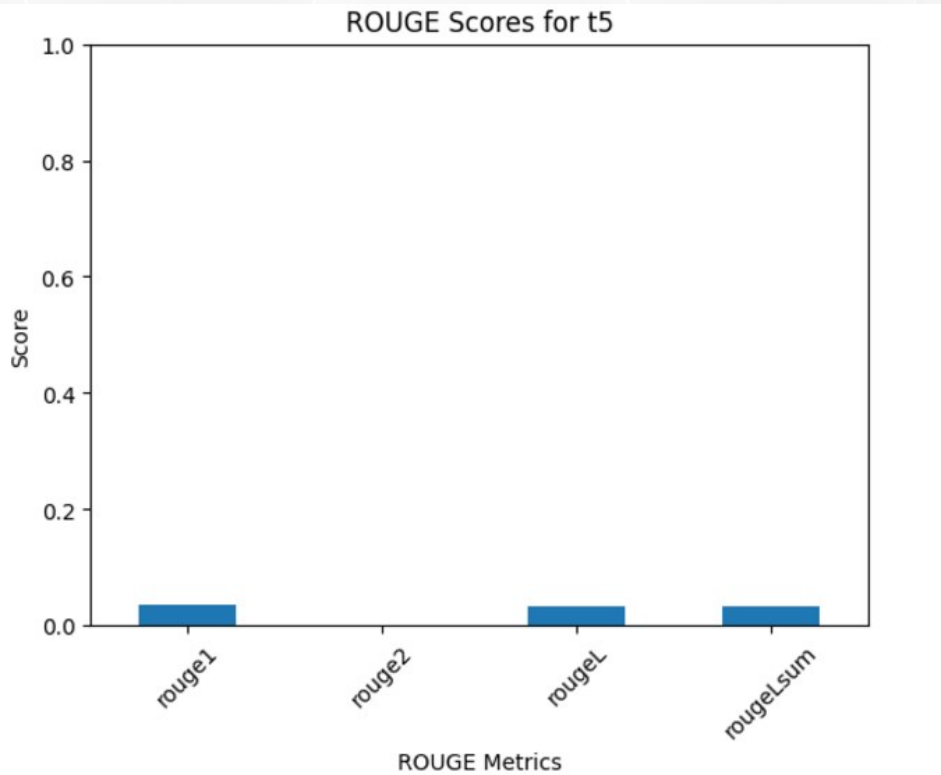
```
score = calculate_metric_on_test_ds(
    dataset_samsum['test'], rouge_metric, trainer.model, tokenizer, batch_size = 2, column_text = 'dialogue', column_summary= 'summary'
)

rouge_dict = dict((rn, score[rn]) for rn in rouge_names)


pd.DataFrame(rouge_dict, index = [f't5'] )
```

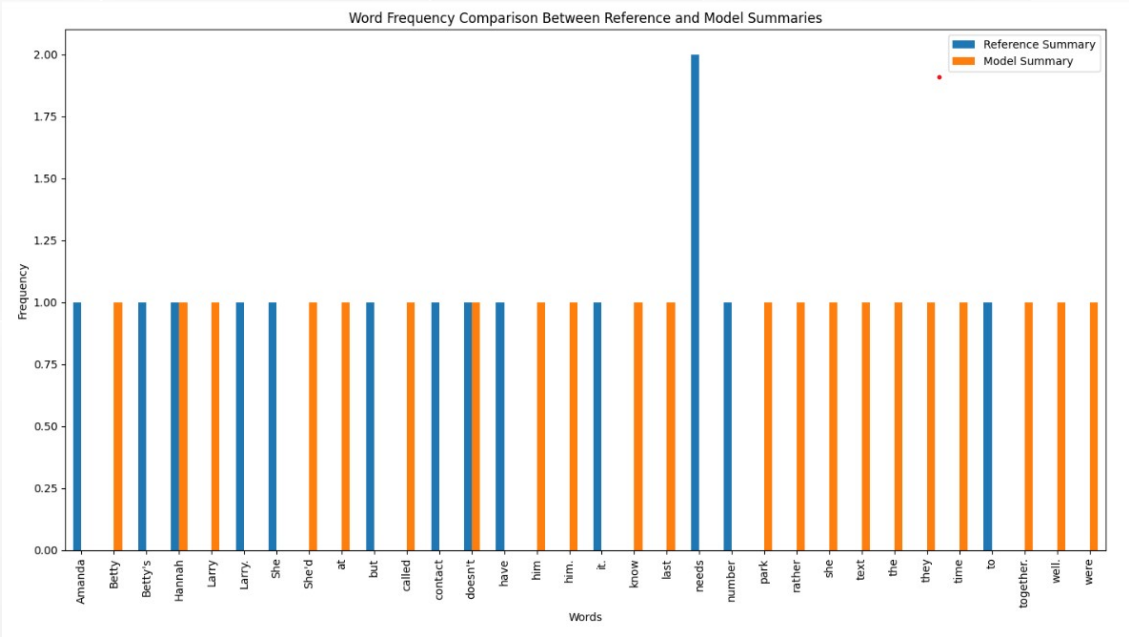
```
100%|██████████| 410/410 [07:42<00:00,  1.13s/it]
```

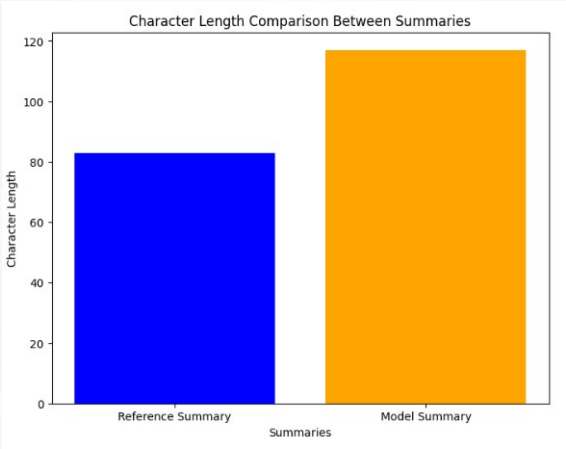
	rouge1	rouge2	rougeL	rougeLsum
t5	0.034551	0.000474	0.033456	0.03346

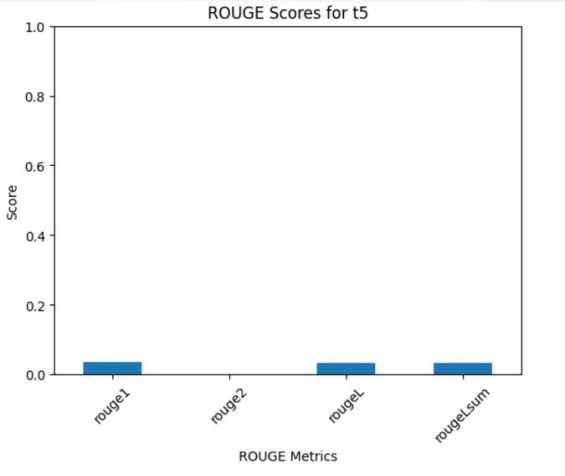


Methodology

Result Visualization







Experimental Setup

- Fine-tuned and tested **Pegasus Model** on **SAMSum corpus** using **Google Colab** with GPU resources.
- Installed essential Python libraries:
  - `transformers (v4.28.0)`, `datasets (v2.7.1)`, `evaluate (v0.2.2)`
  - `sacrebleu (v1.5.1)`, `rouge_score (v0.1.2)`, `py7zr (v0.11.3)`
- Fine-tuned `google/pegasus-cnn_dailymail` model checkpoint with `AutoTokenizer`.
- Pre-processed data with:
  - `nltk (v3.7)` for sentence tokenization.
  - `DataCollatorForSeq2Seq` for batch operations.

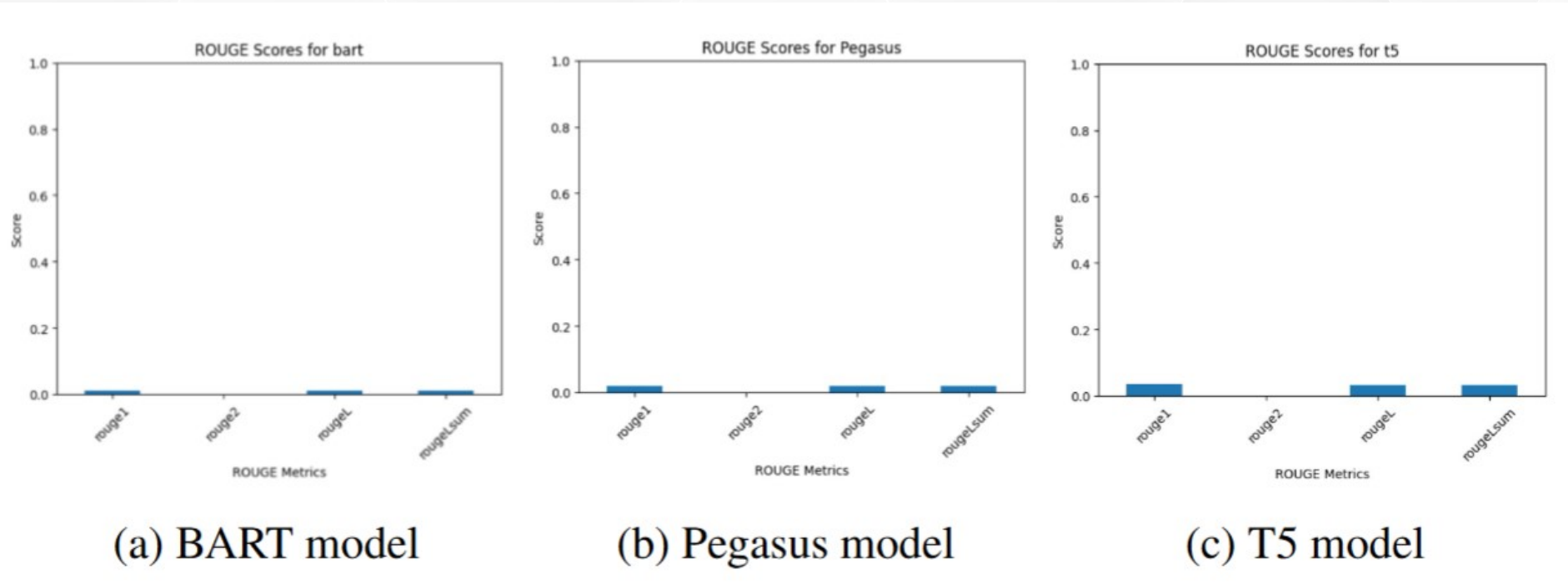
Experimental Setup

- Training parameters:
  - `num_train_epochs=1`, `warmup_steps=500`, `weight_decay=0.01`.
  - `per_device_train_batch_size=1`, `gradient_accumulation_steps=16`.
- Evaluation with ROUGE metric and results visualized using `matplotlib` and `pandas`.
- Saved model and tokenizer as `pegasus-samsum-model` for reproducibility.
- Analyzed word frequencies in generated vs. reference summaries using bar graphs.

The same setup was applied to BART and T5.

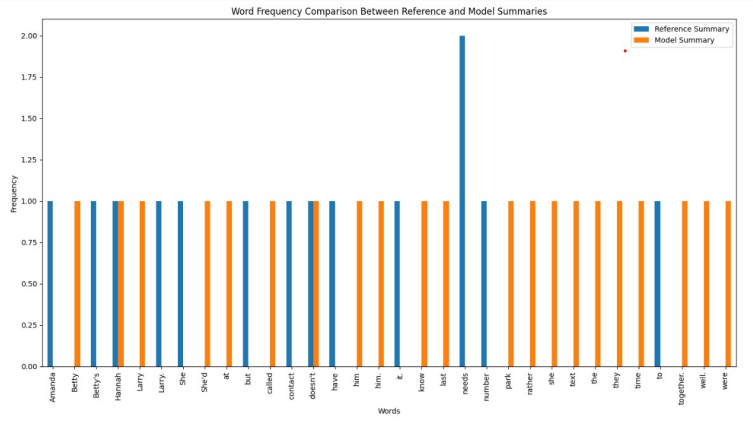
Results and Discussion

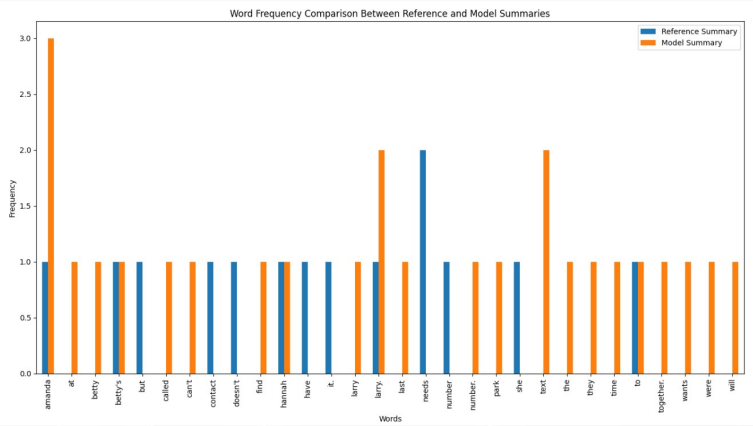
we compared the performance of three models fine-tuned on the SAMSum dataset for text summarization: Pegasus, T5, and BART. Using ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric, the output of each model was assessed, primarily giving importance to ROUGE-1, ROUGE-2, and ROUGE-L ROUGELSUM

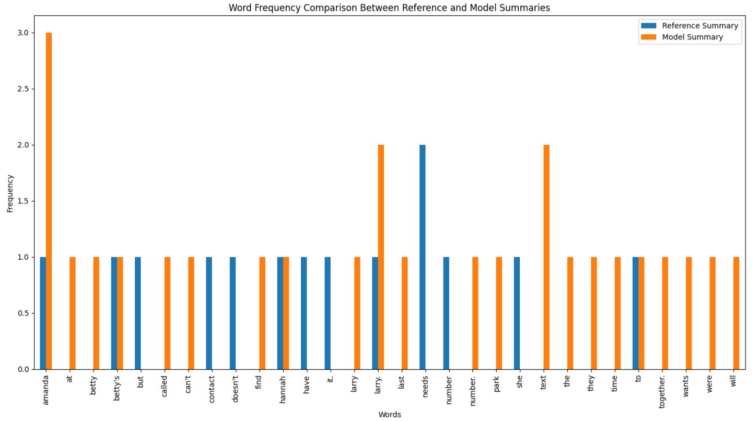


(a) BART model (b) Pegasus model (c) T5 model

Results and Discussion







Results and Discussion

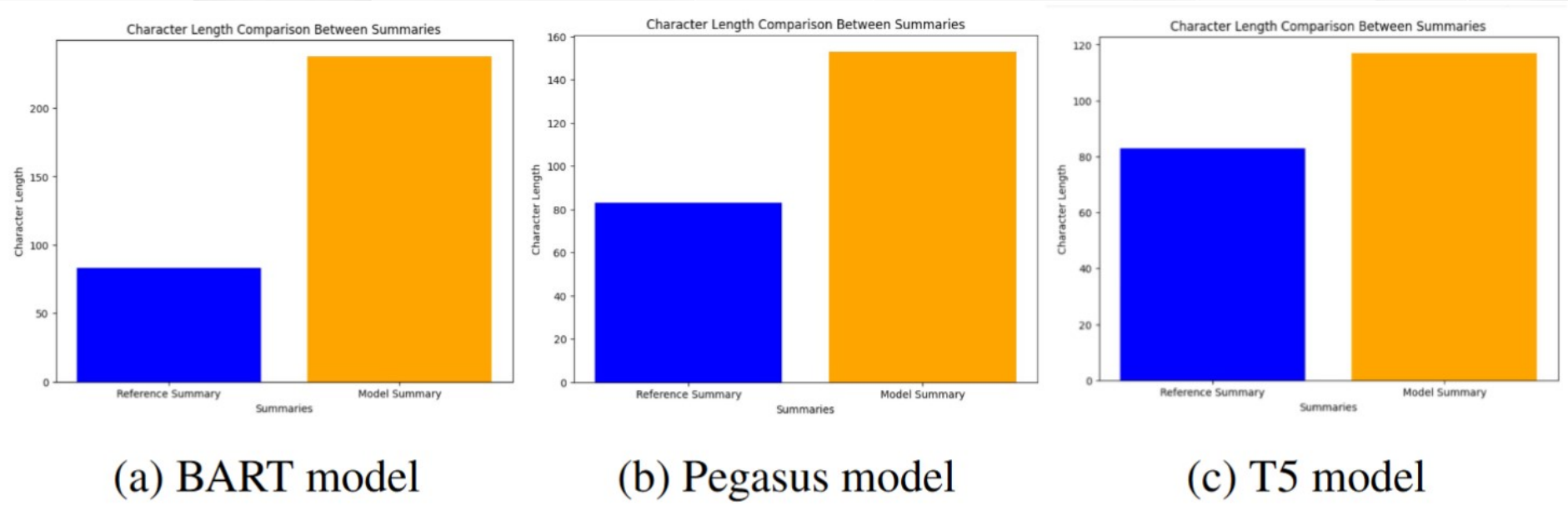
Model	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
BART	0.012645	0.000255	0.012591	0.0126
T5	0.034551	0.000474	0.033456	0.03346
PEGASUS	0.015541	0.000296	0.015525	0.01556

Table 4.1: Overall ROUGE Scores

Model	average ROUGE scores
BART	0.009523
T5	0.025985
PEGASUS	0.011731

Table 4.2: Comparison of average ROUGE scores

Results and Discussion



models are comparing the token lengths of the Reference Summaries with the token lengths of the summary generated by model

Overall, T5 showed superior performance, particularly with ROUGE-L and ROUGE-1, while BART and PEGASUS remain competitive options, depending on specific summarization needs.

Conclusion

This study demonstrates that transformer models, including BART, T5, and PEGASUS, are effective for conversational text summarization, particularly on the SAMSum dataset. Among the models, T5 achieved the highest ROUGE scores, producing clearer and more comprehensive summaries. PEGASUS and BART showed similar performance but were slightly less effective than T5 due to architectural differences. The findings highlight the potential of fine-tuning pre-trained models for conversational summarization, with applications in fields such as customer service, live chats, and content filtering.

Future Works

- Dataset Expansion: Future research can explore a broader range of data sources, such as customer service or interactional chats, to test and improve models across various dialogue types.
- Advanced Preprocessing Techniques: Techniques like dialogue segmentation and entity recognition can enhance summary quality by adding conversational context and speaker relationships, leading to more coherent summaries.
- Integration of Knowledge-Based Summarization: Incorporating external knowledge sources could enrich summaries with additional context, improving model performance by providing more background information.
- Developing User-Centric Evaluation Tools: To better assess summary quality, user feedback on relevance, coherence, and informativeness should be incorporated, particularly from those working with conversational data. This will optimize model performance based on real-world usage.

References

- M. M. Saiyyad and N. N. Patil, “Text summarization using deep learning techniques: A review,” Journal of Machine Learning Research, vol. 24, pp. 145, 2023. [Online]. Available: https : / / jmlr . org / papers / volume24 /saiyyad23a/saiyyad23a.pdf
- H. Zhang, P. S. Yu, and J. Zhang, “A systematic survey of text summarization: From statistical methods to large language models,” arXiv preprint arXiv:2406.11289, 2024.
- B. Lee, J. Park, and S. Kim, “Abstractive and extractive summarization using transformers,” Journal of Computational Linguistics, 2020.
- C.-Y. Lin, “Rouge: A package for automatic evaluation of summaries,” in Text Summarization Branches Out, 2004, pp. 74–81.

Thank You.

For your attention