[AR-T5]

Introduction

The aim of the project is to explore how text mining techniques deal with the task of text summarization, for example across a series of news articles.

The primary objective of this project is to understand the abstractive summarization capabilities of the T5 model.

Abstractive summarization involves understanding the core meaning of a text and generating a concise, coherent summary that may include rephrased or newly generated content. Not simply extract sentences, but rather rearrange the most meaningful content of the text given

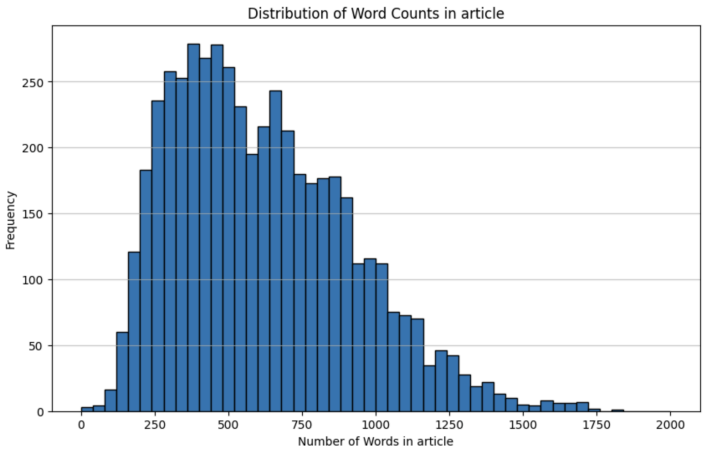
Initially, the project will explore the limitations of the pre-trained T5 base model, a model trained on a large corpus, which still shows difficulties on the given dataset.Then, the focus will be on training the base model and trying to improve its performance, aiming to generate consistent summaries.

To evaluate the model's performance on the task, the traditional ROUGE metric, will be used first, followed by BERTScore, a more modern approach .

Data

The dataset selected for this task is the CNN/DailyMail dataset, a well-known English-language dataset containing over 300,000 unique news articles authored by journalists from CNN and the Daily Mail.

Among the various versions of this dataset, the version 2.0.0 is taken into consideration, as it is best suited to the nature and requirements of the task. A restricted version of the original dataset has been used in the project; the first `5000` articles have been maintained, due to resource limitations.

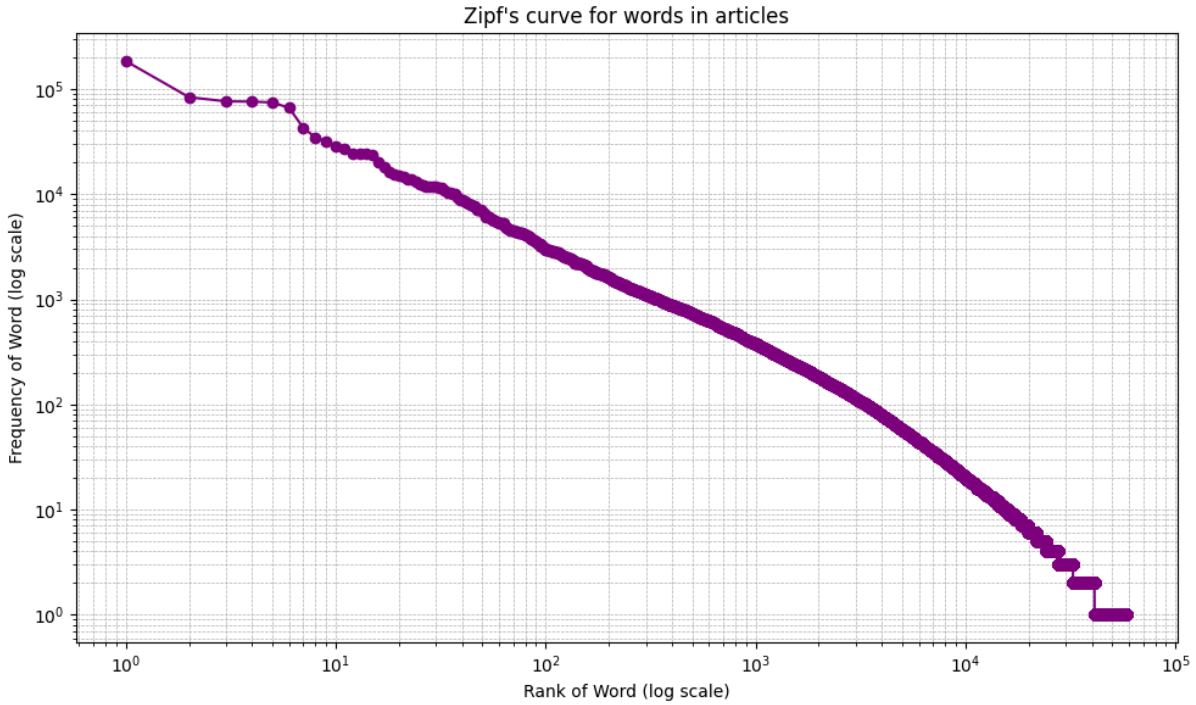


The articles vary in length, the longest is about `1,831` words while the average length is of `615` words. In contrast, the summaries are consistently brief, typically around `50` words, with a maximum of `75` and an average of `43` words.

Valuable insights can be given by the analysis of the distribution of words across articles; it is expected that this distribution will follow Zipf's Law, creating a straight line with a slope close to -1 in the log-log scale.

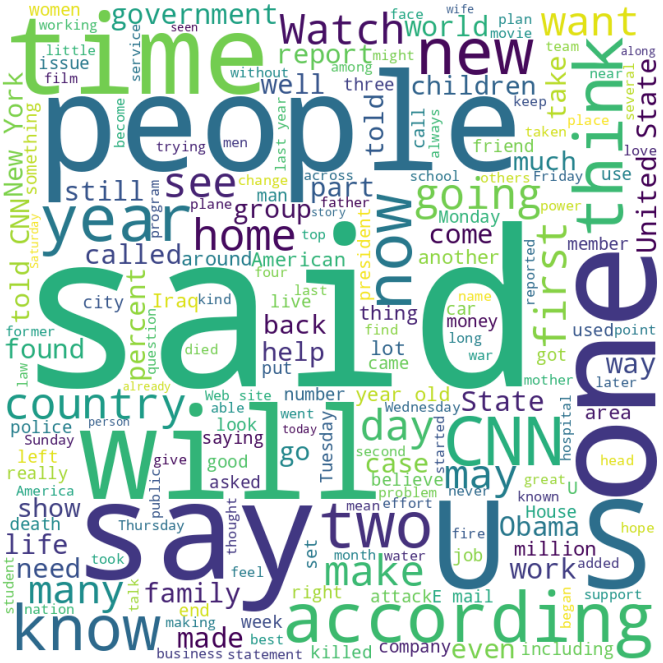
Zipf's Law reflects the natural linguistic phenomenon where common words such as stopwords ("the," "and," "is") dominate the text, whereas specialized or rare terms appear infrequently. Indeed the most frequent word across the articles considered is “the” which appears `184330 times.`

The log-log graph obtained for the text of the articles seems to respect the general law, confirming the general trend which claims that frequency of an item is inversely proportional to its rank.



The creation of the word cloud gives some insights about the nature of the articles : they are news pieces, so it’s natural that they include references to people, events, and time.

Specifically, since CNN is a U.S.-based news outlet, the word cloud contains terms that reflect its American context, such as *New York*, *Obama*, and the United *States*.



Methodology

WHY T5 BASE ?

The first strategy proposed to summarize the articles in the dataset relies on a pre-trained T5 model, specifically in its T5 Base configuration. This version of the model contains approximately 223 million parameters, which, although it seems relatively large, it offers significantly better performance on summarization tasks compared to smaller variants like T5 Small (60 million parameters).

T5, short for Text-to-Text Transfer Transformer, is a versatile encoder-decoder architecture that treats every NLP task as a text-to-text problem. For summarization, it encodes the input text and generates the summary through its decoder. One of the main strengths of T5 lies in its flexibility: the same architecture can be applied to various tasks, such as translation, question answering, and classification, simply by modifying the input format.

```
# example 1 = summarization prompt
summarize: "TEXT"

# example 2  = translation prompt
translate English to French: The weather is nice today.

# example 3  = question answering prompt
question: What is the capital of France?
context: France is a country in Europe. Its capital city is Paris.
```

The choice of exploiting a pretrained model offers several advantages; pretrained models like T5 are trained on large-scale corpora (the C4 dataset, Wikipedia, RealNews..) and are capable of few-shot learning, meaning they can generalize well with minimal task-specific data.

T5 Base is often compared with other models designed for sequence-to-sequence tasks, such as BART (Bidirectional and Auto-Regressive Transformers). While both models share a similar architecture, BART is pre trained using a denoising autoencoder objective, whereas T5 adopts a unified text-to-text approach. In the project it has been preferred to use the T5 model as a baseline because empirical comparisons indicate that T5 frequently achieves stronger results in abstractive summarization, particularly when task instructions are clearly defined via prompts.

PRE PROCESSING

The tokenizer used is the one provided by the T5 model, `T5 tokenizer` manages text normalization and handles any special tokens required by the model. It processes articles using a method called `SentencePiece`, the input is first splitted into smaller units called subword tokens and each token is then mapped to a unique numerical ID from the model's vocabulary. This procedure allows the

```python
tokenizer = T5Tokenizer.from_pretrained(MODEL)

# Function to convert text data into model inputs and targets
def preprocess_function(examples):
    inputs = [f"summarize: {article}" for article in examples['article']]
    model_inputs = tokenizer(
        inputs,
        max_length=MAX_LENGTH,
        truncation=True,
        padding='max_length'
    )

    targets = [summary for summary in examples['highlights']]
    with tokenizer.as_target_tokenizer():
        labels = tokenizer(
            targets,
            max_length=MAX_LENGTH,
            truncation=True,
            padding='max_length'
        )

    model_inputs["labels"] = labels["input_ids"]
    return model_inputs
```

tokenizer to effectively handle rare or unknown words by breaking them into familiar pieces. These token IDs are the actual input that the T5 model receives. When generating output, the tokenizer reverses this process by converting the sequence of token IDs produced by the model back into readable text.

To format the input as required by the T5 model, the tokenizer is used within the `preprocess_function`.

Each article is prepended with the task prefix `"summarize: "` to clearly indicate the summarization task to the model (see example 1 before). The tokenizer then processes the inputs, applying truncation and padding so that all sequences have the same fixed length, `MAX_LENGTH`. The same tokenization, truncation, and padding steps are applied to the reference summaries as well.

T5 BASELINE

To test the base model's summarization capabilities, a few examples were generated using a dedicated function. Summaries were limited to 40–70 tokens to match the length of the reference highlights and simplify comparison. The model output, a tensor of token IDs, is at the end converted by the tokenizer into readable text.

Some observations emerged:

- Comparing the length of the summaries generated by the model and the references, no consistent discrepancy can be observed; the model produces summaries of adequate length.
- Regarding punctuation marks and the use of lowercase/uppercase letters, the model seems not to handle them correctly, which may be due to the fact that the model has not yet been fully refined.
- Indeed, the model captures the semantics and is able to identify the most important chunks of information from the article, although it does not express them with fully correct grammar

	ARTICLE 1	ARTICLE 2	ARTICLE 3
T5 BASE	young actor says he has no plans to fritter his cash away on fast cars, drink and celebrity parties . at 18, he will be able to gamble in a casino, buy a drink in a pub or see "Hostel: Part II" details of how he'll mark his landmark birthday are under wraps WORDS : 44	mentally ill inmates housed on the "forgotten floor" of a pretrial detention facility in florida . inmates with the most severe mental illnesses are incarcerated until they're ready to appear . one-third of all people in florida-dade county jails are mentally ill . WORDS : 36	driver: "the whole bridge from one side of the Mississippi to the other just gave way" "it just gave a rumble real quick, and it all just gave way," survivor says . "there's cars in the water, there's cars on fire. the whole bridge is down," he says . WORDS : 47
REFERENCE	Harry Potter star Daniel Radcliffe gets £20M fortune as he turns 18 Monday . Young actor says he has no plans to fritter his cash away . Radcliffe's earnings from first five Potter films have been held in trust fund . WORDS ; 38	Mentally ill inmates in Miami are housed on the "forgotten floor" Judge Steven LeifMan says most are there as a result of "avoidable felonies" While CNN tours facility, patient shouts: "I am the son of the president" LeifMan says the system is unjust and he's fighting for change . WORDS : 41	NEW: "I thought I was going to die," driver says . Man says pickup truck was folded in half; he just has cut on face . Driver: "I probably had a 30-, 35-foot free fall" Minnesota bridge collapsed during rush hour Wednesday . WORDS : 39

ROUGE AND BERT SCORE

Rouge

To evaluate the initial model outputs, the ROUGE score was primarily used. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a set of metrics designed to assess the quality of automatic text summaries by comparing them to human-written reference summaries. `ROUGE` measures the overlap between the generated summary and the reference summary using various techniques, such as n-gram matching, word sequences, and word pairs.

In this evaluation, three main versions of ROUGE were considered:

- `ROUGE-1`: measures the overlap of unigrams (individual words) between the generated and reference summaries.
- `ROUGE-2`: measures the overlap of bigrams (pairs of consecutive words), which helps capture fluency and basic phrase matching.
- `ROUGE-L`: measures the longest common subsequence between the generated summary and the reference, reflecting sentence-level structure similarity.

The ROUGE metric was computed using the `evaluate` library, together with a custom `compute_metrics` function. This function converts the model's predicted token IDs and reference labels into readable text before calculating the ROUGE scores. It also tracks the average length of the generated summaries to support a fair interpretation of the results.

Evaluation was performed using the Hugging Face `Trainer` class, with metrics computed every 200 steps and a custom `preprocess_logits_for_metrics` function used to extract predictions from the model's output.

BERTScore

To evaluate the semantic quality of the generated summaries, BERTScore was also used as a complementary evaluation metric.Unlike ROUGE, which relies on surface-level n-gram overlaps, BERTScore measures semantic similarity by computing the cosine similarity between contextual embeddings of the generated and reference summaries.

These embeddings are obtained from the pre-trained language model `bert-base-uncased`. BERTScore calculates precision, recall, and F1-score based on how closely the predicted tokens align with the reference tokens in embedding space.

- `precision` : is the model's output semantically similar to parts of the reference?
- `recall` → is most of the reference's meaning captured by the candidate?
- `F1-score` → semantic alignment in both directions.

$$\text{Recall} = \frac{1}{m}\sum_{j=1}^{m}\max_{i}\cos(x_i, y_j) \qquad \text{Precision} = \frac{1}{n}\sum_{i=1}^{n}\max_{j}\cos(x_i, y_j$$

The metric was computed using the `bert_score` library in combination with a custom `compute_metrics2` function; as before evaluation was performed using Hugging Face's `Trainer` class, with quite the same configuration and a slightly different `preprocess_logits_for_metrics2` function.

FINE TUNING

The overall performance, measured by both ROUGE and BERTScore, was unsatisfactory. To improve the quality of the generated summaries, fine-tuning the pre-trained T5 base model proved to be a useful strategy, especially considering the relatively small size of the dataset.

```
training_args3 = TrainingArguments(
    output_dir=OUT_DIR,
    num_train_epochs=EPOCHS,
    per_device_train_batch_size=BATCH_SIZE,
    per_device_eval_batch_size=BATCH_SIZE,
    warmup_steps=500,
    weight_decay=0.01,
    gradient_accumulation_steps = 2,
    logging_dir=OUT_DIR,
    logging_steps=10,
    eval_strategy='steps',
    eval_steps= 625,
    save_strategy='epoch',
    save_total_limit=2,
    report_to='tensorboard',
    learning_rate=0.0001,
    dataloader_num_workers=2,
    fp16=True
)
```

Fine-tuning was performed using Hugging Face's `TrainingArguments` and `Trainer`, not just for evaluation but to actively train the baseline model on the articles in the dataset. Mixed precision, `fp16=True`, was enabled to speed up computation.

Due to memory limitations and available resources, the model was trained for only 5 epochs, which still resulted in noticeable improvements.

A batch size of 4 per device was used, with `gradients accumulated` over 2 steps to effectively increase the `batch size` without requiring additional memory.

The `learning rate` was set to 0.0001, while `weight decay` of 0.01 was applied to reduce overfitting.

Evaluation was performed every 625 steps to enable tracking metric improvements during training, as the total number of samples is 5000.

Step	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Gen Len
625	0.193200	0.271215	0.533100	0.270600	0.494200	50.191000
1250	0.223200	0.273832	0.530900	0.267400	0.491000	50.192000
1875	0.234500	0.268922	0.530700	0.267500	0.491800	50.191000
2500	0.219500	0.274322	0.528100	0.267000	0.489500	50.191000
3125	0.181300	0.276350	0.528000	0.267200	0.488600	50.191000

Result and Analysis

The initial results obtained using both ROUGE and BERTScore were unsatisfactory, indicating that the T5-Base model struggled to generalize effectively to the summarization task in this specific dataset.

In terms of ROUGE, the scores suggest that while the model captured some relevant content, it lacked precision and fluency:

- `eval_rouge1: 0.2817` : 28% overlap in individual words, lots of key terms are misrepresenting
- `eval_rouge2: 0.1351` : this value is close to 0, meaning that the model captures few phrase-level similarity but struggles to generate fluent or accurate word sequences matching the references.

From a semantic perspective, BERTScore results further confirmed the model's poor initial performance:

- `eval_bertscore_precision: 0.062` , only the 6% of the generated content is semantically relevant (compared to reference)
- `eval_bertscore_recall: 0.37`, the 37% of the reference meaning is covered in the prediction
- `eval_bertscore_f1: 0.165`, this is the main metric to consider, it is relatively low, meaning the generated outputs are not strongly aligned semantically with the references.

These findings suggest that the model was not sufficiently optimized for this dataset and required additional training. After performing fine-tuning on the dataset, the quality of the generated summaries improved significantly.
Final results showed substantial gains:

- `ROUGE-1 = 0.58, ROUGE-L = 0.53` → stronger lexical and structural overlap with reference summaries.
- `Precision = 0.52, Recall = 0.54 and F1 = 0.53` → better semantic alignment between generated and reference summaries.

Although the summaries are still not at the level of human-written content, these scores indicate that fine-tuning allowed the model to produce outputs that are substantially more accurate, fluent, and semantically meaningful.

	T5 BASE	T5 TRAINED
ROUGE SCORE	Rouge1 : 0.2810 rouge2 : 0.135 rougeL : 0.259	rouge1 : 0.5801 rouge2 : 0.3111 rougeL : 0.5365

	T5 BASE	T5 TRAINED
BERTScore	precision : 0.062 recall : 0.37 f1 : 0.16	precision : 0.52 recall : 0.54 f1 : 0.53

An analysis of the output format also reveals that the fine-tuned model demonstrates improved handling of capitalization, punctuation, and grammar, producing summaries that are more coherent and stylistically clean.

T5 BASE	Zully Broussard gave one of her kidneys to a stranger . her generosity paired up with big data, and six patients received transplants . "i know this entire journey is much bigger than all of us," she says .
T5 TRAINED	Zully Broussard gave one kidney to a stranger . Data processing of donor-recipient pairs multiplied her gift . Six patients received kidney transplants .

Conclusion

This project explored the use of a pre-trained T5 model for automatic text summarization.

The work unfolds through five key steps, which are also reported in the structure of Colaboratory notebook

The main phases of the project were:

- Importing and applying a pre-trained T5-Base model
- Evaluating the initial outputs using ROUGE and BERTScore
- Fine-tuning the model
- Performing a second evaluation to assess improvements
- Analyzing the final results and drawing conclusions

Initially, a pre-trained model was loaded and used to generate summaries without any additional training, with the aim of creating a baseline to assess the model's general capabilities. The second phase focused on evaluating the outputs using both lexical (ROUGE) and semantic (BERTScore) metrics, which revealed that although the T5-Base model was able to generate summaries, its initial performance on the dataset was limited.

To address this, a fine-tuning process was carried out, resulting in significant improvements in summary quality; these improvements were reflected not only in the evaluation scores but also in the overall structure and linguistic accuracy of the generated text.

The project demonstrates that fine-tuning a general-purpose language model like T5 can lead to substantial improvements in summarization tasks, even with limited data (5000 sample train dataset)

I encountered several challenges throughout the project.

a) Evaluating the actual quality of the summaries is complex.

   Automatic metrics attempt to account for grammar, adequacy, and fluency, but quantifying the quality of a summary is a difficult task; in reality, it can only be done a posteriori, after reading the articles and reasoning about what information is truly meaningful.

   Judging the quality of an “open” text is always tricky. The comparison with human-written references provides a baseline that helps avoid working blindly. However, each reference summary was written by a single person, offering just one subjective interpretation of the article.

   Comparing the generated summaries to multiple human-written versions for each article might have provided a broader perspective and allowed a fairer evaluation of the model's ability to capture relevant information.

b) Integrating different tools, such as pre-trained models, datasets, and training frameworks, was another challenge. In this context, I relied on AI assistance to better understand how these frameworks work, what the different parameters represent, and how specific libraries function in practice.

REMARK : I share the folder where I have saved the final model, so that it can be exploited directly without the need to wait for the end of training.
It just needs to insert the path where you have saved the folder on your drive to use it