Natural Language Processing Project - Master's Degree in Data Science

Abstractive Summarization on WikiHow

Alfio Leanza 910235

July 9, 2025

Abstract

This project benchmarks different abstractive summarization approaches on the WikiHow dataset. The focus is on model architecture, training strategies, and evaluation (with ROUGE metrics and BERTscore). There is the comparison among GRU-based Seq2Seq model with modern transformer architectures, including Distilbart, base and fine-tuned, and the instruction-tuned FLAN-T5. Experiments span zero-shot and few-shot prompting,full fine-tuning, and parameter efficient fine-tuning (PEFT) methods such as LoRA and Prefix Tuning. Performance is assessed using ROUGE and BERTScore metrics. Results show that Distilbart fine-tuned achieves the best balance of lexical and semantic quality (ROUGE-1 equal to 0.46 and BERTscore equal to 0.90).

1 Introduction

Automatic text summarization is an old problem in Natural Language Processing. The goal is to turn a long piece of text into a short, clear version that keeps the main ideas. In abstractive summarization the model writes new sentences instead of copying them, but it still has trouble keeping facts correct, covering every key point, and sounding natural especially on how-to articles, which are very different from the news and Wikipedia pages most models see during training.

WikiHow makes the job even harder. Each article is a step-by-step guide that mixes short commands with longer explanations. The average article is about 34 words, yet the official summary is only one sentence of about 5 words. A model has to shrink the text by roughly ten times, keep the order of the steps, and avoid inventing steps that never appeared in the source.

To see how today's models handle these issues, were tested several systems under the same conditions. It starts with a classic GRU-based Seq2Seq model and then moves

to transformer models of increasing power: DistilBART (both the base version and a fully fine-tuned one) and the instruction-tuned FLAN-T5.

For each model four decoding styles have been tried:

- Greedy decoding;
- Beam search;
- Top-k;
- Top-p.

In particular, for the FLAN-T5 were tested the results experimenting with the prompting strategies of:

- Zero-shot;
- One-shot;
- Few-shot.

Judging the results with ROUGE-1/2/L for word overlap and BERTScore for meaning. The experiments show that a fully fine-tuned DistilBART gives the best mix of word-level accuracy and meaning on WikiHow, scoring 0.46 ROUGE-1 and 0.90 BERTScore.

2 Dataset Exploration

The WikiHow summarization corpus is built from the collaborative WikiHow website, where volunteers post step-by-step tutorials on everyday tasks ("How to change a flat tire", "How to bake sourdough", and so on). Each page contains:

- text - the full body of the article, organized into numbered steps with short explanations;
- summary – a single sentence written by the original author that should capture the overall goal of the tutorial.

Unlike news-centric datasets (e.g., CNN/DailyMail) the language is informal, full of imperatives ("Add the yeast", "Check the bolt pattern") and rich in lists, photos, and side notes.

For these experiments was loaded the official train split only, which holds 128 543 article–summary pairs. Then were made three helper columns:

- len_art – number of words in the article;
- len_sum – number of words in the summary;
- ratio – $len\_sum \div len\_art$.

Document

Statistic	Value
Count	128 543
Mean	34.6
Median	34
25%	28
75%	40
Max	141

Summary

Statistic	Value
Count	128 543
Mean	5.2
Median	5
25%	4
75%	6
Max	24

Length-ratio

Statistic	Value
Count	128 543
Mean	0.17
Median	0.14
25%	0.10
75%	0.20
Max	8.00

WikiHow articles are concise, list-style guides, but their one-sentence summaries are even more compact.

Figures 1 and 2 plot the word-count spread for documents and summaries, while Figure 3 shows the Summary-to-Document Length Ratio (SDLR). The SDLR peak sits around 0 .15, yet a thin tail stretches past 0 .5 and even above 1, marking a handful of mismatched pairs where the "summary" is as long as—or longer than—the source.

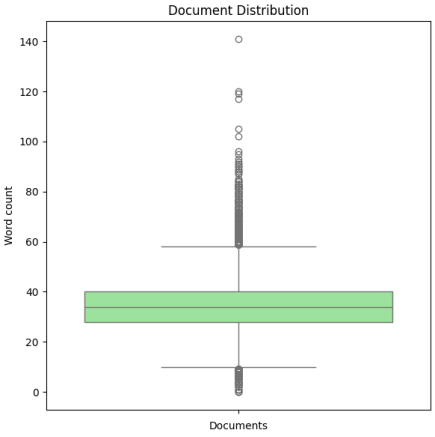

Figure 1: Document Distribution

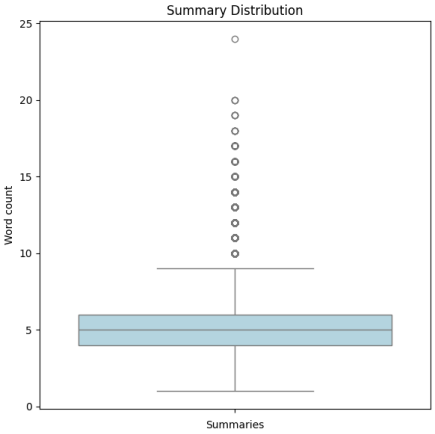

Figure 2: Summary Distribution

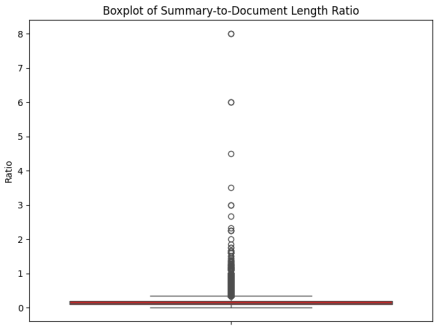

Figure 3: Summary-to-Document Length Ratio

These outliers confirm the need for the extreme 0 .5 ratio filter applied in the data-cleaning step in order to prevent and remove any possible noises. In total the filter caused a loss of about 1900 observations.

3 Experiments

The study relies on three modeling choices. A GRU-based encoder-decoder with attention was built entirely from scratch to establish a lightweight baseline. Then was evaluated DistilBART twice: once in its unchanged base form to asses various decoding strategies, and once after full fine-tuning on the WikiHow corpus to see how much task-specific training helps. Finally, was explored prompt-only strategies with FLAN-T5-small, using the specifics prompting strategies while keeping the model weights frozen.

3.1 Seq2Seq Baseline Model

The starting point is a straightforward encoder–decoder network enhanced with attention. The encoder is a unidirectional GRU that processes the tokenized input document and outputs a sequence of hidden states. The

decoder is also a GRU, which, at each time step consults a scaled dot-product attention mechanism that lets it focus on the most relevant parts of the encoder trace; a small "copy gate" then decides whether to draw the next word from the softmax vocabulary or copy it directly from the source—handy for proper names and rare terms.

Training is deliberately lightweight. We reserve the bulk of the corpus for learning, keep a sliver for validation, and hold out a small slice for final testing. Cross-entropy loss with label smoothing guides the updates, while dropout and gradient clipping keep gradients stable. The learning rate warms up briefly, then follows a gentle cosine decay. Teacher forcing is scheduled: early batches feed the gold token back into the decoder, later batches oblige it to cope with its own previous guess, so the network gradually becomes self-reliant.

Inference and Decoding

- **Greedy decoding**
- **Top-p (nucleus sampling)** with p=0.9
- **Top-k sampling** with k=50
- **Beam search** with b=4

Was implemented a lenght penalty function to rescales beam-search scores so that longer candidate summaries are not automatically penalized for having more tokens. Its sole goal is to balance brevity with completeness, letting the decoder choose the most informative sequence rather than the shortest one.

3.2 DistillBART

BASE

Firstly was ran the standard *sshleifer/distilbart-cnn-12-6* checkpoint in 8-bit mode so it would fit comfortably on Colab GPU. With the original tokenizer we clipped each article to roughly a couple of hundred tokens and forced the decoder to stay within a short, single-sentence budget. Even without task-specific learning the model produced serviceable headlines, giving us a clean zero-shot yard-stick. As before, were generated with greedy, beam, top-k and nucleus search and logged ROUGE + BERTScore for later comparison.

Fine-tuned

For the fine-tuned variant was kept the original *sshleifer/distilbart-cnn-12-6* backbone frozen and grafted small LoRA adapters onto every self-attention projection ($q\_proj, k\_proj, v\_proj, out\_proj$).

The LoRA configuration used:

- rank=16;
- α=32;
- dropout=0.05.

Training relied on a 30 k-example slice of the cleaned WikiHow corpus, in order to successfully complete the training. Each article was capped at 165 tokens and each summary at 12 tokens, based on the distribution of the corpus.

The model was fine-tuned for only one epoch with:

- batch size=4
- gradient accumulation=4
- learning rate=2×10^{-4}

After training we saved only the LoRA weights—just a few megabytes—which can be re-attached to the 8-bit base model at inference time.

With the adapter in place DistilBART produced noticeably sharper and more faithful one-sentence summaries than the zero-shot baseline, especially under beam search.

The model was fine-tuned using LoRA adapters only, to assess its potential, and this configuration achieved the best results.

3.3 Flan-T5

Finally was tested the performance of the Flan-T5 model which is instruction-tuned and prompting to it the instructions as:

"You are an expert WikiHow editor."
"Summarize this article:"

Besides this, three prompting setups was tested:

- **Zero-shot:** we give the model only the instruction (e.g., "Summarize: <document>") plus the document itself—no example summaries are shown.
- **One-shot:** the prompt starts with a single article-headline pair, followed by the new document the model must summarize.
- **Few-shot:** three, or more, article-headline pairs are concatenated ahead of the target document so the model can pick up the desired style before generating its own headline.

Lastly, as the other experiments, were applied the decoding strategies of greedy, beam search, top-k and top-p.

4 Results

4.1 Evaluation Metrics

To evaluate the quality of the generated summaries, were adopted both lexical overlap metrics and semantic similarity measures. These include ROUGE variants and BERTScore, which together allow to evaluate both syntactic and semantic accuracy.

ROUGE-n Measures the exact overlap of n-grams between the candidate summary and the reference, giving quick recall-style evidence of lexical coverage.

$$\text{Rouge-n}_{F_1} = \frac{2\,\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Precision} = \frac{\#\,\text{overlapping } n\text{-grams}}{\#\,n\text{-grams in candidate}}$$

$$\text{Recall} = \frac{\#\,\text{overlapping } n\text{-grams}}{\#\,n\text{-grams in reference}}$$

ROUGE-L Looks at the longest common subsequence(LCS), rewarding candidates that preserve the overall sentence order and structure even if contiguous n-grams differ. Let LCS(X,Y) be the length of the LCS between the candidate X and reference Y:

$$P_{\text{LCS}} = \frac{LCS(X,Y)}{|X|}$$

$$R_{\text{LCS}} = \frac{LCS(X,Y)}{|Y|}$$

$$\text{ROUGE-L}_{F_1} = \frac{(1+\beta^2)\,P_{\text{LCS}}\,R_{\text{LCS}}}{R_{\text{LCS}} + \beta^2\,P_{\text{LCS}}},$$

where β balances Precision and Recall

BERTScore Uses contextual BERT embeddings to align tokens and computes cosine similarity, capturing semantic equivalence and crediting accurate paraphrases. Let x_i and y_j be BERT embeddings of tokens from the candidate and reference, respectively:

$$\text{Precision} = \frac{1}{n}\sum_{i=1}^{n}\max_{j}\cos\big(\mathbf{x}_i, \mathbf{y}_j\big) \qquad \text{BERTScore}_{F_1} = \frac{2\,\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$$

$$\text{Recall} = \frac{1}{m}\sum_{j=1}^{m}\max_{i}\cos\big(\mathbf{y}_j, \mathbf{x}_i\big)$$

4.2 Model performance

Model performance on the WikiHow test set

Model	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore F1
FLAN-T5 base – Zero-shot (top-p)	0.1688	0.0640	0.1581	0.8425
FLAN-T5 base – One-shot (greedy)	0.3663	0.1839	0.3498	0.8832
FLAN-T5 base – Few-shot (beam)	0.3738	0.2029	0.3596	0.8839
DistilBART base (beam)	0.2300	0.0941	0.2143	0.8510
DistilBART fine-tuned (beam)	**0.4607**	**0.2566**	**0.4432**	**0.9043**
GRU Seq2Seq (greedy)	0.2963	0.1065	0.2920	0.5487

How we can see the DistillBART fine-tuned beats each model on each metric.

4.3 Explainability Analysis

To understand *what* parts of the article support the probability of each generated token, we employed **Integrated Gradients** (IG), an attribution method that combines analytical simplicity and theoretical grounding. IG stems from the "*gradient* × *input*" rule, which assigns each feature x_i an importance proportional to the local sensitivity of the model:

$$\text{Attribution}(x_i) = x_i \times \frac{\partial f(x)}{\partial x_i}.$$

The novelty of IG is that, instead of computing the gradient at a single point, it integrates the gradient along the trajectory connecting a neutral baseline (all tokens `[PAD]`) to the actual input. This reduces saturation effects and provides more stable heatmaps: for each pair *token source* → *token output* we obtain a score that quantifies the real contribution of that text fragment, making the behavior of the model more transparent and amenable to critical analysis.

Saliency heat-maps, obtained with this method on the fine-tuned DistilBART, reveal that the generated tokens almost always draw on the same source tokens. When moving from greedy and beam to top-k and top-p, the columns become slightly more blurred, a sign that the context consulted is broadening without altering the information core; thus, the variation is stylistic and not content-related.

These observations converge on one point: the choice of decoder influences form, not substance. Regardless of the generation strategy, the model systematically retrieves the same set of information and distributes attention over coherent text fragments, a signal of semantic robustness and fine-tuning that has effectively aligned summarizer behavior with the needs of the WikiHow domain.

5 Conclusions

This project compared a GRU baseline, DistilBART (base + LoRA fine-tune) and prompt-based FLAN-T5 on the WikiHow dataset. LoRA-tuned DistilBART delivered the strongest ROUGE and BERTScore, while FLAN-T5 closed most of the gap simply by adding three in-context examples—showing that careful prompting can rival task-specific training. The GRU model trailed as a classical reference. Different decoding strategies shifted style more than substance, and Integrated-Gradients heat-maps confirmed that all decoders focus on the same high-salience tokens. Overall, lightweight fine-tuning or few-shot prompting provides an efficient path to high-quality WikiHow summaries without the cost of full model retraining.