

Efficient Fine-Tuning of Transformer Models for Abstractive Summarization

Dontha Chandra Mahesh Goud¹, Prof. Kunjam Nageswara Rao²

¹*MTech Student, Department of Computer Science and Systems Engineering, Andhra University College of Engineering, Visakhapatnam, India*

²*Professor, Department of Computer Science and Systems Engineering, Andhra University College of Engineering, Visakhapatnam, India*

Abstract—Abstractive text summarization with large pre-trained transformers is often constrained by the computational cost of full fine-tuning, which limits accessibility for researchers and practitioners with modest hardware budgets.

This paper presents a lightweight, end-to-end system for abstractive summarization that fine-tunes multiple encoder-decoder architectures, namely BART, T5, FLAN-T5, and mBART, using Low-Rank Adaptation (LoRA), and compares them against their non-adapted counterparts (BART-base, T5-small, DistilBART, and PEGASUS) on the CNN/DailyMail news summarization dataset. We describe the fine-tuning pipeline, the evaluation protocol based on BLEU and ROUGE (ROUGE-1, ROUGE-2, ROUGE-L), and a deployable web interface that exposes the best-performing model as an interactive summarization service.

Preliminary evaluation shows that LoRA-adapted variants of BART, T5, and FLAN-T5 consistently achieve higher ROUGE scores than their frozen or non-adapted baselines, while requiring only a small fraction of trainable parameters compared to full fine-tuning. We also report an anomalous failure mode observed for a pretrained PEGASUS baseline, which produced degenerate outputs under our decoding configuration, underscoring the sensitivity of summarization pipelines to generation hyperparameters. We situate the system relative to recent surveys and domain-specific summarization studies, and discuss the case for evaluating abstractive summaries with more than a single overlap-based metric. We conclude by outlining a path toward large-scale, statistically robust evaluation of the proposed pipeline.

Index Terms—text summarization, LoRA, parameter-efficient fine-tuning, BART, T5, PEGASUS, ROUGE, transformers

I. INTRODUCTION

Automatic text summarization aims to condense a source document into a shorter passage that preserves its most salient information. Two broad paradigms exist: extractive summarization, which selects and concatenates existing sentences from the source, and abstractive summarization, which generates novel phrasing, potentially paraphrasing and synthesizing information across the document, as catalogued in recent surveys of the field [1], [5]. Since the introduction of large pretrained sequence-to-sequence transformers such as BART, T5, PEGASUS, and their multilingual and instruction-tuned derivatives (mBART, FLAN-T5), abstractive summarization quality has improved substantially. However, these models typically contain hundreds of millions to billions of parameters, and full fine-tuning on a downstream summarization dataset requires updating all of these parameters, which is computationally expensive and often infeasible for individual researchers, students, or small organizations.

Parameter-efficient fine-tuning (PEFT) methods have emerged as a practical alternative to full fine-tuning. Among these, Low-Rank Adaptation (LoRA) freezes the pretrained weight matrices of a transformer and injects a small number of trainable low-rank decomposition matrices into selected layers (typically the attention projection layers), reducing the number of trainable parameters by several orders of magnitude while retaining most of the downstream task performance achievable through full fine-tuning. LoRA has been widely adopted for large language model adaptation, but its behavior across multiple

summarization-oriented encoder-decoder architectures, evaluated side by side under an identical protocol, is less commonly reported in a single, unified comparison.

This paper makes the following contributions:

- 1) We implement and compare eight summarization configurations on the CNN/DailyMail dataset: four models fine-tuned with LoRA (BART-LoRA, T5-LoRA, FLAN-T5-LoRA, mBART-LoRA) and four non-adapted baselines used directly from their pretrained or distilled checkpoints (BART-base, T5-small, DistilBART, PEGASUS).
- 2) We describe a reproducible generation and evaluation pipeline using beam search decoding and standard automatic metrics (BLEU, ROUGE-1, ROUGE-2, ROUGE-L).
- 3) We report and analyze a degenerate-output failure case observed with the PEGASUS baseline under our decoding configuration, which is instructive for practitioners deploying pretrained summarization checkpoints without dataset-specific pre- and post-processing.
- 4) We package the best-performing configuration behind a lightweight web application, allowing a user to submit arbitrary text and receive a generated summary through a browser-based interface backed by a REST endpoint.
- 5) We relate our findings to recent extractive, abstractive, and domain-specific summarization studies [3], [4], [6]–[10], and discuss the limitations of relying on a single reference-based metric for evaluating abstractive summaries.

The remainder of this paper is organized as follows. Section II reviews related work on general summarization surveys, extractive summarization advances, abstractive and LLM-assisted summarization, and domain-specific summarization systems.

Section III describes the proposed system, including the model configurations, the fine-tuning procedure, and the web application architecture. Section IV details the experimental setup. Section V presents results and discussion. Section VI states the limitations of the present study, and Section VII concludes with directions for future work.

II. RELATED WORK

A. Survey Perspectives on Text Summarization

Several recent surveys frame the landscape our system draws on. Widyassari et al. [1] review automatic text summarization techniques and methods broadly, cataloguing pre-processing, feature extraction, and summary-generation stages common to both extractive and abstractive pipelines, a taxonomy that motivates our own separation between LoRA-adapted and non-adapted candidate models. Altmami and El Bachir Menai [2] survey summarization specifically for scientific articles, highlighting domain-specific challenges, such as structured discourse and citation handling, that are absent from the general-news domain of CNN/DailyMail used in this work, which reinforces the caution in Section VI against generalizing our results across domains. Yadav et al. [5] survey feature-based (extractive) summarization methods, providing a comprehensive state-of-the-art view of the extractive family that complements the abstractive focus of our own pipeline.

B. Extractive Summarization Advances

Building on the survey line above, Azam et al. [7] review current trends and advances in extractive text summarization, and Yang et al. [8] propose a generative adversarial network for extractive summarization that combines transductive and reinforcement learning to select salient sentences. These works illustrate that extractive techniques remain an active complement to abstractive neural approaches rather than a superseded predecessor, and motivate our inclusion of both LoRA-adapted abstractive models and non-adapted baselines side by side.

C. Abstractive and LLM-Assisted Summarization

Zhang et al. [3] use ChatGPT for extractive summarization under a faithfulness objective, part of a broader trend toward using large language models either as summarizers or as evaluators of summary quality. Fatima et al. [9] propose a multi-encoder architecture for abstractive summarization in Arabic-like scripts, underscoring that summary quality and metric behavior are sensitive to script and language morphology, not only to topical domain. Kadhim et al. [10] combine classical NLP techniques with feature-based scoring in a hybrid summarization model, closer

in spirit to combined extractive-abstractive pipelines such as ours, in which an extractive-style selection stage could complement the purely abstractive LoRA fine-tuning studied here.

D. Domain-Specific Summarization Systems

Two studies are especially relevant to the deployment described in Section III. Chen et al. [6] propose a multi-task learning algorithm specifically for dialogue summarization; this is directly relevant to our deployment, since the checkpoint exposed through our web application, *flan-t5-base-samsum*, is fine-tuned on the SAMSum dialogue corpus rather than the CNN/DailyMail news corpus used for the offline LoRA comparison in Section IV, so its behavior is better contextualized against dialogue-summarization work such as [6] than against news-summarization work. Anand and Wagh [4] similarly show that deep learning summarization approaches require domain-specific adaptation, in their case for legal text, reinforcing that no single fine-tuned checkpoint should be expected to generalize uniformly across domains without further adaptation, a point we revisit qualitatively in Section V-E.

III. PROPOSED SYSTEM

A. Overview

Fig. 1 summarizes the end-to-end system architecture, which consists of four stages: data preparation, candidate model training/selection, offline evaluation, and deployment. In the first stage, article-highlight pairs from CNN/DailyMail are used both to LoRA-adapt four pretrained backbones and to evaluate four non-adapted baselines. In the second stage, all eight configurations generate summaries under an identical beam search decoding configuration. In the third stage, generated summaries are scored against the dataset's reference highlights using BLEU and ROUGE, and the best-performing configuration is selected. In the fourth stage, the selected checkpoint is loaded once at process startup by a FastAPI backend, which exposes a POST /summarize endpoint and also serves the static frontend files, so that the model can be queried interactively through a browser rather than only through offline batch scripts.

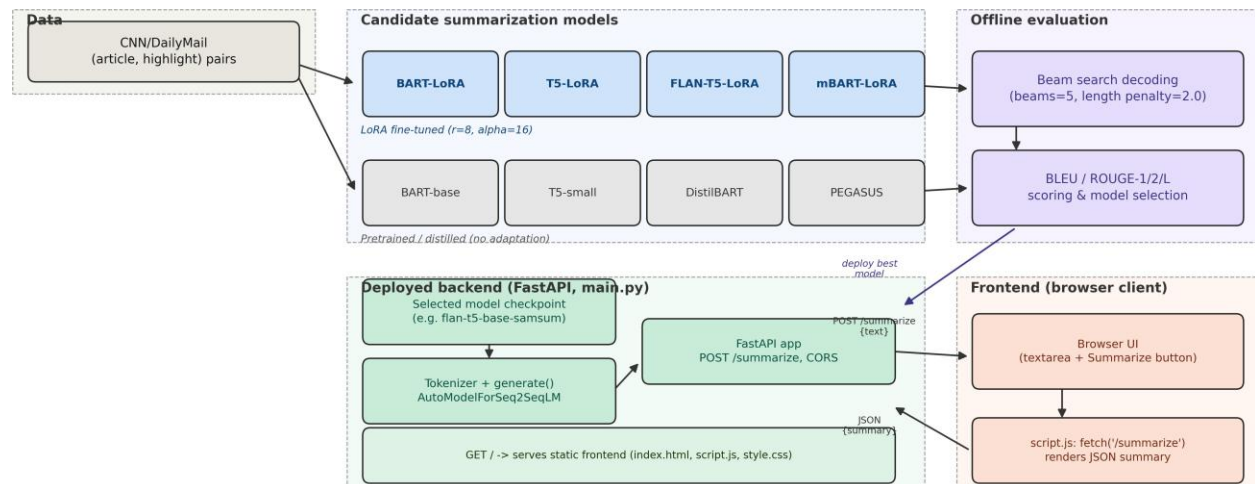


Fig. 1. End-to-end system architecture. CNN/DailyMail article-highlight pairs are used to LoRA-adapt four backbones (BART, T5, FLAN-T5, mBART) and are also used, unmodified, to evaluate four non-adapted baselines (BART-base, T5-small, DistilBART, PEGASUS). All eight configurations are decoded with beam search and scored with BLEU/ROUGE; the selected checkpoint is loaded once at startup by a FastAPI backend (*main.py*), which exposes a POST /summarize endpoint and serves the static frontend (*index.html*, *script.js*, *style.css*). The browser client submits article text and renders the JSON summary returned by the backend.

B. Model Configurations

Table I lists the eight configurations evaluated in this study. Four models are adapted using LoRA: *BART-*

LoRA and *mBART-LoRA* attach low-rank adapters to a BART-family backbone; *T5-LoRA* and *FLAN-T5-LoRA* attach adapters to a T5-family backbone. In each

case, LoRA adapters are inserted into the attention projection layers with rank $r = 8$ and scaling factor $\alpha = 16$, following common practice for encoder-decoder adaptation. The remaining four configurations, *BART-base*, *T5-small*, *DistilBART*, and *PEGASUS*, are used directly from their publicly released, pretrained (and in the case of *DistilBART*, distilled) checkpoints, without any additional dataset-specific training, and serve as reference points for the effect of adaptation.

TABLE I SUMMARIZATION MODEL CONFIGURATIONS

Model	Backbone	Adaptation
BART-LoRA	BART (large)	LoRA ($r=8$, $\alpha=16$)
T5-LoRA	T5 (base)	LoRA ($r=8$, $\alpha=16$)
FLAN-T5-LoRA	FLAN-T5 (base)	LoRA ($r=8$, $\alpha=16$)
mBART-LoRA	mBART (large)	LoRA ($r=8$, $\alpha=16$)
BART-base	BART (base)	None (pretrained)
T5-small	T5 (small)	None (pretrained)
DistilBART	BART (distilled)	None (distilled)
PEGASUS	PEGASUS	None (pretrained, CNN/DM ckpt)

C. Fine-Tuning Procedure

For the four LoRA configurations, only the injected low-rank adapter parameters are updated during training; all pre-trained backbone weights remain frozen. This reduces the number of trainable parameters by roughly two to three orders of magnitude relative to full fine-tuning of the corresponding backbone, which in turn reduces GPU memory consumption and enables fine-tuning on consumer-grade hardware. Training uses article-highlight pairs from the CNN/DailyMail training split, with the article as the encoder input and the reference highlight as the decoder target, optimized with a standard cross-entropy sequence-to-sequence loss.

D. Generation and Decoding

At inference time, all eight configurations generate summaries using beam search with a beam width of 5 and a length penalty of 2.0, applied uniformly across

models to ensure a controlled comparison. No dataset-specific input formatting (e.g., task prefixes for T5-family models) or output post-processing was applied beyond the default tokenizer behavior of each checkpoint, which, as discussed in Section V, has a direct bearing on the PEGASUS results.

E. Web Application

The selected model is deployed behind a lightweight full-stack web application. The backend is implemented with FastAPI and, at process startup, loads the summarization tokenizer and model once via Hugging Face's AutoTokenizer and AutoModelForSeq2SeqLM interfaces, avoiding the latency of reloading model weights on every request. Cross-origin requests are permitted through FastAPI's CORS middleware, and the backend serves two roles concurrently: it returns `index.html` at the root URL (`GET /`) and mounts the remaining static assets (`script.js`, `style.css`) under a `/frontend` prefix, while also exposing a `POST /summarize` endpoint that accepts a JSON payload containing the input text.

On receiving a request, the backend tokenizes the input with truncation to a maximum of 512 tokens and generates a summary using beam search (4 beams, a 3-gram repetition block, a length penalty of 0.9, and early stopping), constrained to between 15 and 60 new tokens, before decoding and returning the result as JSON. On the client side, a static HTML page presents a text area for the source article and a "Summarize" button; on submission, `script.js` issues an asynchronous POST request containing the input text to `/summarize`, displays a loading indicator while awaiting the response, and renders the returned summary text in the result panel, with basic error handling if the backend is unreachable or returns a non-successful response. This component demonstrates that a fine-tuned summarization model can be operationalized as an interactive service rather than remaining confined to offline batch evaluation.

IV. EXPERIMENTAL SETUP

A. Dataset

We use the CNN/DailyMail dataset¹, a standard benchmark for news summarization consisting of online news articles paired with multi-sentence

¹ <https://www.kaggle.com/code/fatimamelsaadny/cnn-dailymail>

highlights written by the articles' editors. The release used in this study is distributed as three CSV files, each with three columns: id (a hexadecimal SHA-1 hash of the source URL), article (the full body text of the news article), and highlights (the reference summary). The training split contains 287,113 unique articles, the validation split contains 13,368 unique articles, and the test split contains 11,490 unique articles. Articles are treated as the summarization input and the corresponding highlights field as the reference summary for evaluation, consistent with the LoRA fine-tuning and offline scoring pipeline described in Section III.

B. Evaluation Metrics

We report BLEU and ROUGE-1, ROUGE-2, and ROUGE-L F-measures for every configuration. ROUGE-1 and ROUGE-2 capture unigram and bigram overlap with the reference, respectively, while ROUGE-L captures the longest common subsequence, which is often interpreted as a proxy for sentence-level fluency and structural alignment with the reference. BLEU is precision-oriented and, as observed in Section V, is highly sensitive to reference and candidate length, which limits its informativeness on short reference summaries.

C. Evaluation Protocol

Each of the eight model configurations generates a summary for a given source article under the beam search settings described in Section III-D. The generated summary is compared against the corresponding reference highlight using the metrics above. We report two evaluation instances: a longer, multi-sentence article-summary pair (Instance A) and a shorter, single-sentence article-summary pair (Instance B). Reporting both instances illustrate how metric behavior, particularly for BLEU, is affected by reference length, and motivates the batch-level evaluation described in Section VI as necessary future work before drawing dataset-level conclusions.

V. RESULTS AND DISCUSSION

A. Instance A: Multi-Sentence Article

Table II reports BLEU and ROUGE scores for all eight configurations on a representative multi-sentence article-summary pair. The LoRA-adapted models occupy five of the top six positions by ROUGE-1, with

T5-LoRA achieving the highest ROUGE-1 (0.3774) and ROUGE-L (0.3396) scores among all configurations, followed closely by DistilBART, BART-LoRA, FLAN-T5-LoRA, and mBART-LoRA, which are closely clustered. The two non-adapted, smaller-capacity baselines, BART-base and T5-small, trail the adapted models on all three ROUGE variants, which is consistent with the expectation that lightweight fine-tuning on in-domain data improves alignment with reference summaries relative to using a general-purpose pretrained checkpoint unmodified.

TABLE II AUTOMATIC EVALUATION SCORES, INSTANCE A

Model	BLEU	R-1	R-2	R-L
T5-LoRA	0.0653	0.3774	0.1569	0.3396
DistilBART	0.0771	0.3667	0.1379	0.3333
FLAN-T5-LoRA	0.0823	0.3448	0.1429	0.3103
BART-LoRA	0.0788	0.3448	0.1429	0.3103
mBART-LoRA	0.0788	0.3448	0.1429	0.3103
BART-base	0.0626	0.2899	0.1194	0.2609
T5-small	0.0484	0.2899	0.1194	0.2609
PEGASUS	0.0000	0.0000	0.0000	0.0000

B. Instance B: Short, Single-Sentence Article

Table III reports results on a shorter article-summary pair. BLEU collapses to 0.0 for every configuration, since BLEU requires overlapping higher-order n-grams (up to 4-grams by default) between candidate and reference, which is difficult to achieve on a single short sentence regardless of summary quality; this instance illustrates why BLEU is poorly suited to short-reference summarization evaluation and is reported here primarily as a cautionary result rather than a meaningful ranking signal. Under ROUGE, the ranking changes noticeably relative to Instance A: BART-base achieves the highest ROUGE-1 (0.3478), followed by mBART-LoRA, DistilBART, T5-LoRA, T5-small, FLAN-T5-LoRA, and BART-LoRA, which are closely spaced.

TABLE III AUTOMATIC EVALUATION SCORES, INSTANCE B

Model	BLEU	R-1	R-2	R-L
BART-base	0.0	0.3478	0.1364	0.3478
mBART-LoRA	0.0	0.3137	0.1224	0.3137
DistilBART	0.0	0.3077	0.1200	0.3077

Model	BLEU	R-1	R-2	R-L
T5-LoRA	0.0	0.2963	0.1154	0.2963
T5-small	0.0	0.2857	0.1111	0.2857
FLAN-T5-LoRA	0.0	0.2759	0.1071	0.2759
BART-LoRA	0.0	0.2623	0.1017	0.2623
PEGASUS	0.0	0.0000	0.0000	0.0000

C. The PEGASUS Failure Mode

Across both evaluated instances, the pretrained PEGASUS baseline scores exactly 0.0 on every metric, which is not a plausible score for a competently trained summarization checkpoint and instead indicates a degenerate output, most likely an empty or malformed generation under the beam search and tokenization settings used uniformly across all eight models. PEGASUS checkpoints are known to be sensitive to input pre-processing and generation configuration (for example, maximum input length truncation and special-token handling), and applying settings tuned for BART- or T5-family models without model-specific adjustment is a plausible cause. We report this result transparently rather than omitting it, because it is instructive: it demonstrates that a pretrained checkpoint cannot be treated as a drop-in replacement across architectures without verifying its generation behavior in isolation, and it motivates a dedicated debugging pass before PEGASUS is included in any dataset-level comparison.

D. Discussion

Two consistent patterns emerge from the two instances. First, the LoRA-adapted models generally match or exceed the ROUGE scores of their non-adapted, smaller counterparts (BART-LoRA and T5-LoRA versus BART-base and T5-small, respectively), which is consistent with the intended effect of lightweight in-domain adaptation. Second, and more cautionary, the relative ranking among models is not stable between the two single-example instances: T5-LoRA leads Instance A but falls to fourth place on ROUGE-1 in Instance B, while BART-base, which trails on Instance A, leads Instance B. This instability is expected from single-example evaluation and should not be interpreted as evidence that any one architecture is definitively superior; it instead illustrates why summarization systems must be evaluated over a sufficiently large and representative

sample before ranking claims are made, a point we return to in Section VI.

Our BLEU results on Instance B further illustrate that overlap-based metrics can be uninformative, or even degenerate to zero, for short references regardless of underlying summary quality, reinforcing the case for reporting multiple metrics rather than relying on a single reference-based score.

E. Qualitative System Demonstration

Beyond the offline comparison in Tables II and III, Fig. 2 shows the deployed web application (Section III-E) summarizing a short encyclopedic paragraph about a university, submitted through the browser interface at inference time. Given a roughly 100-word input describing the institution's founding, accreditation, and key figures, the flan-t5-base-samsum backend returns a concise two-sentence summary retaining the most salient evaluative facts (accreditation status, academic standing) while dropping secondary biographical detail. This confirms the end-to-end pipeline in Fig. 1 functions correctly from browser input to generated output, but it also illustrates the domain-mismatch caveat raised in Section VII and by [4], [6]: the deployed checkpoint was fine-tuned on the dialogue-oriented SAMSum corpus rather than on encyclopedic or news text, so its qualitative behavior here should not be assumed to match the quantitative CNN/DailyMail benchmarks reported for the LoRA-adapted models in Sections V-A and V-B.

AI Text Summarizer

Andhra University, established in 1926, is a premier public state university located on an expansive campus in the coastal city of Visakhapatnam, Andhra Pradesh. As a historic institution that recently marked its centenary, it was originally conceived as the first residential and teaching-cum-affiliating university dedicated to the region. The university's foundational pillars were shaped by visionaries like its founder Vice-Chancellor Sir C.R. Reddy and its second Vice-Chancellor, the former President of India, Dr. Sarvepalli Radhakrishnan. Today, it stands as an elite NAAC A++ accredited institution, highly

Summarize

Andhra University is an elite NAAC A++ accredited institution, highly regarded for its rigorous academic standards and robust research ecosystem

Fig. 2. The deployed web application (Section III-E) summarizing an encyclopedic paragraph. The frontend sends text to POST /summarize, and flan-t5-base-samsum returns a two-sentence abstractive summary.

VI. LIMITATIONS

This study has three main limitations to address before its conclusions are treated as dataset-level findings. First, the results in Section V are drawn from two article-summary instances rather than an aggregated evaluation over the CNN/DailyMail test split; a 200-example subset was prepared for this purpose, but batch-level scoring was not completed at the time of writing, so reported scores should be read as an illustrative pilot rather than a statistically robust comparison. Second, the PEGASUS baseline's degenerate 0.0 score across all metrics indicates a configuration issue requiring isolation before it can be meaningfully compared to the other seven configurations. Third, evaluation is restricted to n-gram overlap metrics (BLEU, ROUGE); no embedding-based or model-based evaluation was performed, and prior summarization studies suggest that such complementary evaluation can materially change system rankings relative to purely lexical metrics [3], [7].

VII. CONCLUSION AND FUTURE WORK

We presented a parameter-efficient fine-tuning pipeline for abstractive summarization that adapts four transformer backbones (BART, T5, FLAN-T5, mBART) using LoRA and compares them against four non-adapted baselines (BART-base, T5-small, DistilBART, PEGASUS) on CNN/DailyMail, alongside a deployable web interface for interactive summarization. Pilot results suggest LoRA adaptation tends to improve ROUGE scores relative to non-adapted counterparts, while also revealing the sensitivity of automatic metrics, and of PEGASUS specifically, to short references and generation configuration.

Future work will (1) complete batch evaluation over the full test subset, reporting mean and variance of BLEU/ROUGE rather than single-instance scores; (2) diagnose and correct the PEGASUS degenerate-output issue; (3) incorporate embedding-based and LLM-as-judge evaluation alongside ROUGE, following the direction of recent LLM-assisted summarization work [3]; and (4) report training efficiency (trainable parameters, GPU memory, wall-clock time) for each LoRA configuration relative to full fine-tuning.

ACKNOWLEDGMENT

The authors thank the maintainers of the open-source libraries used in this work.

REFERENCES

- [1] P. Widyassari, S. Rustad, G. F. Shidik, E. Noersasongko, A. Syukur, A. Affandy, and D. R. I. M. Setiadi, "Review of automatic text summarization techniques & methods," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 4, pp. 1029–1046, 2022.
- [2] N. I. Altmami and M. El Bachir Menai, "Automatic summarization of scientific articles: A survey," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 4, pp. 1011–1028, 2022.
- [3] H. Zhang, X. Liu, and J. Zhang, "Extractive summarization via ChatGPT for faithful summary generation," in *Findings Assoc. Comput. Linguistics: EMNLP 2023*, 2023, pp. 3270–3278.
- [4] D. Anand and R. Wagh, "Effective deep learning approaches for summarization of legal texts," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 5, pp. 2141–2150, 2022.
- [5] D. Yadav, R. Katna, A. K. Yadav, and J. Morato, "Feature based automatic text summarization methods: A comprehensive state-of-the-art survey," *IEEE Access*, vol. 10, pp. 133981–134003, 2022.
- [6] H. Chen, C. Li, J. Liang, and L. Tian, "A dialogues summarization algorithm based on multi-task learning," *Neural Process. Lett.*, vol. 56, no. 3, p. 160, 2024.
- [7] M. Azam, S. Khalid, S. Almutairi, H. A. Khattak, A. Namoun, A. Ali, and H. S. M. Bilal, "Current trends and advances in extractive text summarization: A comprehensive review," *IEEE Access*, vol. 13, pp. 28150–28166, 2025.
- [8] J. Yang, H. Qin, Y. Sun, H. Wang, A. A. Khan, L. Y. Por, and P. Pławiak, "A generative adversarial network-based extractive text summarization using transductive and reinforcement learning," *IEEE Access*, vol. 13, pp. 65490–65509, 2025.
- [9] W. Fatima, S. S. R. Rizvi, T. M. Ghazal, Q. M. Kharma, M. Ahmad, S. Abbas, and K. M. Adnan, "Abstractive text summarization in Arabic-like script using multi-encoder architecture and

semantic extraction techniques,” *IEEE Access*, vol. 13, pp. 104977–104991, 2025.

- [10] E. A. Kadhim, M. R. Feizi-Derakhshi, and H. S. Aghdasi, “Advanced text summarization model incorporating NLP techniques and feature-based scoring,” *IEEE Access*, vol. 13, pp. 19302–19319, 2025.