FREEsum: A Conceptual Framework for Evaluating Text Summarization Approaches

Lucas V. Alves[1], **Cleilton L. Rocha**[1]

[1]Instituto Atlântico – Fortaleza – Ceará – 60811-341 – Brazil

{lucas_alves,cleilton_rocha}@atlantico.com.br

Abstract. *Research Context: The increasing amount of digital information in various areas, combined with the use of large language models (LLMs) in intelligent systems, chatbots, and LLM-based agents, underscores the importance of automatic text summarization for enhancing the efficiency of information processing and dissemination. Scientific and/or Practical Problem: Although advances in summarization techniques have been made, practitioners face difficulties in systematically comparing strategies, as automatic evaluations rely on metrics and tools with limited scope and insufficient integration with reproducible experimental pipelines, which creates challenges for rigorous benchmarking and the transparent selection of the best solutions. Proposed Solution and/or Analysis: This work proposes FREEsum—a conceptual, reproducible framework that enables the creation of end-to-end pipelines for automatic summarization evaluation, covering all stages of the experiment from data ingestion to results analysis, through declarative workflows and traceable artifacts, thus enabling systematic experimentation and direct comparison of summarization strategies. Related IS Theory: The framework applies Design Science Research principles and open-science practices, ensuring transparency, traceability, and reproducibility in Information Systems experiments. Research Method: FREEsum was developed and validated by implementing a configurable, reproducible experimental pipeline, including a proof-of-concept that demonstrates its applicability in realistic evaluation scenarios. Summary of Results: Experiments show that FREEsum standardizes benchmarking, streamlines configuration, supports method-and-metric trade-off analysis, and facilitates auditing across all experimental stages. Contributions and Impact to IS area: This work contributes to the field of Information Systems by connecting AI techniques for summarization (LLM- and NLP-based) with core IS concerns, such as transparency, governance, and technological and economic impacts. A standardized, auditable, and reusable infrastructure for reliable and extensible summarization evaluation across domains, addressing IS demands for empirical rigor and transparent experimentation and supporting evidence-based adoption of AI methods, particularly in large-scale and complex-text settings where summarization mitigates information overload and improves interpretability.*

1. Introduction

Text summarization involves generating a concise version of a text that captures its most important information [Gambhir and Gupta 2017]. An effective summary consists of various key aspects, including coverage, non-redundancy, cohesion, relevance, readability,

and the inclusion of key points [Parveen et al. 2016]. Text summarization methods can be classified as extractive (selecting important sentences from the text), abstractive (creating new sentences covering the main ideas) [Liu and Lapata 2019], or a combination of both as hybrid approaches [Mahajani et al. 2019].

In the current scenario of exponential data growth, which encompasses textual data ranging from news and social media to scientific publications and business communications, the diversity and complexity of information sources are rapidly increasing [Rydning et al. 2018]. In Brazil's healthcare system, for example, the Unified Health System (SUS) conducts roughly 2.8 billion patient encounters per year[1]. Each of these encounters generates clinical notes, reports, and longitudinal records in electronic health records (EHRs). Therefore, the effective implementation of the International Patient Summary (IPS) stands out as a vital tool in health information systems, ensuring that all individuals receive appropriate care that respects their unique needs [de Oliveira et al. 2025]. In this context, automatic summarization can significantly reduce reading load while preserving clinically critical information. Consequently, there is a growing need for effective ATS approaches that enable users and systems to filter, prioritize, and operate on high-value information.

Artificial intelligence (AI), including deep learning and approaches based on Large Language Models (LLMs), has significantly advanced text summarization techniques. However, the reverse is also true, as AI systems, particularly those that deal with text-type data, can benefit from summarization strategies. In conversational AI, for instance, chatbots and virtual assistants, summarization helps to condense prior exchanges, allowing the systems to maintain context during long dialogues. In Retrieval-Augmented Generation (RAG) systems, relevant documents are retrieved from extensive databases and utilized to ground the generation of answers [Arslan et al. 2024]. Concise summaries of retrieved content can enhance grounding and response focus. In agent-based and multi-agent frameworks with large language models (LLMs), summarization plays a role in consolidating and relaying information between different models [Arslan et al. 2024]. In these situations, retrieval of the most relevant information is key, and summarized forms allow AI systems to focus.

Despite the advances, it remains necessary to conduct an appropriate assessment to verify whether the proposed strategy is suitable for the target task and achieves satisfactory results in text generation. Human evaluation is still the gold standard for assessing the quality of synthesis; however, human evaluations can be expensive and time-consuming, especially in domains that require specialised knowledge (e.g., legal decisions, clinical notes, or financial areas) [Vilca and Cabezudo 2017]. Moreover, there is often a lack of consistency in how human evaluations are conducted, which limits the reproducibility and comparability of results across different summarization strategies [El-Kassas et al. 2021].

To evaluate different strategies for summarization, automatic evaluation methods have been developed that align the quality of generated summaries with human evaluations. Traditional lexical methods, including ROUGE [Lin 2004] and BLEU [Papineni et al. 2002], detect n-gram overlaps between system and reference summaries, but struggle to capture paraphrases and more nuanced semantic equivalence. ME-

[1]https://www.gov.br/saude/pt-br/assuntos/noticias/2025/setembro/viva-o-sus-o-maior-sistema-publico-de-saude-do-mundo-e-gratuito-universal-e-do-brasil

TEOR operates between lexical and semantic metrics by aligning unigrams with stemming, synonyms, and paraphrase matches. It blends precision and recall with a fragmentation penalty, enhancing its robustness against lexical variation[Lavie et al. 2004]. In contrast, BERTScore focuses on contextual embeddings and uses sophisticated language models to evaluate semantic similarity and textual coherence [Zhang* et al. 2020]. However, no single method captures overall summary quality; consequently, it is common to analyze multiple metrics to assess systems effectively [Goyal et al. 2023].

Although there exist many libraries and tools for generating and evaluating summaries—such as SacreROUGE [Deutsch and Roth 2020], SummEval [Fabbri et al. 2021], SummerTime [Ni et al. 2021], and FineSurE [Song et al. 2024], which consolidate metrics, models, and evaluation routines and facilitate experimentation—there remains a gap: the lack of a complete and standardized system for assembling end-to-end, reproducible evaluation pipelines that integrate data ingestion, summary generation, and metric computation, and that reliably measure the effectiveness of generated text [Goyal et al. 2023].

To address this challenge, we propose a conceptual framework, FREEsum (**Fr**amework for **E**valuating T**e**xt **Sum**marization), which simplifies the construction of end-to-end pipelines for text summarization and evaluation. FREEsum targets NLP researchers and practitioners who design or evaluate summarization systems, both in academia and in data-driven organizations (e.g., AI teams in healthcare, media, or legal domains). The framework integrates data ingestion, summary generation, and evaluation, supporting both classical methods and LLM-based approaches, and enables systematic comparisons among strategies under consistent experimental conditions. It promotes reproducibility and standardization by expressing the entire experimental configuration in a YAML file and storing results in a unified JSONL format, which also provides an auditable trail from corpus to metrics. In addition, FREEsum accelerates the experimental cycle via built-in corpora and preconfigured routines for preprocessing, summarization, metric computation, and visualization. Although FREEsum can handle texts in multiple languages, it provides ready-to-use settings for Brazilian Portuguese and remains extensible to incorporate new metrics and methods.

The remainder of this paper is organized as follows: Section 2 provides background; Section 3 surveys related work; Section 4 introduces the FREEsum framework; Section 5 presents the evaluation and results; and Section 6 concludes and outlines future directions.

2. Background

2.1. Automatic Text Summarization

Automatic Text Summarization (ATS) is a Natural Language Processing (NLP) task that aims to convert lengthy documents into concise versions while preserving the essential information [Zhang et al. 2024]. The text summarization process can be applied across a wide range of specific domains, including the processing of legal, medical, and scientific documents [Tsirmpas et al. 2024], as well as in more general-purpose domains, such as news monitoring [El-Kassas et al. 2021]. ATS approaches are commonly classified into two categories: extractive and abstractive. Extractive methods create summaries by extracting sentences from the source texts [Liu and Lapata 2019]. In contrast, abstractive

methods generate summaries that may include novel words and sentences, not explicitly found in the source texts [Lewis et al. 2020].

Extractive methods typically employ statistical approaches that are simpler, faster, and more computationally efficient [Zhang et al. 2024]. However, they often produce summaries with limitations regarding readability and compression [Hou et al. 2018]. Lack of semantics and cohesion in the extracted sentences used in the summarization. [Gupta and Lehal 2010]. On the other hand, abstractive strategies generally produce summaries that are more aligned with human-written abstracts [El-Kassas et al. 2021]. Nevertheless, they involve highly complex methods and demand substantial NLP resources, making them computationally intensive [El-Kassas et al. 2021].

In statistical summarization, key information is extracted from preprocessed text using algorithms that select important sentences. Examples of such algorithms include Term Frequency-Inverse Document Frequency (TF-IDF), Latent Dirichlet Allocation (LDA), and TextRank, each employing a distinct strategy. TF-IDF performs keyword-based sentence extraction, LDA focuses on topic-based sentence selection, and TextRank utilizes graph-based ranking.

TF-IDF is a statistical method that assigns weights to terms based on their frequency of appearance in a document, thereby emphasizing terms that are specific to a given text. It helps identify sentences likely to contain relevant or unique content, making it a useful heuristic for extractive summarization [Cranganu-Cretu et al. 2001]. LDA is a topic modelling technique that can be applied to extractive summarization by selecting sentences that best represent the dominant topics identified within the text [Blei et al. 2003]. TextRank is a graph-based method that models the document as a graph, where sentences are represented as nodes and edges encode similarity or lexical relations between them. Thus, extractive summarization is formulated to identify the most central nodes in this graph. The assumption is that sentence centrality indicates its importance in the document, which makes it a valuable criterion for constructing summaries [Mihalcea and Tarau 2004]. Thus, each of these algorithms was chosen to reflect distinct summarization paradigms: frequency-based methods (TF-IDF), optimization-based approaches (LDA), and graph-based methods (TextRank) [Zhang et al. 2024].

Abstractive summarization techniques leverage the advancements in deep learning for natural language generation. A frequently used architecture is the sequence-to-sequence (seq2seq) model, where an encoder processes the source document and a decoder generates the summary, one word at a time. Initial neural summarizers utilized recurrent neural networks (RNNs). They then introduced attention mechanisms, which enabled the model to focus on relevant segments of the original text when generating summaries [Zhang et al. 2025]. Recent advances in deep learning have generated renewed interest in automatic text summarization, especially with the emergence of large pre-trained language models (LLMs) such as ChatGPT. These models have especially advanced the quality of summaries in the abstractive summarization techniques [Bommasani et al. 2021].

In addition, approaches have explored hybrid methods that combine both extractive and abstractive summarization techniques to leverage the strengths of each method [El-Kassas et al. 2021]. Commonly, a hybrid strategy initiates the process with an extrac-

tive method that identifies the most relevant sentences from the input text. This is followed by an abstractive method applied to the filtered content, which generates a refined final summary. As defined by Bhat et al. (2018) [Bhat et al. 2017], this approach leverages the efficiency of extraction in the first stage. Then, it utilizes abstractive generation to enhance the fluency and conciseness of the summary. The hybrid approach has been gaining attention and is considered promising for maximizing performance in different domains by balancing text quality and computational costs [Bhat et al. 2017].

Despite the wide adoption of LLMs, they are prone to hallucinations and factual inconsistencies, which, in the context of summarization, may compromise the purpose of preserving the original meaning of the source texts [Jorge et al. 2024], mainly in abstractive summaries. Thus, using LLMs in text summarization remains a challenging task [Ghinassi et al. 2024]. The choice of summarization method should be guided by the balance between quality, efficiency, and domain specificity, with hybrid approaches increasingly emerging as a promising solution to maximize performance across different tasks [Zhang et al. 2024].

2.2. Automatic Evaluation Metrics

The evaluation of generated summarization can be intrinsic and extrinsic. The former is comparing the generated summarization to reference abstracts (ideally written by human experts). Later, the impact of the summarization on a practical task (e.g., the usefulness of the abstract for decision-making or answering questions) is measured. In this work, the focus was on intrinsic evaluation.

The most established automatic metrics are based on lexical overlap between the generated summarization and the reference summarization. Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is a widely used set of metrics that calculates recall, precision, and F-score in terms of matching n-grams (ROUGE-N) or subsequences and basic content units (ROUGE-L, ROUGE-S, etc.) [Lin 2004]. Similarly, Bilingual Evaluation Understudy (BLEU) computes a similarity index based on the precision of n-grams in the generated text relative to the reference [Papineni et al. 2002]. Metric for Evaluation of Translation with Explicit ORdering (METEOR) expands the lexical comparison by incorporating synonyms and partial matches, weighting differences in word order, which makes it slightly more correlated with human judgments in some situations [Lavie et al. 2004].

These classic metrics have the advantage of being reproducible and objective, but they have known limitations. A central limitation is that different summarizations may use different vocabulary to express the same idea; therefore, metrics based only on exact word matches fail to capture semantic equivalences. For this reason, the literature has started to explore semantics-oriented metrics. For example, BERTScore calculates the similarity between each token (text unit considered) in the generated summarization and tokens in the reference summarization in the BERT embedding space (tokens mapped onto vectors of numbers), recognizing when different words carry similar meaning in context [Zhang* et al. 2020].

Given that each metric focuses on distinct facets (lexical similarity, semantic coherence, factual fidelity, lexical variety, etc.), it is common practice to apply several metrics together when evaluating a summarization system. For example, recent studies have evaluated ROUGE (for content coverage), BERTScore (for semantic similarity), and lex-

ical diversity metrics, among others, simultaneously to obtain a more comprehensive picture of the summary's quality.

3. Related Work

SacreROUGE [Deutsch and Roth 2020] library addresses the fragmentation of evaluation metrics by providing Python wrappers for heterogeneous official implementations (many of which are written in Perl or Java), exposing a unified API through a common interface (Metric) for evaluation by instance or in batches. The supported set includes, among others, ROUGE, BERTScore, METEOR, and AutoSummENG, as well as tools for meta-metrics (i.e., evaluating the correlation and behaviour of the metrics themselves). Both SacreROUGE and FREEsum aim to increase standardisation and reproducibility in summarization evaluation. However, with different scopes: SacreROUGE is a library specialising in metrics and metametrics, while FREEsum is an end-to-end pipeline framework that orchestrates the entire experiment (corpus import/standardisation, pre-processing, summary generation, evaluation, and analysis), treating metric calculation as a module within a broader reproducible infrastructure.

SummEval tool [Fabbri et al. 2021] implements and provides an extensible and unified toolkit in Python, consolidating 14 automatic evaluation metrics into a single package with a high-level API and immediate use (*out-of-the-box*). In addition to the default settings that reflect common practices for using each metric, the toolkit allows for fine-tuning through external configuration files, maintaining a unified interface for batch execution. In contrast, FREEsum positions metric calculation as just one of the modules in a complete and traceable pipeline that covers the entire experiment — from corpus import and standardisation to pre-processing, summary generation, evaluation, and analysis — thus providing an end-to-end execution and reproducibility infrastructure.

SummerTime [Ni et al. 2021] tool brings together models (e.g., TextRank, BART, PEGASUS, Longformer), tasks (e.g., query-based, dialogue, and multi-document summarization), and metrics (ROUGE, BLEU, BERTScore, METEOR), offering simplified APIs, operational explanations, automatic model selection mechanisms (e.g., successive halving), and supporting visualisations. While SummerTime focuses on accessibility for non-specialists, FREEsum emphasizes reproducibility and traceability through declarative artifacts (YAML/JSONL), as well as explicit support for general summarization strategies, pre-processing routines, and evaluations.

FineSurE system represents a significant advance in the automatic evaluation of summaries, especially for LLM outputs, by breaking down the evaluation into dimensions such as factual verification and coverage of key facts, mitigating hallucination effects [Song et al. 2024]. It is, therefore, a state-of-the-art evaluation methodology that FREEsum can incorporate as a metrics module. FREEsum, on the other hand, is an operational framework that orchestrates the complete cycle of experiments — from corpus import and standardisation to pre-processing, summary generation, and analysis — allowing researchers to execute any summarization strategy (extractive, abstractive, or hybrid, configured via YAML) and apply both classic (built-in) metrics and advanced methodologies such as FineSurE.

Unlike SacreROUGE and SummEval, which focus mainly on metrics and benchmarking, FREEsum is an end-to-end operational framework that orchestrates the entire

summarization experiment (corpus ingestion and standardization, summary generation, metric computation, and result analysis), treating metrics as modules within a broader, auditable infrastructure. Moreover, it prioritizes reproducibility and traceability through a single YAML configuration file and append-only JSONL logs that preserve the inputs and outputs of each step, thus going beyond the scope of tools such as SummerTime. FineSurE, in turn, is an automatic evaluation methodology tailored to LLM-generated summaries, whereas FREEsum provides the complete infrastructure into which FineSurE (and other classical or advanced metrics) can be integrated as a module. In summary, them can be viewed as instantiations of components within the Evaluation & Analysis Layer of FREEsum.

4. FREEsum Overview

FREEsum (Framework for Evaluating Text Summarization) was developed to address the identified gaps in the automatic text summarization process, enabling mainly a more systematic and auditable evaluation. It enables comparison of multiple summarization approaches by organizing experiments as reproducible pipelines. Each stage of the framework is explicitly specified and maintains end-to-end traceability of inputs and outputs, from dataset selection through to the final analysis. The framework is organised into three layers: (i) a *Data Layer* comprising corpus import and data aggregation; (ii) a *Processing Layer* covering descriptive statistics, text preprocessing and summarization (extractive, abstractive, or hybrid); and (iii) an *Evaluation & Analysis Layer* encompassing automatic evaluation (metric calculation) and analysis & visualisation, as illustrated in Figure 1. The complete implementation is publicly available in the GitHub repository [2].

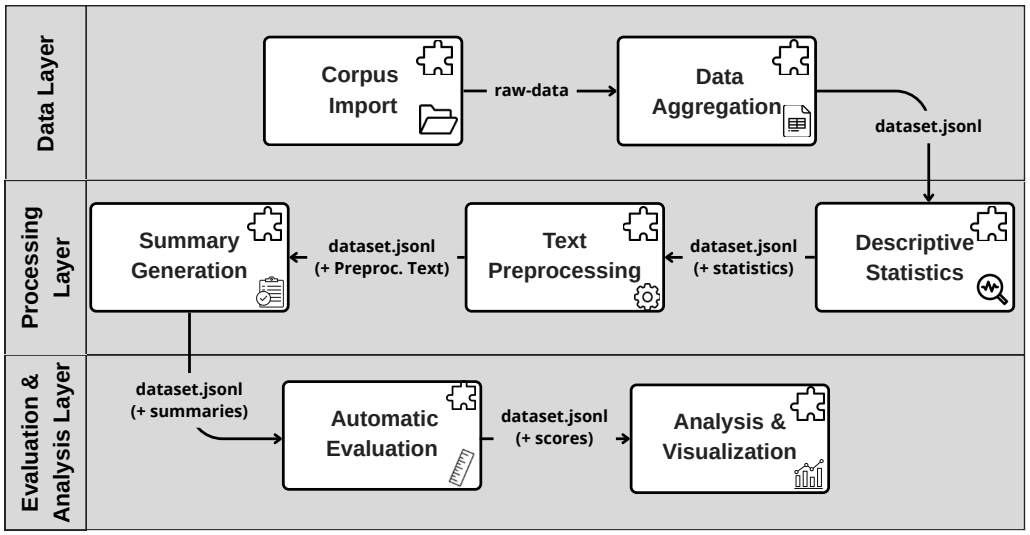

Figure 1. Overview of the FREEsum

4.1. Data Layer

This layer imports, organises, and prepares corpora for subsequent steps. The FREEsum begins with the **Corpus Import** stage, where the datasets for summary and assessment are imported. This import process can be performed by built-in and user-provided datasets. In

[2] https://github.com/lucasvieira123/FREEsum/tree/sbsi

both cases, the imported data must be organized within the framework's defined directory (see Listing 1 in Section 4.4, field $corpus.path$). Typically, each corpus comprises a collection of documents, where each document is an input text accompanied by one or more corresponding reference summaries. These reference summaries are necessary, as they serve as the gold standard for evaluating the quality of a defined summarization approach. An illustration of how a document's data aggregation is represented can be seen in the fields $file\_name$, raw_text, and $raw\_summary$ in Listing 4 (Section 4.4).

The inclusion of built-in datasets accelerates experimentation by providing heterogeneous domains and supporting evaluations in both Brazilian Portuguese and English (see Table 1).

Table 1. Built-in corpora available in FREEsum

Corpus	Description
OpiSums-PT	Human-written abstractive summaries of product reviews from the Buscapé platform [Condori et al. 2015].
TeMário	Professional summaries accompanying Brazilian journalistic articles [Pardo and Rino 2003].
CSTNews 6.0	Multi-document news clusters with human-created summaries in Brazilian Portuguese [Cardoso et al. 2011].
MultiClinSum (PT)	Gold-standard full-text and summary pairs of clinical case reports in Portuguese [Rodríguez-Ortega et al. 2025].
UK-Abs	Abstractive and segment-wise summaries of UK Supreme Court case documents [Shukla et al. 2022].

Each corpus commonly has its own schema; therefore, to facilitate data entry in the next steps, a standardization process is required to map the data into a common schema, representing the **Data Aggregation** stage. The result is a JSON Lines (JSONL) dataset. It is a file containing multiple lines, where each line represents a JSON object. Each record corresponds to a single document with a document identifier ($file\_name$), the source text to be summarized (raw_text), and one or more reference summaries ($raw\_summary$), as represented in Listing 1. The dataset is progressively passed through each stage, and new fields are appended to the same JSON object, preserving earlier information. To facilitate this, the framework already provides aggregation routines for the corpora it supports.

4.2. Processing Layer

The Processing Layer adds details (e.g., statistical information) to each document in the JSONL dataset, preprocesses the source text, and performs summarization (extractive, abstractive, or hybrid). The **Descriptive Statistics** stage provides statistics for each document as well as for the entire corpus, such as token, sentence, and character counts, and simple length indicators (e.g., average sentence length). These indicators help characterise each document and support more controlled evaluation. The script responsible for computing statistics should be specified and configured via the fields $statistics.script$ and $statistics.args$ in Listing 1. An illustration of how a document's descriptive statistics are represented can be seen in the $text\_statistics$ field in Listing 2.

The **Text Preprocessing** stage then prepares the data for the summarization stage. Common tasks include tokenization, normalization, and, when appropriate, stemming or lemmatization, with results recorded in the JSONL (exemplified in the $preprocess$ field in Listing 2). Preprocessing can be configured to meet the needs of various strategies, such as sentence segmentation and lexical statistics for extractive methods, or minimal preparation for neural abstractive models that perform their own tokenization, allowing experiments to be reproduced exactly with the same inputs. In addition, the framework includes preprocessing routines for the built-in corpora, which can be configured via the fields $preprocessing.script$ and $preprocessing.args$ in Listing 1.

The **Summary Generation** stage operates over the preprocessed text using extractive, abstractive, or hybrid approaches. For each method, the generated summary is recorded in the JSONL file, for example, in the fields $extractive\_summaries$ and $abstractive\_summaries$ in Listing 2. If several methods are applied, the record keeps separate entries, which makes comparison in the evaluation stage. FREEsum includes built-in summarization methods, including statistical extractive techniques and modern abstractive approaches (specified in $summararization$ field in Listing 1). The built-in extractive methods include TF–IDF, TextRank, and LDA for identifying salient sentences. For built-in abstractive summarization, the framework supports LLMs via both API-based services (e.g., OpenAI GPT) and locally hosted open-source models (e.g., Mistral).

4.3. Evaluation & Analysis Layer

This layer computes automatic evaluation metrics and aggregates performance per document and per summarization approach, appending the scores to the JSONL artefact for full traceability. Results are then organised into tabular structures and can be explored through visual charts. The **Automatic Evaluation** stage compares system outputs with the reference summaries using the specified metrics. The framework provides built-in implementations of ROUGE, BLEU, METEOR, and BERTScore, reflecting both lexical overlap and semantic similarity. The relevant scripts and parameters are declared via the fields $metrics.script$ and $metrics.args$ (Listing 1), and their outputs are illustrated under the $metrics$ field within each method entry in Listing 2.By storing metric outputs alongside the corresponding document and method metadata, the same JSONL artefact captures the execution flow from input text to metric results, enabling systematic comparisons and the identification of trade-offs among approaches that excel under different indicators.

Finally, the **Analysis & Visualisation** stage aggregates metric outputs across datasets and experimental conditions, producing and saving tables and charts as the concrete results of this phase. These outputs are suitable for reporting and facilitate comparative assessment, especially when multiple summarization approaches are evaluated. In particular, the tool supports side-by-side visualisations of metrics per method, including aggregate statistics such as mean values across the evaluated documents. The relevant scripts and parameters are configured via the fields $analysis.script$, $analysis.args$, $visualization.script$, and $visualization.args$ in Listing 1.

4.4. Configuration & Traceability

FREEsum adopts a configuration-driven design in which the definition of an experiment does not require code changes. All experimental details are specified in a single YAML

file, including the pipeline stages, scripts, parameters, models, and expected outputs (Listing 1). Each stage can be implemented by reusing built-in scripts, extending them, or developing new ones, as long as the corresponding configurations are properly declared in the YAML file. This design simplifies the creation and reuse of summarization and evaluation pipelines while improving reproducibility.

In addition, FREEsum records metadata at each stage of the pipeline, creating audit trails for each document and method. The resulting JSONL file preserves both configurations and results: for each component, it stores the information required for execution and the corresponding outputs, enabling full traceability of experiments.

```yaml
# FREEsum generic pipeline configuration template

corpus:
  path: <corpus_folder_path>   # e.g., ./corpus/dataset_name
  language: <lang_code>        # e.g., "pt" or "en"
statistics:
    script: <statistics_script.py>
    args:
preprocessing:
  script: <preprocess_script.py>
  args:
    # lowercase: true, remove_stopwords: true, any other arguments...
summarization:
  extractive:
    script: <extractive_script.py>
    args:
      # method: textrank  # e.g., tfidf, textrank ...
  abstractive:
    script: <abstractive_script.py>
    args:
      # model: <llm_model> # e.g., gpt4nano, mistral7b, prompt_template:
      ↪ <file.txt> ...
metrics:
  script: <metrics_script.py>
  args:
    # metrics: [rouge_1, bleu, bert_score_f1 ...]
analysis:
  script: <analysis_script.py>
  args:
    # group_by: <column_name>, filter: <condition>...
visualization:
  script: <visualization_script.py>
  args:
    # chart_type: bar # e.g., bar, line, heatmap, save_fig: true
output:
  dir: <output_folder_path>   # e.g., ./outputs
```

Listing 1. FREEsum configuration template (YAML)

Traceability is achieved by storing the inputs and outputs for each stage of the process. At each step, the framework appends metadata to the record—document identifiers, extracted statistical information, preprocessed text, summarization method identifiers, and performance metrics—thereby creating an audit trail at both the document and method levels. The exact configuration used in a run is stored alongside the result dataset, —the JSONL file—evolves in an append-only manner, so earlier fields are preserved and later stages do not overwrite prior outputs (see Listing 2).

In summary, the YAML file serves as the single source for all pipelines, and the JSONL file serves as the single source for all results. Together, these files allow for easy

reruns of experiments, configuration checks, and validation of claimed results.

```
[
  {
    "file_name": "<string>", // Original filename of the document
    "raw_text": "<string>", // Full input text
    "raw_summary": "<string>", // Human or gold reference summary
    "text_statistics": {
      "num_words": "<int>", // Number of words in the document
      "size_bucket": "<string>" // Size class (e.g., 'short', 'medium', 'long')
      // ... more statistics can be added (number of sentences, tokens, etc.)
    },
    "preprocess": [
      {
        "raw_sentence": "<string>", // Original sentence
        "preprocessed_sentence": "<string>" // Preprocessed/cleaned sentence
      }
      // ... more sentences
    ],
    "extractive_summaries": {
      "<method_name>": {
        "summary": "<string>", // Extractive summary produced by the method
        "metrics": {
          "rouge_1": "<float>", // ROUGE-1 score
          "rouge_2": "<float>", // ROUGE-2 score
          "rouge_l": "<float>", // ROUGE-L score
          "bleu": "<float>",    // BLEU score
          "bert_score_f1": "<float>" // BERTScore F1
          // ... add any additional evaluation metrics
        }
      }
      // ... other extractive methods
    },
    "abstractive_summaries": {
      "<method_name>": {
        "summary": "<string>", // Abstractive summary produced by the method
        "metrics": {
          "rouge_1": "<float>", // ROUGE-1 score
          "rouge_2": "<float>", // ROUGE-2 score
          "rouge_l": "<float>", // ROUGE-L score
          "bleu": "<float>",    // BLEU score
          "bert_score_f1": "<float>" // BERTScore F1
          // ... add any additional evaluation metrics
        }
      }
      // ... other abstractive methods
    }
  }
]
```

Listing 2. Schema of the framework's JSONL output

5. Evaluation

5.1. Experimental Setup

The evaluation was conducted as a proof of concept (PoC) aimed at demonstrating the flexibility, modularity, and reproducibility of the FREEsum framework by benchmarking two distinct summarization strategies on the built-in TeMário corpus: (i) a purely abstractive approach based exclusively on large language models (LLMs), and (ii) a hybrid method that combines statistical extraction (TextRank) with LLM-based abstraction. The entire experimental pipeline was configured through a YAML configuration file (provided in Section 4.4), which defines all stages, from corpus ingestion to the output directory. All

configuration files, scripts, and results are available at GitHub Repository [3].

As illustrated in Figure 2, the specified pipelines systematically process the entire corpus, going through each document and executing all steps of the framework. For each text, the pipeline performs data aggregation and statistical analysis, applies the two summarization strategies, and calculates the relevant evaluation metrics for each summary generated. All outputs—including summaries, metrics, and intermediate statistics—are stored in a traceable JSONL format. Finally, the results are aggregated and visualised in charts, allowing for a comprehensive comparative analysis of the evaluated approaches.

It is important to note that in this PoC, we reused almost the entire FREEsum structure. In the **Data Layer**, we relied on the built-in TeMário corpus, the standard import/aggregation mechanisms, and the JSONL schema. In the **Processing Layer**, we reused the scripts for descriptive statistics, the TeMário preprocessing pipeline, and the built-in components for purely LLM-based summarization and TextRank-based summarization, configuring only hyperparameters and prompts. In the **Evaluation & Analysis Layer**, we reused the scripts for metric computation, result aggregation, and the generation of tables and plots. The only component that required extension was the hybrid LLM-based summarization module, implemented by extending the purely LLM-based module to inject, at runtime, the anchor sentences produced by TextRank into the prompt.

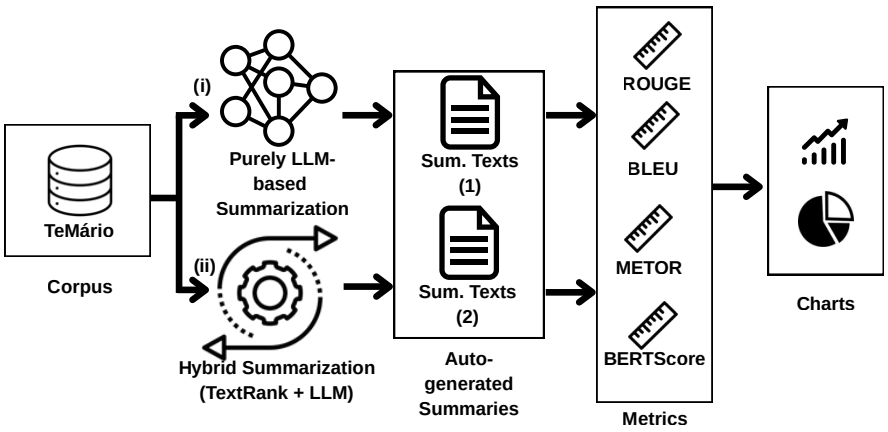

Figure 2. Overview of Evaluation Strategy

5.2. Summarization Strategies

To demonstrate the extensibility of the framework and the possibility of comparing different approaches, two summarization strategies were proposed, as previously described. As both approaches are based on LLMs, they were implemented with prompt templates, as shown in Table 2. In the purely LLM-based method, the model receives the entire document as input. At the same time, the hybrid approach provides anchor phrases extracted by the TextRank algorithm, providing a more focused context for the LLM during summary generation. Other relevant hyperparameters are specified in the YAML configuration file, as illustrated in Section 5.3.

The templates presented in Table 2 contain placeholders, such as $full\_text$ and $anchor\_sentences$, which must be dynamically filled at runtime. The $full\_text$ field is

[3] https://github.com/lucasvieira123/FREEsum/tree/sbsi

Table 2. Prompt templates for LLM-based and hybrid sum. in FREEsum

Purely LLM-based Prompt	Hybrid Prompt
Instruction: Create an **abstractive summary** from the full text provided below. An abstractive summary conveys the main ideas using your own words and sentence structures. Do not copy sentences directly. Instead, rewrite the key information into a concise, coherent, and fluent summary that reflects the essence of the original content. **Input Text:** {full_text} **Abstractive Summary:** final summary text.	**Instruction:** Create an **abstractive summary** based on the sentences provided below. An abstractive summary conveys the main ideas using new wording and sentence structure. Use the anchor sentences as a guide and rewrite the content in a coherent, fluent, and concise form. Do not copy the sentences verbatim; reformulate them while preserving the original meaning. **Input Text:** {full_text} **Anchor Sentences:** {anchor_sentences} **Abstractive Summary:** final summary text.

populated with the raw document text, while the *anchor_sentences* field is filled with the extractive summarization produced by the TextRank algorithm. The process of filling these placeholders, as well as adding the appropriate hyperparameters for each LLM request, is handled by the scripts defined for the summarization stages of the pipeline.

5.3. Framework configurations

To implement the pipeline and execute the two evaluation approaches shown in Figure 2, it is necessary to specify all steps and parameters in the configuration file (Listing 3). This file organises the pipelines into modular sections: the *corpus* field defines the path to the TeMário dataset and the language of the texts; the *preprocessing* field specifies the preprocessing script to be applied to each document.

```yaml
corpus:
  path: ./corpus/temario
  language: pt
preprocessing:
  script: temario_preprocess.py
  args:
summarization:
  extractive:
    script: textrank_extr_sum.py
    args:
  abstractive:
    script: pure_GPT_abst_sum.py
    args:
      model: gpt-4.1-nano-2025-04-14
      temperature: 0.7
      max_tokens: 300
      prompt_template: pure_llm_temp.txt
    script: hybrid_GPT_abst_sum.py
    args:
      model: gpt-4.1-nano-2025-04-14
      temperature: 0.7
      max_tokens: 300
      prompt_template: hybrid_llm_temp.txt
output:
  dir: ./outputs
```

Listing 3. FREEsum YAML configuration file

In the *summarization* field, two approaches are configured: extractive summa-

rization, based on the TextRank algorithm for selecting anchor sentences, and abstractive summarization, which can be performed in two ways — using LLMs exclusively or adopting a hybrid strategy that combines anchor sentences with LLM input. Each summarization technique requires specifying hyperparameters, including model version, temperature, max token length, and prompt template file. The *output* field concludes with the path to the directory containing the results.

Since no custom scripts were specified for statistical analysis, metric calculation, analysis, or visualization in the configuration file, the framework automatically uses its built-in scripts for each of these stages. The default statistical analysis scripts perform basic descriptive statistics on input documents. For evaluation, the built-in scripts calculate key metrics, including ROUGE, BLEU, METEOR, and BERTScore. Similarly, the analysis and visualization steps are handled by default scripts, which aggregate metric results and generate the comparative charts and tables presented in this work.

5.4. Results and Analysis

The FREEsum pipeline stores each summarization output in a JSONL file, containing all information structured and traceable, as illustrated in Listing 4. Each record contains the document identifier in the corpus, the source text, the reference summary, document statistics (such as the number of words), detailed pre-processing records (including original and normalized sentences), as well as summaries generated by each summarization strategy and the respective automatic evaluation metrics.


```
[{
  "file_name": "cf94ab13-a.txt",
  "raw_text": "Uma mulher iraniana de 40 anos apresentou-se ...",
  "raw_summary": "Este caso descreve uma mulher iraniana de 40 anos com uma ...",
  "text_stats": {"num_words": 1620,"size_bucket": "long"},
  "textrank": {
    "preprocess": [
      {
        "raw_sentence": "Uma mulher iraniana de 40 anos apresentou-se num ...",
        "preprocessed_sentence": "mulher iraniano 40 ano apresentar ..."
      }...],
    "summary": [{
          "score": 0.0209,
          "sentence": "A tomografia computadorizada ...",
          "preprocessed": "tomografia computadorizado ..."},]},
  "purellm": {
    "summary": "A paciente iraniana de 40 anos apresentou, inicialmente...",
    "metrics": {"rouge_1": 0.40,"rouge_2": 0.10,"rouge_l": 0.20, "bleu": 0.06,
    ↪  "meteor": 0.27, "bert_score_f1": 0.81}
  },
  "hybrid": {
    "summary": "A paciente apresentou melhora nos sintomas neurológicos...",
    "metrics": {"rouge_1": 0.40, "rouge_2": 0.10,"rouge_l": 0.17, "bleu": 0.029,
    ↪  "meteor": 0.31, "bert_score_f1": 0.82}
  }}...]
```


Listing 4. Example snippet of FREEsum results in JSONL

Using the JSONL dataset, automatic analyses can be produced. Table 3 shows the average of the metrics results of the automatic evaluation, comparing the two summarization strategies. It can be observed that the hybrid strategy achieved slightly higher scores in all metrics considered.

Table 3. Extractive Summarization Metrics (StatLLM vs. LLM-Based)

Metric	Purely LLM-based Sum.	Hybrid Sum. (TextRank + LLM)
ROUGE-1	0.417985	0.430991
ROUGE-2	0.124908	0.132715
ROUGE-L	0.213788	0.221056
BLEU	0.043524	0.047861
METEOR	0.232389	0.247843
BERT-Score	0.71787	0.71891

Figure 3 illustrates the distribution of these scores (using boxplots) for different instances of the validation set. It can be observed that the distributions of metrics are broadly similar between the two approaches, with similar medians and interquartile ranges.

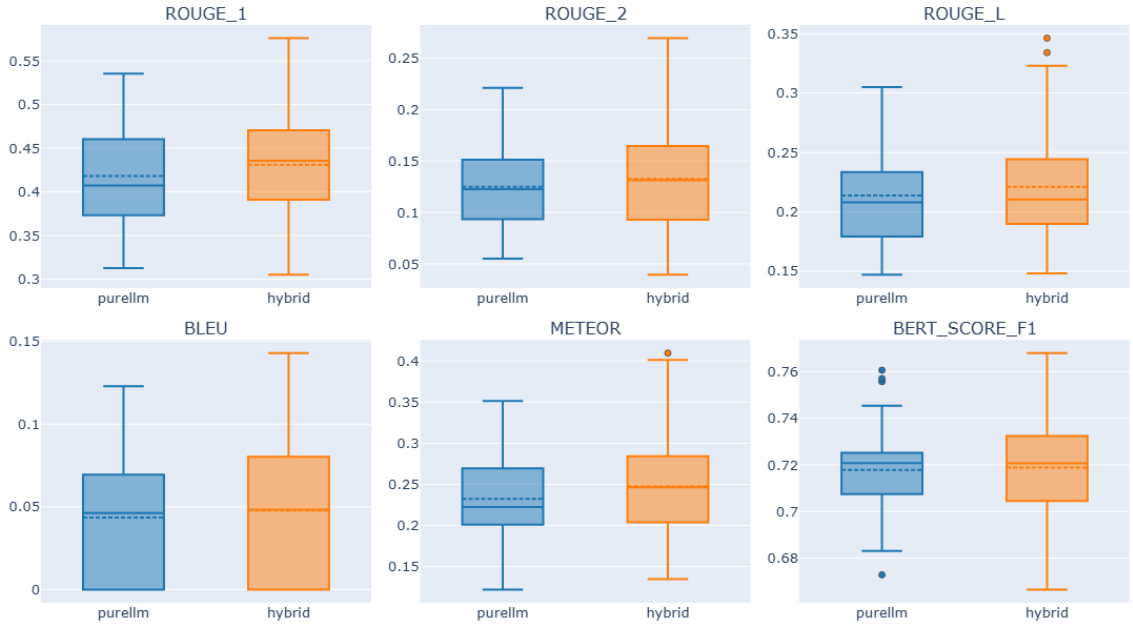

Figure 3. Visualization (box-plots) of metrics by approach

In summary, this investigation does not aim to detail each approach, but rather to demonstrate that FREEsum is capable of executing, recording, and facilitating the analysis of complete summarization experiments—from preprocessing to the generation of both quantitative and visual evaluation metrics.

6. Conclusion

FREEsum is an extensible and reproducible conceptual framework that allows the construction of evaluation pipelines for automatic summarization strategies. It ensures experimental reproducibility and enables standardised comparison of multiple summarization approaches, integrating support for various evaluation metrics (both lexical and semantic). Its modular architecture facilitates extensibility to new summarization methods, new evaluation metrics, and new processing steps. Although designed to be multilingual, FREEsum provides ready-to-use configurations, preprocessing resources, and built-in corpora to simplify experiments in Brazilian Portuguese, facilitating both summarization and evaluation processes. These features make FREEsum particularly relevant for

researchers and developers interested in applying and validating automatic summarization strategies in real-world applications, such as agents based on large language models. As a limitation, it is worth noting that, to date, the framework has only been validated through a proof of concept, and its effectiveness has not yet been confirmed in broader practical scenarios. The main adoption costs arise from adapting data and scripts when built-in resources are insufficient, as well as from the computational load and analytical effort required to run more complex summarization models and conduct careful analyses of their quality metrics. As future work, we intend to apply the Technology Acceptance Model (TAM) to measure the acceptability of the tool among its potential users, thereby complementing the technical evaluation already performed with a usability and acceptance analysis.

References

Arslan, M., Ghanem, H., Munawar, S., and Cruz, C. (2024). A survey on rag with llms. *Procedia computer science*, 246:3781–3790.

Bhat, I. K., Mohd, M., and Hashmy, R. (2017). Sumitup: A hybrid single-document text summarizer. In *Soft Computing: Theories and Applications: Proceedings of SoCTA 2016, Volume 1*, pages 619–634. Springer.

Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2021). On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Cardoso, P. C., Maziero, E. G., Jorge, M. L. C., Seno, E. M., Di Felippo, A., Rino, L. H. M., Nunes, M. d. G. V., and Pardo, T. A. (2011). Cstnews-a discourse-annotated corpus for single and multi-document summarization of news texts in brazilian portuguese. In *Proceedings of the 3rd RST Brazilian Meeting*, pages 88–105. sn.

Condori, R. L., Pardo, T., Avanço, L., Balage Filho, P., Bokan, A., Cardoso, P., Dias, M., Nóbrega, F., Cabezudo, M., Souza, J., et al. (2015). A qualitative analysis of a corpus of opinion summaries based on aspects. In *Proceedings of the 9th Linguistic Annotation Workshop*, pages 62–71.

Cranganu-Cretu, B., Chen, Z., Uchimoto, T., and Miya, K. (2001). Automatic text summarizing based on sentence extraction: A statistical approach. *International Journal of Applied Electromagnetics and Mechanics*, 13(1-4):19–23.

de Oliveira, A. A., Frade, S., Vieira-Marques, P., Jacinto, T. A. Q., Homem-Silva, P., Lemos-Sebasteão, S., de Oliveira, C. M. M., Ferreira, L. F., Rocha, J. C. O., Cruz-Correia, R. J., et al. (2025). Challenges and strategies in the implementation of the international patient summary in accordance with international standards: A systematic review. *Simpósio Brasileiro de Sistemas de Informação (SBSI)*, pages 211–220.

Deutsch, D. and Roth, D. (2020). SacreROUGE: An open-source library for using and developing summarization evaluation metrics. In Park, E. L., Hagiwara, M., Milajevs, D., Liu, N. F., Chauhan, G., and Tan, L., editors, *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 120–125, Online. Association for Computational Linguistics.

El-Kassas, W. S., Salama, C. R., Rafea, A. A., and Mohamed, H. K. (2021). Automatic text summarization: A comprehensive survey. *Expert systems with applications*, 165:113679.

Fabbri, A. R., Kryściński, W., McCann, B., Xiong, C., Socher, R., and Radev, D. (2021). Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Gambhir, M. and Gupta, V. (2017). Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66.

Ghinassi, I., Catalano, L., and Colella, T. (2024). Efficient aspect-based summarization of climate change reports with small language models. In *Proceedings of the Third Workshop on NLP for Positive Impact*, pages 123–139.

Goyal, R., Kumar, P., and Singh, V. (2023). A systematic survey on automated text generation tools and techniques: application, evaluation, and challenges. *Multimedia Tools and Applications*, 82(28):43089–43144.

Gupta, V. and Lehal, G. S. (2010). A survey of text summarization extractive techniques. *Journal of emerging technologies in web intelligence*, 2(3):258–268.

Hou, L., Hu, P., and Bei, C. (2018). Abstractive document summarization via neural model with joint attention. In *Natural Language Processing and Chinese Computing: 6th CCF International Conference, NLPCC 2017, Dalian, China, November 8–12, 2017, Proceedings 6*, pages 329–338. Springer.

Jorge, G. A. Z., Bezerra, D. A., Xavier, C. C., and Pardo, T. A. S. (2024). Multilingual extractive summarization: Investigating state-of-the-art methods for english and brazilian portuguese. In *Brazilian Conference on Intelligent Systems*, pages 212–223. Springer.

Lavie, A., Sagae, K., and Jayaraman, S. (2004). The significance of recall in automatic metrics for mt evaluation. In *Conference of the Association for Machine Translation in the Americas*, pages 134–143. Springer.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Liu, Y. and Lapata, M. (2019). Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Mahajani, A., Pandya, V., Maria, I., and Sharma, D. (2019). A comprehensive survey on extractive and abstractive techniques for text summarization. *Ambient Communications and Computer Systems: RACCCS-2018*, pages 339–351.

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Ni, A., Azerbayev, Z., Mutuma, M., Feng, T., Zhang, Y., Yu, T., Awadallah, A. H., and Radev, D. (2021). SummerTime: Text summarization toolkit for non-experts. In Adel, H. and Shi, S., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 329–338, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Pardo, T. A. S. and Rino, L. H. M. (2003). Temário: Um corpus para sumarização automática de textos. *São Carlos: Universidade de São Carlos, Relatório Técnico*.

Parveen, D., Mesgar, M., and Strube, M. (2016). Generating coherent summaries of scientific articles using coherence patterns. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 772–783.

Rodríguez-Ortega, M., Rodríguez-Lopez, E., Lima-López, S., Escolano, C., Melero, M., Pratesi, L., Vigil-Giménez, L., Fernandez, L., Farré-Maduell, E., and Krallinger, M. (2025). Overview of multiclinsum task at bioasq 2025: evaluation of clinical case summarization strategies for multiple languages: data, evaluation, resources and results. In *CLEF*.

Rydning, D. R.-J. G.-J., Reinsel, J., and Gantz, J. (2018). The digitization of the world from edge to core. *Framingham: International Data Corporation*, 16:1–28.

Shukla, A., Bhattacharya, P., Poddar, S., Mukherjee, R., Ghosh, K., Goyal, P., and Ghosh, S. (2022). Legal case document summarization: Extractive and abstractive methods and their evaluation. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1048–1064.

Song, H., Su, H., Shalyminov, I., Cai, J., and Mansour, S. (2024). FineSurE: Fine-grained summarization evaluation using LLMs. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 906–922, Bangkok, Thailand. Association for Computational Linguistics.

Tsirmpas, D., Gkionis, I., Papadopoulos, G. T., and Mademlis, I. (2024). Neural natural language processing for long texts: A survey on classification and summarization. *Engineering Applications of Artificial Intelligence*, 133:108231.

Vilca, G. C. V. and Cabezudo, M. A. S. (2017). A study of abstractive summarization using semantic representations and discourse level information. In *International Conference on Text, Speech, and Dialogue*, pages 482–490. Springer.

Zhang, H., Yu, P. S., and Zhang, J. (2024). A systematic survey of text summarization: From statistical methods to large language models. *ACM Computing Surveys*.

Zhang, H., Yu, P. S., and Zhang, J. (2025). A systematic survey of text summarization: From statistical methods to large language models. *ACM Computing Surveys*, 57(11):1–41.

Zhang*, T., Kishore*, V., Wu*, F., Weinberger, K. Q., and Artzi, Y. (2020). Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.