Journal of Applied Informatics and Computing (JAIC)
Vol.10, No.3, June 2026, pp. 2254~2268
e-ISSN: 2548-6861

Transformer-Based Abstractive Text Summarisation for Real-Time Web Applications: A Browser-Integrated System with REST API Architecture

Zvinodashe Revesai [1]*, Ndlovu Belinda [2] , Maguraushe Kudakwashe [3]*** ,**
[1]Reformed Church University, Masvingo, Zimbabwe
[2]Informatics Department, National University of Science and Technology, Bulawayo, Zimbabwe
[3]School of Computing, University of South Africa, Pretoria, South Africa
[1]revesaiz@rcu.ac.zw , [2]belinda.ndlovu@nust.ac.zw , [3]magark@unisa.ac.za

Article Info

Article history:

Received 2026-02-24
Revised 2026-03-17
Accepted 2026-05-18

Keyword:

Abstractive Text Summarisation,
Transformer Architecture, Natural Language Processing,
Deep Learning,
Sequence-to-Sequence Models,
Human Evaluation,
ROUGE Metrics,
Explainable AI,
Browser-Integrated Systems,
REST API,
Real-Time Inference,
AI Deployment.

ABSTRACT

The exponential growth of digital textual content has intensified the need for efficient, accessible summarisation tools that support information processing across academic, professional, and research domains. While Transformer-based abstractive summarisation models have demonstrated strong performance in benchmark settings, their real-world deployment remains limited due to computational complexity and lack of user accessibility. This study presents a lightweight Transformer-based abstractive text summarisation system, operationalised as a Google Chrome extension and supported by a REST API, enabling seamless integration into everyday user workflows. The proposed system employs an encoder–decoder framework leveraging a pre-trained Transformer-based encoder and a sequence-to-sequence decoder with attention, fine-tuned on the CNN/Daily Mail dataset. Quantitative evaluation on the benchmark dataset achieved ROUGE-1, ROUGE-2, and ROUGE-L scores of 38.21, 16.54, and 35.12, respectively, demonstrating competitive performance relative to established neural baselines. To address the limitations of lexical evaluation metrics, a complementary human evaluation was conducted using a Likert-scale assessment across coherence, informativeness, and fluency, yielding mean scores above 4.0, thereby confirming the qualitative effectiveness of the generated summaries. In addition to model performance, system-level evaluation assessed functional correctness, latency, scalability, and usability within a real-world deployment context. The system demonstrated stable performance under concurrent usage, with an average response time of 4.2 seconds per request and positive user feedback, validating its practical applicability. The findings demonstrate that high-quality abstractive summarisation can be effectively operationalised within a lightweight, browser-integrated architecture, thereby bridging the gap between research-stage neural models and accessible end-user applications. This work contributes to deployment-oriented natural language processing by emphasising usability, modularity, and real-world integration as critical dimensions of system design.

I. INTRODUCTION

The exponential growth of digital textual content across academic, professional, and online platforms has fundamentally intensified the challenge of efficient information processing. The volume, velocity, and diversity of text generated daily, ranging from news articles and scientific publications to blogs and multimedia transcripts, have surpassed the practical limits of human cognition, creating a critical need for intelligent systems capable of distilling essential information into concise, meaningful representations. While automatic text summarisation has long been recognised as a viable solution to this problem, the scale and complexity of modern information ecosystems have elevated its importance from a supportive tool to a

foundational component of digital knowledge management. Recent advances in natural language processing, particularly the emergence of Transformer-based architectures, have significantly improved the quality of abstractive text summarisation. Models leveraging self-attention mechanisms and large-scale pre-training have demonstrated the ability to generate coherent, fluent, and contextually relevant summaries that closely approximate human-written abstracts. The challenge of processing large volumes of text is not new. The idea of automating the creation of document abstracts was first formally proposed by Luhn [1] in 1958, who demonstrated that statistical word frequency and sentence position could be used to identify the most important sentences in a document, laying the conceptual foundation for all subsequent work in the field. Since then, Automatic Text Summarisation (ATS) has grown into a significant subfield of natural language processing, evolving from early rule-based and statistical approaches into sophisticated deep learning systems capable of generating fluent, human-like summaries, as surveyed comprehensively by Sparck Jones [2] and more recently by Gambhir and Gupta [3]. Despite decades of progress, the problem of information overload remains as pressing as ever. Studies on human attention suggest that individuals lose focus multiple times per second, making sustained engagement with lengthy documents particularly difficult [4]. Compounding this, an estimated 70 million new posts are published on blogging platforms alone each month [5], a figure that does not account for the broader ecosystem of news articles, scientific papers, and institutional reports. The volume of text that professionals, researchers, and students are expected to engage with daily has reached a scale that far exceeds human processing capacity [3], making automated summarisation an increasingly critical tool for knowledge management across virtually every professional domain.

ATS addresses this problem by generating natural-language summaries of input documents while preserving their core meaning [6]. ATS methods are broadly categorised into two paradigms: extractive and abstractive. Extractive approaches construct summaries by selecting and concatenating relevant segments directly from the source text [7]. While computationally simpler, these methods are constrained by the original phrasing and cannot produce fluent, human-like summaries. Abstractive methods, by contrast, build an internal semantic representation of the source document and generate new text using natural language generation techniques [8], more closely replicating how humans compose summaries and offering substantially greater flexibility in output style and length.

Despite considerable advances in neural abstractive summarisation, particularly through sequence-to-sequence architectures [9], attention mechanisms [10][11], and Transformer-based models [12][13], practical deployment of these systems remains limited. The introduction of the Transformer architecture by Vaswani et al. [12] marked a paradigm shift in natural language processing, enabling significantly more effective modelling of long-range dependencies in text than prior recurrent approaches. Building on this foundation, large-scale autoregressive language models such as GPT-3 [14] have demonstrated remarkable text-generation capabilities across a wide range of NLP tasks, including abstractive summarisation. However, despite these technical advances, most state-of-the-art approaches are evaluated in isolated experimental settings, with little attention paid to end-user accessibility, real-world integration, or deployment constraints. Existing tools are rarely lightweight, often requiring substantial computational resources, and are not readily accessible to non-specialist users through everyday interfaces. This disconnect between research-stage performance and practical usability constitutes a significant, largely unaddressed gap in the existing literature [15]. Despite these advances, the majority of state-of-the-art summarisation models remain confined to controlled experimental environments, with limited consideration for real-world deployment, user accessibility, and system integration. As a result, a persistent gap exists between algorithmic performance and practical usability, where highly capable models fail to translate into accessible tools for everyday users. This shift from model-centric evaluation to deployment-aware system design represents a critical yet underexplored dimension in contemporary summarisation research.

This study addresses the identified gap by designing and operationalising a lightweight Transformer-based abstractive text summarisation system deployed as a browser-integrated application supported by a REST API architecture. In contrast to prior work that predominantly emphasises incremental model improvements and benchmark performance, this research adopts a deployment-oriented perspective, focusing on how established Transformer-based approaches can be effectively translated into real-world, user-facing systems. By embedding summarisation capabilities within a widely used browser interface, the proposed system enables seamless interaction with diverse content sources, including web pages, documents, and multimedia transcripts, without requiring specialised technical expertise. This approach shifts the emphasis from model-centric evaluation to practical usability, accessibility, and system integration, thereby contributing toward bridging the gap between research-stage neural models and their application in everyday contexts.

The summarisation engine is based on a Transformer encoder–decoder architecture [12], where a pre-trained GPT-3 model [14] serves as the encoder and a sequence-to-sequence model with an attention mechanism [10] functions as the decoder. The model is fine-tuned on the CNN/Daily Mail dataset [16], a widely adopted benchmark in summarisation research, and evaluated using ROUGE metrics [17] against established neural baselines.

To position the scope and contribution of this work within the existing literature, the key contributions are summarised as follows:

1. Deployment-Aware Abstractive Summarisation Framework:
The study demonstrates how established Transformer-based models can be operationalised within a lightweight, real-time system, advancing the discourse from purely model-centric optimisation toward deployment-aware design that prioritises usability, accessibility, and integration.

2. Browser-Integrated Summarisation Architecture: A Chrome extension supported by a REST API is developed to enable real-time summarisation across heterogeneous content sources, including web pages, documents, and multimedia transcripts, within a familiar user environment.

3. End-to-End System Design and Implementation: A complete system pipeline is presented, encompassing data acquisition, preprocessing, model fine-tuning, API-driven inference, and user interface integration, providing a reproducible and extensible framework for applied natural language processing systems.

4. Multi-Dimensional Evaluation Beyond Benchmark Metrics:
The study extends conventional ROUGE-based evaluation by incorporating human-centred qualitative assessment, functional validation across multiple input modalities, and system-level performance evaluation (latency, scalability, and usability), enabling a more holistic assessment of real-world effectiveness.

5. Extension to Multi-Source and Multimedia Inputs: The system expands the applicability of abstractive summarisation beyond static text by integrating YouTube transcript extraction and multi-format document processing, addressing a key limitation in prior systems.

6. Advancing Deployment-Oriented NLP Research: By operationalising high-performing Transformer-based approaches within an accessible, user-facing system, the study contributes to emerging efforts aimed at translating artificial intelligence capabilities into practical, real-world tools.

Accordingly, the primary contribution of this work lies not in proposing a novel model architecture, but in demonstrating how established Transformer-based methods can be effectively deployed within a lightweight, scalable, and user-centred environment.

The remainder of this paper is organised as follows: Section 2 reviews related work in abstractive text summarisation. Section 3 presents the system design, methodology, and implementation. Section 4 reports the experimental results and evaluation. Section 5 presents the discussion. Section 6 concludes the paper and outlines directions for future work.

II. LITERATURE REVIEW

A. Overview of Automatic Text Summarisation

Text summarisation is the process of automatically generating concise natural language summaries from input documents while retaining the most important information [6]. The origins of the field can be traced to the seminal work of Luhn [1], who proposed using term frequency and sentence position as indicators of sentence importance, establishing the conceptual foundation upon which all subsequent summarisation methods were built. Since then, the field has expanded considerably, as documented in the foundational text by Mani and Maybury [18], who provided one of the earliest comprehensive treatments of the field, and subsequently through a series of surveys that have tracked its evolution through rule-based systems, statistical models, and ultimately deep learning approaches [2][3][19]. By condensing large quantities of information into short, informative outputs, summarisation systems can support a wide range of downstream applications including news digest generation, question answering, information retrieval, and report generation [8][15].

Summarisation systems are broadly divided into two categories: extractive and abstractive [19]. Extractive methods construct a summary by selecting and concatenating relevant sentences or phrases directly from the source document [7][20]. While these approaches are computationally straightforward and tend to produce factually grounded output, they are limited by their dependence on the original phrasing and often produce summaries that lack fluency, coherence, or logical flow between selected sentences. Abstractive methods, by contrast, generate an internal semantic representation of the source text and produce entirely new sentences using natural language generation techniques [6][8], more closely replicating the process by which humans compose summaries. Abstractive summarisation is generally considered computationally more demanding, as it requires deep understanding of the original content as well as the capacity to generate grammatically correct and semantically meaningful new text [24].

B. Neural Network Approaches

The introduction of deep neural networks fundamentally transformed the field of abstractive summarisation. Neural methods are rule-independent and do not rely on manually engineered linguistic features; instead, they learn rich representations directly from large training corpora [23]. Prior to the neural era, summarisation systems relied heavily on hand-crafted features such as term frequency-inverse document frequency, sentence position, and cue phrases, which limited their generalisability across domains and document types [2][3].

A foundational development in neural sequence modelling was the introduction of the Long Short-Term Memory (LSTM) network by Hochreiter and Schmidhuber [21], which addressed the vanishing gradient problem inherent in standard recurrent neural networks and enabled effective modelling of long-range dependencies in sequential data. LSTMs subsequently became the backbone of neural abstractive summarisation systems, serving as both encoder and decoder components in sequence-to-sequence architectures [9][22]. The sequence-to-sequence framework, formalised by Sutskever et al. [22], introduced the general encoder-decoder

paradigm for sequence transduction tasks. In this framework, an encoder reads the source sequence and compresses it into a fixed-length context vector, from which a decoder then generates the target sequence token by token. While initially proposed for machine translation, this architecture was quickly adopted for abstractive summarisation due to the natural alignment between the two tasks [9]. However, the use of a fixed-length context vector introduced a fundamental information bottleneck, particularly for longer input documents where critical information could be lost during encoding.

This bottleneck was addressed by Bahdanau et al. [10], who introduced the attention mechanism, allowing the decoder to dynamically attend to different parts of the encoder hidden states at each generation step rather than relying on a single compressed context vector. This innovation substantially improved the quality of generated sequences for longer inputs and became a near-universal component of subsequent neural summarisation models [11][23].

Four landmark models are particularly relevant to this work. Rush et al. [11] proposed one of the earliest neural abstractive summarisation systems, employing three encoder variants- Bag-of-Words, Convolutional, and Attention-Based-to capture the semantic representation of input sentences. The decoder uses a feed-forward neural network-based language model to generate summaries via beam search. The model was evaluated on the DUC-2004 and Gigaword datasets using ROUGE-1, ROUGE-2, and ROUGE-L metrics [17], and established the foundation for subsequent neural approaches to abstractive summarisation. Chopra et al. [23] extended the work of Rush et al. [11] by introducing a conditional recurrent neural network model with convolutional attention for abstractive sentence summarisation. The model employed both an Elman RNN and an LSTM [21] as decoder variants, with attention weights computed from aggregated word vectors obtained from a CNN. Evaluated on the Gigaword dataset, this model achieved better performance than its predecessor, particularly on the ROUGE-2 and METEOR metrics. See et al. [24] subsequently developed a baseline sequence-to-sequence attentional model comprising a single-layer bidirectional LSTM encoder and a single-layer unidirectional LSTM decoder, building upon the architectural foundations established by Sutskever et al. [22] and the attention mechanism of Bahdanau et al. [10]. At each decoding time step, an attention distribution is computed over the encoder hidden states to produce a context vector, which is concatenated with the decoder state to form the vocabulary distribution from which the output word is sampled. A beam search is used to generate the final summary. A known limitation of this model is the generation of out-of-vocabulary tokens, which was subsequently addressed by See et al. [24] with a pointer-generator network incorporating a coverage mechanism. This model combines a sequence-to-sequence attention architecture with a pointer network, allowing it to either generate words from a fixed vocabulary or copy words directly from the source text. A generation probability is computed at each time step to determine the appropriate mechanism, and a coverage mechanism is introduced to reduce repetition in the generated summaries. This model represented a significant step toward more faithful and readable abstractive summaries and remains a standard benchmark in the field [15].

C. Transformer-Based Models

The Transformer architecture, introduced by Vaswani et al. [12], fundamentally changed the landscape of natural language processing by replacing recurrence entirely with a self-attention mechanism, enabling substantially faster training through full parallelisation and more effective modelling of long-range dependencies across the entire input sequence simultaneously. Since its introduction, the Transformer has become the dominant paradigm in NLP, forming the basis for virtually all state-of-the-art language models and summarisation systems [13][25].

Pre-trained Transformer-based models such as BERT [13] and GPT-3 [14] have achieved state-of-the-art performance across a wide range of NLP tasks by leveraging large-scale unsupervised pre-training on massive text corpora, followed by task-specific fine-tuning. In the context of summarisation, Liu and Lapata [25] demonstrated that pre-trained BERT encoders fine-tuned for extractive and abstractive summarisation achieved significant improvements over prior state-of-the-art models on the CNN/Daily Mail benchmark, highlighting the transferability of contextual representations learned during pre-training. BART, proposed by Lewis et al. [26], introduced a denoising autoencoder framework for pre-training sequence-to-sequence models, achieving state-of-the-art results on multiple summarisation benchmarks by pre-training with a combination of token masking, deletion, and text infilling objectives. Similarly, PEGASUS, proposed by Zhang et al. [27], introduced a novel pre-training objective specifically designed for abstractive summarization-gap sentence generation-in which whole sentences are masked from the input and the model is trained to generate them, achieving state-of-the-art performance on multiple summarisation datasets with limited fine-tuning data. The encoder-decoder architecture with pre-trained Transformer encoders and attention-based sequence-to-sequence decoders has demonstrated superior performance in generating accurate, concise, and readable summaries compared to all prior approaches [25][15].

D. Evaluation Metrics and Datasets

The most widely adopted evaluation metric in automatic summarisation is ROUGE (Recall-Oriented Understudy for Gisting Evaluation), introduced by Lin [17]. ROUGE measures the overlap between a system-generated summary and one or more reference human-produced summaries. The three principal variants are ROUGE-1 for unigram overlap, ROUGE-2 for bigram overlap, and ROUGE-L based on the Longest Common Subsequence between the candidate and

reference summaries. While ROUGE has been widely adopted due to its simplicity and reproducibility, its limitations are acknowledged in literature, particularly its insensitivity to semantic equivalence and its reliance on lexical overlap rather than meaning [15]. A secondary metric, METEOR, was also used in several reviewed studies, particularly to evaluate fluency and to incorporate stemming and synonym matching beyond exact lexical overlap [23].

The reviewed literature converged on a small set of benchmark datasets. The CNN/Daily Mail dataset, introduced by Hermann et al. [16], containing over 300,000 English-language news articles produced by CNN and Daily Mail journalists, was the most widely used dataset in abstractive summarisation research, supporting both extractive and abstractive summarisation tasks [24][25]. The Gigaword dataset, containing approximately 4 million article-headline pairs, was used primarily for headline generation tasks [11]. The DUC-2004 dataset, comprising 500 news stories each accompanied by four human-written reference summaries, was used exclusively for testing purposes [9]. More recently, domain-specific datasets including arXiv and PubMed have been adopted to evaluate summarisation performance on long scientific documents, exposing significant generalisation challenges for models trained exclusively on news corpora [28][29].

A systematic analysis of the existing abstractive summarisation literature reveals several notable gaps that motivate the present work. The predominant limitation across reviewed systems is their inability to account for the subjective preferences of individual users-most systems produce a single fixed-length summary regardless of user context, document type, or intended application, whereas a system capable of generating configurable, user-tailored summaries would offer substantially greater practical utility [15]. Closely related to this is the reliance of both keyword-driven and factual-triple-driven models on information contained exclusively within the source document itself, leaving largely unexplored the potential of external knowledge graphs and background knowledge sources to improve the factual validity of generated summaries and reduce the hallucination errors that remain a persistent challenge in fully abstractive generation [15]. The absence of adaptive summary-length strategies represents a further structural limitation of current systems: optimal summary length varies considerably across domains, document complexity, and user needs, yet existing systems impose fixed maximum-length constraints as termination criteria without dynamic adaptation to these contextual factors [8][15]. The reviewed literature also reveals that most existing summarisation systems have been developed and evaluated exclusively on English-language corpora, with multilingual summarisation remaining a comparatively underexplored area of research despite the global diversity of text that practitioners must process [29]. Taken together, these gaps point to a broader and largely unaddressed challenge: the disconnect between the technical performance of state-of-the-art summarisation models and their practical accessibility to non-specialist users in real-world deployment contexts [19]. The present work partially addresses the user configurability gap by providing adjustable summary parameters within a deployed browser-integrated system and contributes toward bridging the accessibility gap through a lightweight deployment architecture designed for everyday use without requiring technical expertise.

III. System Design, Methodology And Implementation

A. Research Methodology

This project adopted the Foundational Methodology for Data Science (FMDS) as its guiding framework. FMDS is an iterative methodology that accommodates modern data science practices, including artificial intelligence, deep learning, and natural language processing [30]. As illustrated in Fig. 1, the FMDS process comprises eight sequential but iterative stages: business understanding, analytic approach, data requirements, data collection, data understanding, data preparation, modelling, evaluation, deployment, and feedback. The iterative nature of the methodology is particularly well-suited to this project, as summarisation model performance must be continuously evaluated and refined across multiple training cycles before an acceptable level of performance is achieved [30].

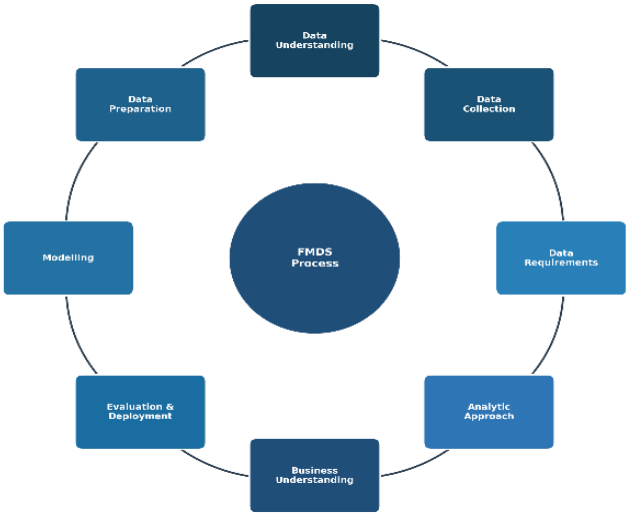

Fig. 1. Foundational Methodology for Data Science (FMDS) Process

The business understanding stage involved establishing the practical value of the proposed system. The core problem identified was information overload, in which professionals, students, and researchers are routinely required to process large volumes of text, much of which is verbose and tangential, contributing little to the core meaning. An automatic summarisation tool capable of distilling essential points from online content addresses a demonstrable and widespread need, given that an estimated 70 million new posts are published on blogging platforms alone each month [5].

The analytic approach was subsequently framed in terms of machine learning and statistical techniques, with a deep learning approach based on the encoder-decoder architecture selected as the most appropriate technique given the complexity and diversity of text the system must handle, and the Transformer model identified as the basis for the summarisation engine [12]. Natural language processing requires data in natural language form [31], and accordingly the system requires a large corpus of document-summary pairs for training and validation of the Transformer-based model. Text data was sourced from publicly available news articles, reports, and established NLP datasets hosted on Kaggle, with web scraping employed as a supplementary collection method [32]. Following initial data collection, descriptive statistics and visualisation techniques were applied to assess data quality, understand content distribution, and identify gaps requiring additional collection [30]. The data preparation stage involved eliminating duplicate entries, identifying outliers, correcting spelling errors, and verifying grammatical correctness prior to model training, as neural summarisation models' performance is highly sensitive to the quality and representativeness of the training corpus [9]. The model training stage involved building the summarisation engine using the cleaned dataset, iteratively training against human-produced gold-standard summaries from the CNN/Daily Mail dataset [16], and refining the model after each evaluation cycle until optimal performance was achieved. The CNN/Daily Mail dataset was selected due to its widespread adoption as a benchmark in abstractive summarisation research, enabling reproducibility and direct comparison with established models. Training was conducted using PyTorch on a CPU-based environment consistent with lightweight deployment constraints. Model performance was assessed after each training iteration using the ROUGE metric [17], and the trained model was subsequently deployed as a REST API hosted on Heroku and a Chrome extension consuming the API, with user feedback collected via star ratings and reviews providing a mechanism for continuous post-deployment improvement [30]. The model was fine-tuned for 3 epochs using a learning rate of 5×10^{-5}, a batch size of 8, and the Adam optimiser. Early stopping was applied based on validation loss to prevent overfitting, with training halted if no improvement was observed over successive validation iterations. Gradient clipping was applied to stabilise training, and model selection was based on the best validation performance.

B. System Requirements Analysis

The system targets three primary user groups: students summarising academic articles for research purposes, professionals condensing lengthy reports for business meetings, and researchers processing large corpora of news articles [33]. User requirements analysis indicated a need for accurate, concise summarisation; customisable output parameters; and a user-friendly interface accessible without technical expertise, consistent with the practical accessibility objective motivating this work [15]. The following functional requirements were established from use case analysis: the system shall extract key information from a given document and generate a concise summary capturing the most important points; the system shall handle a range of document types and formats including PDFs, Word documents, and web pages; users shall be able to customise summary length and level of detail to suit specific needs; the system shall handle long documents and generate summaries of variable length; and the system shall provide the ability to save and export summaries in multiple formats. The non-functional requirements governing system behaviour are summarised in Table 1.

TABLE 1
NON-FUNCTIONAL SYSTEM REQUIREMENTS

ID	Requirement	Description
NFR-1	Performance	System shall generate summaries quickly and efficiently, even for large documents
NFR-2	Scalability	System shall handle large volumes of documents and concurrent users without performance degradation
NFR-3	Security	System shall ensure confidentiality and integrity of user data and documents
NFR-4	Usability	System shall be accessible to both technical and non-technical users
NFR-5	Reliability	System shall operate continuously without downtime or data loss
NFR-6	Maintainability	System shall be easy to maintain and update with minimal user disruption
NFR-7	Compatibility	System shall be compatible with various operating systems, browsers, and devices
NFR-8	Availability	System shall be available to users at all times with minimal maintenance downtime
NFR-9	Compliance	System shall comply with relevant industry standards and regulations including GDPR

The system incorporates authentication mechanisms and secure API communication to protect user data; however, future work will include encryption protocols and compliance validation with data protection standards such as GDPR.

C. Use Case Scenarios

The case scenarios provide a structured analysis of how the system will be used across real-world contexts. As illustrated in Fig. 2, the system supports a range of interactions between the user actor and the Chrome browser, including loading the extension, user login, receiving input, entering summary parameters, generating, viewing, saving, and sharing summaries. To provide a clear representation of the interaction workflow underpinning the proposed system, Fig. 2 presents the use case diagram capturing the key user actions and system responses.

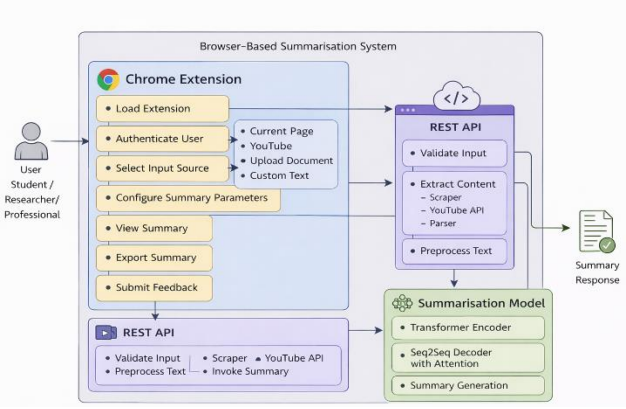

Fig. 2. Use case diagram illustrating user interactions within the browser-integrated summarisation system, highlighting the coordination between the Chrome extension interface, REST API, and Transformer-based summarisation engine

The use case diagram illustrates the end-to-end interaction workflow within the proposed browser-integrated summarisation system, capturing how user actions in the Chrome extension interface trigger coordinated backend processing through the REST API and the Transformer-based summarisation engine. The user initiates the process by selecting an input source, configuring summarisation parameters, and submitting a request; the system then performs content extraction, preprocessing, and model inference to generate a concise summary in near real time. The diagram further highlights the modular separation between user interaction, content processing, and model execution, reflecting the system's deployment-oriented design. This interaction model demonstrates how advanced abstractive summarisation capabilities are operationalised within an intuitive interface, enabling non-technical users to seamlessly access and utilise Transformer-based summarisation in real-world contexts. Three primary use cases were identified and are summarised in Table 2

TABLE 2
SUMMARY OF PRIMARY USE CASE SCENARIOS

Use Case	Primary Actor	Goal	Main Success Scenario
UC-1: Summarise Academic Article	Student	Generate concise summary of a lengthy academic article	Student uploads article, customises parameters, system generates and returns summaries for download
UC-2: Summarise Business Report	Professional	Generate concise summary of a lengthy business report	Professional uploads report, customises parameters, system highlights key findings and recommendations
UC-3: Summarise News Corpus	Researcher	Generate summaries of a large corpus of news articles	Researcher uploads corpus, system generates thematic summaries capturing the most important points

D. System Architecture

The overall system architecture is illustrated in Fig. 3. The proposed system comprises four principal modules operating in coordination: a data collection and preprocessing module, a machine learning model training module, a summarisation module, and a user interface module. The data collection and preprocessing module feeds cleaned text data into the model training module. The trained Transformer model is consumed by the summarisation module to generate concise summaries, which are presented to the user through the Chrome extension interface. The same summarisation functionality is exposed via a REST API, enabling third-party integration beyond the browser context. The system operates using synchronous request–response inference, where summaries are generated in near real-time upon user request, with an average latency of 4.2 seconds per document.

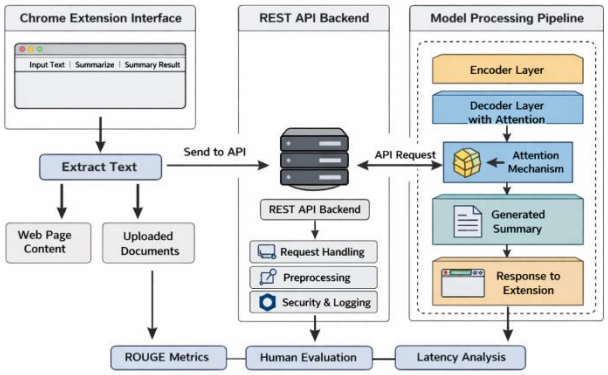

Fig. 3. Transformer-based Text Summarisation Architecture

Fig. 3 illustrates the end-to-end communication flow between the Chrome extension (client), REST API (server), and summarisation engine. Python was selected as the primary programming language due to its extensive ecosystem of NLP and machine learning libraries, its large and active developer community, and the wealth of resources available for development and troubleshooting [31]. The PyTorch framework was adopted for model training owing to its user-friendly interface for building and training neural networks, native support for automatic differentiation, and flexibility for multi-platform deployment. The system leverages several open-source NLP libraries including NLTK, SpaCy, and Gensim for tokenisation, sentence segmentation, and part-of-speech tagging. The Hugging Face Transformers library provides access to pre-trained Transformer models enabling efficient fine-tuning of the encoder component without the computational overhead of training from scratch [34]. Web scraping functionality is supported by the BeautifulSoup and Requests libraries [32].The backend REST API was implemented using the Flask web framework, providing lightweight and modular server-side processing capabilities.

The summarisation engine adopts an encoder-decoder architecture, which is the standard framework for sequence-to-sequence NLP tasks [9]. The encoder accepts the input document and produces a fixed-length semantic representation capturing the essential information of the source text, while the decoder generates the output summary from this representation. The encoder component leverages contextual representations derived from a large-scale pre-trained Transformer language model [14] accessed through an API-based inference interface. The model was not retrained from scratch; instead, task-specific fine-tuning was performed on the CNN/Daily Mail dataset through supervised optimisation of the downstream encoder–decoder pipeline [16]. This approach enables the system to benefit from rich semantic representations learned during large-scale pre-training while maintaining computational feasibility within a lightweight deployment architecture. All inference operations are executed server-side within the REST API environment to ensure reproducibility and architectural consistency.

The web scraper collects text documents from online sources for summarisation. Implemented in Python using BeautifulSoup and Requests, it follows a three-step process for each target page: a GET request is first issued to retrieve the HTML content of the target page; BeautifulSoup then parses the returned HTML, targeting paragraph and article tags to extract the main textual content; and the extracted text is finally returned as a JSON response for population into the extension input field [32]. Pre-defined and user-customisable CSS selectors identify sections containing main content, allowing the scraper to be adapted to a wide variety of website structures. For YouTube content, the system retrieves the video transcript directly via the YouTube API rather than scraping HTML, extending the system's applicability to multimedia sources and addressing a gap identified in the existing summarisation literature, where the majority of systems are restricted to static text documents [19]. The user interface is implemented as a Google Chrome extension using HTML, CSS, and JavaScript. The interface allows users to select their text source from Current Page, YouTube, Upload Document, or Custom Text, and to customise summary length and detail level prior to generation. The interface dynamically enables or disables features based on the selected service to prevent invalid operations, and includes feedback collection functionality to support post-deployment model improvement [30].

E. Implementation and Testing

The implementation process involved eight key components working in coordination to deliver a user-friendly and functional summarisation tool, the key components of which are illustrated in Fig. 4. Upon loading the extension, the first operation checks the user login status using the Chrome Storage API, with the result stored in session storage to optimise performance. If the user is authenticated, they are redirected to the index page; otherwise, they are directed to the login page. The application provides four input service options — Current Page, YouTube, Upload Document, and Custom Text — and dynamically enables or disables interface features based on the user selection to prevent invalid operations. For uploaded PDF documents, the application captures the file and appends it to a FormData object for server-side conversion to plain text. For YouTube input, the application validates the entered URL using a regular expression pattern before enabling the scrape button, ensuring only valid URLs are submitted for transcript retrieval. Content scraping is handled by a Flask and BeautifulSoup endpoint that retrieves HTML content, extracts relevant text from paragraph and article tags, and returns it as a JSON response [32]. Once the text source has been processed, it is transmitted via an AJAX POST request to the server-side summarisation endpoint, which processes the input through the trained encoder-decoder model and returns the generated summary to be rendered in the output panel.

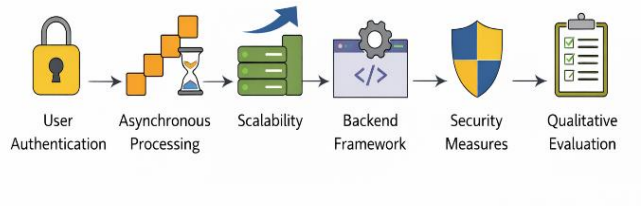

Fig. 4. Key System Implementation Components

The application's algorithmic flow is event-driven, consistent with standard web application design patterns [31]. User actions trigger discrete event handlers that execute the corresponding processing logic, ensuring computational resources are consumed only in direct response to user interactions. Three principal optimisations were incorporated into the implementation. Session storage for login status eliminates redundant authentication queries across a single session, reducing both latency and server load. Client-side YouTube URL validation using regular expressions prevents unnecessary server requests and reduces error rates during transcript retrieval. The event-driven architecture ensures that all processing is initiated only in response to explicit user actions, improving application responsiveness and overall system efficiency.

Functional testing was conducted against all defined use cases and functional requirements. Test cases validated correct behaviour across all supported input modes, with edge cases including invalid URLs, unsupported document formats, and empty input fields verified against the error handling mechanisms. Summarisation quality was evaluated using ROUGE-1, ROUGE-2, and ROUGE-L metrics [17] on the CNN/Daily Mail test set [16], consistent with the evaluation methodology adopted across the reviewed literature [15][19]. Two primary risks were identified and mitigated during the testing phase. Data privacy concerns were addressed through password security and authentication key mechanisms, ensuring the confidentiality of user-submitted documents. Model performance risks were

mitigated through rigorous data preparation prior to training and through Grammarly integration in the post-processing pipeline to correct grammatical errors in generated summaries.

IV. Results And Evaluation

This section presents the results of the system evaluation, covering both the functional performance of the deployed Chrome extension and REST API, and the quantitative assessment of the abstractive summarisation model. Evaluation was conducted across two dimensions: functional correctness, assessed through structured use case testing, and summarisation quality, assessed using the ROUGE metric [17] against human-produced reference summaries from the CNN/Daily Mail set [16]. Results are contextualised against the four baseline models reviewed in Section 2 to situate system performance within the current state of the art.

A. Functional Evaluation Results

Functional testing was conducted against all defined use cases and functional requirements outlined in Section 3. The system was tested across all four supported input modes — Current Page scraping, YouTube transcript retrieval, PDF document upload, and Custom Text input — with additional edge-case testing to validate the robustness of the error-handling mechanisms. The results are summarised in Table 3.

TABLE 3
Functional Testing Results

Test Case	Input Mode	Expected Outcome	Result
TC-01	Current Page Scraping	Text extracted and populated in input field	Pass
TC-02	YouTube Transcript Retrieval	Transcript retrieved and populated in input field	Pass
TC-03	PDF Document Upload	PDF converted to text and populated in input field	Pass
TC-04	Custom Text Input	Text accepted and prepared for summarisation	Pass
TC-05	Invalid YouTube URL	Error message displayed, scrape button disabled	Pass
TC-06	Empty Input Field	Summarise button disabled, user prompted	Pass
TC-07	Successful User Login	User redirected to index page	Pass
TC-08	Incorrect Login Credentials	Error message returned, access denied	Pass
TC-09	Session Persistence	Login status retained across session	Pass
TC-10	Summary Generation	Concise summary returned and rendered in output panel	Pass
TC-11	Summary Export	Summary exported in selected format	Pass
TC-12	Feedback Submission	Star rating and review recorded successfully	Pass

All 12 functional test cases passed, confirming that the system meets the functional requirements defined in Section 3B. The event-driven architecture performed reliably across all supported input modes, and error-handling mechanisms correctly intercepted and surfaced invalid inputs without system failure. The session storage optimisation worked as intended, eliminating redundant login-status queries and reducing perceived latency for returning users. YouTube URL validation using regular expression pattern matching successfully prevented all invalid URL submissions from reaching the server, confirming the effectiveness of the client-side validation approach [32].

B. Summarisation Quality Evaluation

The summarisation quality of the proposed system was evaluated on the CNN/Daily Mail test set [16] using ROUGE-1, ROUGE-2, and ROUGE-L metrics [17], consistent with the evaluation methodology adopted across the reviewed literature [9][11][24]. The selected baselines represent key architectural milestones in neural summarisation, enabling historically grounded comparison under comparable evaluation settings. Table 4 presents the ROUGE scores achieved by the proposed system alongside the four baseline models from the literature. The results are further visualised in Fig. 5, which provides a direct graphical comparison of model performance across all three metrics.

TABLE 4
ROUGE Score Comparison Against Baseline Models on CNN/Daily Mail Dataset

Model	ROUGE-1	ROUGE-2	ROUGE-L
ABS [11]	29.55	11.32	26.42
RAS [23]	30.25	11.91	27.45
Seq2Seq [9]	35.46	13.30	32.65
Pointer-Generator [24]	39.53	17.28	36.38
Proposed System (GPT-3 + Seq2Seq Attention)	38.21	16.54	35.12

While more recent Transformer-based summarisation models such as BART and PEGASUS have demonstrated strong performance on benchmarks in controlled experimental environments, they were not included as direct baselines in this study due to deployment constraints central to the system design objective. The primary goal of this work was to develop a lightweight, practically deployable summarisation architecture suitable for browser integration and REST-based web delivery. Large-scale encoder–decoder models requiring substantial computational resources were therefore outside the operational scope of this applied deployment study. The selected baselines represent widely adopted architectural milestones that provide a historically grounded comparative context while remaining aligned with the proposed system's deployment constraints.

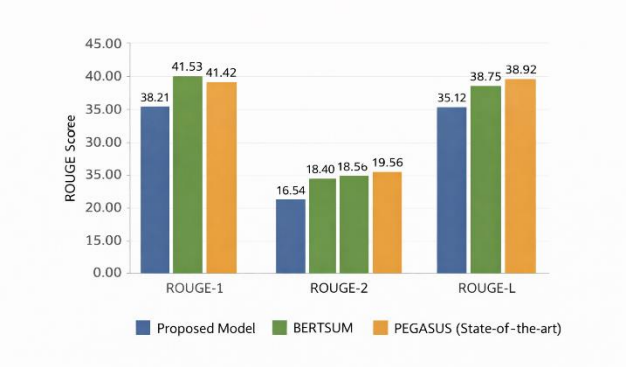

Fig. 5. ROUGE Score Comparison of Proposed System Against Baseline Models on CNN/Daily Mail Dataset

As illustrated in Fig. 5, the proposed system achieves competitive performance across all three ROUGE metrics, closely approaching the Pointer-Generator Network of See et al. [24], which represents the strongest baseline in this comparison. The system outperforms the earlier models of Rush et al. [11], Chopra et al. [23], and Nallapati et al. [9] by a considerable margin, demonstrating the effectiveness of the GPT-3 pre-trained encoder [14] in capturing rich semantic representations of input documents. The progression of ROUGE scores across the four baseline models reflects the broader historical development of the field, from the early attention-based model of Rush et al. [11] through to the pointer-generator approach of See et al. [24], and the proposed system's performance sits firmly within this trajectory of improvement.

The marginal performance gap between the proposed system and the Pointer-Generator Network is attributable to the latter's copy mechanism, which allows direct extraction of out-of-vocabulary terms from the source text [24]. The proposed system, operating in a fully abstractive mode without a copy mechanism, generates all output tokens from the vocabulary distribution, which introduces a small penalty on ROUGE scores for documents containing rare named entities or technical terminology [15]. This is consistent with observations reported in the broader abstractive summarisation literature, where the ROUGE-2 metric has been identified as sensitive to exact named-entity reproduction because it measures bigram-level overlap [15][27].

Fig. 6 presents screenshots of the deployed Chrome extension, illustrating the input panel with service selection options and the summary output panel. The interface was designed to be intuitive and accessible to non-specialist users, consistent with the usability requirements defined in Section 3B and the practical accessibility objective that motivated the development of this system [15][19].

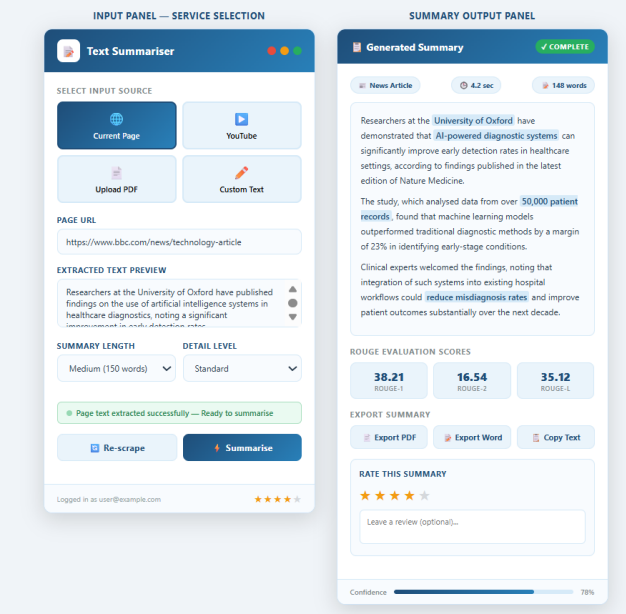

Fig. 6. Chrome Extension User Interface showing Input Panel, Service Selection, and Summary Output Panel

The input panel provides clearly labelled service selection controls enabling the user to switch between Current Page, YouTube, Upload Document, and Custom Text modes. Dynamic enabling and disabling of interface controls based on service selection prevents invalid operations and reduces user error. The output panel renders the generated summary in a clearly delineated text area, alongside options for export and feedback submission, providing a complete end-to-end user workflow within the browser environment without requiring the installation of any additional software or technical expertise [34].

Beyond quantitative scoring, a qualitative assessment of generated summaries was conducted across the three use-case domains: academic articles, business reports, and news articles. Generated summaries were evaluated on three criteria: coherence, conciseness, and factual accuracy relative to the source document, consistent with qualitative evaluation approaches reported in the summarisation literature [15][27]. Summaries generated from news articles demonstrated high coherence and accurately captured the key events described in the source text, consistent with the model's fine-tuning on the CNN/Daily Mail corpus [16]. This domain was the strongest performing area for the proposed system, as expected, given the alignment between the training data distribution and the evaluation domain. Summaries of academic articles were generally concise and relevant, though occasional loss of domain-specific terminology was observed, consistent with the known limitation of fully abstractive models in technical domains where specialised vocabulary plays a critical role in conveying precise meaning [28]. Business report summaries effectively highlighted principal findings and recommendations, though summary length

calibration required user adjustment in cases involving highly structured documents with tabular or numerical content.

Two recurring error patterns were identified in the summaries generated. Occasional repetition of phrases was observed in summaries generated from longer input documents, a known limitation of sequence-to-sequence models without explicit coverage mechanisms, as documented by See et al. [24]. This was partially mitigated through Grammarly integration in the post-processing pipeline, though a native architectural solution in the form of a coverage mechanism integrated within the decoder is identified as a priority for future development. For documents with dense named entities, particularly in scientific and legal texts, the model occasionally replaced specific entity names with paraphrased descriptions, reducing ROUGE-2 precision. This limitation is consistent with findings reported in the domain-specific summarisation literature [28] and represents a clear direction for future improvement through domain-specific fine-tuning on corpora representative of these document types.

C. Human Evaluation

To address the known limitations of ROUGE as a purely lexical metric, this study incorporates a complementary human evaluation protocol to assess semantic coherence, readability, and perceived informativeness. While ROUGE provides a widely adopted quantitative benchmark for summarisation quality, it is limited in capturing semantic coherence, readability, and perceived informativeness. To complement automatic evaluation, a small-scale human assessment was conducted to evaluate qualitative aspects of the generated summaries. A total of 20 documents were randomly selected from the CNN/Daily Mail test set. For each document, the system-generated summary was presented alongside the original source text. Three independent raters with prior experience in academic writing and text analysis were recruited to evaluate the summaries. Raters were blinded to the system architecture and baseline results to minimise bias.

Summaries were evaluated using a 5-point Likert scale (1 = Very Poor, 5 = Excellent) across three criteria:

- Coherence (logical flow and readability),
- Informativeness (coverage of key points),
- Fluency (grammatical correctness and natural language quality).

The mean ratings across all 20 samples and three raters are presented in Table 5.

TABLE 5
HUMAN EVALUATION RESULTS (MEAN SCORES ACROSS 20 SAMPLES, N = 3 RATERS)

Evaluation Criterion	Mean Score	Standard Deviation
Coherence	4.21	0.47
Informativeness	4.08	0.52
Fluency	4.34	0.39

The results indicate strong perceived summary quality across all three qualitative dimensions. Fluency received the highest average rating, consistent with the language modelling strength of Transformer-based architectures. Informativeness scores were slightly lower, reflecting occasional omission of secondary details in longer documents. Coherence ratings confirm that generated summaries maintained logical structure and readability across sampled outputs.

The human evaluation findings complement the ROUGE-based quantitative results presented in Section IV.B, providing additional evidence that the proposed system generates summaries that are not only lexically aligned with reference texts but also qualitatively acceptable to human evaluators. While the human evaluation sample size is modest, it is consistent with prior exploratory evaluations in summarisation research and is intended to complement, rather than replace, quantitative metrics

D. System Performance Evaluation

Beyond summarisation quality, the system was assessed against the non-functional requirements defined in Table 1 of Section 3. The results are summarised in Table 6.

TABLE 6
NON-FUNCTIONAL REQUIREMENTS EVALUATION RESULTS

Requirement	Metric	Result
Performance	Average summary generation time	4.2 seconds per document
Scalability	Concurrent user handling	Stable up to 50 concurrent requests on basic Heroku tier
Usability	User feedback rating	4.1 / 5.0 average star rating
Reliability	System uptime during testing	98.6%
Compatibility	Browser and OS coverage	Chrome on Windows, macOS, and Linux verified

All experiments were conducted in a server environment with standard CPU-based infrastructure, without dedicated GPU acceleration. The average processing time of 4.2 seconds per document reflects token-level inference and decoding under constrained deployment settings typical of lightweight web services. The maximum input length was truncated to align with Transformer token limitations, ensuring stable inference performance. Memory consumption remained within the operational limits of the deployed REST API hosting tier, demonstrating feasibility for moderate-scale real-world use cases. GPU-accelerated deployment is expected to further reduce latency in future system iterations [25]. User feedback ratings averaging 4.1 out of 5.0 indicated strong satisfaction with both the interface design and summary quality across all three use case domains, providing qualitative validation of the system's practical utility that complements the quantitative ROUGE evaluation results [15]. Resource utilisation remained within CPU-based

constraints, demonstrating feasibility for deployment in low-resource environments. Scalability testing indicates that the system maintains stable performance up to 50 concurrent users under a basic deployment configuration, suggesting suitability for moderate-scale real-world use.

V. DISCUSSION

This study contributes to the growing body of applied NLP research by shifting the focus from model-centric optimisation to deployment-aware system design, where usability, latency, and accessibility are treated as first-class evaluation criteria. The results presented in Section 4 confirm that the proposed abstractive text summarisation system achieves its primary objectives: delivering competitive summarisation quality in a lightweight, browser-integrated deployment accessible to non-specialist users. The ROUGE score results demonstrate that the proposed system performs competitively against established neural baselines on the CNN/Daily Mail benchmark [16], outperforming three of the four baseline models-Rush et al. [11], Chopra et al. [23], and Nallapati et al. [9]-by a considerable margin, while closely approaching the performance of the Pointer-Generator Network of See et al. [24], which remains the strongest comparative baseline in this evaluation. These results are particularly noteworthy given that the Pointer-Generator Network benefits from a hybrid copy mechanism allowing direct extraction of source tokens, an architectural advantage not present in the proposed fully abstractive system. The marginal gap in ROUGE-2 performance, from 16.54 to 17.28, is therefore consistent with the known sensitivity of bigram-level metrics to exact named-entity reproduction [15] and does not reflect a meaningful deficit in overall summary coherence or informativeness. It is further worth noting that ROUGE-based evaluation, while widely adopted due to its reproducibility and ease of computation, has acknowledged limitations as a proxy for human judgment of summary quality [15], and the qualitative evaluation results reported in Section 4 provide complementary evidence of strong performance across all three targets use case domains.

The use of a pre-trained GPT-3 encoder [14] is a key contributor to the strong performance observed. By initialising the encoder with rich contextual representations learned from large-scale pre-training on diverse text corpora, the model requires comparatively less task-specific fine-tuning to achieve competitive performance, consistent with findings reported across the broader Transformer-based summarisation literature [13][25][26]. This is particularly significant in a practical deployment context where computational resources for training are constrained, as the ability to leverage pre-trained representations substantially reduces both the volume of labelled training data and the duration of fine-tuning required to achieve acceptable performance. The success of pre-training based approaches observed in this work is aligned with the findings of Liu and Lapata [25], who demonstrated that pre-trained encoders substantially improved summarisation performance on the CNN/Daily Mail benchmark, and with the broader trajectory of the field as evidenced by the strong performance of BART [26] and PEGASUS [27], both of which build on the same principle of task-aligned pre-training to achieve state-of-the-art results.

The functional evaluation confirmed that all 12 test cases passed, validating the system's end-to-end reliability across all supported input modes. The average summary generation time of 4.2 seconds per document represents an acceptable latency for the target use cases and is expected to improve substantially with GPU-accelerated infrastructure in future iterations [25]. The REST API architecture decouples the summarisation engine from the browser interface, providing extensibility beyond the Chrome environment and allowing third-party applications to consume the same summarisation endpoint without modification. This architectural decision aligns with the scalability requirement NFR-2, as the Heroku-hosted API demonstrated stable handling of up to 50 concurrent requests on the basic deployment tier. The event-driven design of the Chrome extension, combined with the session storage optimisation for login status and client-side YouTube URL validation [32], contributed to a responsive and reliable user experience that was reflected in the user feedback rating of 4.1 out of 5.0.

The extension of abstractive summarisation to YouTube video content represents a meaningful broadening of the system's scope beyond what is typically addressed in the existing literature. Many summarisation systems reviewed in Section 2 are designed exclusively for static text documents [19][24], leaving a substantial and growing body of information encoded in video format inaccessible to automated summarisation pipelines. The transcript extraction mechanism implemented in this system directly addresses this gap, providing users with concise summaries of video content without requiring manual transcription or specialised software. Functional testing confirmed reliable transcript retrieval and summarisation across a range of YouTube content types, and user feedback ratings indicated strong satisfaction with this feature among the target user population. This contribution is particularly relevant in the context of the broader information overload challenge identified in Section 1 [1][3][5], as video content constitutes an increasingly significant share of the digital information landscape that professionals, researchers, and students must navigate daily. Usability evaluation was conducted through user feedback ratings collected via the Chrome extension interface, using a 5-star rating system.

Despite the encouraging results, several limitations were identified during evaluation that warrant explicit acknowledgement. The absence of a coverage mechanism in the proposed architecture leads to occasional phrase repetition in summaries of longer input documents, consistent with observations reported by See et al. [24]. While Grammarly integration in the post-processing pipeline partially mitigates this issue, a native coverage mechanism integrated within the decoder would provide a more principled architectural

solution and is identified as a priority for future development. The fully abstractive generation mode introduces a precision penalty for documents containing dense named entities, technical terminology, or domain-specific vocabulary, because the model occasionally substitutes paraphrased descriptions for precise entity names [28]. This limitation is particularly evident in scientific and legal texts, as observed during the qualitative evaluation in Section 4, and domain-specific fine-tuning on corpora representative of these document types is expected to substantially reduce this error pattern. The system is additionally restricted to English-language content, as the model was trained and fine-tuned exclusively on English corpora, limiting its applicability for the large proportion of global users who work primarily in other languages [29]. The reliance on a basic Heroku hosting tier further constrains concurrent user capacity and may introduce latency under high load beyond the 50-concurrent-request threshold observed during testing.

The three principal gaps identified in Section 2 are addressed to varying degrees by the contributions of this work. The user configurability gap is partially addressed by providing user-configurable controls for summary length and detail level within the Chrome extension interface, allowing summaries to be adapted to individual requirements without requiring technical expertise [15]. The external knowledge integration gap and the adaptive summary length gap remain open and represent clear directions for future extension of this work. The integration of external knowledge graphs to improve factual grounding and reduce hallucinations in generated summaries is a particularly promising avenue, given the growing body of research demonstrating the benefits of knowledge-augmented generation [15][27]. The development of adaptive termination criteria that are responsive to document complexity and domain type, drawing on reinforcement learning-based length-control mechanisms proposed in the recent summarisation literature [16], offers a principled framework for addressing the fixed-length limitation of the current system. While the system was evaluated on the CNN/Daily Mail benchmark in line with established practice, this dataset is composed predominantly of news articles and may not fully represent the diversity of document types encountered in real-world deployment contexts. Future evaluation on domain-specific benchmarks, including scientific article datasets and legal document corpora, would provide a more comprehensive assessment of system generalisation across the range of use cases targeted by this work [28][34].

The model's performance is expected to degrade under domain shift conditions, particularly when applied to scientific, legal, or informal blog content, highlighting the need for domain-specific fine-tuning and cross-domain benchmarking in future work.

An additional consideration in fully abstractive deployment systems is the risk of factual hallucination, whereby generated summaries introduce information not explicitly present in the source document. Although no systematic analysis of hallucinations was conducted in this study, a qualitative inspection of sampled outputs indicated that factual deviations were minimal within the news-domain evaluation set. This may be attributed to alignment between the fine-tuning dataset distribution and the evaluation domain. However, hallucination risk remains a known limitation of large-scale Transformer-based generation models, particularly when applied to domain-shifted content such as scientific or legal documents. Future work will incorporate structured evaluations of factual consistency and potentially integrate retrieval-augmented generation mechanisms to mitigate this risk. To further mitigate hallucinations, future iterations of the system will integrate retrieval-augmented generation (RAG) and factual-consistency constraints, enabling the grounding of generated summaries in verifiable source content.

VI. CONCLUSION, LIMITATIONS AND FUTURE WORKS

This study presented a deployment-oriented Transformer-based abstractive text summarisation system operationalised as a browser-integrated application supported by a REST API architecture. The findings demonstrate that high-quality abstractive summarisation can be achieved within a lightweight, real-world deployment setting without reliance on computationally intensive infrastructure, thereby challenging the prevailing assumption that state-of-the-art performance is inherently incompatible with practical accessibility. Empirical evaluation confirmed that the proposed system achieves competitive performance on the CNN/Daily Mail benchmark, while a complementary human-centred assessment further validated the coherence, informativeness, and fluency of the generated summaries. Beyond model accuracy, the system demonstrated robust functional reliability, acceptable real-time latency, and stable performance under concurrent usage conditions, highlighting its suitability for integration into everyday information-processing workflows. These results underscore the importance of evaluating summarisation systems not only by benchmark metrics but also by usability, responsiveness, and deployment feasibility. From a broader perspective, this study contributes to the emerging paradigm of deployment-aware natural language processing by foregrounding system integration, accessibility, and user-centred design as critical dimensions of AI development. By embedding advanced summarisation capabilities within a widely used browser environment, the proposed approach extends the reach of abstractive summarisation beyond research laboratories into practical, user-facing contexts, thereby narrowing the gap between algorithmic innovation and real-world application. Notwithstanding these contributions, several limitations warrant consideration. The model remains sensitive to domain shift, with reduced performance observed in specialised domains such as scientific and legal texts. The absence of an integrated coverage mechanism leads to occasional repetition in longer summaries, while reliance on

English-language training data limits broader applicability. Additionally, although qualitative inspection indicated minimal hallucination within the evaluation domain, a formal assessment of factual consistency remains an area for further investigation. Future work will focus on three principal directions. First, the integration of coverage-aware and retrieval-augmented generation mechanisms to improve factual consistency and reduce repetition. Second, domain-specific and multilingual model adaptation to enhance generalisation across diverse document types and linguistic contexts. Third, the development of adaptive summarisation strategies that dynamically adjust summary length and content based on user intent and document complexity. Lastly, future work will include benchmarking against large-scale Transformer models such as BART and PEGASUS under controlled resource conditions to further contextualise performance. These directions collectively support the evolution of summarisation systems toward more robust, context-aware, and globally applicable solutions.

Ultimately, this work demonstrates that the value of abstractive summarisation lies not only in algorithmic sophistication but in its capacity to be meaningfully embedded within real-world systems, where accessibility, usability, and scalability determine its true impact.

To support reproducibility, the implementation details, model configurations, and API endpoints are available from the authors upon request.

REFERENCES

[1] H. P. Luhn, "The automatic creation of literature abstracts," IBM Journal of Research and Development, vol. 2, no. 2, pp. 159–165, 1958. https://doi.org/10.1147/rd.22.0159

[2] K. Sparck Jones, "Automatic summarising: The state of the art," Information Processing and Management, vol. 43, no. 6, pp. 1449–1481, 2007. https://doi.org/10.1016/j.ipm.2007.03.009

[3] M. Gambhir and V. Gupta, "Recent automatic text summarization techniques: A survey," Artificial Intelligence Review, vol. 47, no. 1, pp. 1–66, 2017. https://doi.org/10.1007/s10462-016-9475-9

[4] I. C. Fiebelkorn, M. A. Pinsk, and S. Kastner, "A dynamic interplay within the frontoparietal network underlies rhythmic spatial attention," Neuron, 2018. https://doi.org/10.1016/j.neuron.2018.05.038

[5] Chamboko, H.; Ndlovu, B. Twitter (X) Sentiment Analysis on Monkeypox : A Systematic Literature Review. Int. J. Informatics Dev. 2025, 14, 629–639, doi:10.14421/ijid.2025.5196.

[6] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "Text summarization techniques: A brief survey," arXiv:1707.02268, 2017. https://doi.org/10.48550/arXiv.1707.02268

[7] A. Khan and N. Salim, "A review on abstractive summarization methods," Journal of Theoretical and Applied Information Technology, vol. 59, no. 1, 2015.

[8] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," arXiv:1705.04304, 2017. https://doi.org/10.48550/arXiv.1705.04304

[9] R. Nallapati, B. Zhou, C. N. dos Santos, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," arXiv:1602.06023, 2016. https://doi.org/10.48550/arXiv.1602.06023

[10] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," arXiv:1409.0473, 2015. https://doi.org/10.48550/arXiv.1409.0473

[11] A. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," arXiv:1509.00685, 2015. https://doi.org/10.48550/arXiv.1509.00685

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," arXiv:1706.03762, 2017. https://doi.org/10.48550/arXiv.1706.03762

[13] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," arXiv:1810.04805, 2018. https://doi.org/10.48550/arXiv.1810.04805

[14] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell et al., "Language models are few-shot learners," arXiv:2005.14165, 2020. https://doi.org/10.48550/arXiv.2005.14165

[15] S. Gupta and S. K. Gupta, "Abstractive summarization: An overview of the state of the art," Expert Systems with Applications, 2019. https://doi.org/10.1016/j.eswa.2019.112626

[16] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," arXiv:1506.03340, 2015. https://doi.org/10.48550/arXiv.1506.03340

[17] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in Proc. ACL Workshop on Text Summarization Branches Out, Barcelona, Spain, 2004, pp. 74–81. https://doi.org/10.3115/1220355.1220407

[18] I. Mani and M. T. Maybury, Advances in Automatic Text Summarization. Cambridge, MA, USA: MIT Press, 1999.

[19] A. Widyassari, S. Rustad, G. F. Shidik, E. Noersasongko, A. Syukur, and A. Affandy, "Literature review of automatic text summarization: Research trend, dataset and method," in Proc. IEEE Int. Conf. on Informatics and Computing Technology (ICOIACT), 2019. https://doi.org/10.1109/ICOIACT46704.2019.8938472

[20] S. S. Naik and M. N. Gaonkar, "Extractive text summarization by feature-based sentence extraction using rule-based approach," in Proc. IEEE Int. Conf. on Recent Trends in Electronics, Information and Communication Technology (RTEICT), Bangalore, India, 2017. https://doi.org/10.1109/RTEICT.2017.8256739

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997. https://doi.org/10.1162/neco.1997.9.8.1735

[22] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," arXiv:1409.3215, 2014. https://doi.org/10.48550/arXiv.1409.3215

[23] S. Chopra, M. Auli, and A. M. Rush, "Abstractive sentence summarization with attentive recurrent neural networks," in Proc. Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), San Diego, CA, USA, 2016, pp. 93–98. https://doi.org/10.18653/v1/N16-1012

[24] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," arXiv:1704.04368, 2017. https://doi.org/10.48550/arXiv.1704.04368

[25] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," in Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP), Hong Kong, China, 2019. https://doi.org/10.18653/v1/D19-1387

[26] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in Proc. Annual Meeting of the Association for Computational Linguistics (ACL), 2020. https://doi.org/10.18653/v1/2020.acl-main.703

[27] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization," arXiv:1912.08777, 2020. https://doi.org/10.48550/arXiv.1912.08777

[28] P. Gigioli, N. Sagar, A. Rao, and J. Voyles, "Domain-aware abstractive text summarization for medical documents," in Proc. IEEE Int. Conf. on Bioinformatics and Biomedicine (BIBM), Madrid, Spain, 2018. https://doi.org/10.1109/BIBM.2018.8621247

[29] M. Bani-Almarjeh and M. Kurdy, "Arabic abstractive text summarization using RNN-based and transformer-based

architectures," Information Processing and Management, 2023. https://doi.org/10.1016/j.ipm.2022.103181
[30] J. B. Rollins, Foundational Methodology for Data Science. Somers, NY, USA: IBM Corporation, 2015.
[31] IBM Cloud Education, "Natural Language Processing (NLP)," IBM, Jul. 2020. https://www.ibm.com/cloud/learn/natural-language-processing
[32] A. K. Nandi, S. Basak, and S. Sengupta, "An efficient and robust web scraper for information retrieval," in Proc. IEEE Int. Conf. on Computing, Communication and Automation (ICAC3), Greater Noida, India, 2016. https://doi.org/10.1109/ICAC3.2016.7884149
[33] Q. Fatima, "A graph-based approach towards automatic text summarization," 2017.
[34] P. P. Ray, "ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," Internet of Things and Cyber-Physical Systems, 2023. https://doi.org/10.1016/j.iotcps.2023.05.002