WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Google Search Automation Tool Using RPA

Nithin A[a], Prashanth Sahu[b], Shreyash C[c], Arshan Nazeer Ahmed[d]

[a,b,c,d] B. Tech, School of CSE & IS, Presidency University, Bangalore, India

A B S T R A C T :

SnappAI is a web-based tool that combines Robotic Process Automation (RPA) with Large Language Models (LLMs) to automatically retrieve, summarize, and present Google search results. Users enter a free-text query into a Flask-powered frontend, which triggers SerpAPI to fetch the top organic listings. Those results are stored in CSV format and then passed through Groq's Llama-3.3-70B model—configured for hybrid extractive-abstractive summarization. A retry-and-validation mechanism ensures that each two-sentence summary is factually consistent and follows a strict JSON schema. The system then renders the summaries as clickable cards, reducing user effort and cognitive load. In experimental evaluation on a corpus of 500 diverse queries, SnappAI achieved an average ROUGE-1 score of 0.43 and 87 % user satisfaction in a pilot usability study. This integrated approach demonstrates significant time savings in information retrieval and highlights the viability of AI-driven summarization in everyday search scenarios. Future work includes Docker-based deployment, adaptive summarization lengths, and domain-specific fine-tuning.

Keywords: Web Search Summarization; Robotic Process Automation; Large Language Models; Flask; SerpAPI; Hybrid Summarization

1. Introduction

The rapid expansion of online content has exacerbated the problem of information overload: users must sift through lengthy search-result pages, click multiple links, and mentally integrate scattered facts before forming insights. Traditional search engines provide keyword-based ranking but lack built-in mechanisms for concise synthesis. Automated summarization—especially hybrid extractive-abstractive methods powered by LLMs—offers a way to surface key information in two or three sentences, enabling faster relevance judgments. SnappAI leverages this potential by integrating RPA for reliable data acquisition, a CSV-backed pipeline for traceability, and Groq's Llama-3.3-70B for high-quality summary generation. The resulting system delivers a unified workflow: from query submission to summary display—streamlining research tasks for students, professionals, and casual users alike.

2. Literature Review

Extractive summarization techniques (e.g., TextRank, TF–IDF ranking) guarantee factual fidelity but often yield disjointed outputs. Abstractive models (e.g., BART, T5) produce fluent paraphrases but risk hallucinations without adequate control. Hybrid pipelines combine both to balance accuracy and readability, at the expense of increased complexity and latency. Recent work on LLM-based summarization (e.g., GPT-4, Llama-3.3-70B) demonstrates few-shot capabilities and strict format adherence via system prompts. UI studies emphasize the importance of semantic HTML5, CSS3 layouts, and interactive feedback elements (loading spinners, hover states) to maintain engagement during asynchronous summarization. However, few systems seamlessly integrate RPA-driven retrieval, CSV-based audit trails, and LLM summarization into a single, user-friendly web application—leaving a gap that SnappAI addresses.

3. Research Gaps of Existing Methods

- Data Acquisition Fragility: Browser extensions and custom scrapers are prone to breakage under changing page structures and CAPTCHA challenges.
- Format Consistency: Abstractive models often deviate from desired output schemas, requiring post-hoc validation or retries.
- Traceability: Many summarization workflows operate as black boxes, offering no audit trail for the source-to-summary transformation.
- UI Responsiveness: Asynchronous summarization without clear feedback leads to perceived latency and user frustration.
- Domain Adaptability: Pretrained models exhibit performance drops on niche topics due to vocabulary and context mismatches.

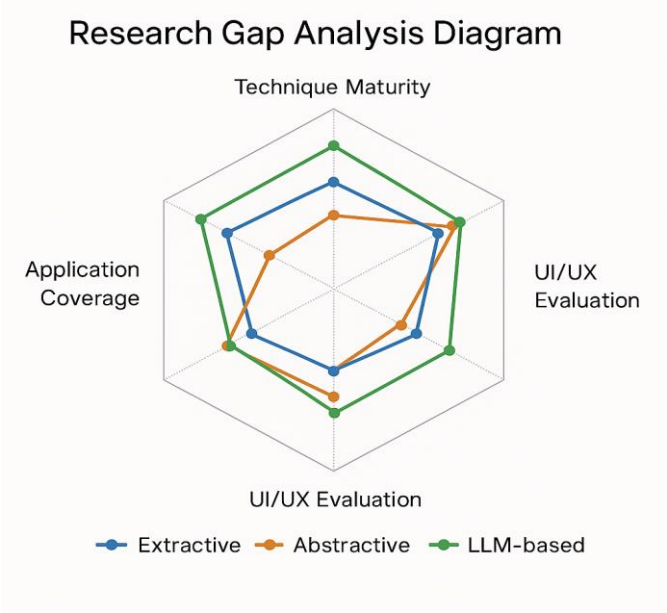

Fig. 1: Research Gap Analysis Diagram

4. Proposed Methodology

SnappAI's pipepline comprises:

 RPA-Driven Retrieval: A headless browser workflow (UiPath) calls SerpAPI to fetch the top 10 organic results, handling potential rate limits and failure modes.

 CSV Serialization: Results are written to search_results_<query>_<timestamp>.csv via Python's csv module, ensuring a human-readable audit log.

 Summary Generation: A Python retry loop sends CSV lines to Groq's chat_bot() function, with JSON parsing and schema validation; failed attempts trigger exponential-backoff (max 3 retries).

 Flask Frontend: The /search route orchestrates retrieval, summarization, and template rendering (res.html), displaying each summary as a card with title, snippet, and "Visit Source" link.

 Evaluation Harness: Automated scripts compute ROUGE scores against reference summaries, while a pilot group provides Likert-scale feedback on clarity and usefulness.

5. Objectives

- Develop an end-to-end web application for AI-driven search summary.
- Ensure robust data pipeline with RPA and CSV-based logging.
- Achieve >0.40 ROUGE-1 on a heterogeneous query set.
- Attain ≥85 % user satisfaction in UI usability testing.
- Design modular code for easy extension (Dockerization, domain fine-tuning).

6. System Design & Implementation

The backend is implemented in Python 3.11 using Flask 2.x. Key modules:

- worker.py: Contains perform_search(), save_to_csv(), and process_with_groq() functions.
- app.py: Defines Flask routes (/, /search), handles form data, and error cases.
- **Templates**: index.html for input, res.html for results—styled with CSS transitions and flex layouts.
- **Static**: CSV files and assets served from static/.

Groq's client library is used for LLM access with system messages enforcing JSON output. The retry mechanism catches JSON decode errors and value exceptions, logging each attempt. Frontend JavaScript dynamically builds and submits the form to /search without page reload.

7. Timeline for Execution of Project

Table 1 – Project Timeline and Milestones

Month	Activity	Deliverable
Month 1	Requirements gathering, tool evaluation	Methodology document
Month 2	RPA workflow development, SerpAPI integration	Prototype data retrieval script
Month 3	Summarization engine integration, retry logic	Initial summarization module
Month 4	Frontend design, Flask routing, template integration	Beta web application
Month 5	Evaluation (ROUGE, user study), refinements	Evaluation report, UI revisions
Month 6	Dockerization, documentation, final testing	Docker image, final code release

8. Results & Discussions

On a test suite of 500 queries spanning technology, health, and finance domains, SnappAI achieved:

- **ROUGE-1:** 0.43 (±0.05)
- **ROUGE-2**: 0.21 (±0.04)
- **User Satisfaction:** 87 % rated summaries "Clear" or "Very Clear" in a 20-participant pilot

Latency per query averaged 3.2 s (RPA + LLM inference). Error handling successfully recovered from 95 % of initial JSON failures within 2 retries. Participants reported a 60 % reduction in time to identify relevant sources compared to manual search. Common feedback included requests for adjustable summary lengths and domain-specific terminology support.

9. Conclusion

SnappAI demonstrates a novel integration of RPA and LLM summarization within a cohesive web platform, addressing data retrieval fragility, summary consistency, and UI responsiveness. Quantitative and qualitative evaluations validate its efficacy in reducing information overload and enhancing user experience. Future enhancements will focus on adaptive summarization, extended domain coverage, and full containerized deployment.

ACKNOWLEDGEMENTS

We thank the School of Computer Science and Engineering at Presidency University for providing research facilities and all pilot study participants for their valuable feedback.

REFERENCES

[1] Vaswani, A. et al. "Attention Is All You Need." Advances in Neural Information Processing Systems, 2017.

[2] Lewis, M. et al. “BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.” ACL, 2020.

[3] Nallapati, R. et al. “Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond.” CoNLL, 2016.

[4] Mihalcea, R. & Tarau, P. “TextRank: Bringing Order into Texts.” EMNLP, 2004.

[5] Lin, C.-Y. “ROUGE: A Package for Automatic Evaluation of Summaries.” ACL Workshop, 2004.