Towards Autonomous Network Management: AI-Driven Framework for Intelligent Log Analysis, Troubleshooting and Documentation

Shaghayegh Shajarian
North Carolina Agricultural and Technical State University, Greensboro, NC, USA
sshajarian@aggies.ncat.edu

Abstract

As modern network management grows increasingly complex, administrators are tasked with navigating vast volumes of log data, often resulting in inefficiencies, errors, and operational challenges. My doctoral research addresses these pressing issues by leveraging advanced AI techniques to minimize human intervention and pave the way for fully automated network operations. I propose a novel AI-driven framework that integrates Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG) and a human-in-the-loop process to effectively automate key network management tasks, including log analysis, troubleshooting recommendations, and documentation generation. By enhancing the accuracy and efficiency of these tasks, this study aims to improve network reliability, reduce operational complexity, and contribute to the evolution of self-running networks.

Introduction

Today's networks comprise both vendor-specific and vendor-agnostic equipment operating in physical and virtual environments. These networks are deployed across a wide range of settings, from data centers and cloud infrastructures to IoT devices and mobile core networks. The devices within each domain generate large volumes of technical logs in various formats and granularity. Traditional network management approaches require administrators to navigate a maze of these log files, which often leads to misconfigurations, security vulnerabilities, and high operational overhead.

Specifically, log analysis methods struggle with the high volume and complexity of data, which hinders the extraction of important information and effective diagnosis (He et al. 2022). Despite the advancements in tools, network operators must learn proprietary query languages like Splunk's SPL or Google's Logging Query Language to analyze logs. Additionally, troubleshooting requires correlating events across logs. Accurate technical records are also crucial for decision-making and future troubleshooting (Benson, Akella, and Maltz 2009). The emergence of self-running networks, which aim to automate many network operations, further underscores the need for advanced solutions (Feamster and Rexford 2017).

Given the advancements in Artificial Intelligence (AI), particularly Large Language Models (LLMs), there is a significant opportunity to enhance network management for operators handling log analysis, troubleshooting, and documentation. Therefore, the central research question guiding my dissertation is: **How can I develop an AI-driven framework that automates network operations—including log analysis, troubleshooting, and documentation generation—while adapting to the dynamics of self-running networks?**

Proposed Framework

Motivated by the outlined challenges and research question, I propose a *"human assisted AI-driven closed-loop framework"* with four main components: ***1) Log Analyst Agent***, ***2) Troubleshooting Recommender Agent***, ***3) Human Validator***, and ***4) Documenter Agent***.

The proposed framework leverages multimodal LLMs for task execution. Despite their effectiveness, LLMs are known to generate hallucinated information and often lack domain-specific knowledge. To mitigate these limitations, the framework incorporates the Retrieval-Augmented Generation (RAG) technique (Lewis et al. 2020), which enhances LLM capabilities by coupling the model's generative output with a retrieval mechanism that accesses predefined knowledge sources. This ensures the responses are grounded in relevant, accurate information.

As shown in Figure 1, the framework is initiated with preprocessed logs—gathered from network devices and environments—which serve as input for the Log Analyst Agent. These logs, structured as prompts, contain technical details such as error messages, event sequences, and system states. The Log Analyst Agent, integrated with RAG, analyzes these logs to detect and identify network issues or failures. It cross-references the logs with historical data, best practices, and other relevant documentation from the RAG knowledge base to produce a comprehensive analysis.

Once the Log Analyst Agent identifies a potential issue, it outputs the detected issue's details. The Troubleshooting Recommender Agent interprets the nature of the problem and generates actionable, natural language recommendations for resolving the identified issue.

The recommendations generated by this agent are validated through a human-in-the-loop process, where network

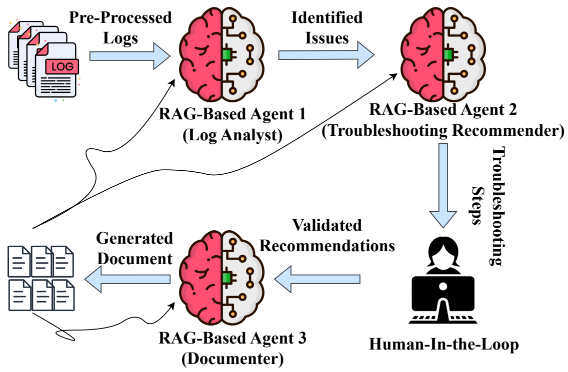

Figure 1: Proposed Research Framework

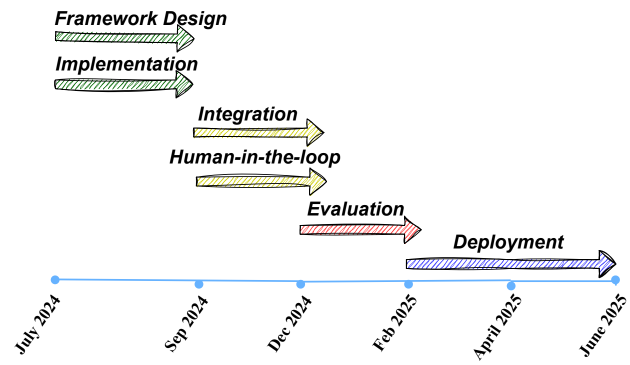

Figure 2: Research Timeline

operators provide feedback. They can either approve or modify the recommendations based on their expertise. This component has two benefits for the proposed framework. First, it ensures that AI-generated recommendations are vetted by experts and the accuracy and relevance of the recommendations will be checked before they are documented. Additionally, by tracking how often users modify or accept the recommendations, I will assess the overall accuracy and effectiveness of the proposed framework.

After validating the recommendation, the last agent, the Documentor, generates the well-documented report. Thanks to its multi-modality ability, it can generate both textual and visual documentation, such as diagrams or network topology visualizations, to create more comprehensive documentation. The generated documents are utilized in post-hoc analysis and support. As this framework is closed-loop, these documents also serve as entries in the RAG knowledge base, where each LLM agent can access detailed records of past network issues and their solutions and thus be able to obtain accurate and up-to-date information.

Research Timeline

As shown in Figure 2, by September 2024, I had covered the crafting of my research question and the design of the framework, alongside the preliminary implementation of components. The implementation, built using Langchain [1], handled efficient retrieval of relevant information from various document formats, including PDFs, text files, and JSON files. The documents were loaded and split into manageable chunks for efficient retrieval and generation. The developed system generated vector embeddings and used a vector index to store these embeddings, allowing for rapid similarity-based search.

By February 2025, I expect to complete the integration of this framework. I will also ensure human-in-the-loop functionality to validate AI-generated recommendations. Moreover, I will present a comprehensive evaluation of the framework's performance, using both quantitative metrics (e.g., precision, recall, F1-score) and qualitative user feedback to measure the framework's effectiveness and ensure the framework's applicability in real-world environments. After the performance evaluation, by June 2025, the framework will be deployed in a real-world network environment for practical testing and further refinement, and its paper will be submitted.

[1] https://github.com/langchain-ai

Future Work

For the next step of this research, I intend to enhance the framework by integrating predictive analytics alongside ReAct (Yao et al. 2023), enabling it to anticipate and autonomously resolve network issues through structured, step-by-step reasoning and proactive self-healing actions.

Acknowledgments

This work is being supported in part by National Science Foundation (NSF) Grants #2113945 and #2200538. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agency.

References

Benson, T.; Akella, A.; and Maltz, D. A. 2009. Unraveling the complexity of network management. In *NSDI*, 335–348.

Feamster, N.; and Rexford, J. 2017. Why (and how) networks should run themselves. *arXiv preprint arXiv:1710.11583*.

He, S.; Zhang, X.; He, P.; Xu, Y.; Li, L.; Kang, Y.; Ma, M.; Wei, Y.; Dang, Y.; Rajmohan, S.; et al. 2022. An empirical study of log analysis at Microsoft. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 1465–1476.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. arXiv:2210.03629.