EXTRACTING FACTS FROM FICTION

A Summarization Pipeline for Narrative Text

Patrick Conan

Data Science Capstone Project

ADVISOR Ali Baheri

YEAR 2025

RIT Rochester Institute of Technology

Software Engineering
Rochester Institute of Technology

Objective

Summarizing fiction is a challenging task due to the complexity of long narratives. This project develops a scalable pipeline in Python to write book summaries using Large Language Models and Knowledge Graphs. Strong system performance is characterized by factual accuracy, narrative coherence, and low processing time.

Knowledge Graph

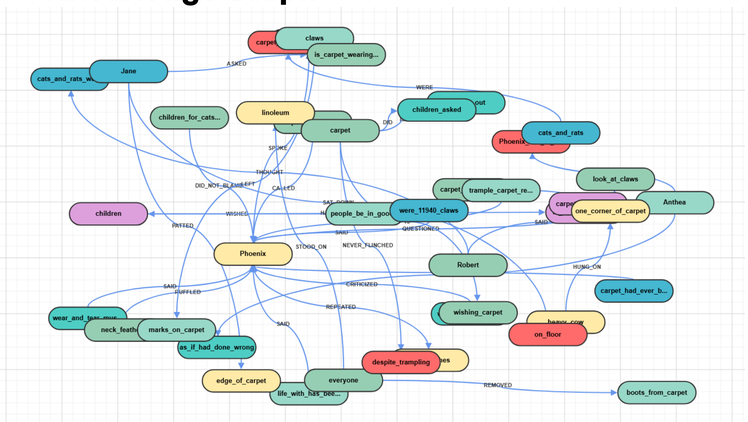

Figure 2. Graph describing a single chunk.

Relation Extraction is performed on the input text using TextaCy and OpenIE to generate a list of semantic triples (Subject Verb Object). Triples are cleaned, classified, and verified using LLM prompts (GPT-5), then sent to the appropriate fact graph in our Neo4j database.

Scalability Analysis

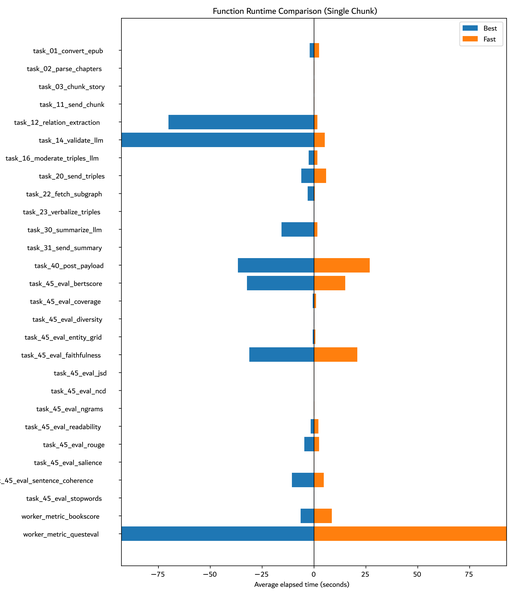

Figure 4. Time elapsed for pipeline jobs.

Architecture

python neo4j MongoDB. Blazor Flask

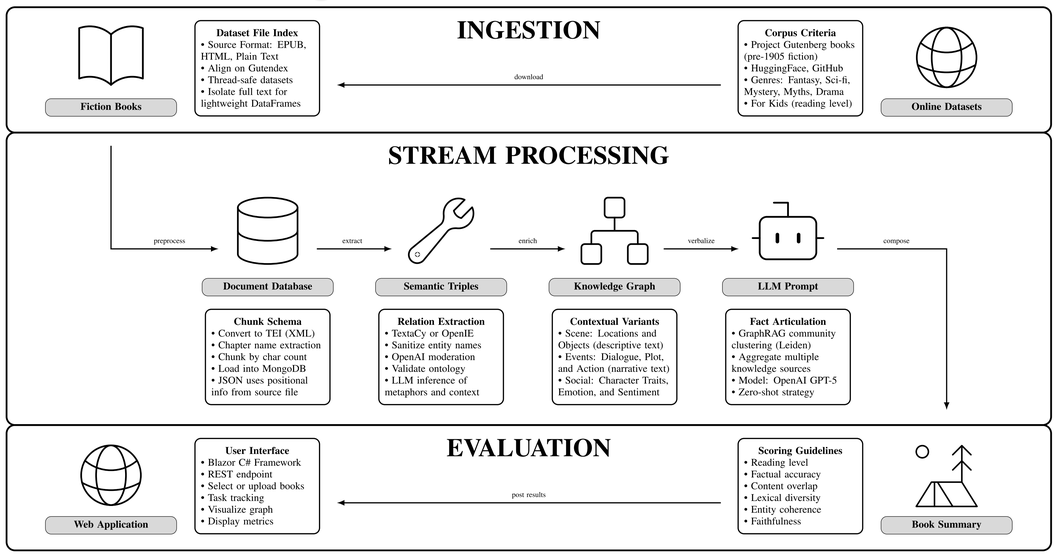

Figure 1. A visualization of data flow through the stream processing pipeline.

Methodology

This data science project was implemented using Test Driven Development and smoke tests to ensure robust, reliable code, while leveraging Agile, LLM code generation, and Git for efficient iteration. Development was conducted solo, with progress tracked through GitHub milestones, issues, and pull requests, in addition to lightweight task management in Jira. Documentation was maintained with Doxygen, and we utilized a CI/CD pipeline to execute automated PyTests within Docker containers.

Efficient Summary

"A phoenix is said to live for about five hundred years in the wilderness. It builds a pile of sweet wood and aromatic gums, and the pile is set on fire. The phoenix burns itself, as does another self described as "it." The phoenix has an egg and lays eggs, and one egg is placed on its own pile. The creature's tail is presented to children, and it eventually goes to sleep and wakes up inside the egg. Afterward, it lives again forever and ever."

High-Quality Summary

"The Phoenix, ruffling its golden feathers, denounces an inaccurate book about it and calls for its destruction. It rejects the purple-body and "worm-from-the-ashes" myths, shows its tail to children as proof, and explains the true cycle: like other respectable birds, it builds a pile of sweet woods and aromatic gums, kindles it by wafting its wings, lays an egg, goes to sleep, then wakes inside the egg, emerges renewed, and repeats this forever. At times it admits weariness and a longing for repose."

Experimental Results

To ensure the feasibility of running this pipeline on a full book dataset, our goal was a throughput of one chunk processed in 60 seconds. This benchmark assumes an average book size of 150,000 words for a typical novel, corresponding to 400 chunks with 500 tokens each. Performance analysis revealed that the highest quality pipeline 'Best' requires 17 minutes per chunk, with the QuestEval metric and Blazor interface consuming a significant portion of that time (5 minutes and 90 seconds, respectively).

However, by excluding the computationally heavy QuestEval metric and selecting the lowest-quality generation settings, the core processing time for the 'Fast' configuration drops from 8 minutes to approximately 60 seconds per chunk. This results in a projected runtime of 7 hours per book, or roughly four weeks of continuous runtime to linearly process a complete dataset of 100 books.

Datasets

Book texts are downloaded before the first run from sources like Project Gutenberg, HuggingFace, GitHub repositories, and wiki articles. Reference summaries and question-answer pairs from BookSum, LiteraryQA, and LitBank establish standardized evaluation benchmarks for future comparative analysis.

Input Chunk

'The size of an eagle,' Cyril went on, 'its head finely crested with a beautiful plumage, its neck covered with feathers of a gold colour, and the rest of its body purple; only the tail white, and the eyes sparkling like stars. They say that it lives about five hundred years in the wilderness, and when advanced in age it builds itself a pile of sweet wood and aromatic gums, fires it with the wafting of its wings, and thus burns itself; and that from its ashes arises a worm, which in time grows up to be a Phoenix. Hence the Phoenicians gave—'
'Never mind what they gave,' said the Phoenix, ruffling its golden feathers. 'They never gave much, anyway; they always were people who gave nothing for nothing. That book ought to be destroyed. It's most inaccurate. The rest of my body was never purple, and as for my—tail—well, I simply ask you, IS it white?'
It turned round and gravely presented its golden tail to the children.
'No, it's not,' said everybody.
'No, and it never was,' said the Phoenix. 'And that about the worm is just a vulgar insult. The Phoenix has an egg, like all respectable birds. It makes a pile—that part's all right—and it lays its egg, and it burns itself; and it goes to sleep and wakes up in its egg, and comes out and goes on living again, and so on for ever and ever. I can't tell you how weary I got of it—such a restless existence; no repose.'
'But how did your egg get HERE?' asked Anthea.

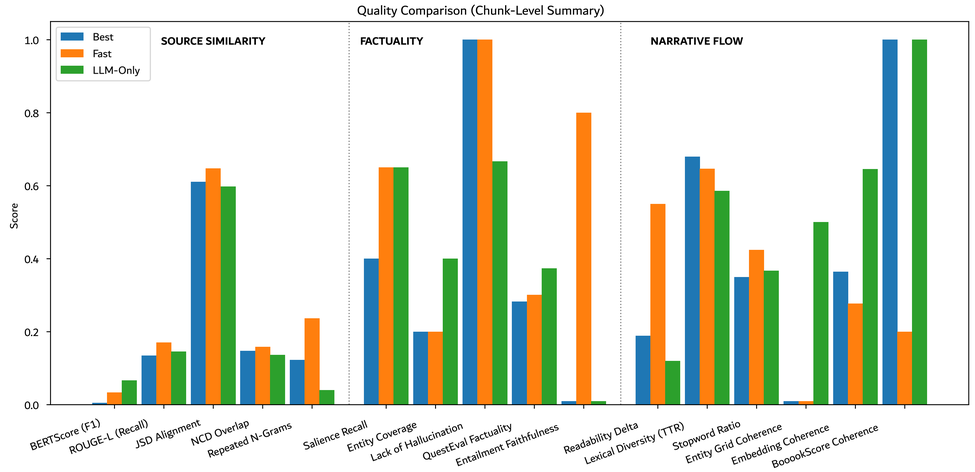

Figure 3. Comparison of Summary Quality Scores

Discussion

Evaluation reveals three distinct performance profiles. 'Fast' achieves the highest source similarity through extractive copying, which sacrifices narrative flow. The baseline 'LLM-Only' strategy produces very coherent summaries but introduces entity hallucinations. Our 'Best' configuration achieves strong coherence using GraphRAG community retrieval to reduce hallucinations, demonstrating that knowledge graph context can successfully ground LLM outputs in source material.

Strict factual grounding comes at significant computational expense. Our pipeline can match the baseline in terms of metrics, but simply prompting the LLM remains superior for rapid processing. The current architecture can be improved by fully integrating the proposed stream pipeline. This parallelization strategy could reduce the infeasible dataset runtime from weeks to days, enabling corpus-level summarization.