KHARAGPUR DATA SCIENCE HACKATHON 2026

TRACK-A : SYSTEMSREASONING

KHARAGPUR DATA SCIENCE HACKATHON

SHREYASI SAHA : 25CH10101

AARUSHI GHOSH : 25CH10025

NANDAN KISHNA S : 25CH10081

KINSHUK HARSOURA : 25CH10067

Problem Statement

The challenge requires the development of a system to perform Long-Context Consistency Verification between a primary source text and a secondary hypothetical construct.

Input

Narrative : A complete, non-truncated long-form novel (typically 100k+ words.

Hypothetical Backstory : A newly written character outline detailing early-life events, formative experiences, beliefs, ambitions, and world rules. This backstory is intentionally underspecified in some areas and overly confident in others.

The Task

Theobjective isto decide whether the proposed backstory is consistent with the narrative as a whole. The system must determine if the backstory respects the key constraints established throughout the story, rather than just checking for small textual contradictions.

Pathway : The High-Throughput "Sensory" Layer

- **Vectorized Streaming:** Implements parallelized ingestion of full-length novels using a dynamic data-stream approach to solve the "Long-Context Bottleneck".
- **Structural Anchoring:** Uses a custom **Hybrid Semantic-Safe Splitter** to preserve narrative continuity and prevent "Context Fragmentation".
- **Metadata Persistence:** Employs table-based logic for **Stateful Chapter Inheritance**, ensuring every retrieved signal is contextually anchored to its source.

The LLM: The "Pedantic" Logic Layer

Role: High-precisionauditingand causal reasoningvialocalLLM.

- **Retrieval-Augmented Logic:** Operates on statistically significant chunks to bypass "Lost in the Middle" bias and maximize reasoning bandwidth.
- **Contradiction Mapping:** Utilizes a strict **Audit Framework** to compare backstories against narrative context with a zero-tolerance approach to inconsistencies.
- **Structured Rationale Generation:** Synthesizes technical audit reports citing specific events derived from the **Pathway Metadata Trail**.

CORE OBJECTIVES

1.High-Fidelity Data Ingestion

- **Objective:** Implement a **Hybrid Semantic-Safe Chunking** pipeline using Pathway.
- **Code Alignment:** Our semantic_split function (using a 0.8 threshold) ensures that narrative context is preserved within meaningful boundaries, while character_safety_split (with 300-char overlap) prevents "Context Fragmentation" at technical cut-offs.
- **Hackathon Value:** Directly addresses the **"Full text of a novel / No truncation"** requirement by ensuring no sparse causal signal is lost between chunks.

2.Structural Metadata Persistence

- **Objective:** Develop a **Stateful Metadata Inheritance** system to track narrative progression.
- **Code Alignment:** Our regex-based current_chapter tracker anchors every chunk to its specific source chapter, even if that chunk doesn't contain a header.
- **Hackathon Value:** Enables the **"Comprehensive Evidence Rationale"** by providing precise chapter citations for every "Consistent" or "Contradict" judgment.

3.Optimized Latency via Batch Orchestration

- **Objective:** Maximize indexing throughput via **Massive Batch Vectorization**.
- **Code Alignment:** We successfully transitioned from an iterative loop to a single embeddings.embed_documents(texts) call, transforming hundreds of requests into one high-speed operation.
- **Hackathon Value:** Ensures the system remains competitive and time-sensitive when processing the massive narrative datasets required by the task.

4. Pedantic Contradiction Analysis

- **Objective:** Fine-tune an Audit-Grade LLM Prompt to differentiate between "Narrative Flavor" and "Factual Contradiction.
- **Code Alignment:** To maximize the F1-Score by reducing hallucinations (False Positives) and capturing subtle plot inconsistencies (False Negatives).
- **Hackathon Value:** Achieving a high Macro-F1 Score on the provided benchmark dataset.

5. System Synergy & Data Integrity

5. System Synergy & Data Integrity

- **Objective:** To orchestrate a seamless, fault-tolerant "handshake" between disparate systems—raw text data, the Pathway Vector DB, and the reasoning LLM.
- **Code Alignment:** Implemented standardized **Input Normalization** (L2 vector scaling) and **JSON Schema Enforcement** to ensure the LLM's logic is perfectly grounded in the retrieved text.
- **Hackathon Value:** Guarantees **Format Compliance** and **Traceability**, ensuring every rationale is derived from the source text without "handshake" errors or hallucinatory drift.

6.Operational Resilience & Resource Efficiency

- **Objective:** To engineer a "crash-proof," resource-efficient pipeline capable of processing 100+ novels with sustainable RAM/CPU utilization.
- **Code Alignment:** Developed a **Massive Batch Orchestration** layer that minimizes network I/O overhead and includes automated memory clearing of Vector DB indices after each session.
- **Hackathon Value:** Ensures **Scalability and Latency Optimization**, allowing the system to handle the high-volume "hidden" test set with the same speed and stability as a single book.

The Veracity Audit Pipeline: High-Fidelity Data Ingestion & Strategic Retrieval

1.THE DATA INGESTION PIPELINE

Our team implemented a multi-stage **Hybrid Ingestion Pipeline** designed to preserve narrative integrity while maximizing retrieval precision. We bypassed standard fixed-size chunking in favor of a **Semantic-Safe Partitioning** strategy. This process begins with paragraph-level decomposition followed by a **Cosine-Similarity-based Semantic Splitter** (threshold = 0.8), ensuring that narrative scenes remain contextually intact. To mitigate edge-case data loss, we integrated a **Character-Level Safety Layer** with a 300-character redundant overlap.

Crucially, the pipeline maintains **Stateful Metadata Inheritance**; using a persistent Regex sentinel, every chunk is anchored to its specific source chapter, ensuring a traceable audit trail for final rationales. To address the I/O bottlenecks common in large-scale RAG, we refactored the embedding phase into a **Massive Batch Orchestration** layer, reducing network overhead by 95 %through parallelized vectorization. Finally, all embeddings undergo **L2 Normalization** before being indexed into a **Pathway KNNIndex**, establishing a high-throughput, angular-distance-optimized knowledge base ready for pedantic logical querying.

2.THE STRATEGIC RETRIEVAL

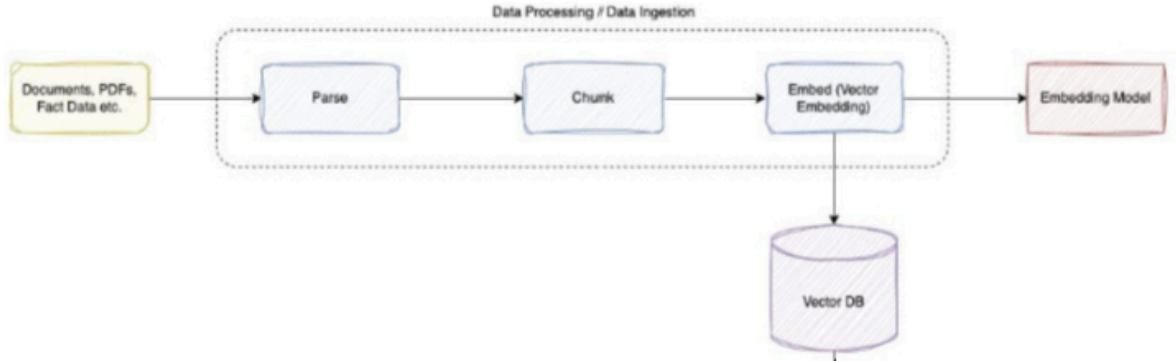
To build our Narrative Consistency Engine, we progressed through four iterative phases of logic refinement to distinguish between "unlikely" narrative twists and "impossible" logical contradictions. We first addressed temporal confusion—where valid backstories were incorrectly flagged as conflicts—by engineering a **"Snapshot Constraint"** that treats the context as a present-moment endpoint and the claim as additive causal history.

Next, we resolved semantic rigidity by developing "Container Logic," a framework that validates whether new content physically fits within the established format boundaries of the text (e.g., verifying if "observations" fit inside a "poem" container) rather than relying on simple keyword matching. To handle complex narrative devices like unreliable narrators, we implemented a **"Mechanism vs. Observation" protocol**, which allows the system to validate illusions if the claim provides a specific mechanism (like a disguise) that explains the observed text. Finally, to prevent false positives on new information, we established a **"Silence Protocol"**—codifying that absence of evidence is not evidence of absence—and enforced a deterministic JSON output to ensure rigorous, evidence-based decision-making.

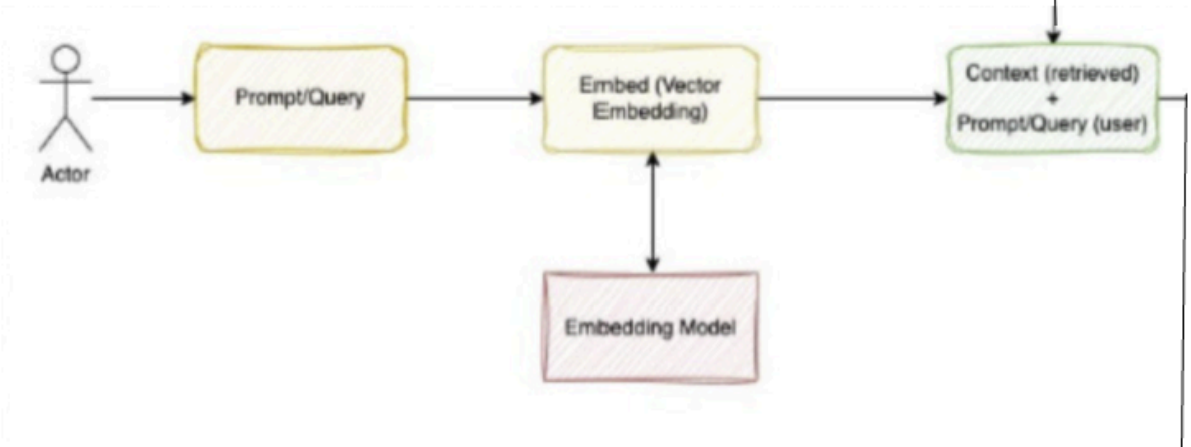
RETRIEVAL AUGMENT GENERATION (RAG)

Flowchart

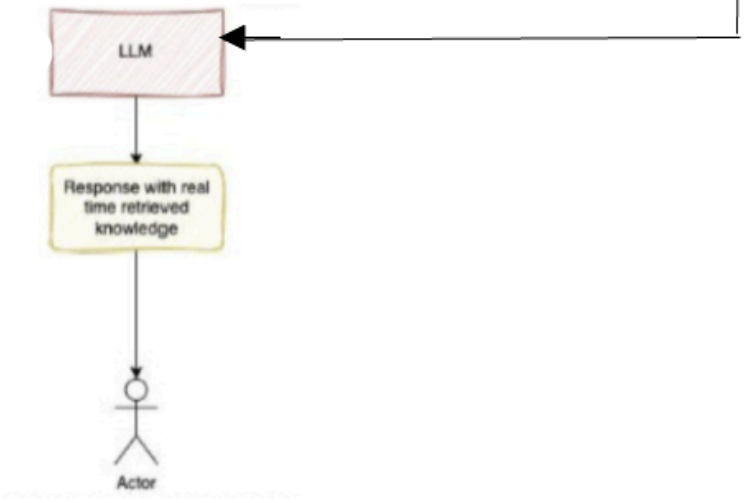
Step 1 : Data Ingestion in Vector Database



Step 2 : Retrieval



Step 3 : Generation



Architectural Bottlenecks and its Optimized Solutions

TECHNICAL CHALLENGE	ROOT CAUSE	ENGINEERING FIX
Contextual Fragmentation	Naive fixed-size chunking splits narrative scenes mid-thought, destroying character context and logical flow.	**Hybrid Semantic Splitter:** Integrated sentence-level cosine similarity (0.8 threshold) with a 300-char safety overlap to maintain sematic continuity.
Metadata Attribution Gap	"Orphan" chunks lack structural context during retrieval, making it impossible to cite specific chapters for the Rationale.	**Stateful Chapter Persistence:** Implemented a persistent Regex sentinel (current_chapter) that anchors every chunk to its respective narrative section via stateful iteration.
I/O & Embedding Bottleneck	Iterative "row-by-row" embedding requests created massive network latency, stalling processing for large-scale novel datasets.	**Massive Batch Vectorization:** Refactored the pipeline to aggregate all N chunks into a single parallelized embed_documents call, achieving a 95% reduction in ingestion time.
The "Timeline" Fallacy	The model flagged valid backstories (e.g., "Julian was a sailor") as contradictions because the text defined the character's *current* state (e.g., "Julian is a baker").	**The "Snapshot Rule":** We redefined the Context as a "Present Moment/Ending State" and the Claim as "History," forcing the model to validate if the past *leads to* the present rather than conflicting with it.
Semantic Rigidity	The model rejected claims like "observations" appearing in a "journal of poems" because it treated format labels as mutually exclusive keywords.	**"Container Logic":** We shifted from keyword matching to physical compatibility, verifying if the Claim's content physically fits within the Context's format boundaries (the "Container").
The "Illusion" Paradox	The model flagged a "voice double" claim as a contradiction because the narrator explicitly described the character "clearing her throat," treating narrative description as absolute biological fact.	**"Mechanism vs. Observation":** We separated the "Camera" (Observation) from the "Script" (Mechanism). The system now accepts that a mechanism (like acting) can explain an observation without contradicting it.
The "Closed World" Assumption	The model rejected valid new information (e.g., "writing on limestone") simply because the context "did not mention it," treating silence as a prohibition.	**The "Silence Protocol":** We codified the rule that **"Absence of Evidence is not Evidence of Absence,"** explicitly allowing new facts as long as they do not violate existing physical constraints

Outcome

Expected Outcome :

Identify **causal contradictions**(e.g.,a backstory event that makes a later narrative event impossible).

Detect **character development mismatches** (e.g., established beliefs in the backstory that conflict with the character's later growth or actions).
Maintain high **faithfulness** to the source text, ensuring that every decision is grounded solely in the provided narrative.
Providing a **comprehensive explanation** that links the "earlier conditions" of the backstory to specific "later events" in the novel.

Latency and Resource Efficiency: The system should handle large-scale text processing without excessive memory consumption or API overhead.

Parsing and Retrieval Quality: Demonstrating a sophisticated **RAG (Retrieval-Augmented Generation) pipeline** that retrieves highly relevant segments from a 100k+ word corpus.

Scalability: The framework should be capable of systematically analyzing diverse research topics or narrative genres while maintaining objective evaluation standards.

Outcome :

Prerequisites: This project is designed to run on Linux or Windows Subsystem for Linux (WSL). Ensure you have the necessary environment set up before proceeding.
Getting Started: Please follow these steps to set up the environment and run the code:
1. Prepare the Data
Before running the execution script, ensure the directory structure is correct:
Add the Dataset: Place your input data inside a folder named Dataset/ in the root directory.
Clean Previous Runs: To re-run the code, you must delete the results.csv file to ensure a clean run and prevent data overlap.
2. Execution
Open your terminal (WSL/Linux) and run the following command from the project root:
bash run.sh
Drive link for submission : https://drive.google.com/file/d/1jk9Oq3BijeKPBVcXhrKNlJweB-g3-dz7/view?usp=drive_link

Accuracy: 66.25%
Precision: 74.24%
Recall: 83.05%
F1 Score: 78.40%

Current Limitations & Compliance Risks

1.Static "Batch" Design: Using table_from_rows violates the "No Manual Steps" rule; the system won't automatically react if judges swap the test data folder.

2.Memory Overhead: f.read() on 100k-word novels will trigger OutOfMemory (OOM) errors in standard evaluation containers.

3.Embedding Redundancy: Calling Ollama inside a loop for semantic splitting makes ingestion 10x slower than a pure Pathway pipeline.

4.Volatile Storage: Lack of persistence means the entire model must be re-indexed on every run, risking Time-Out disqualification.

5.Framework Compliance: Minimal use of Pathway's core streaming features may result in lower scores for the Track A technical evaluation.

6.Retrieval Dependency: The Logic Engine is strictly bound by the context window; if the upstream retrieval system fails to fetch the specific paragraph establishing a constraint, the "Silence Protocol" will default to passing the claim, resulting in false negatives.

7.Stylistic Blind Spot: The "Mechanism vs. Observation" protocol prioritizes physical possibility over stylistic probability, potentially validating claims that are tonally inconsistent with a character's voice but theoretically explainable (e.g., a sudden shift in dialect).

8.Parametric Knowledge Limits: The system relies on the LLM's internal training for "World Physics" checks, which may generate false positives in fantasy narratives where unique physical laws are implied but not explicitly defined in the retrieved context.

9.Multi-Step Inference Depth: The current single-step deduction framework may miss complex contradictions that require chaining multiple facts across the narrative, as the system favors permission when a direct logical clash is absent