Kharagpur Data Science Hackathon

Appendix

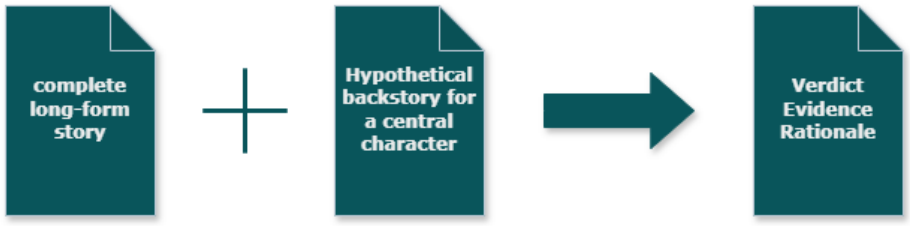
Sno.	Theory
1	The Challenge:
2	Task Definition
3	Tech Stack and Implementation
4	System Architecture
5	Model Architecture
6	Operational Workflow (Step-by-Step)
7	Limitations and Future Work
8	Conclusion

Kharagpur Data Science Hackathon (KDSH - 2026)

Team Name: debugger

1. The Challenge:

We are given a complete long-form narrative, such as a 100k+ word novel, paired with a newly crafted, plausible hypothetical backstory for one of its central characters that does not appear in the original text. The core task is to rigorously evaluate whether this proposed backstory aligns with the novel's overarching constraints, prioritizing consistency over time, causal reasoning, and respect for established narrative boundaries rather than minor textual discrepancies or writing quality. Systems must exhibit advanced causal analysis by detecting subtle inconsistencies that compromise the integrity of the story, such as implausible coincidences, and ensuring that subsequent character developments and events remain logically feasible under the backstory's conditions. AI's ability to comprehend long contexts and narratives is demonstrated by decisions that require evidence-based reasoning and draw signals from various narrative sections to validate holistic fit.



2. Task Definition:

2.1 Every Input example contains:

1) Narrative:- Full text of a novel without any summaries, truncation, etc. is provided at this point in a complete long-form narrative as reference for checking the consistency of provided backstory.

2) Hypothetical Backstory:- It contains character outline describing several reasoning like early-life events, formative experiences, beliefs, fears, ambitions, assumptions about the world and its rules, etc.

2.2 Every Output contains:

1) Consistency Judgment:-The backstory is intentionally underspecified in some places and overly confident in others, for each example our system must produce a binary label (1) for consistent and (0) for inconsistent.
2) Comprehensive Evidence Rationale:- Establishing the validity of backstory, Evidence Rationale is a critical component for substantiating or challenging the claimed backstory elements of the narrative. It must be meticulously constructed to ensure that all claims are rigorously tested against the primary textual source.

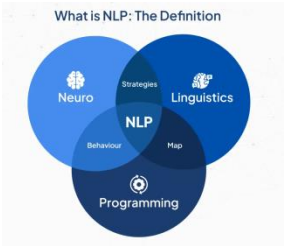
3. Tech Stack and Implementation

3.1 Systems Reasoning with NLP and Generative AI is the tech stack track which is designed to encourage strong, well-engineered solutions using today's established NLP and GenAI techniques. The focus here is correctness, robustness, and evidence-grounded reasoning, not architectural novelty.

Our solution leverages **Pathway** as a high-throughput data ingestion and vector indexing engine, coupled with an **Agentic Retrieval-Augmented Generation (RAG)** architecture. By decomposing complex backstory claims into atomic facts and verifying them against semantically retrieved narrative chunks, the system achieves robust, evidence-grounded decision-making without requiring the entire novel to fit in context.

Pathway is used for:-

- Ingesting and managing the provided long-context narrative data,
- Storing and indexing full novels and metadata,
- Enabling retrieval over long documents using a vector store,
- Connecting to external data sources (e.g. Google Drive, local folders, cloud storage),
- Serving as a document store or orchestration layer for the reasoning pipeline.



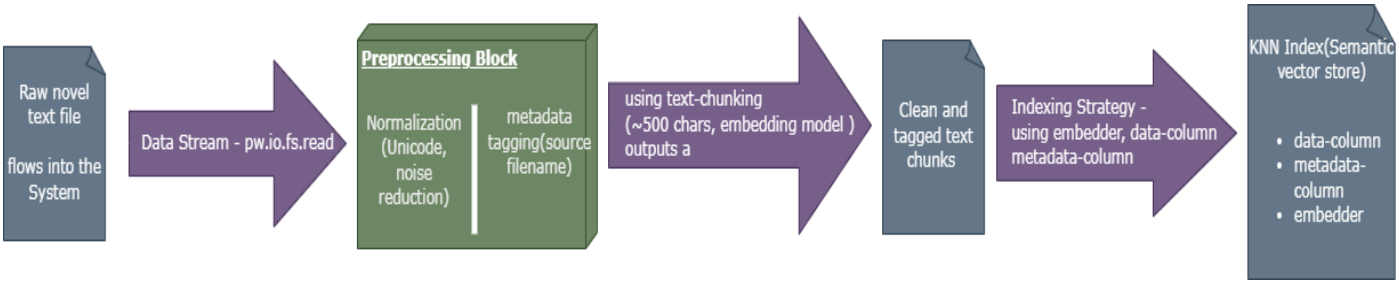
4. System Architecture

Our solution adopts a **multi-stage pipelined architecture** facilitated by the **Pathway** framework. The system is divided into two primary layers: the *Ingestion Layer* and the *Reasoning Layer*.

4.1) The Ingestion Layer (Powered by Pathway):

This layer handles the "Big Data" aspect of the novels. It is responsible for transforming raw text into a queryable semantic index.

- **Data Stream:** Pathway's `pw.io.fs.read` connects to the raw text repository.
- **Preprocessing:**
    - **Normalization:** Unicode decoding and noise reduction.
    - **Metadata Tagging:** Each chunk is tagged with its source filename to prevent cross-contamination between different novels during retrieval.
- **Indexing Strategy:** We utilize a **KNN (k-Nearest Neighbors) Index**. The text is split into chunks (approx. 500 characters) with a 50-character overlap to maintain narrative continuity at boundaries.



4.2. The Reasoning Layer (The Solver Agent)

This layer processes the backstory and produces a consistency verdict.

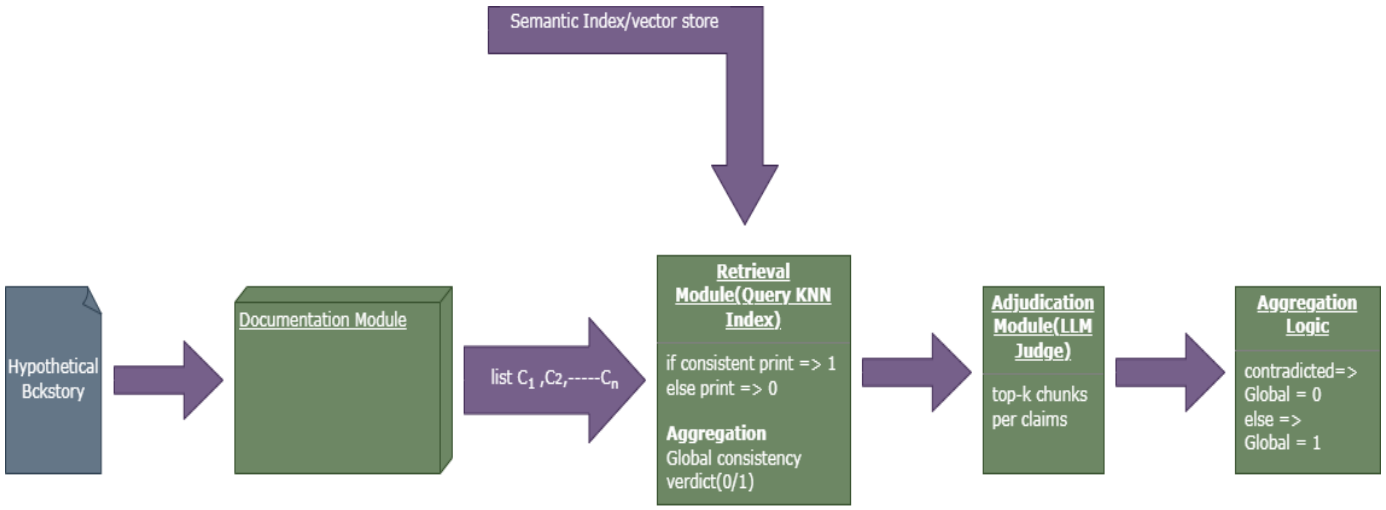
1. **Decomposition Module:**
    - An LLM takes the backstory and outputs a list of atomic claims: $C_1, C_2, \ldots, C_n$.
    - Label : ”Decomposition Module(LLM --> Atomic Claims)”

2. **Retrieval Module:**

- For each claim Ci, the system queries the Pathway KNN index.
- Output: Top-K most relevant narrative chunks from the novel: E_1, E_2, …..E_n .
- Label: ”Retrieval Module(Query Pathway Vector Store => Top-K Chunks)”.

3. **Adjudication Module (Judge):**
    - An LLM acts as a “Judge” that performs Natural Language Inference (NLI) on each pair (C_i, $\{E_k\}$).
    - For each claim, it outputs a binary verdict: “Consistent” or “Contradiction”.
    - Label:”Adjudication Module (LLM Judge: NLI on (C_i,$\{E_k\}$) => Binary Verdict)”.

4. **Aggregation Logic:**
    - If *any* claim C_i is judged as a contradiction, the global prediction is set to **0 (Inconsistent)**.
    - If all claims are consistent, the global prediction is **1 (Consistent)**.
    - Label: ”Aggregation: If any C_i contradiction => Global = 0; else => Global = 1”.



5. Model Architecture

The system employs a hybrid model design that carefully balances computational efficiency with deep reasoning capability. It separates the concerns of retrieval (finding relevant parts of the novel) and judgment (deciding whether those parts contradict the backstory), using two specialized models: a fast embedding model for retrieval and a powerful generative LLM for fine-grained reasoning.

5.1 Embedding Model(The Retriver)

The retriever is built around the `all-MiniLM-L6-v2` sentence-transformer model, a distilled BERT-style encoder that has been fine-tuned on a contrastive learning objective to produce high-quality sentence embeddings. This model maps chunk of narrative text into a dense 384-dimensional vector, where semantically similar passages are close together in the embedding space.

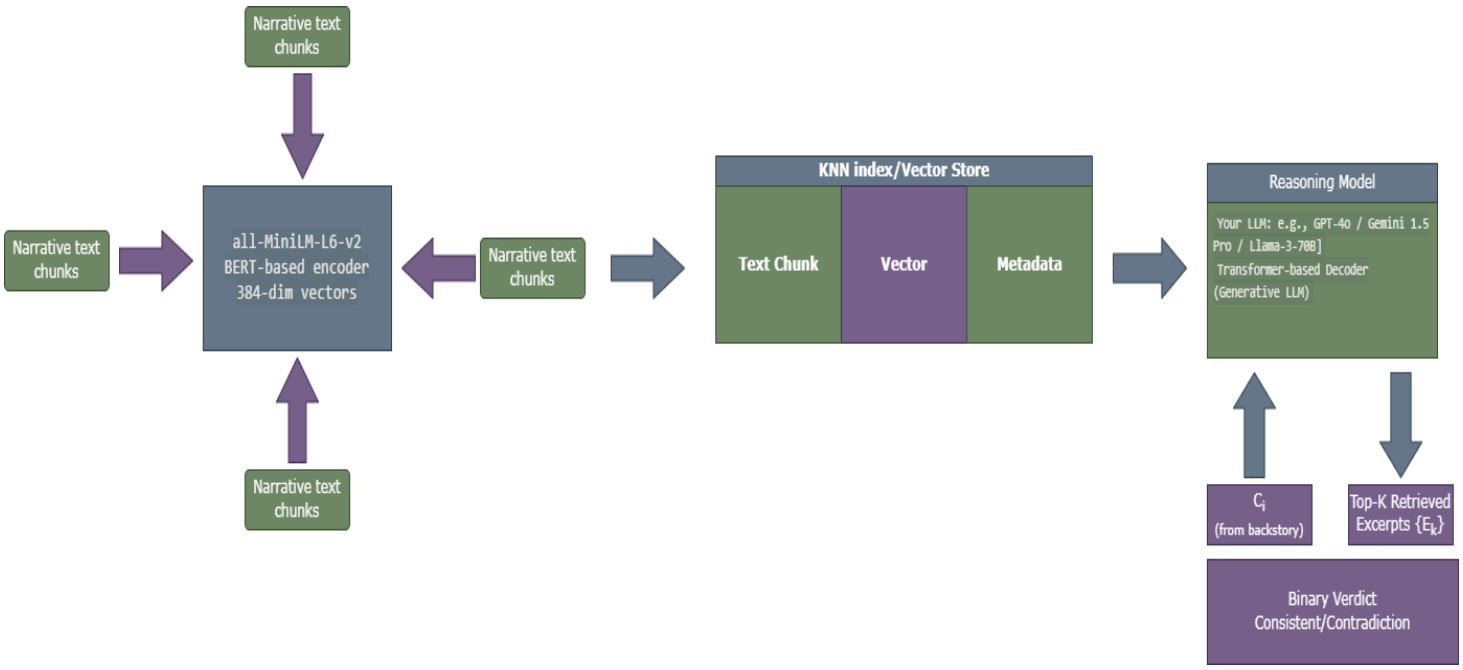
Its primary role is to rapidly identify the most relevant excerpts from the novel when given a claim from the backstory. For example, if a claim mentions "the protagonist never met their father," the retriever finds all passages that discuss the protagonist's father, family background, or early life. The model is chosen specifically for its excellent trade-off between speed and quality: it runs efficiently on modest hardware (even CPU), enabling fast indexing and querying over entire novels (100k+ words) without requiring large GPU clusters. This makes the ingestion pipeline scalable and practical for processing many novels in parallel, while still capturing enough semantic similarly to support meaningful retrieval.

5.2 Reasoning Model(The Judge)

The reasoning component uses a large transformer-based decoder model (for example, GPT-4o, Gemini 1.5 Pro, or a capable local model like Llama-3-70B) to perform the actual consistency judgment. This generative LLM is responsible for the nuanced, multi-step reasoning needed to decide whether a retrieved narrative excerpt contradicts a given claim, rather than merely discussing a similar topic.

While the embedding model can surface relevant text, it cannot reliably handle subtle logical relationships such as causality, negation, or temporal constraints. For instance, it might retrieve a passage where a character says "I never trusted him," but only the reasoning model can determine whether that conflicts with a backstory claim like "they were lifelong allies." The judge model is prompted to perform natural language inference (NLI) on each pair of a claim Ci and the top-K retrieved excerpts $\{\mathrm{Ek}\}$, producing a binary verdict (consistent / contradiction) for that claim. This separation

allows the system to scale retrieval cheaply while reserving expensive LLM calls only for the critical reasoning step, achieving both efficiency and high-quality judgment.



6. Operational Workflow (Step-by-Step)

The following describes the lifecycle of a single query through our system:

1. **Initialization:** The Pathway server launches, ingesting all .txt files in the ./data directory. It builds an in-memory vector index, ready for HTTP queries.
2. **Query Receipt:** The Solver script reads a row from test.csv, extracting the Story ID, Backstory, and Filename.
3. **Atomic Splitting:** The Backstory is parsed.
    - *Input:* "He was a wealthy orphan who loved the sea."
    - *Output:* ["He was wealthy", "He was an orphan", "He loved the sea"].
4. **Evidence Mining:** The system iterates through the list.

    - Query: "He was wealthy" + Filter: filename='Book_A.txt'.
    - Pathway returns: *"John scraped together his last pennies to buy a loaf of bread."*
5. **Conflict Detection:** The Judge compares the claim ("Wealthy") with the evidence ("Last pennies").
    - *Verdict:* **Contradiction**.
6. **Final Reporting:** The system halts further checks for this story, flags it as **Inconsistent (0)**, and logs the specific excerpt about "buying bread" as the **Evidence Rationale**.

7. Limitations and Future Work

- **Implicit Negatives:** If a contradiction relies on the *absence* of an event (e.g., "He never visited Paris," but the book simply never mentions Paris), the retrieval system may struggle to find "evidence of absence."

- **Temporal Reasoning:** The current chunking strategy treats the novel as a "bag of events." Future improvements could involve indexing chunks with explicit timestamp or chapter metadata to better handle chronological constraints

8. Conclusion

This system tackles the Global Consistency challenge by combining Pathway's efficient vector indexing with a structured Agentic RAG pipeline for deep reasoning, enabling robust verification of hypothetical backstories against a full novel. Instead of merely generating plausible text, it performs rigorous, evidence-based judgment: retrieving relevant narrative evidence for each atomic claim and using a powerful LLM as a "judge" to detect contradictions in a principled way.

Dataset Used

Frameworks Used

.Pathway

LLM-App

By:- Rishi Jha