KHARAGPUR DATA SCIENCE HACKATHON (KDSH) 2026

Title : Backstory Consistency Verification

Team Name: HackHers

Team Members:

Mrunali Kamerikar

Riddhima Taose

IIT Kharagpur

Abstract

This project addresses the task of verifying whether character backstory claims are consistent or contradictory with long-form literary narratives. Instead of relying on black-box LLMs or off-the-shelf Retrieval-Augmented Generation (RAG), we design a transparent, evidence-driven NLP pipeline that explicitly models claim extraction, long-context retrieval, and reasoning-based consistency scoring. The approach emphasizes interpretability, robustness, and thoughtful handling of long documents, in alignment with the evaluation criteria of the IIT Kharagpur Data Science Hackathon.

1. Problem Understanding

Given:

- A claim about a character's backstory (from caption or content)
- The full text of a novel (long narrative)

Goal:

- Predict whether the claim is Consistent (1) or Contradictory (0) with respect to the novel.

Key challenges:

- Handling very long documents (hundreds of thousands of characters)
- Avoiding shallow keyword matching or naive RAG pipelines
- Producing a reasoning-aware and robust consistency decision

2. Overall Pipeline

Flow:

Novel Text

↓

Novel Chunking

(Split full book into fixed-size text chunks)

↓

Lightweight Vector Store

(Mock Pathway: chunk_id + chunk_text)

↓

Backstory Claim Extraction

(From caption or content field)

↓

Evidence Retrieval

(Keyword overlap scoring to find top-K relevant chunks)

↓

Evidence-Aware Reasoning

(Analyze overlap between claim words and retrieved evidence)

↓

Consistency Score

(Quantifies claim–evidence alignment on a 0–1 scale)

↓

Final Label

(1 = Consistent, 0 = Inconsistent)

Each step is explicitly implemented and interpretable.

Project Directory Structure

```
hackathon-project/
|
├── project/
|   └── main.py
|
├── dataset/
|   ├── train.csv
|   ├── test.csv
|   └── books/
|       ├── In search of the castaways.txt
|       └── The Count of Monte Cristo.txt
|
├── output/
|   └── submission_v1.csv
|
├──KHARAGPUR DATA SCIENCE HACKATHON Report
|
├── requirements.txt
|
└── README.md
```

3. Backstory Claim Extraction

Claims are extracted from the dataset as follows:

- If the caption field exists, it is treated as the claim
- Otherwise, the content field is used

This ensures robustness to missing or incomplete annotations while preserving semantic intent.

4. Long-Context Handling via Chunking

Novels are split into fixed-size chunks (800 words each). This design:

- Preserves local narrative coherence
- Allows scalable processing of very long texts
- Avoids truncation or loss of important evidence

Chunking is deterministic and reproducible, ensuring consistent retrieval behavior.

5. Mock Pathway Vector Store

Instead of using an opaque external vector database, we simulate Pathway functionality with a structured in-memory table:

- Each chunk is stored with a unique identifier
- This abstraction mirrors real vector stores while remaining lightweight and debuggable

This choice prioritizes clarity and control, aligning with the hackathon's focus on reasoning over tooling.

6. Evidence Retrieval Strategy

Relevant chunks are retrieved using a stopword-filtered lexical overlap score:

- Stopwords are removed to focus on semantic content
- Claims and chunks are compared using word-level intersections
- Top-K (K=3) chunks with highest overlap are selected

This retrieval strategy:

- Avoids brittle exact matching
- Acts as a transparent proxy for semantic similarity
- Enables reproducible and explainable evidence selection

7. Evidence-Aware Reasoning (LLM-Inspired)

Instead of generating text, we simulate LLM-style reasoning using structured signals:

Signals used:

- Claim coverage ratio (how much of the claim is supported)
- Strength of evidence (number of distinct supporting words)

The final consistency score is computed as:

Score = $0.6 \times$ claim coverage + $0.4 \times$ evidence strength

The score is bounded between 0 and 1, ensuring numerical stability and interpretability.

8. Threshold-Based Classification

A calibrated threshold converts the consistency score into a label:

- Score $\geq 0.25 \rightarrow$ Consistent (1)
- Score $< 0.25 \rightarrow$ Contradictory (0)

This threshold allows the system to:

- Detect weak or unsupported claims
- Avoid predicting all samples as consistent
- Exhibit realistic and human-like judgment behavior

9. Robustness and Generalization

The entire pipeline is wrapped into a reusable prediction function and applied uniformly across the test dataset.

Observed behavior:

- Both consistent and contradictory labels are produced
- Predictions vary meaningfully across samples
- The system avoids degenerate outputs (all-0 or all-1)

10. Novelty and Evaluation Alignment

Why this approach stands out:

- Not a black-box LLM or end-to-end RAG
- Custom-designed reasoning and scoring logic
- Explicit long-context management
- Transparent, explainable decisions

This aligns strongly with the evaluation focus on:

- Thoughtful NLP design
- Long-context reasoning
- Novel, interpretable approaches

11. Limitations and Future Work

- Replace lexical overlap with embeddings for semantic similarity
- Introduce contradiction-specific signals (negation, temporal mismatch)
- Incorporate selective LLM validation only when evidence is ambiguous

12. Conclusion

This project demonstrates that effective long-context consistency verification can be achieved without relying on heavy black-box models. By combining structured retrieval, evidence-aware reasoning, and transparent scoring, the system delivers robust and interpretable predictions well-suited for literary narrative analysis and the goals of the IIT Kharagpur Data Science Hackathon.