RAG System for Low- Resource NLP

Author: Iteoluwa Ibitoye

Course: AI Systems Management

1. Introduction & Corpus Design

1.1 Domain Scope

This Research Portal focuses on Low-Resource Natural Language Processing (NLP), specifically targeting African languages. The domain presents unique challenges such as data scarcity, code-switching, and the need for culturally grounded evaluation metrics.

1.2 Data Manifest

I created a dataset for this RAG that's made up of 34 high quality academic papers, that collectively cover three major parts of Multicultural Natural Language Processing:

- **Benchmarks:** This section provides the essential scaffolding for evaluating LLMs on African languages. Papers such as **AfroBench**, **IrokoBench**, and **NaijaSenti** move beyond standard English-centric metrics, introducing novel datasets for 60+ African languages. These benchmarks cover a diverse range of tasks from sentiment analysis and machine translation to complex reasoning, and establish the first standardized baselines to quantify the performance gaps between high-resource and low-resource languages.
- **Models & Architectures:** This category focuses on architectural innovations designed to overcome data scarcity. It includes foundational work on **XLM-R** (cross-lingual transfer learning) and **No Language Left Behind (NLLB)**, which utilizes Sparsely Gated Mixture-of-Experts (MoE) to scale translation to 200+ languages. Additionally, papers like **Cheetah** explore specialized natural language generation (NLG) for 517 African languages, demonstrating how massive multilingualism can compensate for the lack of monolingual data.
- **Cultural & Ethical Analysis:** This focuses on the socio-technical implications of AI deployment. It includes **Masakhane**, which advocates for participatory research and data sovereignty to prevent digital colonialism. Theoretical works like **The Bitter Lesson** are analyzed in the context of resource allocation, while **Preserving Cultural Identity** ensures that the RAG system considers not just model *accuracy*, but also model *fairness* and cultural alignment.

A metadata manifest (data_manifest.csv) was maintained to ensure reproducibility. Each entry includes the filename and a formal citation (e.g., (Adebara et al., 2022)), which the system uses to generate structured, academic-style references in its output.

2. Methodology: The RAG Architecture

The system implements a robust Retrieval-Augmented Generation pipeline with specific enhancements designed for academic synthesis and auditability.

- **Ingestion:** Documents were processed using a standard sliding window chunking strategy and embedded using **OpenAI Embeddings** to capture semantic meaning.
- **Vector Store: ChromaDB** was used for local persistence, enabling efficient similarity search across the vectorized corpus.
- **Retrieval (Baseline):** The system utilizes Cosine Similarity as the base metric, enhanced by **Maximal Marginal Relevance (MMR)** to diversify results and reduce redundancy in retrieved chunks.
- **Generation:** The generation layer uses **GPT-4o** with a strict prompt that enforces a Grounding Rule that requires every claim to be supported by retrieved context.
- **Logging:** A dual-logging mechanism was implemented. The system captures a clean summary for reporting (Question, Answer, Citations) and a detailed **machine-readable log** (retrieval_logs.json) containing the full text of retrieved chunks, model versions, and latency metrics for debugging and grading.
- **Trust Behavior:** The system is engineered for intellectual honesty. It is explicitly instructed to refuse to invent citations and to output "Insufficient Evidence" when the retrieved context does not contain the answer.

3. Enhancements

To improve the system's ability to handle complex Synthesis questions, I implemented and tested Maximal Marginal Relevance (MMR) Reranking.

1. Maximal Marginal Relevance (MMR) Reranking

The Standard retrieval (k=5) i tried at first often suffered from redundancy, fetching multiple chunks from the same section (e.g., five Introduction snippets) while missing crucial details needed for synthesis, so I implemented MMR (search_type="mmr") with a fetch pool of 20 (fetch_k=20) and a final selection of 12 (k=12) to force the retriever to select semantically diverse chunks.

2. Structured Citations (Metadata Usage)

I integrated a build_citation_map function that links vector store IDs to the data_manifest.csv. This converts raw filenames (e.g., source_05) into formal academic citations (e.g., (Adebara et al., 2022)) and ensures that every claim is backed by a verifiable source, which is essential for academic integrity. Currently, the system mostly cites the whole paper. Phase 3 would improve granularity by citing specific page numbers or section headers if available in the metadata.

4. Query Set Design

To test the RAG system's capabilities beyond simple keyword matching, I designed an evaluation set of 20 queries grouped into three difficulty levels. This ensures the system is tested on recall, reasoning, and safety.

1. **Direct Retrieval (10 Queries):**
    - *Goal:* Test the system's ability to find specific facts, numbers, or definitions within a single document.
    - *Example: "What specific failures does AfroBench identify?"*
    - *“According to Conneau (2020), how does XLM-R compare to mBERT?”*
    - *Success Criteria:* The answer must contain the exact metric or list from the paper and cite the correct source.
2. **Synthesis & Comparison (5 Queries):**
    - *Goal:* Test the system's ability to retrieve chunks from multiple document**s** and synthesize a cohesive answer. This is the "Research Assistant" capability.
    - *Example: "Compare the approaches of 'Masakhane' and 'NLLB' regarding community involvement."*
    - *"Do 'AfroBench' and 'IrokoBench' agree on the performance of GPT-4 for African languages?"*
    - *Success Criteria:* The answer must discuss *both* entities and cite *both* sources
3. **Edge Cases & Hallucination Tests (5 Queries):**
    - *Goal:* Test the system's "Safety Rails." These queries ask about topics *not* in the corpus or fake facts to see if the model lies.
    - *Example: "Does the 'WAXAL' paper discuss speech synthesis for Martian languages?"*
    - *"What does the corpus say about 'Quantum Computing in Yoruba'?"*
    - *Success Criteria:* The system must explicitly state "Insufficient Evidence**"** and refuse to answer.

5. Evaluation Metrics

The system was evaluated using a Human-in-the-Loop review of the generated logs (evaluation_results.json). I utilized three primary metrics:

1. **Retrieval Recall (Hit Rate):**
    - Did the retrieval system find the relevant chunk?
    - *Pass:* The correct chunk appears in the top 12 results.
    - *Fail:* The chunk is missing, leading to "Insufficient Evidence" for a valid question.
2. **Groundedness (Citation Accuracy):**
    - Does the answer rely *only* on the retrieved text?
    - *Pass:* Every claim is backed by a [source_id] that actually contains the information.
    - *Fail:* The model uses outside knowledge or hallucinates a citation.
3. **Refusal Accuracy (Safety):**
    - Did the system correctly refuse to answer out-of-scope questions?
    - *Pass:* Output is "Insufficient Evidence."
    - *Fail:* The system generates a plausible-sounding but fake answer.

6. Results & Analysis

Category	Queries	Success Rate	Notes
Direct Retrieval	10	**90% (9/10)**	Great recall on factual direct retrieval questions like ″
Synthesis	5	**80% (4/5)**	Strong performance on comparison questions.
Edge Cases	5	**100% (5/5)**	0% Hallucination rate.
Total	**20**	**90% (18/20)**	**Good performance for a Baseline RAG**

The Link to the complete breakdown of my queries is found here: Phase 2 Report_iibitoye and more complete query logs are in evaluation_results.json

Despite the high accuracy (90%), my evaluation identified some failures.

- **Failure Case 1: The Semantic Gap**
  - **Query:** *"What is the 'Bitter Lesson' described by Wu et al. (2025)?"*
  - **Result:** *"Insufficient Evidence"*
  - **Root Cause:** The system failed to link the specific named concept ("Bitter Lesson") to the paper's content, which likely discussed "compute scaling" or "hardware efficiency" without repeating the specific phrase enough times to trigger a vector match.
  - **Fix: Query Expansion** (Phase 3) is needed for better mapping.

- **Failure Case 2: Metadata Disconnect**
  - **Query:** *"Synthesize findings from Terblanche (2024)..."*
  - **Result:** *"Insufficient Evidence"*
  - **Root Cause:** The user queried by *citation key* (Author + Year), but the vector store indexes the *text content*. The system couldn't find the name "Terblanche" in the text body of the chunks it retrieved.
  - **Fix:** Implement **Metadata Filtering** in Phase 3 so users can explicitly filter by author="xxx" rather than relying on semantic search.

- **Failure Case 3: Surface Level Responses**

  Query: *"What **specific failures** does the 'AfroBench' paper identify in current LLMs?"*

- **Result:** The answer correctly identified broad themes: *"biased towards English," "large performance gaps,"* and *"data scarcity."* (Can be seen in my Q1 Log). but it failed to list the **specific downstream tasks** (e.g., "Named Entity Recognition," "Sentiment Analysis") or specific error types (e.g., "hallucination in translation") that the paper contains in its Results section.
- **Root Cause: Abstract Bias.** Academic papers summarize their findings in the Abstract (Chunk 1). The retrieval system ranked the Abstract chunk highest because it contained the words "failures" and "LLMs" densely. This crowded out the results chunks because they were likely ranked lower. The system gave a correct summary answer but failed on my request for specifics.

7. Conclusion and Recommendations for Phase 3

My phase 2 transitioned the system from a basic search script to a more grounded research assistant.To address the identified failures of a Semantic Gap and a Metadata Disconnect, the following would be implemented in Phase 3:

1. **Implement Query Expansion :**Users sometimes ask abstract questions that don't match exact paper keywords, as can be seen in Failure Case 1. I would implement an LLM to rewrite user queries into 3-5 specific search terms before retrieval. This was prototyped in this phase (query.py file) and showed promise for increasing recall.

2. **Hybrid Retrieval with RRF:**The system sometimes misses specific entities that a keyword search would catch. I tried implementing **BM25** (Keyword Search) with vector search but it wasn't as effective as my current MMR reranking. In the next Phase, I would re-introduce this method alongside **Reciprocal Rank Fusion (RRF)** to better balance the keyword and semantic signals.

3. **Improved citation recalls and Metadata Filtering :** I noticed from failure Case 2 above how necessary this enhancement is so I would upgrade the retrieval pipeline to allow structured filters (e.g., filter={'year': 2024}) alongside semantic search. I also observed that many of the responses failed to retrieve the proper citations but rather just stated the (source_id) instead.

Github Repo: https://github.com/IIbitoye/NLP_RAG