NLP Textbook RAG System

Colin Sidberry
sidberry.c@northeastern.edu

December 7, 2025

1. Project Objective

Built a Retrieval-Augmented Generation (RAG) system that enables semantic search and question-answering across NLP textbook content (Jurafsky & Martin's *Speech and Language Processing*), providing students with accurate, cited answers to conceptual questions.

2. Technical Architecture

Full System Diagram: github.com/ColinSidberry/nlp-textbook-rag/blob/main/System_Diagram.png

System Components:

- **Data Pipeline:** 35 chapters (2.2MB) → 10,170 text chunks → 384-dim embeddings
- **Vector Database:** ChromaDB with persistent storage
- **Embedding Model:** sentence-transformers/all-MiniLM-L6-v2
- **Language Model:** Mistral 7B-Instruct (via Ollama)
- **User Interface:** Streamlit web application
- **Deployment:** Hybrid architecture (Streamlit Cloud + Google Cloud VM)
- **Quality Assurance:** Automated testing (11 quality tests + 20 edge cases, 100% & 90% pass rates)

Query Pipeline:

1. User question → acronym expansion (RAG → retrieval-augmented generation)
2. Query embedding vector generation (sentence-transformers)
3. Cosine similarity search across 10,170 chunks (ChromaDB)
4. Similarity threshold filtering (blocks out-of-scope questions)
5. Top-5 relevant chunks retrieved (∼1-2s)
6. LLM generates comprehensive answer with strong anti-hallucination guardrails (∼17-23s)
7. Display answer, citations, and source chunks

3. Key Design Decisions & Trade-offs

Quality over Speed

Chose Mistral 7B-Instruct over smaller models (e.g., Llama 3.2:3b) to generate comprehensive 400-500 word educational responses. This results in 19-second average query times, but provides thorough explanations appropriate for learning contexts.

Zero-Loss Chunking Strategy

Implemented 280-character chunks with 20% overlap (56 characters) using raw text sliding window to achieve 10,170 chunks from 35 chapters. Key innovation: Complete text preservation (100% coverage verified) with no stripping or text manipulation, preventing information loss between chunks while maintaining semantic continuity.

Hybrid Deployment Architecture

Deployed UI on Streamlit Cloud (free tier) and LLM on GCP VM (n1-standard-4). Provides professional public interface while leveraging dedicated compute for model inference.

4. Performance Metrics & Validation

Quality Test Suite (11 tests, 100% pass rate):

- **Out-of-scope handling:** 4/4 tests passed (current events, general knowledge correctly declined)
- **Valid NLP questions:** 4/4 tests passed (transformers, embeddings, n-grams, RAG)
- **Context grounding:** No hallucination detected
- **Citation accuracy:** 100% match between citations and retrieved chunks

Edge Case Test Suite (20 tests, 90% pass rate):

- Multi-acronym queries (2/2), case sensitivity (2/2), typos (2/2)
- Comparison questions, special characters, negative questions (all passed)
- Very long queries (50+ words), multiple questions in one (passed)
- Empty/invalid queries handled gracefully

Text Coverage Verification:

- 100% text coverage across all 35 chapters (no missing text)
- Perfect 20% overlap between consecutive chunks
- 10,170 chunks indexed with complete metadata

5. Technical Challenges & Solutions

Challenge 1: Hallucination & Out-of-Scope Questions
Solution: Implemented multi-layer defense: (1) Similarity threshold filtering (avg $>$ 0.4 OR max $>$ 0.42), (2) Strengthened prompt with 7 critical anti-hallucination instructions, (3) 15 common NLP acronym expansions (RAG, BERT, LSTM, etc.), (4) Comparison query enhancement. Result: 100% pass rate on quality tests, 90% on edge cases.

Challenge 2: Text Loss Between Chunks
Solution: Eliminated all text stripping/cleaning operations. Implemented pure sliding window (280 chars, step 224) with completely raw text. Verification script confirms 100% text coverage across all 35 chapters with perfect 20% overlap.

Challenge 3: Deployment without committing large database
Solution: Implemented automatic index rebuilding from source data (2.2MB JSON) on first run. ChromaDB stays in .gitignore; rebuilt from normalized data when deployed.

Challenge 4: Connecting Streamlit Cloud to GCP-hosted Ollama
Solution: Configured Ollama to accept external connections (0.0.0.0:11434), created GCP firewall rule for port 11434, passed VM external IP via Streamlit secrets as OLLAMA_BASE_URL environment variable.

6. Demonstration Highlights

- **Meta-query:** "What is retrieval-augmented generation?" (0.835 similarity) — system explains its own architecture
- **Semantic understanding:** "How does backpropagation work?" vs. "How do neural networks propagate errors backwards?" (0.725 vs. 0.715 similarity) — demonstrates vocabulary-independent matching
- **Technical depth:** "How do you evaluate n-gram language models?" (0.838 similarity) — shows system handles methodology questions

7. Future Enhancements

- Implement conversation history for follow-up questions
- Add query caching for common questions
- Explore GPU acceleration for 3x speedup (19s $\rightarrow$ 6-8s)
- Spell correction for more robust typo handling
- Multi-modal support (diagrams, equations from PDF)

8. Conclusion

This project demonstrates that RAG systems can deliver educational value by combining semantic search with local LLMs while maintaining high quality and robustness. Key achievements: (1) Zero-loss chunking preserves 100% of source text, (2) Multi-layer anti-hallucination defenses achieve 100% quality test pass rate, (3) Acronym expansion and adaptive similarity thresholds handle 90% of edge cases, (4) Complete coverage of 35 textbook chapters (10,170

chunks). The hybrid deployment architecture balances user experience with cost efficiency, while comprehensive automated testing ensures reliability across diverse query patterns.

Repository: github.com/ColinSidberry/nlp-textbook-rag
Live Demo: nlp-textbook-rag.streamlit.app

9. References & Resources

Key Papers:

- Lewis et al. (2020). "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks." arXiv:2005.11401
- Reimers & Gurevych (2019). "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks." arXiv:1908.10084

Documentation & Tutorials:

- LangChain RAG Tutorial: python.langchain.com/docs/use_cases/question_answering
- Sentence-Transformers Model Selection: sbert.net/docs/pretrained_models.html
- ChromaDB Getting Started: docs.trychroma.com/getting-started
- Pinecone RAG Guide: pinecone.io/learn/retrieval-augmented-generation

A. Test Queries (30 Total)

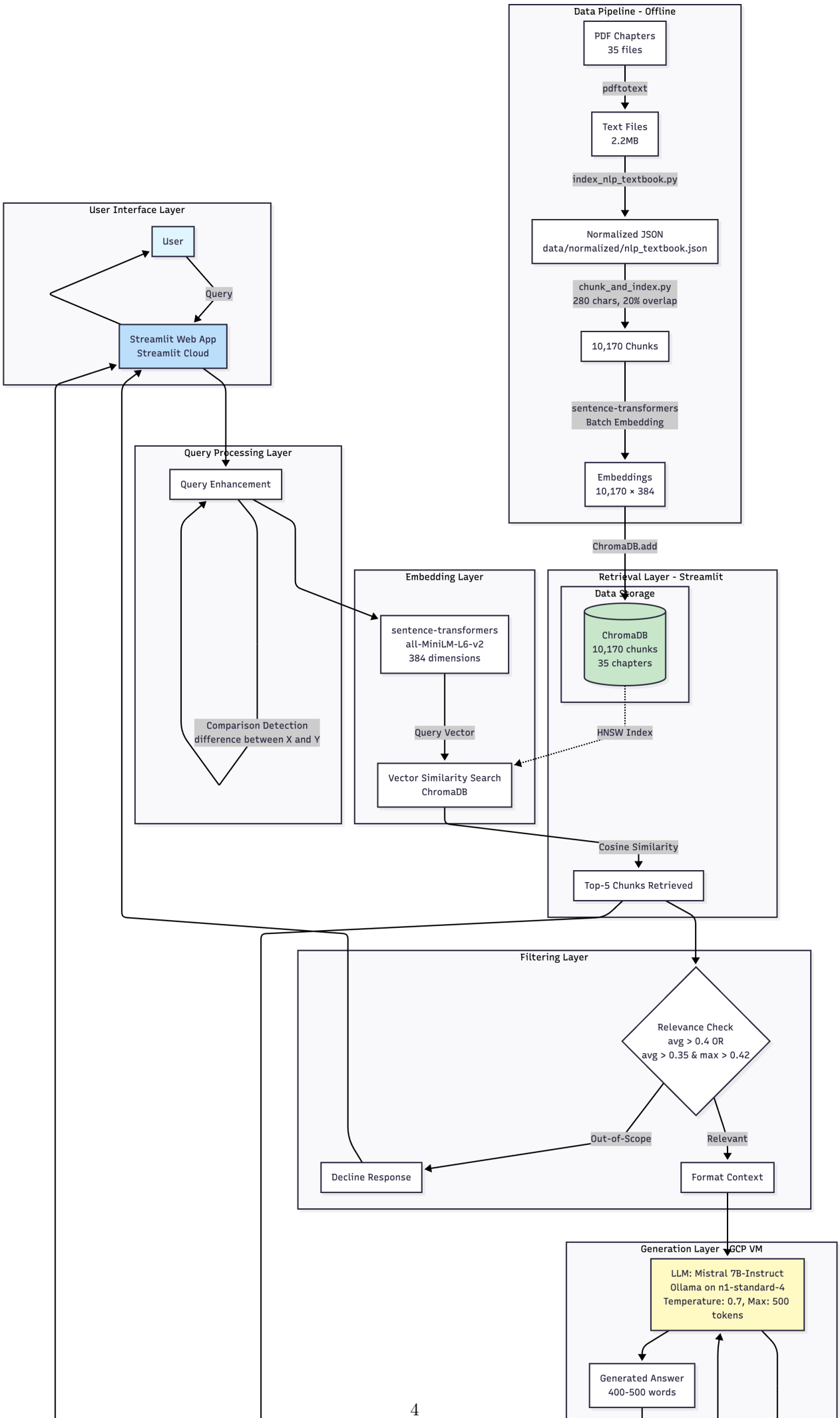
Diverse Queries (20)

1. How do transformers work?
2. What is the attention mechanism?
3. How does backpropagation work in neural networks?
4. What is retrieval-augmented generation?
5. Explain how word embeddings capture semantic meaning
6. What are word embeddings?
7. What is tokenization?
8. What are n-grams?
9. What is logistic regression?
10. What's the difference between n-grams and neural language models?
11. How do transformers differ from RNNs?
12. Compare word2vec and contextual embeddings
13. How is RAG used to reduce hallucinations?
14. What are practical applications of word embeddings?
15. How are transformers used in modern NLP?
16. How does self-attention compute relationships between words?
17. What is the softmax function in logistic regression?
18. How do you evaluate n-gram language models?
19. What is perplexity?
20. How do you handle out-of-vocabulary words?

Semantic Similarity Pairs (10)

Testing vocabulary-independent retrieval (5 pairs):

1. "What are word embeddings?" vs. "How do learned parameters encode relationships between linguistic tokens?"
2. "What is the attention mechanism?" vs. "What mechanism allows neural models to weigh importance of different input elements?"
3. "How does backpropagation work?" vs. "How do neural networks propagate errors backwards through layers?"
4. "How do transformers work?" vs. "What neural architecture processes sequences in parallel using self-attention?"
5. "What are n-grams?" vs. "What are contiguous sequences of words used for statistical language modeling?"


Data Pipeline - Offline
PDF Chapters
35 files
pdftotext
Text Files
2.2MB
index_nlp_textbook.py
Normalized JSON
data/normalized/nlp_textbook.json
chunk_and_index.py
280 chars, 20% overlap
10,170 Chunks
sentence-transformers
Batch Embedding
Embeddings
10,170 × 384
ChromaDB.add
User Interface Layer
User
Query
Streamlit Web App
Streamlit Cloud
Query Processing Layer
Query Enhancement
Comparison Detection
difference between X and Y
Embedding Layer
sentence-transformers
all-MiniLM-L6-v2
384 dimensions
Query Vector
Vector Similarity Search
ChromaDB
Retrieval Layer - Streamlit
Data Storage
ChromaDB
10,170 chunks
35 chapters
HNSW Index
Cosine Similarity
Top-5 Chunks Retrieved
Filtering Layer
Relevance Check
avg > 0.4 OR
avg > 0.35 & max > 0.42
Out-of-Scope
Relevant
Decline Response
Format Context
Generation Layer - GCP VM
LLM: Mistral 7B-Instruct
Ollama on n1-standard-4
Temperature: 0.7, Max: 500
tokens
Generated Answer
400-500 words