Hybrid RAG System with Automated Evaluation

Comprehensive Evaluation Report

Project	Hybrid RAG System with Automated Evaluation
Course	BITS Pilani - Conversational AI
Date	February 07, 2026
Questions	100 Questions × 3 Methods = 300 Evaluations

GitHub: https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation

Table of Contents

1. Executive Summary

This report presents a comprehensive evaluation of a **Hybrid Retrieval-Augmented Generation (RAG)** system that combines dense vector retrieval (ChromaDB + all-MiniLM-L6-v2), sparse keyword retrieval (BM25), and Reciprocal Rank Fusion (RRF) to answer questions from Wikipedia articles.

Key Finding: BM25 (Sparse) achieves the highest MRR of 0.4392, outperforming Dense (0.3025) by 45% and Hybrid (0.3783) by 16%.

Performance Summary:

Method	MRR	Recall@10	Avg Time (s)
Dense (ChromaDB)	0.3025	0.33	5.86
Sparse (BM25)	0.4392	0.47	5.53
Hybrid (RRF)	0.3783	0.43	6.37

2. System Architecture

2.1 Components

Component	Technology	Details
Dense Retrieval	ChromaDB + MiniLM	384-dim embeddings, 7,519 chunks
Sparse Retrieval	BM25 + NLTK	Tokenization, stopwords, stemming
Fusion	RRF	k=60 parameter
Generation	FLAN-T5-base	Text-to-text transformer
Interface	Streamlit	Web UI with method selection

2.2 Data Flow

- 1. Query Input: User enters question via Streamlit UI
- 2. Dense Retrieval: Query embedded, similarity search in ChromaDB
- 3. Sparse Retrieval: BM25 keyword matching on tokenized corpus
- 4. Fusion: RRF combines rankings from both methods
- 5. Generation: Top chunks fed to FLAN-T5 for answer generation
- 6. Display: Answer, sources, and metrics shown to user

Source: `https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/blob/main/chromadb_rag_system.py`

3. Dataset Description

3.1 Corpus Statistics

Metric	Value
Total Articles	~501 Wikipedia articles
Fixed URLs	200 unique URLs
Total Chunks	7,519 chunks
Avg Chunk Size	~160 tokens
Overlap	50 tokens
Corpus Size	14.5 MB (JSON)
Vector DB Size	212 MB (ChromaDB)

3.2 Evaluation Questions

Question Type	Count	Description
Factual	59	Direct fact-based questions
Comparative	15	Questions comparing concepts
Inferential	11	Reasoning-based questions
Multi-hop	15	Questions requiring multiple sources
Total	100	-

4. Evaluation Methodology

The evaluation framework tests each of the 100 questions against three retrieval methods: Dense-only (ChromaDB), Sparse-only (BM25), and Hybrid (RRF fusion). For each query:

- 1. Retrieve top-10 chunks using the selected method
- 2. Generate answer using FLAN-T5 with retrieved context
- 3. Calculate MRR based on ground truth URL ranking
- 4. Calculate Recall@10 for source retrieval
- 5. Calculate Answer F1 for generation quality
- 6. Record timing metrics for performance analysis

Script: `https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/blob/main/evaluate_chromadb_fast.py`

5. Metric Definitions & Justifications

5.1 Mean Reciprocal Rank (MRR)

Definition: Average of the reciprocal of the rank at which the first relevant document appears.

Formula: MRR = (1/Q) × Σ(1/rank_i) where Q is number of queries

Why MRR? Ideal for QA systems where users care most about the first correct result. A score of 1.0 means perfect retrieval; 0.5 means the correct document is typically second.

5.2 Recall@10

Definition: Proportion of relevant documents retrieved in the top 10 results.

Formula: Recall@K = |Relevant ∩ Retrieved@K| / |Relevant|

Why Recall@10? In RAG systems, multiple chunks are passed to the LLM. Recall@10 ensures we capture relevant context even if not ranked first.

5.3 Answer F1 Score

Definition: Token-level F1 score measuring overlap between generated and expected answers.

Why Answer F1? Unlike exact match, F1 gives partial credit for overlapping content, important when generated answers may be correct but phrased differently.

```
Full Documentation: https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/blob/main/docs/METRIC_JUSTIFICATION.md
```

6. Results & Analysis

6.1 Detailed Results

Metric	Dense	Sparse	Hybrid	Best
MRR	0.3025	0.4392	0.3783	Sparse (+45%)
Recall@10	0.33	0.47	0.43	Sparse (+42%)
Answer F1	~0.05	~0.05	~0.05	Tie
Retrieval Time	0.09s	0.006s	0.09s	Sparse (15x)

6.2 Key Observations

- BM25 Dominance: Sparse retrieval outperforms dense by 45% on MRR
- Hybrid Underperformance: RRF doesn't exceed sparse with current k=60
- Low Answer F1: All methods have ~0.05 F1, suggesting model limitations
- Speed: BM25 is 15x faster than dense retrieval

7. Error Analysis

Error Type	Count	%	Description
Retrieval Failure	~53	53%	Correct doc not in top 10
Partial Match	~20	20%	Doc found, wrong chunk
Generation Error	~15	15%	Correct context, wrong answer
Ambiguous Query	~12	12%	Question unclear

Details: `https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/blob/main/docs/ERROR_ANALYSIS.md`

8. Ablation Studies

Configuration	MRR	Recall@10	vs Best
Dense Only	0.3025	0.33	-31%
Sparse Only (Best)	0.4392	0.47	Baseline
Hybrid (RRF k=60)	0.3783	0.43	-14%

Future Ablation Studies:

- K parameter: K=5, 10, 15, 20 for top-K retrieval
- RRF k: k=30, 60, 100 for fusion weighting
- Embedding models: MiniLM vs larger models
- Chunk sizes: 100, 200, 300, 400 tokens

9. Conclusions & Recommendations

9.1 Key Findings

- BM25 (sparse) outperforms dense vector retrieval by 45% for Wikipedia QA
- Hybrid RRF doesn't exceed sparse performance with current parameters
- Answer generation limited by FLAN-T5-base model
- Retrieval failures account for 53% of errors

9.2 Recommendations

- Tune RRF k parameter for better fusion
- Add cross-encoder re-ranking for precision
- Fine-tune FLAN-T5 on Wikipedia QA
- Implement query expansion for complex questions

9.3 Repository

Main: https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation

RAG System: https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/blob/main/chromadb_rag_system.py

Evaluation: https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/blob/main/evaluate_chromadb_fast.py

UI: https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/blob/main/app_chromadb.py

Docs: https://github.com/vishalvishal099/Hybrid_RAG_System_with_Automated_Evaluation/tree/main/docs