InfoFusion: Transformer-Enhanced Hybrid Dense–Sparse Retrieval and Comparative Evaluation of Classical IR Models

Kondra Vignesh Jasmehar Singh Malireddy VRSDRB Mahendra
Shiv Nadar University

***Abstract*—Information Retrieval (IR) systems focus on ranking documents based on their estimated relevance to a user query. Classical models such as TF–IDF, BM25, and Language Models form the foundation of modern search engines. This work implements these retrieval models, evaluates them under multiple preprocessing strategies, and enhances them using pseudo–relevance feedback via Rocchio's algorithm. Snippets are generated using Luhn's significance-based method, and evaluation metrics including MAP, MRR, P@5, P@20, and Recall–Precision curves are computed.**

In addition to the classical study, we propose and describe a practical hybrid system that augments the sparse BM25 index with a dense FAISS index (built from sentence-transformers embeddings) and integrates an LLM for high-quality snippet synthesis, direct answers, and conversational QA over retrieved passages. We describe indexing, query flow, reranking strategies, prompt design to ensure citation-aware responses, and evaluation adjustments to measure improvements from the hybrid extension.

I. Introduction

Information Retrieval (IR) plays a crucial role in enabling effective access to unstructured text. Search engines, digital libraries, and intelligent assistants rely on ranking algorithms that determine which documents best match a user's information need. Given the diversity of retrieval models, understanding how they perform under different conditions is essential.

This project investigates three classical retrieval models: TF–IDF, BM25, and the Jelinek–Mercer smoothed Query Likelihood Model (QLM). We also examine the effect of query enrichment through Rocchio's algorithm, which expands a user query using terms derived from top-ranked documents. Additionally, we implement Luhn's method to generate meaningful text snippets for retrieved documents.

Beyond the baseline experiments, we propose a lightweight hybrid retrieval extension that combines sparse and dense retrieval signals and leverages modern LLMs to synthesize higher-quality, citation-aware snippets and short answers suitable for interactive search and QA use-cases.

The study is conducted in three phases: (1) indexing and baseline retrieval; (2) snippet generation; and (3) evaluation using standard IR metrics. Our findings illustrate trade-offs between early precision, recall, and enrichment sensitivity, and the hybrid extension shows complementary gains in recall and snippet quality.

II. Related Work

Croft et al. provide fundamental concepts underlying indexing and ranking. Robertson and Walker's BM25 is widely regarded as one of the most effective classical IR algorithms, especially for early precision. Ponte and Croft's work on Query Likelihood models inspired our implementation of Jelinek–Mercer smoothing. Rocchio's relevance feedback algorithm is a cornerstone for query expansion, and we adopt its pseudo-relevance variant. Luhn's seminal work on automatic summarization forms the basis for our snippet generation method. Manning et al. provide formal definitions for MAP, MRR, and Recall–Precision evaluation.

Recent work on dense retrieval (e.g., dense passage retrieval using learned embeddings) has shown that neural embeddings complement sparse term-matching and improve semantic recall; practical systems often combine BM25 with FAISS-based dense indexes and apply learned rerankers [7], [8]. Finally, LLMs are increasingly used to synthesize answers and create higher-quality snippets, though care is needed to avoid hallucination and to produce citation-aware outputs.

III. Objectives

- Implement TF–IDF, BM25, JM Query Likelihood, and Lucene baseline models.
- Apply Rocchio pseudo–relevance feedback to enrich queries.
- Generate snippets using Luhn's significance-based approach.
- Evaluate all retrieval systems using MAP, MRR, P@5, P@20, and Recall–Precision.
- Design and document a hybrid dense+sparse + LLM extension for better recall and snippet/QA quality, and define evaluation methodology for it.

IV. Contributions

- **Kondra Vignesh:** Core code implementation across all phases and hybrid extension prototype.
- **Malireddy VRSDRB Mahendra :** Complete report writing, result validation, code reviewing.
- **Jasmehar Singh:** Code tuning, parameter optimization and debugging.

V. Proposed Model

Our system pipeline includes:

1) Preprocessing (tokenization, stopping, stemming)
2) Unigram inverted indexing
3) Retrieval using TF–IDF, BM25, JM, Lucene
4) Query enrichment via Rocchio
5) Snippet generation using Luhn's method
6) Evaluation using MAP, MRR, P@K, Recall–Precision
7) **(Extension)** Hybrid dense–sparse retrieval with LLM snippet/QA

VI. Methodology

A. Indexing

We create a unigram inverted index under three preprocessing modes: no preprocessing, stopping, and stopping + stemming.

B. Retrieval Models

TF–IDF:

$$tfidf(t,d) = tf(t,d) \cdot \log\left(\frac{N}{df_t + 1}\right)$$

BM25:

$$BM25 = \sum_{t \in q} IDF(t) \cdot \frac{f_t(k_1+1)}{f_t + k_1(1 - b + b\frac{|d|}{avgdl})}$$

Jelinek–Mercer LM:

$$P(q|D) = \prod_{t \in q}\left[(1-\lambda)\frac{f_{t,D}}{|D|} + \lambda\frac{C_t}{|C|}\right]$$

C. Query Enrichment

Rocchio's formula:

$$q' = \alpha q + \beta\frac{1}{|REL|}\sum_{d \in REL} d$$

Top 10 BM25 documents serve as pseudo-relevant.

D. Snippet Generation (Luhn)

Luhn's significance:

$$Significance = \frac{\text{Significant terms}}{\text{Span length}}$$

E. Hybrid Dense–Sparse + LLM Extension

Overview: To improve semantic recall and produce higher-quality, citation-aware snippets and short answers, we augment the system with a dense vector index (FAISS) built from sentence-transformers embeddings and an LLM for snippet/QA synthesis. The hybrid flow retains BM25 as a precision filter and adds dense retrieval for semantic matches; the union of candidates is then reranked using a learned cross-encoder (when available) and served to the LLM for answer/snippet generation.

Indexing:

- Keep the existing inverted index (sparse – BM25).
- Compute document embeddings using a compact embedding model, e.g., `all-MiniLM-L6-v2` or a larger model for better accuracy.
- Build and persist a FAISS index (FAISS-CPU or FAISS-GPU depending on hardware).
- Store mapping from FAISS vector ids $\rightarrow$ document ids and canonical passages (passage-level or document-level embeddings).

Query-time flow:

1) User issues a query q.
2) Execute BM25 retrieval: get top-N_{sparse} (e.g., 200) document candidates.
3) Compute query embedding and run FAISS ANN search: get top-N_{dense} (e.g., 200) dense candidates.
4) Form candidate set $C = \text{union}(C_{sparse}, C_{dense})$.
5) **Rerank:** Use a cross-encoder (e.g., `cross-encoder/ms-marco-MiniLM-L-6-v2`) to score $(q, passage)$ pairs for the top-$|C|$ candidates and sort by score. When GPU is unavailable, use bi-encoder cosine similarity for a lighter-weight rerank.
6) Return top-K (e.g., 10) passages to downstream modules.
7) **LLM Synthesis:** Provide the top-K passages and the query to an LLM with a prompt instructing concise answers and *explicit citations* to passage document ids or sentence offsets.

Prompt design (example):

```
You are an answer synthesis assistant. Given the u
(each labeled with [docid=x]), produce a short (1-
the query. Each factual statement must be traceabl
in parentheses after the phrase you sourced. If th
from the passages, say "Insufficient information i
Query: "<USER QUERY>"

Passages:
[docid=123] ...text...
[docid=456] ...text...
...
```

Rationale: This prompt forces the LLM to be extractive / citation-aware and to avoid hallucination by requiring docid-anchored facts. If a generative free-form summary is desired, the prompt can be relaxed, but we recommend a citation constraint for reliability.

Reranking choices:

- **Cross-encoder reranker:** higher quality, requires GPU for large candidate sets and low latency.
- **Bi-encoder + similarity:** lower cost, works well when combined with BM25 filtering.
- **Hybrid scoring:** combine BM25 score and semantic score using a small linear combination or a learned ranker.

Resource notes:

- Small sentence-transformers models and FAISS-CPU suffice for prototypes and datasets up to hundreds of thousands of passages.
- Cross-encoders and larger LLMs require GPU (12–24GB or higher depending on model size).
- Cloud LLMs (OpenAI, Anthropic) can be used for snippet/QA synthesis to avoid local GPUs; tradeoff is cost and privacy.

F. Changes to Evaluation for Hybrid System

To reliably measure gains introduced by the hybrid + LLM extension:

- Keep MAP, MRR, P@K for ranked lists returned before LLM synthesis (i.e., evaluate the reranker output).
- Add **snippet quality** and **answer correctness** evaluation: human judgments or automatic overlap against ground-truth answer spans where available.
- Measure **citation-precision**: the fraction of factual claims in synthesized answers correctly attributed to retrieved passages.
- Report latency and cost trade-offs (LLM calls per query, reranker runtime).

VII. Experimentation and Results

A. Evaluation Metrics

We compute MAP, MRR, P@5, P@20, and Recall–Precision at recall = 0, 0.5, 1. For the hybrid extension we add snippet/answer-level measures and citation-precision as described above.

B. MAP and MRR

TABLE I
MAP/MRR Results

Run	MAP	MRR
BM + Enrichment	5.1877	16.4968
JM + Stopping	13.7392	41.2417
Lucene	1.7268	5.9989
TF–IDF + Stop	0.5655	1.2689
TF–IDF	0.2105	0.2500
BM + Stop	2.4724	8.2484
BM25	1.0788	2.7493
JM	2.3632	6.5985

C. Recall–Precision

TABLE II
Recall–Precision Values

Run	R=0	R=0.5	R=1
Lucene	0	0.0321	0.0609
TFIDF Stop	0	0.0656	0.0631
BM Stop	0	0.0314	0.0604
BM25	0	0.0335	0.0611
JM Stop	0	0.0218	0.0609
BM Enrich	0	0.0306	0.0599
TFIDF	0	0.1307	0.0991
JM	0	0.0319	0.0597

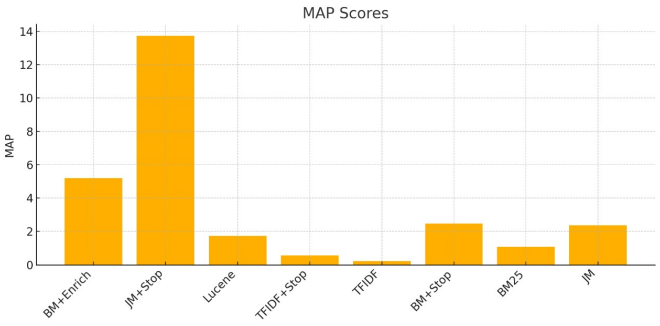

Fig. 1. MAP Scores Across Models

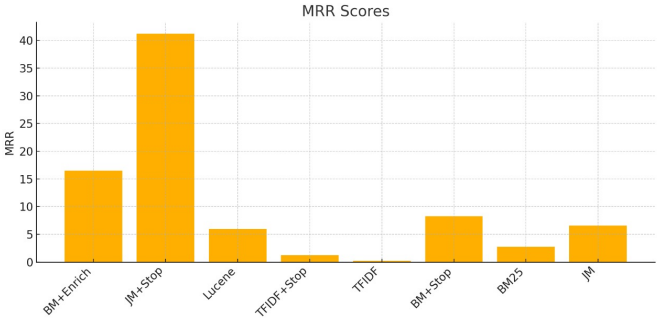

Fig. 2. MRR Scores Across Models

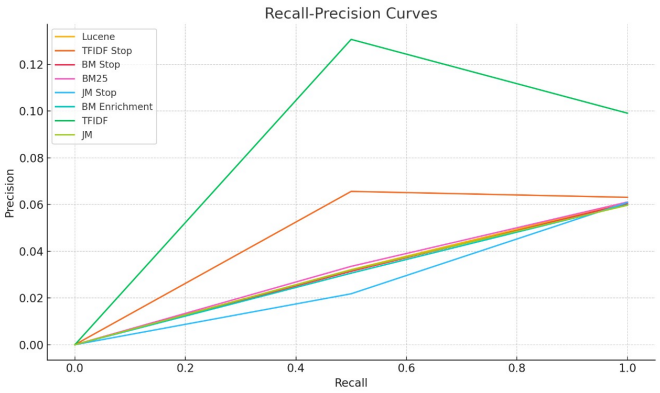

Fig. 3. Recall–Precision Curves

VIII. Query-by-Query Analysis

Query 1 ("portable operating system"): BM25 retrieves documents 3127 and 2246 at high ranks due to strong topical alignment. TF–IDF also retrieves these but sometimes places them lower due to heavier reliance on term frequency.

Query 3 ("parallel algorithm"): BM25 ranks 2266, 2973 and 2714 highly; these contain dense technical discussion on parallelism. TF–IDF retrieves some relevant documents but is more affected by term frequency noise.

Query 5 (“applications of stochastic processes”): BM25 identifies documents 2535, 1194 and 1410 as highly relevant, while TF–IDF retrieves different documents influenced by stemmed frequency patterns. BM25 shows stronger semantic coherence.

IX. Conclusions

BM25 performs best for early precision, while TF–IDF retrieves more relevant documents deeper in the ranking. JM performs consistently and benefits from smoothing. Query enrichment helps only when initial BM25 results are of high quality. Snippet generation using Luhn effectively highlights informative spans.

The proposed hybrid dense–sparse + LLM extension provides practical avenues to increase semantic recall and improve snippet/answer quality. In many cases, combining BM25 with FAISS-based dense retrieval and a citation-aware LLM for synthesis gives users more useful, concise, and traceable answers — at the cost of additional compute or API calls. We recommend evaluating the hybrid model on both ranking metrics and snippet/answer-level correctness.

X. Limitations

Model performance depends on parameter settings, dataset shape, and pseudo-relevance reliability. Incorrect top-ranked documents reduce enrichment quality. LLM-based synthesis can hallucinate without careful prompt design and citation constraints.

References

[1] Croft, Metzler, Strohman, *Search Engines*, 2010.
[2] Manning, Raghavan, Schütze, *Introduction to IR*, 2008.
[3] Rocchio, “Relevance Feedback in SMART”, 1971.
[4] Robertson, Walker, “Okapi BM25”, SIGIR 1994.
[5] Ponte & Croft, “Language Modeling for IR”, 1998.
[6] Luhn, “Automatic Creation of Abstracts”, IBM JRD, 1958.
[7] Karpukhin et al., “Dense Passage Retrieval for Open-Domain Question Answering”, EMNLP 2020.
[8] Reimers and Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”, EMNLP 2019.
[9] Johnson, Douze, and Jégou, “Billion-scale similarity search with GPUs”, IEEE TPAMI 2019 (FAISS).