engineering proceedings

MDPI

Proceeding Paper

Enhancing Candidate Generation in Recommendation Systems Through LLM-Powered Semantic Enrichment in a Distributed Environment †

Balagangadhar Reddy Kandula * and Lija Jacob

Department of Data Science, CHRIST (Deemed to be University), Lavasa 412112, India; lija.jacob@christuniversity.in

* Correspondence: balagangadhar.reddy@res.christuniversity.in

† Presented at the 6th International Electronic Conference on Applied Sciences, 9–11 December 2025; Available online: https://sciforum.net/event/ASEC2025.

Abstract

Effective candidate generation is a critical component of two-stage recommender systems; however, traditional methods such as Term Frequency–Inverse Document Frequency (TF-IDF) often fail to capture deep semantic context. This limitation leads to suboptimal recall rates, particularly for new or niche items—a challenge commonly referred to as the cold start problem—thereby degrading overall recommendation quality and user experience. This study proposes a semantically aware approach to improve the initial recall phase of recommendation pipelines. The methodology integrates Large Language Models (LLMs) into a distributed Apache Spark pipeline for large-scale content enrichment, generating 768-dimensional vector embeddings and concise, context-aware summaries for each content item. These enriched representations are indexed in Elasticsearch to enable efficient vector-based retrieval during candidate generation. Quantitative evaluation on a corpus of 143,000 Wikipedia articles demonstrates that the LLM-enriched method achieves a Recall@10 of 62%, representing a 37% relative improvement over the TF-IDF baseline (45%). When relevance is measured using only embedding-independent signals (category overlap and keyword similarity), the method still achieves a Recall@10 of 58%, confirming that gains are not an artifact of the evaluation metric. The resulting candidate pools exhibit improved semantic diversity and broader category coverage, delivering richer input for downstream ranking models.

Keywords: recommender systems; large language models; candidate generation; semantic enrichment; Apache Spark; vector embeddings; cold start problem

Academic Editor: Lucia Billeci

Published: 6 March 2026

1. Introduction

Large-scale recommender systems commonly adopt a two-stage architecture comprising a recall stage that retrieves a broad set of potentially relevant candidates and a ranking stage that precisely orders these candidates for presentation [1]. Although ranking algorithms have received substantial attention, the recall stage remains a critical bottleneck that constrains overall system performance.

Conventional candidate generation relies on lexical matching methods such as TF-IDF, which measure similarity through term overlap. While effective for documents sharing similar vocabularies, these methods fail when semantic relationships extend beyond surface-level term matching. Additionally, the cold start problem—where new items lack

interaction history—prevents collaborative filtering from being applied, disproportionately affecting knowledge platforms with long-tail content [2].

Recent advances in Large Language Models (LLMs) such as BERT [3] and Gemini offer promising solutions by generating vector embeddings that encode semantic meaning, enabling similarity computation beyond lexical boundaries. However, deploying LLMs at production scale presents significant engineering challenges.

This study proposes a methodology integrating LLM-powered semantic enrichment into candidate generation. The research objectives are: (1) to design a scalable architecture leveraging LLM-based content enrichment within a distributed Apache Spark pipeline; (2) to evaluate semantic embedding-based retrieval against lexical baselines; and (3) to assess the impact on retrieval diversity and cold start performance. Unlike prior dense retrieval work targeting ad hoc search [4], this study focuses on a production-oriented recommendation pipeline combining distributed batch processing via Spark with real-time serving through Elasticsearch, processing 143,000 Wikipedia articles using Google's text-embedding-004 model.

2. Related Work

2.1. Two-Stage Recommendation Architecture

The two-stage paradigm has emerged as the dominant architecture for large-scale systems [5]. Lightweight recall methods reduce millions of candidates to hundreds or thousands, while sophisticated ranking models produce final recommendations [6]. Conventional recall approaches include collaborative filtering via matrix factorization [7] and content-based methods using TF-IDF or BM25 [8]. More recent approaches incorporate ANN search over learned embeddings for sub-linear retrieval complexity [9].

2.2. Dense Retrieval and Semantic Embeddings

Word2Vec [10] demonstrated that dense vectors capture semantic relationships through distributional patterns. Sentence-BERT [11] adapted the BERT architecture for efficient sentence embedding generation and remains a widely used open-source baseline for dense retrieval. Dense Passage Retrieval (DPR) [4] showed that learned dense representations can outperform BM25, while ColBERT [12] introduced late interaction for fine-grained token-level matching. General-purpose models such as E5 [13] have achieved state-of-the-art results across diverse benchmarks. More recently, Anthropic's Contextual Retrieval [14] demonstrated that LLM-based content enrichment before embedding—at the chunk level—combined with hybrid retrieval reduces retrieval failures by 49–67%, validating the premise that semantic enrichment improves retrieval quality. In recommender systems, transformer-based embeddings consistently improve over conventional methods [15], though generating embeddings at scale remains a practical barrier.

2.3. Cold Start Problem

The cold start problem manifests in three forms: new user, new item, and system cold start [16]. Content-based methods provide the primary solution for new-item cold start, as item features remain available without interaction history. Recent work has explored leveraging language models to generate pseudo-interactions or transfer knowledge from auxiliary domains [17], though these approaches often require substantial training data. Pre-trained LLM embeddings offer a simpler alternative deployable without domain-specific fine-tuning.

https://doi.org/10.3390/engproc2026124055

2.4. Distributed Processing for Recommendations

Apache Spark has become the standard framework for distributed data processing in recommendation pipelines [18], with UDFs enabling incorporation of specialized components such as embedding models. Elasticsearch provides dense vector search through HNSW indexing for efficient ANN queries on high-dimensional embedding spaces [19].

3. Methodology

3.1. System Architecture

The proposed system architecture comprises three primary components: a distributed content enrichment pipeline, a vector indexing layer, and a retrieval service. Figure 1 illustrates the overall system design. Raw article content enters the pipeline through Apache Spark (version 3.4.1), undergoes semantic enrichment via LLM integration, and is indexed into Elasticsearch for efficient retrieval. The retrieval service accepts seed articles as queries and returns semantically similar candidates.

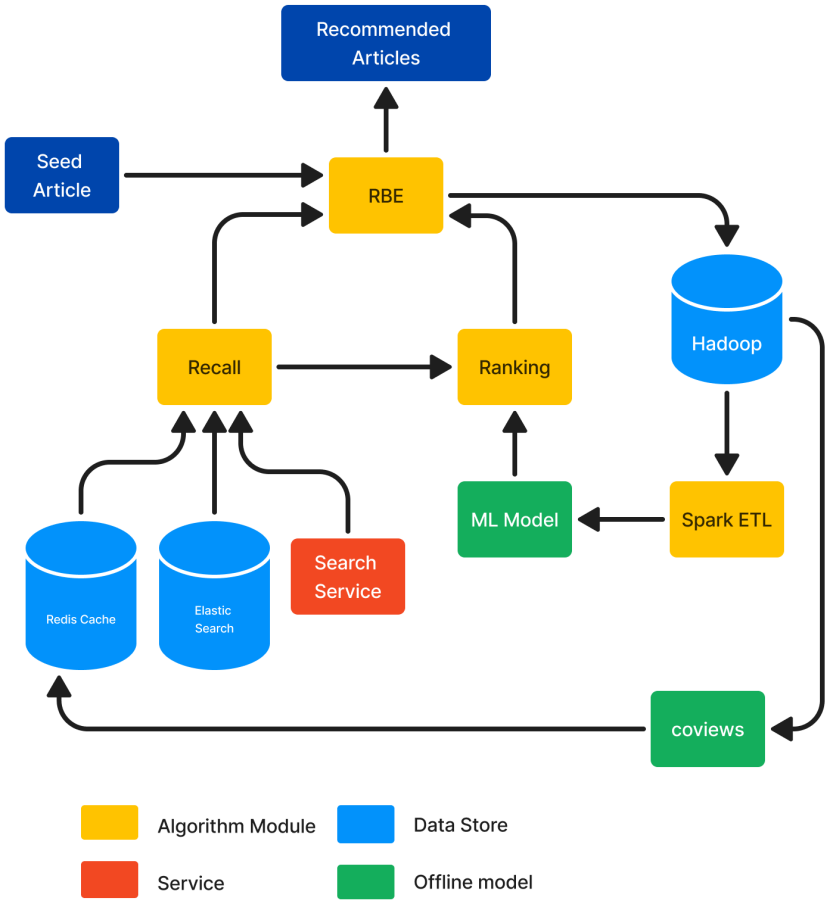

Figure 1. System architecture overview.

Batch processing through Spark handles the computationally intensive embedding generation phase, while Elasticsearch provides sub-second retrieval latency for production queries, enabling independent scaling of processing and serving components.

3.2. Dataset

The experimental evaluation employed a large-scale Wikipedia corpus comprising 143,000 articles obtained through the Wikipedia API. Articles were selected from multiple knowledge domains including Artificial Intelligence, Machine Learning, Natural Language Processing, Computer Science, Data Science, Physics, Mathematics, and related interdis-

ciplinary fields. The average article length of 2847 words provides sufficient content for meaningful semantic analysis, and the category structure (averaging 4.2 categories per article) offers ground truth signals for relevance assessment.

3.3. Pre-Processing Pipeline

The pre-processing pipeline comprised four stages within Spark: tokenization, stopword removal, and lemmatization using NLTK; TF-IDF keyword extraction retaining the top 20 keywords per article; and named entity recognition via part-of-speech tagging. Summary generation employed the Gemini 1.5 Flash model (version gemini-1.5-flash-001, temperature 0.2, max output tokens 300) to produce context-aware summaries of approximately 200 words per article, with articles exceeding 8000 tokens truncated prior to summarization.

3.4. LLM-Powered Semantic Enrichment

The semantic enrichment process integrates Google's text-embedding-004 model (768 dimensions, task type: RETRIEVAL_DOCUMENT) through the Vertex AI API to generate dense vector embeddings. Embeddings were generated for two inputs per article: the LLM-generated summary and a concatenation of the title and extracted keywords, with summary embeddings serving as the primary retrieval vectors. Embedding generation was distributed across Spark worker nodes using a custom UDF with connection pooling, rate limiting at 300 requests per minute per node, and exponential backoff for transient failures.

3.5. Vector Indexing and Candidate Generation

Enriched article data was indexed into Elasticsearch using the dense vector field format with HNSW indexing (m = 32, ef_construction = 200). Each indexed document contains original metadata, extracted features, a generated summary, and an embedding vector, with hybrid retrieval supported through inverted indexes in the category and keyword fields. The candidate generation process accepts a seed article, generates an embedding using the same pipeline, and performs an ANN search in Elasticsearch, returning the top-k candidates (default 100) ranked by cosine similarity.

3.6. Baseline Implementation

The TF-IDF baseline computed sparse vectors (50,000 dimensions) across the corpus vocabulary, with candidate retrieval via cosine similarity. BM25 retrieval through Elasticsearch's native text search served as an additional baseline with default parameters (k1 = 1.2, b = 0.75) [8]. Sentence-BERT (all-mpnet-base-v2) [11] served as a modern dense retrieval baseline, generating 768-dimensional embeddings of raw article text with retrieval via cosine similarity in the same Elasticsearch infrastructure.

4. Results and Analysis

4.1. Experimental Setup

The evaluation employed 200 randomly selected seed articles stratified across knowledge domains. Ground truth relevance required sharing at least two categories or at least three keywords with the seed article. To address potential bias from embedding-based criteria, the results are reported both with and without an embedding similarity threshold (cosine similarity > 0.5). All experiments were conducted over five independent runs; paired t-tests across the 200 seed articles (199 degrees of freedom) assessed significance at $\alpha = 0.05$, with Cohen's d for effect size.

https://doi.org/10.3390/engproc2026124055

4.2. Recall Performance

Table 1 presents recall performance. The LLM-enriched method achieved Recall@10 of 62.3%, outperforming TF-IDF (45.2%) and BM25 (48.7%) by 37.8% and 27.9% respectively. Using embedding-independent ground truth only, Recall@10 was 58.1%, confirming that gains are not an artifact of circular evaluation.

Table 1. Recall performance comparison. Values in parentheses indicate results using embedding-independent ground truth only.

Method	Recall@10	Recall@20	Recall@50
TF-IDF Baseline	45.2% (44.8%)	58.4%	72.1%
BM25 Baseline	48.7% (48.3%)	61.2%	75.8%
Sentence-BERT	54.1% (53.2%)	66.8%	80.3%
LLM-Enriched (Proposed)	62.3% (58.1%)	74.6%	87.2%

The improvement was statistically significant across all metrics. Paired t-tests comparing the LLM-enriched method to TF-IDF yielded $t(199) = 5.82$, $p < 0.001$, and Cohen's $d = 0.84$, with a 95% confidence interval of [14.8%, 19.4%] for the Recall@10 difference. The comparison against BM25 was also significant: $t(199) = 4.37$, $p < 0.001$, $d = 0.62$. Crucially, the improvement over Sentence-BERT was also significant ($t(199) = 3.91$, $p < 0.001$, $d = 0.55$), confirming that commercial LLM embeddings on enriched content offer measurable advantages over open-source dense retrieval alternatives.

4.3. Semantic Diversity Analysis

Semantic diversity was measured as the average pairwise cosine distance among candidates (higher = more diverse). Coverage indicates the proportion of unique categories in the candidate set relative to the evaluation corpus (Table 2).

Table 2. Diversity and precision analysis.

Method	Precision@10	Diversity	Coverage
TF-IDF Baseline	0.312	0.287	34.2%
BM25 Baseline	0.345	0.302	38.7%
Sentence-BERT	0.398	0.358	43.9%
LLM-Enriched (Proposed)	0.478	0.423	52.4%

The LLM-enriched method demonstrated 47.4% higher diversity than TF-IDF (0.423 versus 0.287), and category coverage increased from 34.2% to 52.4%, indicating retrieval across a wider range of topical areas through conceptual rather than lexical relationships.

4.4. Cold Start Performance

Recall was measured separately for articles in the lowest quartile of page view counts (Table 3). Note that TF-IDF and BM25 are also content-based methods; this comparison evaluates the advantage of LLM-based semantic understanding over lexical analysis, not cold start mitigation relative to collaborative filtering.

Table 3. Cold start performance comparison (Recall@10).

Method	Popular Items	Cold Start Items
TF-IDF Baseline	48.7%	38.2%
BM25 Baseline	49.1%	40.5%
Sentence-BERT	55.8%	48.3%
LLM-Enriched (Proposed)	64.1%	58.9%

https://doi.org/10.3390/engproc2026124055

The LLM-enriched method showed a 54.2% relative improvement for cold start items versus 31.6% for popular items over TF-IDF. Sentence-BERT occupies a middle ground (55.8% popular, 48.3% cold start), confirming that while dense representations improve over lexical methods, the additional semantic enrichment through LLM summarization provides further gains for under-represented content.

4.5. Ablation: Summarization Impact

To isolate the contribution of LLM summarization, we compared Gemini embeddings generated from raw article text (truncated to 8000 tokens) against embeddings of LLM-generated summaries. Raw-text Gemini embeddings achieved Recall@10 of 56.8%, compared to 62.3% with summary-based embeddings—a 9.7% relative improvement ($t(199) = 2.89$, $p = 0.004$, $d = 0.41$). This indicates that summarization contributes meaningfully by focusing semantic content before embedding, though the embedding model itself accounts for the majority of gains over lexical baselines (56.8% vs. 45.2% for TF-IDF).

4.6. Processing Performance

A five-node Spark cluster (each node: 16 vCPUs, 64 GB RAM) processed the full corpus in approximately 14 h, with embedding generation achieving a throughput of 42 articles per minute per worker node. The Gemini API cost for processing 143,000 articles was approximately $85 USD (at $0.0006 per 1000 characters). Elasticsearch indexing completed in 2.3 h, producing an index of 18.7 GB. The average query latency for retrieving 100 candidates was 87 ms, with a p99 latency of 142 ms.

5. Discussion

The 37% relative improvement in Recall@10 over TF-IDF translates directly to higher-quality input for downstream ranking models. Crucially, the proposed method also significantly outperforms Sentence-BERT (62.3% vs. 54.1%, $p < 0.001$), demonstrating that commercial LLM embeddings with content enrichment offer measurable advantages over open-source dense retrieval alternatives—not merely over lexical baselines. The persistence of gains under embedding-independent ground truth confirms genuine semantic matching rather than circular evaluation bias.

The ablation analysis reveals that both the embedding model and the summarization step contribute to the observed gains. Raw-text Gemini embeddings (56.8%) outperform Sentence-BERT (54.1%), while LLM summarization provides an additional 9.7% relative improvement by focusing semantic content before embedding. This aligns with findings from Anthropic's Contextual Retrieval [14], which showed that content enrichment before embedding reduces retrieval failures by 49–67%. Our document-level summarization approach differs from their chunk-level contextualization but validates the same underlying principle: enriching content representations before embedding improves retrieval quality. The diversity and cold start analyses further confirm that semantic embeddings capture conceptual relationships beyond vocabulary boundaries, with specialized-vocabulary items benefiting disproportionately.

From an operational perspective, pre-trained LLM embeddings eliminate the need for domain-specific training data, the batch architecture scales horizontally, and the one-time cost ($85 for 143 K articles) is modest. Query-time costs are negligible, with storage growing linearly (130 KB per article).

Several limitations remain. Firstly, the ground truth relies on proxy signals rather than human judgments; future work should incorporate user studies. Secondly, the evaluation is limited to STEM-adjacent Wikipedia articles. Thirdly, while we include Sentence-BERT as a dense baseline, comparisons with supervised models such as DPR [4] or Col-

BERT [12] should be carried out in future work. Finally, as a commercial versioned API (text-embedding-004), reproducibility may be affected by model updates; we recommend archiving embeddings as research artifacts.

6. Conclusions

This study demonstrates that LLM-powered semantic enrichment substantially improves candidate generation: Recall@10 improved by 37.8% over TF-IDF and 15.2% over Sentence-BERT (62.3% vs. 54.1%), with gains persisting (58.1%) under embedding-independent evaluation. Ablation analysis confirms that both the embedding model and LLM summarization contribute meaningfully (raw-text 56.8% vs. summary-based 62.3%). Candidate diversity increased by 47.4% with 53.2% greater category coverage, and cold start items showed 54.2% relative improvement over TF-IDF. The distributed Spark pipeline processes 143,000 articles with $85 in API costs and with sub-100 ms query latency. These contributions provide empirical evidence that commercial LLM embeddings with content enrichment outperform both lexical methods and open-source dense retrieval alternatives in a production-ready architecture. Future work will extend evaluation to additional domains and conduct user studies.

Author Contributions: Conceptualization, B.R.K. and L.J.; methodology, B.R.K.; software, B.R.K.; validation, B.R.K. and L.J.; formal analysis, B.R.K.; investigation, B.R.K.; resources, L.J.; data curation, B.R.K.; writing—original draft preparation, B.R.K.; writing—review and editing, L.J.; visualization, B.R.K.; supervision, L.J.; project administration, L.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The Wikipedia dataset used in this study is publicly available through the Wikipedia API. Generated embeddings and summaries are available upon request to support reproducibility.

Acknowledgments: The authors would like to thank CHRIST (Deemed to be University) for providing the computational resources and infrastructure support necessary for this research.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Covington, P.; Adams, J.; Sargin, E. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016*; ACM: New York, NY, USA, 2016; pp. 191–198.
2. Schein, A.I.; Popescul, A.; Ungar, L.H.; Pennock, D.M. Methods and Metrics for Cold-Start Recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference, Tampere, Finland, 11–15 August 2002*; ACM: New York, NY, USA, 2002; pp. 253–260.
3. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 4171–4186.
4. Karpukhin, V.; Oğuz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; Yih, W. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of EMNLP, Online, 16–20 November 2020*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 6769–6781.
5. Wang, J.; Huang, P.; Zhao, H.; Zhang, Z.; Zhao, B.; Lee, D.L. Billion-Scale Commodity Embedding for E-Commerce Recommendation in Alibaba. In *Proceedings of KDD, London, UK, 19–23 August 2018*; ACM: New York, NY, USA, 2018; pp. 839–848.
6. Davidson, J.; Liebald, B.; Liu, J.; Nanez, P.; Van Vleet, T.; Gargi, U.; Sampath, D. The YouTube Video Recommendation System. In *Proceedings of the 4th ACM Conference on Recommender Systems, Barcelona, Spain, 26–30 September 2010*; ACM: New York, NY, USA, 2010; pp. 293–296.

7. Koren, Y.; Bell, R.; Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer* **2009**, *42*, 30–37. [CrossRef]
8. Robertson, S.; Zaragoza, H. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* **2009**, *3*, 333–389. [CrossRef]
9. Johnson, J.; Douze, M.; Jégou, H. Billion-Scale Similarity Search with GPUs. *IEEE Trans. Big Data* **2019**, *7*, 535–547. [CrossRef]
10. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781. [CrossRef]
11. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. In *Proceedings of EMNLP, Hong Kong, China, 3–7 November 2019*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 3982–3992.
12. Khattab, O.; Zaharia, M. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of SIGIR, Virtual Event, 25–30 July 2020*; ACM: New York, NY, USA, 2020; pp. 39–48.
13. Wang, L.; Yang, N.; Huang, X.; Jiao, B.; Yang, L.; Jiang, D.; Majumder, R.; Wei, F. Text Embeddings by Weakly-Supervised Contrastive Pre-training. *arXiv* **2022**, arXiv:2212.03533.
14. Anthropic. Introducing Contextual Retrieval. Anthropic Engineering Blog. September 2024. Available online: https://www.anthropic.com/engineering/contextual-retrieval (accessed on 3 March 2026).
15. Guo, H.; Tang, R.; Ye, Y.; Li, Z.; He, X. DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction. In *Proceedings of IJCAI, Melbourne, Australia, 19–25 August 2017*; IJCAI: Marina del Rey, CA, USA, 2017; pp. 1725–1731.
16. Lika, B.; Kolomvatsos, K.; Hadjiefthymiades, S. Facing the Cold Start Problem in Recommender Systems. *Expert Syst. Appl.* **2014**, *41*, 2065–2073. [CrossRef]
17. Ding, H.; Ma, Y.; Szummer, M.; Young, S.; Shum, H.P.; Yang, L. Cold Start Similar Artists Ranking with Gravity-Inspired Graph Autoencoders. In *Proceedings of RecSys, Amsterdam, The Netherlands, 27 September–1 October 2021*; ACM: New York, NY, USA, 2021; pp. 443–452.
18. Zaharia, M.; Xin, R.S.; Wendell, P.; Das, T.; Armbrust, M.; Dave, A.; Meng, X.; Rosen, J.; Venkataraman, S.; Franklin, M.J.; et al. Apache Spark: A Unified Engine for Big Data Processing. *Commun. ACM* **2016**, *59*, 56–65. [CrossRef]
19. Elasticsearch B.V. Dense Vector Field Type. Version 8.x. 2023. Available online: https://www.elastic.co/docs/reference/elasticsearch/mapping-reference/dense-vector (accessed on 3 March 2026).