Links
- Notebook: RAG_RapidFire.ipynb
- Repo: Github_Repo_RapidFire-AI-Winter-LLM-Challenge
- Screenshots:

1 to 4 of 4 entries Filter

index	run_id	model_name	search_type	rag_k	top_n	chunk_size	chunk_overlap	sampling_params	model_config	Samples Processed	Processing Time	Samples Per Second	Total	Hit Rate	Precision	Recall	NDCG@5	MRR
0	1	Qwen/Qwen2.5-0.5B-Instruct	similarity	20	5	256	32	{'temperature': 0.7, 'top_p': 0.95, 'max_tokens': 128}	{'dtype': 'half', 'gpu_memory_utilization': 0.25, 'enforce_eager': True, 'max_model_len': 2048, 'disable_log_stats': True}	100	438.55 seconds	0.23	100	0.0	0.0	0.0	0.0	0.0
1	2	Qwen/Qwen2.5-0.5B-Instruct	similarity	20	10	256	32	{'temperature': 0.7, 'top_p': 0.95, 'max_tokens': 128}	{'dtype': 'half', 'gpu_memory_utilization': 0.25, 'enforce_eager': True, 'max_model_len': 2048, 'disable_log_stats': True}	100	351.66 seconds	0.28	100	0.0	0.0	0.0	0.0	0.0
2	3	Qwen/Qwen2.5-0.5B-Instruct	similarity	20	5	128	32	{'temperature': 0.7, 'top_p': 0.95, 'max_tokens': 128}	{'dtype': 'half', 'gpu_memory_utilization': 0.25, 'enforce_eager': True, 'max_model_len': 2048, 'disable_log_stats': True}	100	325.98 seconds	0.31	100	0.0	0.0	0.0	0.0	0.0
3	4	Qwen/Qwen2.5-0.5B-Instruct	similarity	20	10	128	32	{'temperature': 0.7, 'top_p': 0.95, 'max_tokens': 128}	{'dtype': 'half', 'gpu_memory_utilization': 0.25, 'enforce_eager': True, 'max_model_len': 2048, 'disable_log_stats': True}	100	305.03 seconds	0.33	100	0.0	0.0	0.0	0.0	0.0

Show 25 per page

RAG Experiment Summary

Dataset + use case (3–6 sentences)

Use case / user:
The goal of this experiment is to build and evaluate a Retrieval-Augmented Generation (RAG) system for answering **electronics-related questions** using user-written product reviews. The intended user is a shopper or analyst seeking grounded answers based on prior customer experiences.

Datasets used (exact sources + roles):

- **Corpus:** Amazon Reviews Multilingual Dataset (buruzaemon/amazon_reviews_multi, English split), filtered to the *electronics* product category. Review bodies are treated as retrievable documents.
- **Eval queries / labels:** Review titles are used as queries, and relevance labels are constructed by grouping documents that share the same product_id (expanded QRELS).

What does "good" look like? Which metrics reflect that?
A good system retrieves reviews belonging to the same product as the query, ensuring factual grounding and relevance. Success is measured using **Precision, Recall, NDCG@5, and MRR**, which collectively capture retrieval accuracy, ranking quality, and early relevance.

Setup

- **Chunking (size/overlap):** RecursiveCharacterTextSplitter with sizes {256, 128} and overlap 32

- **Embeddings:** sentence-transformers/all-MiniLM-L6-v2 (GPU-accelerated, normalized embeddings)
- **Retriever:** FAISS similarity search, top-k = 20
- **Reranker:** Cross-encoder ms-marco-MiniLM-L6-v2 (CPU), top-n ∈ {5, 10}
- **Generator + prompt notes:** Qwen2.5-0.5B-Instruct with a system prompt instructing grounded answers based on retrieved reviews
- **Compute:** GPU for embeddings + FAISS search, CPU for reranking, constrained GPU memory utilization (0.25)

Experiment dimensions (knobs varied + why)

- **Chunking:** {128, 256} → tradeoff between finer semantic matching and increased chunk noise
- **top-k retrieval:** k = 20 → higher recall given multiple relevant documents per product
- **Embeddings:** Fixed MiniLM model → strong semantic quality with low latency
- **Reranker:** On/off implicitly tested via top-n ∈ {5, 10} → precision vs recall vs latency tradeoff
- **Prompt:** Single grounded-answer prompt → minimize hallucinations while keeping generation concise

Results

Variant	Key change(s)	Retrieval metric(s)	Answer metric(s)	Notes
Baseline	Chunked docs + FAISS	Precision = 0.0	MRR = 0.0	ID mismatch between chunks and QRELS
A	Larger chunk size	Precision = 0.0	NDCG@5 = 0.0	Chunking granularity unchanged at eval
B	Reranker top-n = 10	Recall = 0.0	MRR = 0.0	Reranker amplified mismatch
Best	***Conceptual best:* no chunking**	***Not executed***	***Expected > 0***	**Document-level alignment fixes metrics**

Why “Best” should win in this case (metrics + tradeoffs)

- **Best config (1 line):**
  Document-level retrieval without chunking, aligned directly with QRELS.

- **Biggest metric gains (expected to be):**
  - Precision: 0 → >0.3 (exact ID matching)
  - Recall: 0 → high (product-level relevance groups)
  - MRR: 0 → >0.2 (earlier relevant hits)

- **Tradeoffs (latency/tokens/failure modes):**
  Removing chunking reduces retrieval granularity but improves evaluation correctness. Chunking increases recall in theory but introduces evaluation failure when relevance is defined at the document level.

- **Why it outperformed:**
  The evaluation labels (corpus_id) were defined per document, while retrieval returned chunk-level IDs. This mismatch caused all relevance intersections to be empty, resulting in zero scores across all retrieval metrics.

RapidFire AI's contribution

- **What it accelerated:**
  RapidFire AI enabled fast iteration over chunk sizes, reranking strategies, and evaluation logic using a unified RAG + metrics pipeline.
- **What insight it surfaced:**
  The evaluation framework made it immediately visible that retrieval and QRELS were misaligned, revealing a common RAG failure mode: **chunk-level retrieval vs document-level relevance labels**.
- **Net impact:**
  Significant time saved on orchestration and metric aggregation, while increasing confidence in diagnosing failure modes rather than silently reporting misleading scores.

Key Reflection from my end:

Although all reported metrics were zero, this result reflects an **evaluation design mismatch rather than retrieval failure**, highlighting the importance of aligning chunking strategy with relevance annotations in RAG systems. I will be working on getting this aspect right in the upcoming days.