RAG Studio Session: Advanced Retrieval Analysis

Date: February 5, 2026 **Subject:** Transitioning from Baseline to Advanced Retrieval using Cross-Encoder Re-ranking

GitHub Repo: `https://github.com/anshium/alm-studio-sessions`

Note: The API key was compromised and is no longer valid now.

---

1. Bi-Encoders vs. Cross-Encoders: A Conceptual Comparison

The fundamental difference between these two architectures lies in how they process the relationship between a user query and the document chunks.

- **Bi-Encoders (The Baseline):** These models encode the query and the documents independently into fixed-sized vectors. Similarity is calculated using simple vector math, such as Cosine Similarity. This is computationally efficient and allows for pre-indexing documents, making it ideal for the initial retrieval from large datasets. However, because the model doesn't see the query and document together during encoding, it can miss fine-grained semantic nuances.
- **Cross-Encoders (The Advanced):** These models process the query and a document chunk simultaneously as a single input pair. This allows for full self-attention, where every token in the query can interact with every token in the document. This results in significantly higher accuracy in identifying relevance but at a much higher computational cost, as a full model inference is required for every query-document pair.

---

2. Qualitative Analysis of Ranking and Generation

For this activity, I used the technical document: `writing-best-practices-rag.pdf`.

Query: *"What are the best practices for handling tables and graphical information in source documents to improve RAG performance?"*

- **Observation:** The **Baseline Retrieval** identified 10 chunks related to "best practices" and "challenges" based on general vector similarity. While it successfully pulled chunks mentioning tables and graphics, the most direct answer from the "Documentation best practices" section was not necessarily the top result.
- **Re-ranking Impact:** The **Cross-Encoder** evaluated these 10 candidates and effectively moved the most relevant chunks from page 13 to the top of the list.
- **Final Answer Impact:** The final answer generated by Gemini Flash was highly specific: it correctly identified that table information should be presented in a **flat-level syntax** or bulleted lists to help models process information more easily. This precision demonstrates that the re-ranking step provided the LLM with the exact context needed to answer the specific technical question.

---

3. Latency Analysis: Baseline vs. Advanced

Efficiency is a critical trade-off when implementing advanced retrieval mechanisms.

Mechanism	Average Latency (Seconds)	Observation
Baseline Retrieval	**0.9604s**	Fast and efficient for initial filtering.
Advanced (Re-Ranking)	**1.0115s**	Adds ~0.0511s of overhead to process the top 10 candidates.

Conclusion: The advanced retrieval took slightly longer than the baseline. While the overhead for 10 chunks is minimal in this experiment, this latency would scale linearly in a production environment, highlighting the necessity of the initial Bi-Encoder filtering step.

---

4. Why not use a Cross-Encoder for the entire dataset?

While Cross-Encoders provide superior accuracy, they are not used for searching an entire dataset for several reasons:

1. **Computational Complexity:** A Cross-Encoder must run a full transformer inference for *every* document in the collection for *every* query. This is computationally prohibitive for large-scale datasets.
2. **Scalability:** Searching thousands or millions of documents with a Cross-Encoder would take seconds or minutes, whereas a Bi-Encoder combined with vector search can do it in milliseconds.
3. **The Hybrid Solution:** The industry standard is the "Two-Stage" paradigm: using a fast Bi-Encoder to narrow the field to the top k candidates and then using a "slower but smarter" Cross-Encoder to refine that small subset.