Short Description:

Hybrid Search and Re-Ranking: From Retrieval to Reliable Answers

As AI-powered search and RAG systems become central to modern applications, the ability to build retrieval that actually works is becoming a critical skill. This course empowers you to go beyond basic vector search by teaching you how to design, implement, and evaluate hybrid retrieval and re-ranking pipelines that deliver accurate, grounded answers.

Inside Content:

Course Overview

This course will teach you how to build reliable retrieval systems for RAG (Retrieval-Augmented Generation) applications using hybrid search and re-ranking. You will learn why pure vector search breaks in predictable and dangerous ways and how to fix it.

We will start by identifying the three structural failure modes of pure vector search: identifier confusion, precision leakage, and negation blindness. Then, we will explore hybrid search strategies that combine lexical and semantic signals to recover what vector search misses including Score Fusion, Result Union, and Filtered Semantic Search.

Next, we will implement three re-ranking techniques like heuristic, cross-encoder, and LLM-as-judge and learn when each approach is the right fit. We will then connect everything into a complete end-to-end RAG pipeline, from user query to generated answer, and see how ranking quality directly affects output quality.

Finally, we will build evaluation and debugging tools using standard retrieval metrics and a structured four gate diagnostic framework, giving you a repeatable way to measure and improve any retrieval system.

Course Objectives

By the end of this course, you will be able to:

- Identify why pure vector search fails and recognize when a system needs augmentation beyond embeddings
- Implement three hybrid search strategies and choose the right one based on recall vs. precision trade-offs
- Apply three re-ranking techniques (heuristic, cross-encoder, LLM-as-judge) and evaluate their trade-offs

- Build a complete RAG pipeline from retrieval through generation with proper context construction and prompt design
- Measure retrieval quality using Recall@K, Precision@K, MRR, Hit@K, and NDCG@K
- Debug RAG failures systematically using a layered diagnostic framework

Prerequisites

This course is designed for developers and learners with a basic understanding of what vector embeddings are and how RAG pipelines retrieve context for LLMs. No machine learning engineering experience is required. All code is provided in hands-on Jupyter Notebooks that you run step by step.

Syllabus

Hybrid Search and Re-Ranking: From Retrieval to Reliable Answers

Welcome to the Hybrid Search and Re-Ranking course for developers and AI practitioners! In today's RAG-driven landscape, retrieval quality and not model quality is the dominant failure mode in LLM-powered applications. This course empowers you to build retrieval systems that actually work by teaching you hybrid search strategies, re-ranking techniques, and structured evaluation methods. You will learn how to diagnose why a RAG system gives wrong answers and fix the root cause, from the retrieval layer up, not the answer down.

Learning Outcomes

By the end of this course, you will be able to:

Understand the three structural failure modes of pure vector search and why they are not fixable with better embeddings alone.

Design and implement hybrid search strategies that combine lexical and semantic signals for robust retrieval.

Apply re-ranking techniques, from hand-coded rules to learned models to LLM-as-judge and select the right approach for a given use case.

Build a complete end-to-end RAG pipeline with proper context construction, grounding instructions, and citation formatting.

Evaluate retrieval and ranking quality using standard metrics (Recall@K, Precision@K, MRR, Hit@K, NDCG@K).

Debug RAG failures systematically using a four-gate diagnostic framework.

Reason about trade-offs between cost, latency, recall, and precision when designing production retrieval systems.

Prerequisites:

Basic familiarity with what vector embeddings are and how RAG pipelines work.

Comfort running Python code in Jupyter Notebooks.

Curiosity about why AI search systems fail and how to fix them.

Course Syllabus

Chapter	Content	Hands-on
Chapter 1: Why Pure Vector Search Breaks	Understanding the three structural failure modes of vector search: identifier confusion, precision leakage, and negation blindness. Why these failures persist even with dense embeddings. Recognizing when a retrieval system needs augmentation beyond vector similarity.	Stress-test vector search on intentionally difficult queries and observe each failure mode firsthand.
Chapter 2: What Is Hybrid Search? + Hybrid Search Strategies	The core idea of hybrid search as a system design pattern. Three concrete strategies: Score Fusion (balanced), Result Union (recall-first), and Filtered Semantic Search (precision-first). The role of the alpha parameter, BM25, and Reciprocal Rank Fusion. Side-by-side comparison of all four systems on the same query.	Implement all three hybrid strategies, tune parameters, and compare outputs directly against pure vector search.
Chapter 3: Re-Ranking — Why Retrieval Is Not Enough	Why retrieval scores are not decision scores. Three re-ranking approaches: heuristic	Build all three re-rankers, run them on the same retrieved set, and compare rankings side by

	(rule-based), cross-encoder (learned model), and LLM-as-judge (instruction-following). Trade-offs between speed, accuracy, generalizability, and cost. Cross-encoder logit-scale scores and cross-attention mechanics.	side. Real cross-encoder model and real LLM API call included.
Chapter 4: From Retrieval to Generation — The Full RAG Flow	The five-stage RAG pipeline. Context construction with source attribution and token budgets. Prompt assembly with grounding instructions, citation rules, and refusal behavior. Positional bias and the "lost in the middle" problem. Negation and insufficiency testing.	Build a complete rag pipeline function. Compare generated answers from re-ranked vs. raw context. Test negation handling and context insufficiency with a real LLM.
Chapter 5: Evaluation and Debugging + Assignment	Standard retrieval metrics: Recall@K, Precision@K, MRR, Hit@K, NDCG@K. Context window stress testing. A reusable four-gate evaluation pipeline (Recall → Precision → Context Stability → Faithfulness). Multi-query test suites with aggregated metrics. The debugging hierarchy: always work upstream to downstream.	Run the full evaluation pipeline, build a test suite, and complete a capstone assignment designing a hybrid retrieval system for a real-world use case.