Information Retrieval Primer

Mat Kelly
mkelly@drexel.edu
Assistant Professor, Information Science
Drexel University College of Computing and Informatics

INFO825 Week 5
February 3, 2026

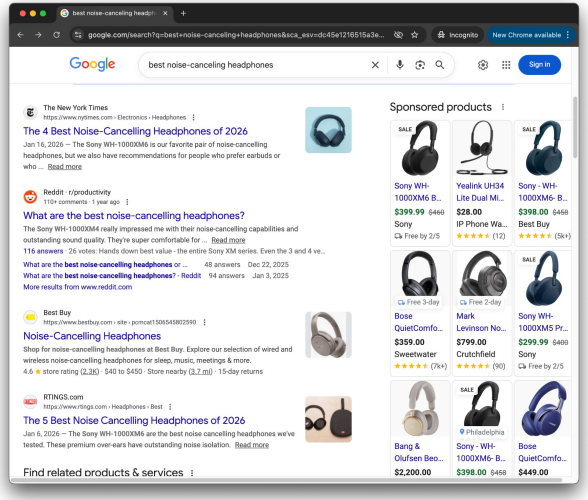
https://matkelly.com

Slides: https://matkelly.com/info825

What Is Information Retrieval?

- Finding useful information in large collections
- Handles unstructured or semi-structured data
- Optimizes for relevance, not exact correctness
- Powers search, recommendations, and assistants
- Unlike databases, IR assumes ambiguity and user intent

- Example: Google ranking web pages for:
  “best noise-canceling headphones”



IR vs Databases

- Databases: exact matches, structured schemas
- IR: approximate matches, free text
- SQL vs keyword queries
- Different evaluation criteria

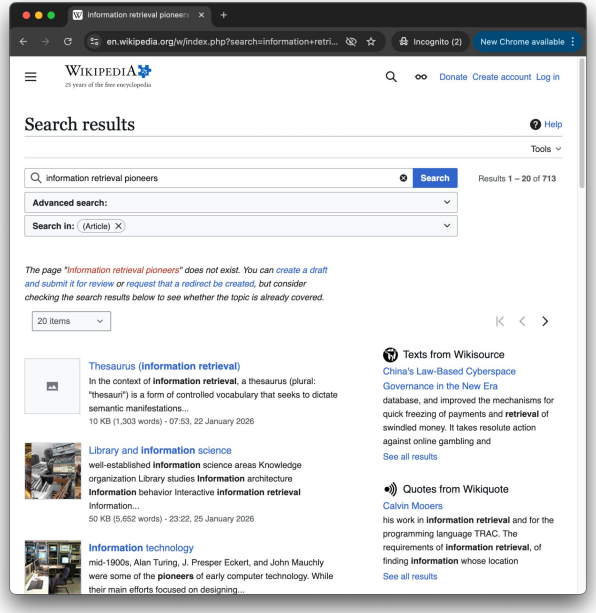
- IR trades exactness for flexibility and scale

SQL → `SELECT * FROM products WHERE price < 100`
IR → “cheap headphones with good bass”
LM → “I have a $100 budget. Recommend 3 headphones known for strong bass, explain why each is a good fit, list key tradeoffs (comfort, mic quality, wired/wireless), and end with your single best pick.”

End-to-End IR Pipeline

From raw text to ranked results

- Document collection
- Text processing
- Indexing
- Query processing
- Ranking
- Evaluation and feedback loops
- Example: Searching “information retrieval pioneers” on Wikipedia articles



- Ranking: orders results by estimated usefulness, not correctness

Text Processing

- Tokenization
- Normalization (case, punctuation)
- Stopword removal:
- removes words like *the*, *and*, *of*
- Stemming vs lemmatization

- Example: “Running runs ran” → run

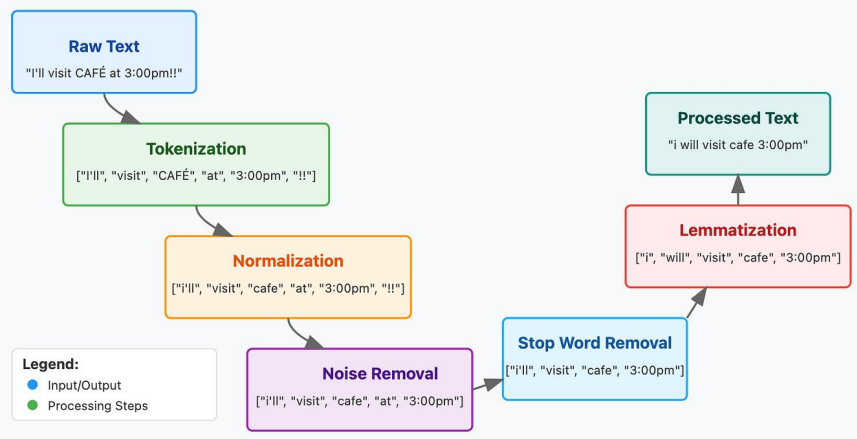


Image by Nishant Gupta

INDEXING

Why search is fast

The Inverted Index

- Core data structure of search engines
- Maps terms to documents
- Stores frequencies and positions
- Optimized for fast query-time access
- Enables fast lookup instead of scanning all documents
- Example:
  “search” → {D1, D3, D7}
  “engine” → {D1, D2}

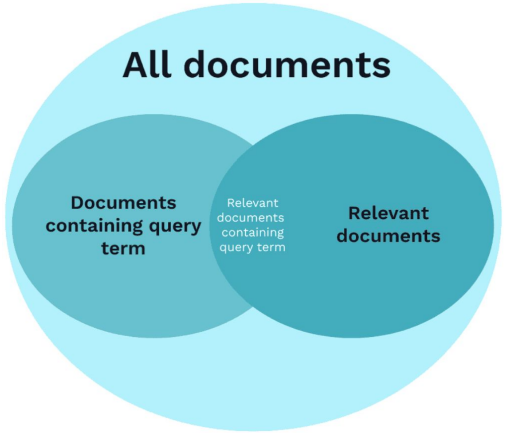
Why Indexing Matters

- Web-scale collections are massive
- Query-time efficiency is critical
- Enables ranking and filtering

Classical Ranking Models

Deciding what comes first

- Boolean retrieval
- TF–IDF
- Vector Space Model
- BM25 (industry standard)



The BM25 formula (annotated)

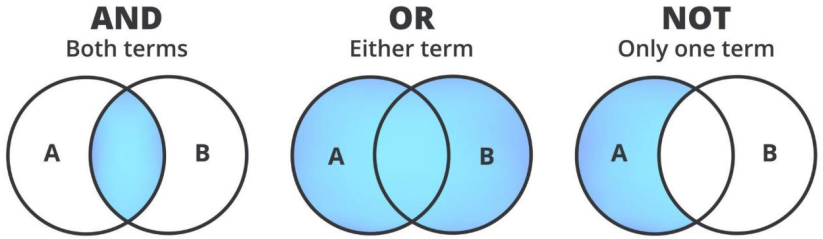
$$\text{score}(D,Q) = \sum_{t \in Q} \text{IDF}(t) \cdot \frac{f(t,D) \cdot (k_1 + 1)}{f(t,D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}$$

What each symbol means

- $f(t, D)$ → how many times term *t* appears in document *D*
- $|D|$ → document length
- **avgdl** → average document length in the corpus
- **k_1** → term frequency saturation (usually ~1.2–2.0)
- **b** → length normalization strength (usually ~0.75)

Boolean Retrieval in IR

- Classic Information Retrieval (IR) model
- Documents represented as sets of terms
- Queries expressed using Boolean logic
- Uses AND, OR, NOT operators
- Documents either match or do not match
- No notion of partial relevance or ranking



How Boolean Retrieval Works

- Build an inverted index (term → document list)
- Parse and evaluate the Boolean query
- Apply AND, OR, NOT on posting lists
- Returns an unordered set of matching documents
- No ranking or relevance scoring
  - A set of relevant documents is returned that satisfies the query

Worked Boolean Query Example

- Query: (information AND retrieval) NOT web
- Documents:
  - D1: information retrieval systems
  - D2: web information retrieval
  - D3: information extraction
- Result: D1 (matches query logic)

TF-IDF in Information Retrieval

- TF-IDF = Term Frequency × Inverse Document Frequency
- Weights terms by importance to a document
- Common words get low weight even if frequent
  - (A word appearing in every document carries little information)
- High TF → frequent in the document
- High IDF → rare across the collection
- Used in vector space models for ranking

Intuition and Use of TF-IDF

- Boosts terms frequent in one document but rare overall
- Downweights common and stop words
- Documents ranked by similarity (e.g., cosine similarity)
- Simple, effective, and widely used in IR
- Foundation for many modern retrieval systems

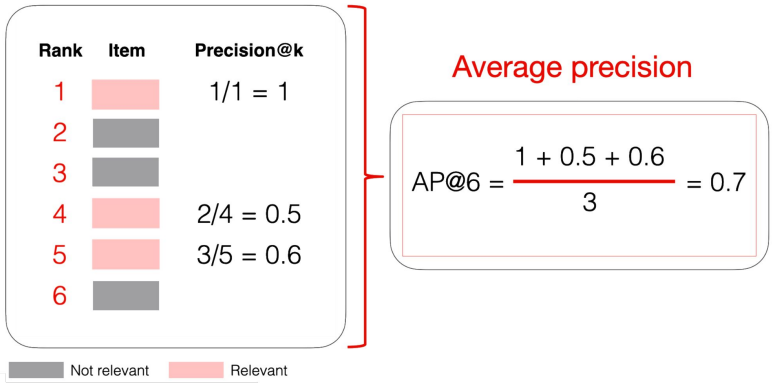
Ranking Intuition

- Term frequency indicates importance
- Rare terms are more informative
- Documents become vectors
- Similarity determines rank
- Documents closer to the query vector rank higher

Evaluating IR Systems

Is your search engine any good?

- Relevance judgments
- Precision vs Recall
  - Precision: “How many results were useful?”
  - Recall: “Did we miss anything important?”
- Precision@k
- MAP and NDCG
- Retrieval quality bounds generation quality



What Hasn’t Changed Since Classic IR

Why IR Is Still a Research Field

- A query represents an information need
- Documents are imperfect proxies for knowledge
- Relevance is *estimated*, not *computed*
- Ranking is always approximate
- Evaluation is task-dependent
- Every system is an approximation under uncertainty

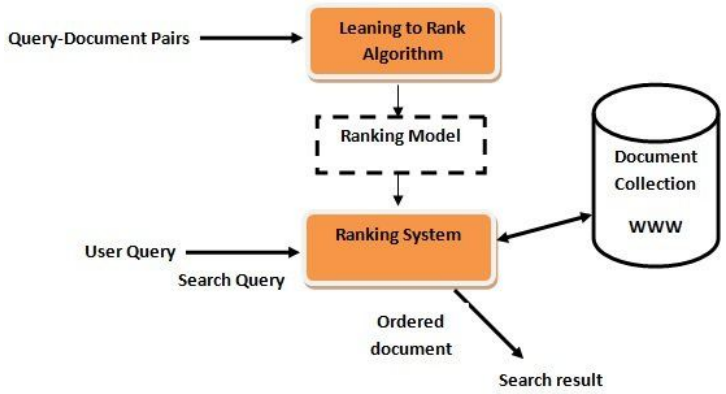
Why IR Still Matters in the Age of LLMs

Search meets machine learning

- LLMs do not know facts, they predict text
- Retrieval determines what information the model sees
- Most real AI systems are retrieval-first, generation-second
- IR controls cost, accuracy, bias, and freshness

Learning to Rank

- Uses features from documents and queries
- Trained on user interaction data
  - e.g., Clicks, dwell time, skips, explicit relevance labels
- Optimizes ranking directly
- Learning paradigms:
  - Pointwise (score each document)
  - Pairwise (learn preferences between documents)
  - Listwise (optimize whole rankings)
- Potential features:
  - BM25 score
  - Query–document term overlap
  - Document length
  - Freshness / popularity



src: An evolutionary strategy with machine learning for learning to rank in information retrieval by Ibrahim and Landa-Silva, 2018

Neural Retrieval

A New Representation, Same Problem

- Documents represented as dense vectors (embeddings)
- Queries embedded into the same vector space
- Similarity ≠ shared keywords
- Captures paraphrase and semantic equivalence
- Still solves the same IR task: match a query to relevant documents
  - Ranking remains the central decision

Concrete Intuition

How Neural Search Differs from BM25

- BM25: exact term overlap
- Neural: meaning overlap
- “heart attack” ≈ “myocardial infarction”
- “cheap flights” ≈ “low-cost airfare”

IR in AI Systems

- Retrieval-Augmented Generation (RAG)
- Chatbots grounded in documents
- Enterprise and academic search
- Retrieved documents ground LLM responses in facts
  - e.g., Prevents hallucination by citing source material

Retrieval-Augmented Generation (RAG)

- RAG: IR as Context Selection
- Pipeline: Query → Retrieve → Rank → Generate
  - User query
  - Retrieve relevant documents (IR)
  - Inject documents into prompt
  - LLM generates grounded answer
- LLM + external knowledge source
- Retrieval selects context
- Generation conditions on retrieved text
- No fine-tuning required
- Important: Retrieval happens before generation

RAG as an IR System

- Retrieval selects what knowledge is visible
- Generation is downstream of ranking
- Poor IR guarantees poor answers
- RAG reframes IR from “results to users” to “results to models”

- Classic IR: query → ranked documents → human 🧑
- RAG: query → ranked documents → model 🤖
- Same failure modes, different consumer
- IR handles finding relevant evidence, RAG handles synthesis and language generation

Why RAG Exists

- Reduces hallucination
  - Grounds generation in retrieved documents
- Keeps knowledge up-to-date
  - Update data without retraining the model
- Improves transparency
  - Answers can cite sources
- Scales better than fine-tuning
  - Cheaper, faster, modular
- RAG separates knowledge from reasoning
- Rationale: Don’t make the model remember, let it look things up

RAG Failure Modes

- Poor retrieval → confident wrong answers
- Semantic drift in embeddings
- Context window limitations
- Bias amplification via ranking

Why IR Evaluation Still Applies to RAG

- Retrieval metrics still matter
- End-to-end answer correctness
- Faithfulness to retrieved sources
- Human-in-the-loop evaluation

Open Evaluation Questions

- How do we measure faithfulness automatically?
- How much retrieval error is tolerable?
- When does reranking help vs harm?
- What does “relevance” mean for generated text?

Dense Vector Embeddings

- Text mapped to high-dimensional vectors (e.g., 768-4096 dimensions)
- Learned via contrastive or self-supervised training
- Distance ≈ semantic similarity
- Enables nearest-neighbor search at scale

- Query: “effects of climate change”
  - nearest neighbors include documents without the word effects

768 is common for BERT encoders while larger models will have more dims for often better semantic resolution (upward of **4096**)

Neural IR Pipeline

1. Encode documents into embeddings (offline)
2. Store vectors in a vector index
3. Encode query at search time
4. Retrieve top-k nearest vectors
5. (Optional) re-rank with a neural model

APPLIED IR

How this works in practice

Industry Tools & Demos

- Elasticsearch / OpenSearch
- Lucene
- Vector databases
- Hybrid pipelines in production

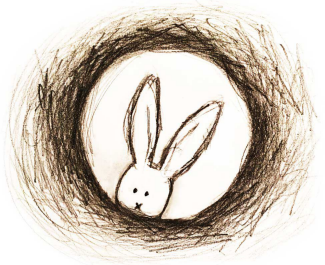
Hybrid Retrieval (Industry Standard)

- Sparse retrieval (BM25) for recall
- Dense retrieval for semantic matching
- Neural re-ranking for precision
- Combines robustness + meaning
- Hybrid systems exist because no single relevance signal is sufficient
  - Combines multiple relevance signals: a core IR idea since the 1970s
- (Pure neural systems rarely operate alone)

Example:

- Elasticsearch + vector search + cross-encoder

Ethics & Failure Modes



When ranking has consequences

- Bias → embedding bias, source selection
- Opacity → opaque similarity, ranking logic
- Power → authority amplification, filter bubbles
- Downstream harm → hiring, credit, policing, medical triage, education
- Feedback loops → popularity bias, rich-get-richer dynamics
  - High-ranked items get more clicks
  - Clicks become training data
  - Model reinforces its own past decisions

Summary

- IR formalizes how systems estimate usefulness under uncertainty
- Indexing and ranking are the irreducible core
- Neural methods change representations, not goals
- RAG changes who consumes retrieval, not why retrieval exists
- Evaluation and ethics remain unsolved (by design)