CS60092: Information Retrieval

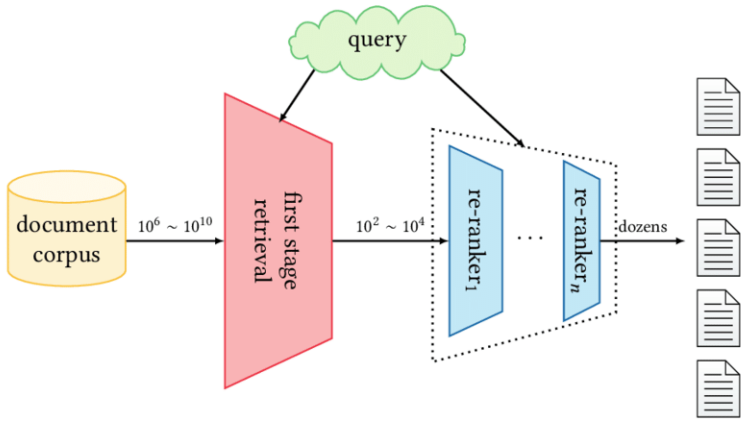
PageRank, Latent Semantic Indexing, and Learning to Rank

Debaditya Roy

What Is the Ranking Problem?

Given

- A **query** (e.g., *"machine learning courses"*)
- A **large document collection**

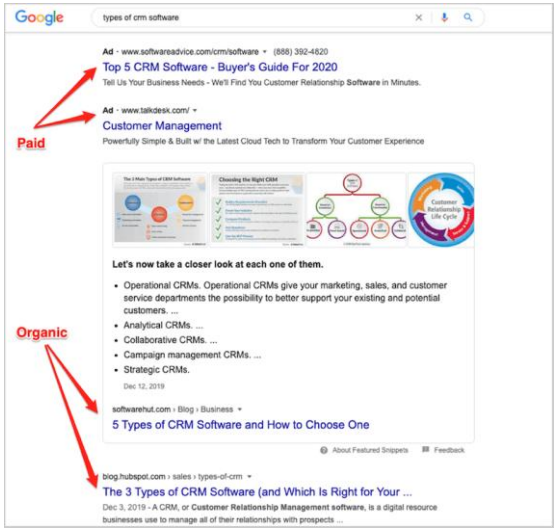


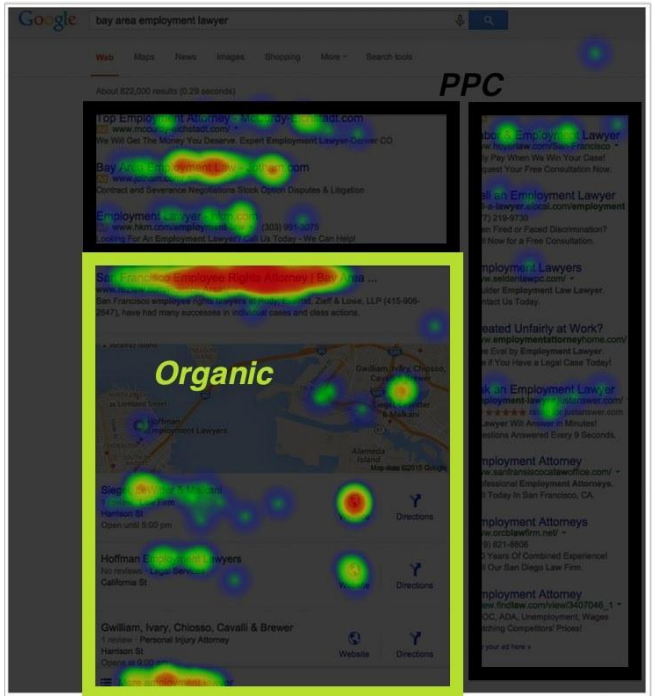
We must

- Order documents so that **the most useful ones appear first**

Why ranking matters?

- Users rarely look past the top 5-10 results
- Ranking errors directly impact
  - User satisfaction
  - Revenue (ads, e-commerce)
  - Trust in the system





PageRank – Authority from Links

Intuition

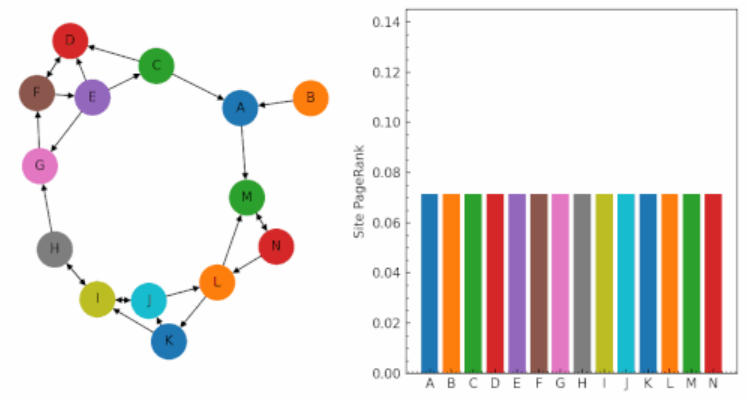
The web behaves like a **citation network**:

- A link is an implicit endorsement
- Being linked by important pages matters more

Human analogy

Academic papers:

- A paper cited by *Nature* is more influential than one cited by an obscure blog

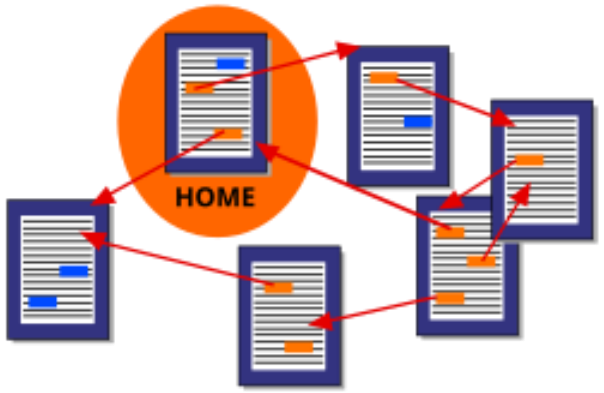


https://en.wikipedia.org/wiki/PageRank

Random Surfer Mental Model

Imagine a user who:

- Starts on a random page
- Keeps clicking links randomly
- Occasionally jumps to a random page



PageRank = probability the surfer is on a page in the long run

https://en.wikipedia.org/wiki/Random_surfing_model

Web Graph as a Markov Chain

- $G = (V, E)$ be the directed web graph
- $N =$| V | number of nodes
- L_i = number of outgoing links from node i

Define transition matrix $P \in \mathbb{R}^{N \times N}$

$$P_{ij} = \begin{cases} \frac{1}{L_i} & \text{if } i \rightarrow j \\ 0 & \text{otherwise} \end{cases}$$

- Each row sums to 1 → **row-stochastic matrix**.

The Teleportation Fix

Real web graphs are

- Not strongly connected
- Contain dangling nodes
- Modify P to avoid dead ends

$$G = \alpha P + (1 - \alpha)\mathbf{1}v^{\mathsf{T}}$$

- $\alpha \in (0,1)$ (typically 0.85)
- v = teleportation distribution
- $\mathbf{1}v^{\mathsf{T}}$ makes every row equal to v^{T}
- With probability α, follow links
- With probability $1 - \alpha$, teleport

PageRank Equation

Let $r^{(t)}$ be the probability distribution over pages at time t

- Update rule: $r^{(t+1)} = r^{(t)}G$
- A **stationary distribution** satisfies $r = rG$
- If the system is in distribution r, applying one transition step leaves it unchanged

$$\therefore r = \alpha rP + (1-\alpha)v$$

- PageRank solves

$$(I - \alpha P^{\top})r = (1-\alpha)v \qquad \Rightarrow \qquad r = G^{\top}r$$

- r is the eigenvector of $G^{\top}$ with eigenvalue 1

What PageRank Really Computes

A **global importance score**

- Independent of any query
- Based only on graph structure

Why it was revolutionary

- Before PageRank:
  - Search engines ranked mostly by keyword frequency
  - Easy to spam
    - e.g., repeat keywords with white text on a white background
- After PageRank:
  - Importance was *earned*, not faked easily

What PageRank cannot compute

- Does **not understand content**
- Cannot distinguish *why* a page is relevant
- **Only 1 signal**

This leads us to **semantic relevance**.

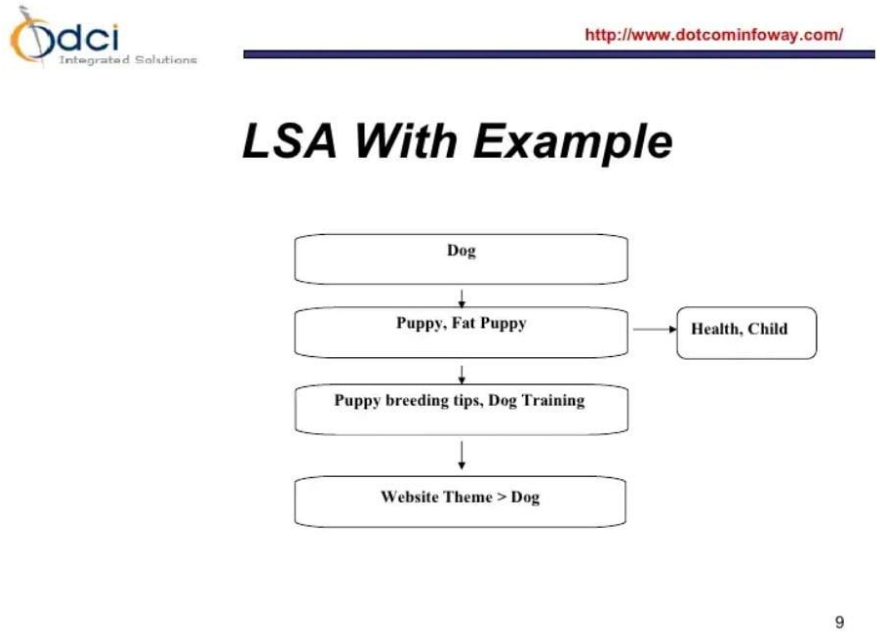
Latent Semantic Indexing – Meaning Beyond Words

The Vocabulary Mismatch Problem

Users say:
“automobile repair”

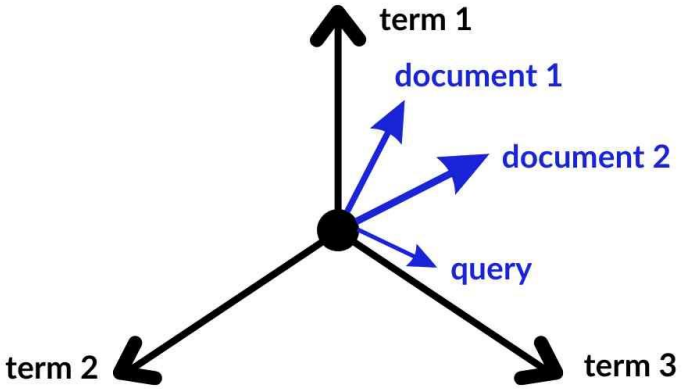
Documents say:
“car engine maintenance”

Keyword matching fails but humans see them as related



Term-Document Space

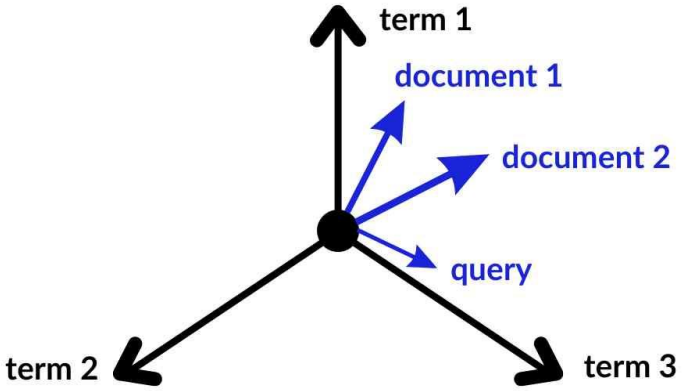
- Each document is a point in a **very high-dimensional space**
- Each dimension = a term
- Space is
  - Sparse
  - Noisy
  - Redundant



https://spotintelligence.com/2023/09/07/vector-space-model/

Term-Document Matrix

- $X \in \mathbb{R}^{m \times n}$
- m terms
- n documents
- $X_{ij} = \text{TF–IDF}(t_i, d_j)$



https://spotintelligence.com/2023/09/07/vector-space-model/

$$\text{TF}(t, d) = \frac{\text{Number of times term t appears in document d}}{\text{Total number of terms in document d}}$$

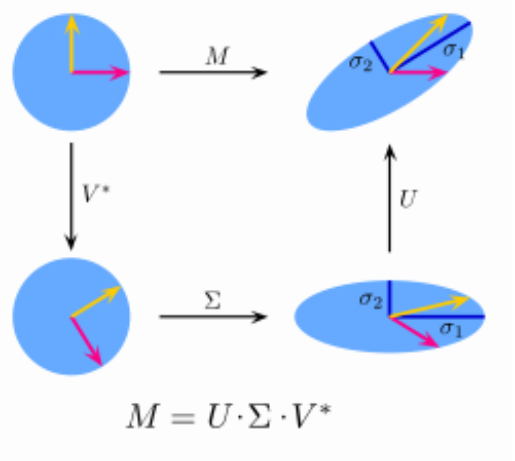
$$\text{IDF}(t, D) = \log \frac{\text{Total number of documents in corpus D}}{\text{Number of documents containing term t}}$$

$$\text{TF - IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$$

What LSI Does

1. Rotates the space
2. Keeps only the most important directions
3. Merges correlated terms into **latent concepts**

Example latent concept:
“vehicles” ← car, automobile, engine, repair



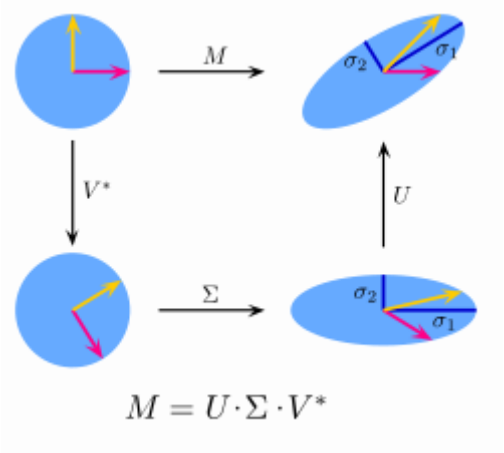
Why This Works

- Documents about similar topics cluster together
- Queries retrieve documents even without exact word matches

Singular Value Decomposition

$$X = U\Sigma V^{\top}$$

- $U \in \mathbb{R}^{m \times m}$
- $V \in \mathbb{R}^{n \times n}$
- $\Sigma = \mathrm{diag}(\sigma_1 \geq \sigma_2 \geq \cdots)$



Rank-k Approximation

$$X_k = U_k \Sigma_k V_k^{\top}$$

$$k < rank(X)$$

Query Projection

- Given query vector $q \in \mathbb{R}^m$
- Project into latent space:

$$q' = \Sigma_k^{-1} U_k^\top q$$

- Document representation:

$$d_j' = \Sigma_k V_k^\top (:, j)$$

- Similarity:

$$\text{sim}(q, d_j) = \frac{q'^\top d_j'}{\|q'\| \|d_j'\|}$$

Limitations of LSI

- Linear
- Cannot model context (“bank” of river vs finance)
- Does not learn from user behavior

How do we combine

- Authority (PageRank)
- Semantics (LSI)
- User feedback?

Learning to Rank - Learning What Matters

Instead of asking

“What formula should we use?”

We ask

“What ordering do users prefer?”

Learning to Rank - Learning What Matters

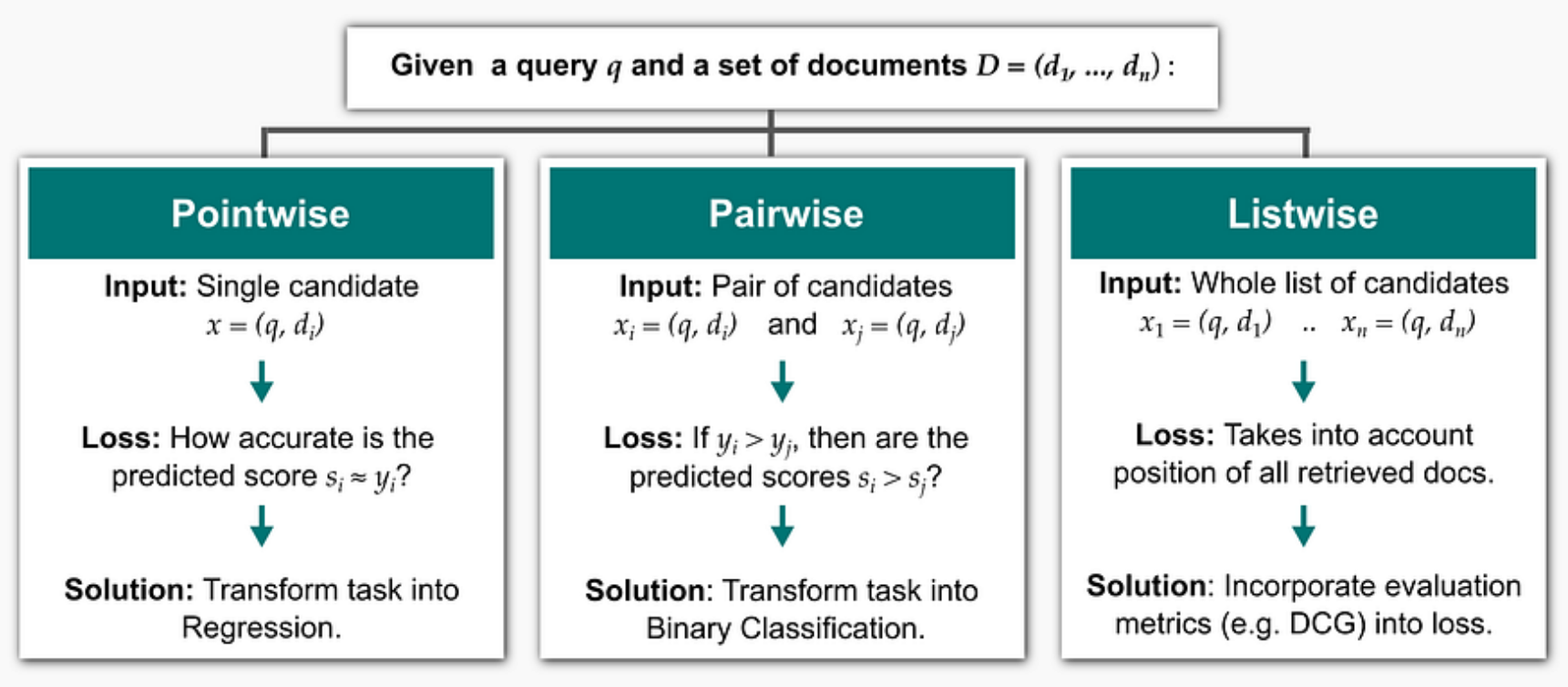
Training Signal

- Human relevance judgments
- Clicks
- Dwell time

We learn

- How to **combine signals automatically**

Learning to Rank Types



https://medium.com/data-science/learning-to-rank-a-complete-guide-to-ranking-using-machine-learning-4c9688d370d4

Pointwise Learning to Rank

- Feature map: $x_{qi} = \phi(q, d_i)$
- Scoring function: $f(q, d_i) = w^\top x_{qi}$
- **Pointwise Learning**
  - Regression problem

$$\min_w \sum_{q,i} \left(y_{qi} - w^\top x_{qi}\right)^2$$

Pairwise Learning to Rank

- Feature map: $x_{qi} = \phi(q, d_i)$
- Scoring function: $f(q, d_i) = w^\top x_{qi}$
- **Pairwise Learning**
  - Preference pairs: $P(d_i \succ d_j) = \frac{1}{1+e^{-(f(q,d_i)-f(q,d_i))}}$
  - Binary Cross-entropy Loss **(RankNet)**

$$\mathcal{L} = -\sum_{(i,j)} \left[y_{ij} \log P_{ij} + (1 - y_{ij}) \log(1 - P_{ij}) \right]$$

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. (2005, August). Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning* (pp. 89-96).

Listwise Learning to Rank

- Feature map: $x_{qi} = \phi(q, d_i)$
- Scoring function: $f(q, d_i) = w^\top x_{qi}$
- **Listwise Learning**
  - Define probability over permutations: $P_i = \frac{e^{f_i}}{\sum_j e^{f_j}}$
  - Softmax Cross-Entropy Loss

$$\mathcal{L} = -\sum_i P_i^* \log P_i$$

where P_i^* is derived from relevance labels

Where Classical Models End

Learning to Rank:

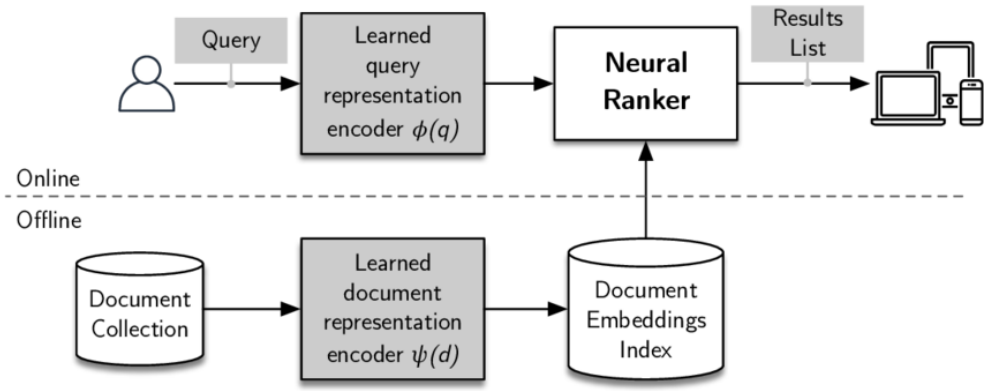
- Still relies on **hand-engineered features**
  - Human relevance judgments
  - Clicks
  - Dwell time
- Cannot deeply understand language

This motivates **neural ranking**

Neural Ranking = Representation + Ranking Together

Neural models

- Learn **representations**
- Learn **ranking**
- Learn **interaction**
  → all jointly



https://www.researchgate.net/figure/Dense-retrieval-architecture-for-representation-focused-neural-IR-systems_fig5_362300551

How Neural Ranking Thinks?

Instead of Counting words

Neural Ranker

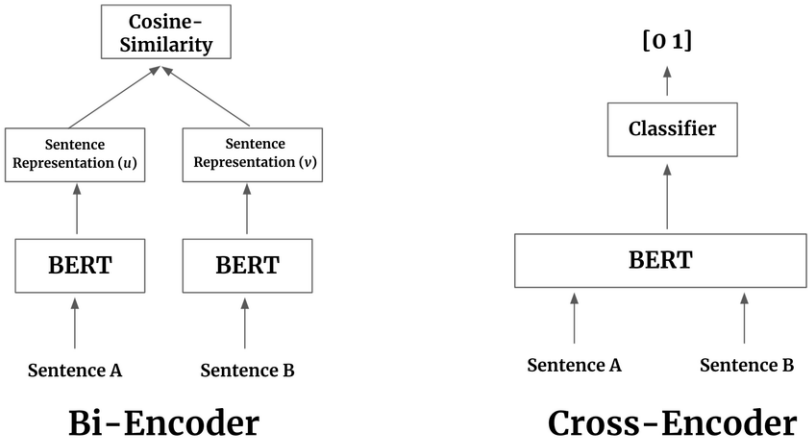
- Compares meaning
- Model context
- Understands word roles

Example:

"Apple stock price"

vs.

"Apple fruit price"



https://www.researchgate.net/figure/Difference-between-Bi-Encoder-and-Cross-Encoder-architectures_fig7_364814907

Neural Ranking

Neural models replace

$$f(q, d) = w^\top x$$

with

$$f(q, d) = g_\theta(q, d)$$

where

- g_θ is a neural network
- θ learned via gradient descent

Representation-Based Model

- Encode separately

$$q' = f_\theta(q)\ d' = f_\theta(d)$$

- Score:

$$f(q, d) = q'^{\top} d'$$

- Training with contrastive loss:

$$L = -\log \frac{e^{q'd^+}}{e^{q'd^+} + \sum_{d^-} e^{q'd^-}}$$

Interaction-Based Models

- Encode together

$$f(q,d) = Transformer_{\theta}([q;d])$$

```
[CLS] query tokens [SEP] document tokens [SEP]
```

- Transformer layer

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^{\top}}{\sqrt{d}}\right)V$$

- Tokens from query attend to document tokens and vice versa.
- Final score:

$$f(q,d) = W^{\top}h_{[\text{CLS}]}$$

Properties of Interaction-Based Models

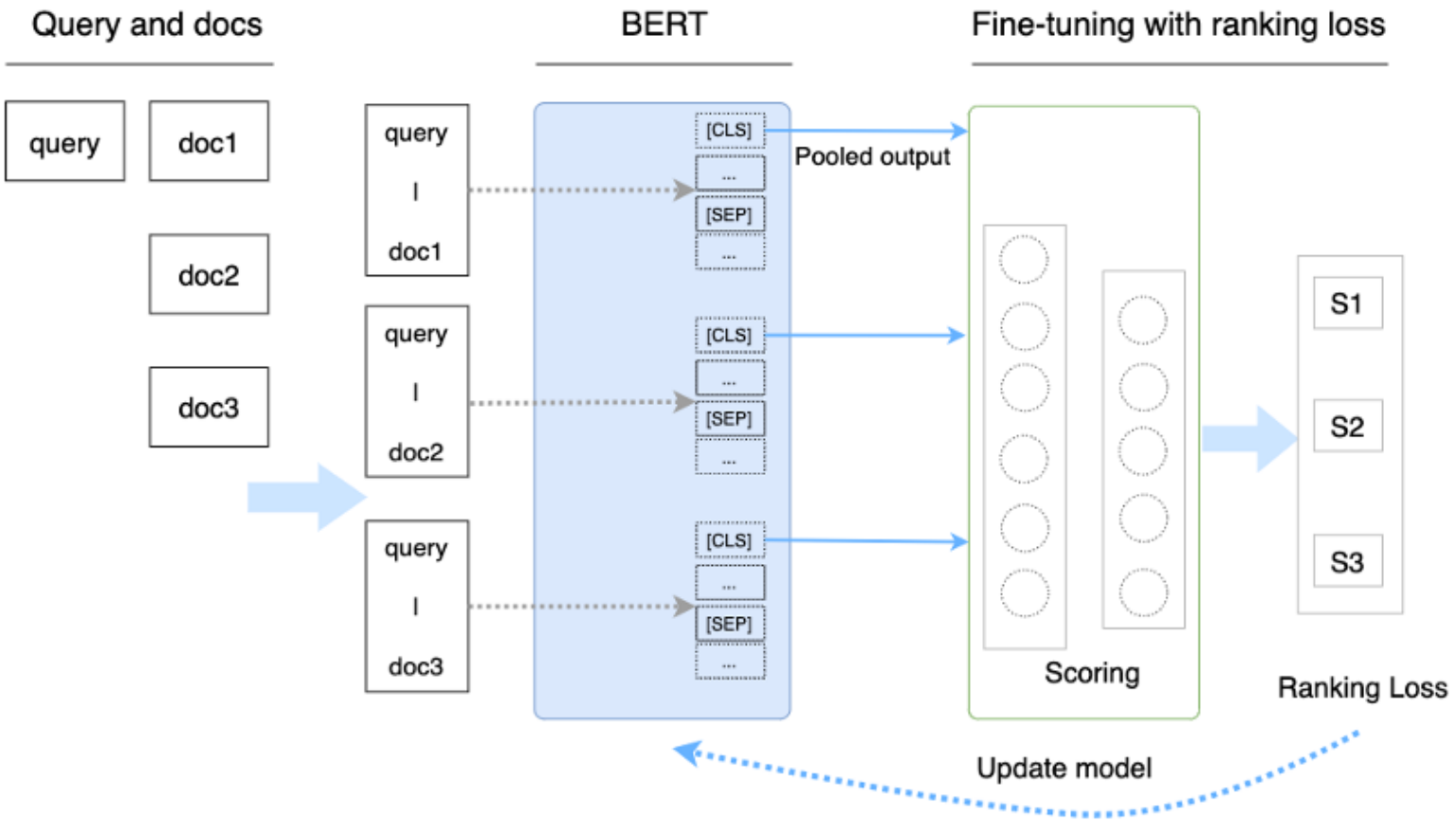
- Full token-level interaction
- Captures
  - Exact matches
  - Phrase alignment
  - Context
  - Polysemy

Complexity

$$O\left((|q| + |d|)^2\right)$$

Not scalable for large retrieval — only usable for **reranking**.

BERT based reranking



Han, S., Wang, X., Bendersky, M., & Najork, M. (2020). Learning-to-Rank with BERT in TF-Ranking. *arXiv preprint arXiv:2004.08476*.

Late Interaction Models

Encode separately

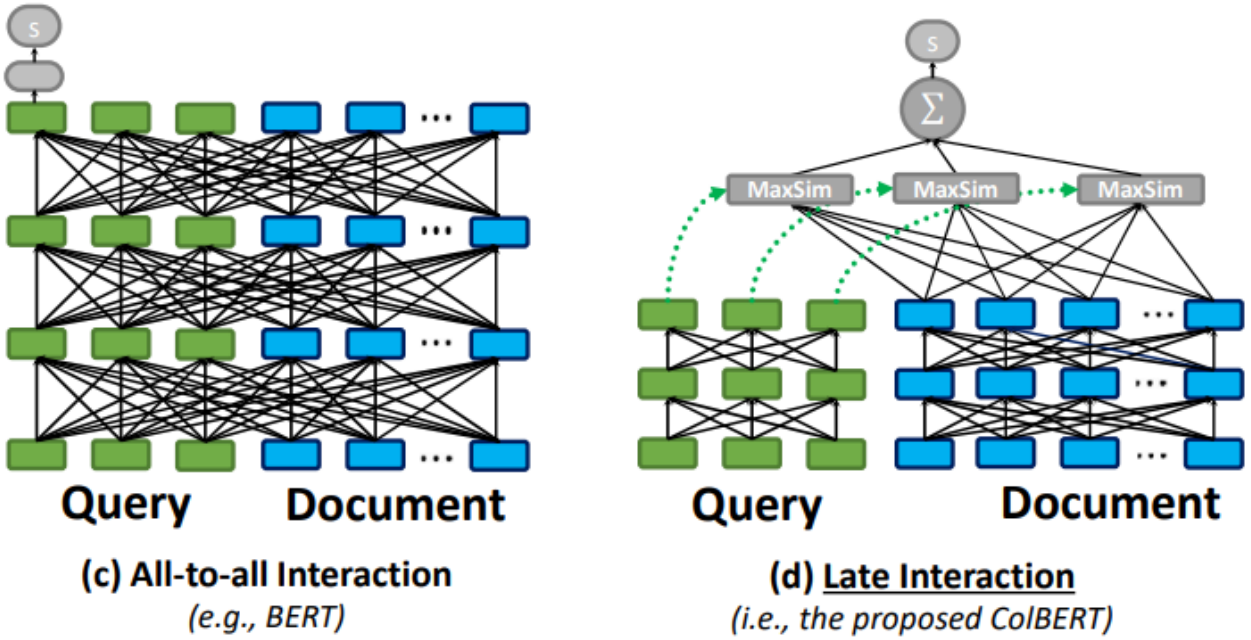
$$q' = \{q'_1, \dots, q'_m\}$$
$$d' = \{d'_1, \dots, d'_n\}$$

Score

$$f(q, d) = \sum_{i=1}^{m} \max_j \left({q'_i}^\top d'_j \right)$$

Each query token finds its best matching document token

Late Interaction Models



Complexity

Offline: Precompute document embeddings

Online: $O(m \cdot n)$ →Much cheaper than cross-encoder.

Khattab, O., & Zaharia, M. (2020, July). Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval* (pp. 39-48).

Conceptual Unification

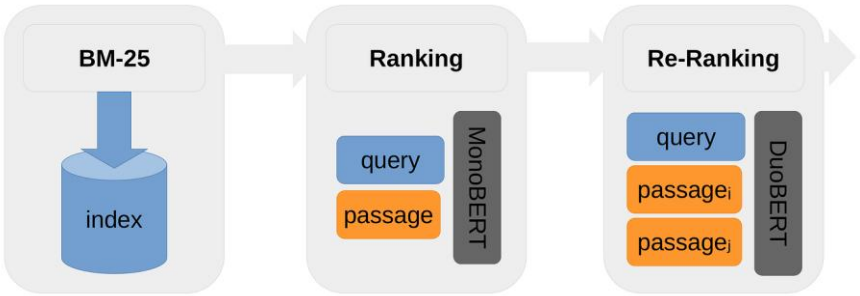
All ranking methods optimize

$$\min_{\theta} \mathbb{E}_q[\mathcal{L}(\text{ordering under } f_\theta)]$$

Method	Representation	Objective
PageRank	Graph eigenvector	Stationary distribution
LSI	Linear SVD	Frobenius norm minimization
LTR	Linear model	Hinge / logistic
Neural	Deep networks	Cross-entropy / contrastive

Modern Search Pipeline

- Fast retrieval (BM25 / dense embeddings)
- Learning-to-rank or neural reranking
- PageRank-like signals as priors



https://itnext.io/deep-learning-in-information-retrieval-part-iii-ranking-da511f2dc325

References

Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze
Introduction to Information Retrieval
Cambridge University Press, 2008

Tie-Yan Liu
Learning to Rank for Information Retrieval
Springer, 2011

Topic	Book	Chapter
PageRank	Manning et al.	Ch. 21
LSI	Manning et al.	Ch. 14
Classical LTR	Liu	Ch. 3–6
Neural ranking losses	Liu	Ch. 4–5