CADT

Cambodia Academy of Digital Technology

# Document/Text Representation

# TF-IDF, Vector Space Model, and BM25

**Lecturer**

Mr. Natt KORAT

*Lecturer-Researcher,*

*Institute of Digital Research & Innovation (IDRI),*

*Cambodia CADT Academy of Digital Technology (CADT)*

# Recap: what boolean retrieval cannot do

- **Lab 2:** Boolean **OR** for “travel Phnom Penh Siem Reap” returned 4 documents, unordered.
- Only 1 of them is truly about that trip.
- An unordered set can’t tell us which.

# Why ranking matters?

- **Real-world scale**: Real collections return thousands or millions of Boolean matches.
- **User behavior**: End users rarely look beyond the first few search results.
- **Core IR problem**: We must order documents by ESTIMATED relevance relative to the user's information need.

# Term frequency (TF)

- How many times a term (*t*) appears in a single document (*d*)

$$tf(t, d)$$

- **Core intuition**: A higher term frequncy suggests that the document is more about that specific concept.
- Example:
  - Doc *A* mentions “python” 1 time.
  - Doc *B* mention “python” 8 times.
  - => Document *B* is likely more relevant to a query on Python.

# The problem with raw TF

- **Ubiquitous words**: Common words (like stop words) or generic domain terms (e.g., offer, system) appear frequently across many documents.
- **Lack of discrimintaiton**: High *tf* in a single document is not special if the term appears everywhere across the corpus.
- **Need for balance**: We must scale term inpact based on how information/rare the term is overall.

# Document frequency (DF)

- The number of documents in the collection *N* that contain term *t* at least once.

$$df(t)$$

- Example:
  - Doc *A* mentions “python” 1 time.
  - Doc *B* mention “python” 8 times.
  - => $df(\text{“}python\text{”}) = 2$

# Inverse document frequency (IDF)

$$idf(t) = \log\left(\frac{N}{df(t)}\right)$$

- N = Total number of documents in the corpus
- Weight effect:
  - Rare terms -> small *df* -> **HIGH** *idf* weight.
  - Common terms -> Large *df* -> **LOW** *idf* weight.

# TF-IDF weight

- Combine formula:

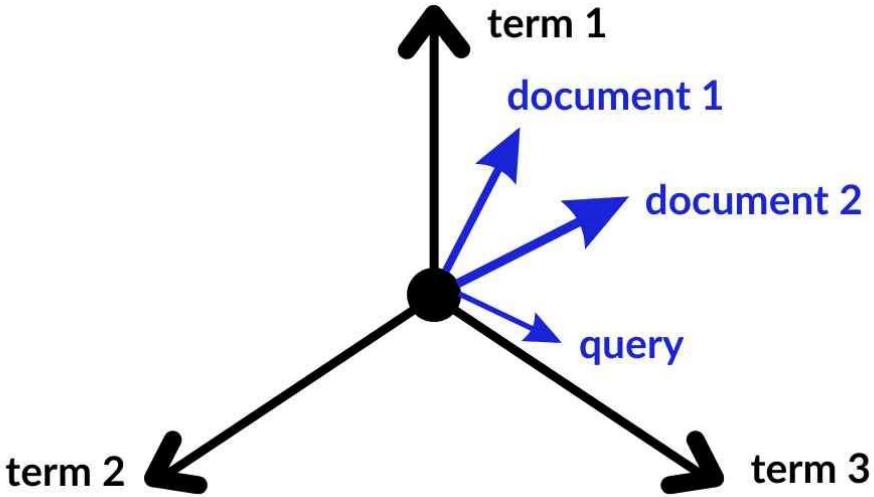
$$weight(t,d) = tf(t,d) \times idf(t)$$

- *Characteristics of a High TF-IDF score:*

1. High term frequency tf in this specific document.
2. Low document frequency df across the entire collection.

# Documents and queries as vectors

- *Each unique term in the collection = one dimension.*
- *A document's vector: its TF-IDF weight in each dimension (0 if the term is absent).*
- *A query is just a very short "document".*




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

# Cosine similarity

- **Geometric score**: Measure the ANGLE between two vectors, not their length.

$$\cos(q, d) = \frac{q \cdot d}{\|q\| \times \|d\|}$$

- ***Score scale**:*
  - *1.0* = Same direction (very similar).
  - *0.0* = *Orthogonal (no shared terms)*.

$$\|x\| = \sqrt{\sum_{i=1}^{n} {x_i}^2}$$

# Ranked Retrieval

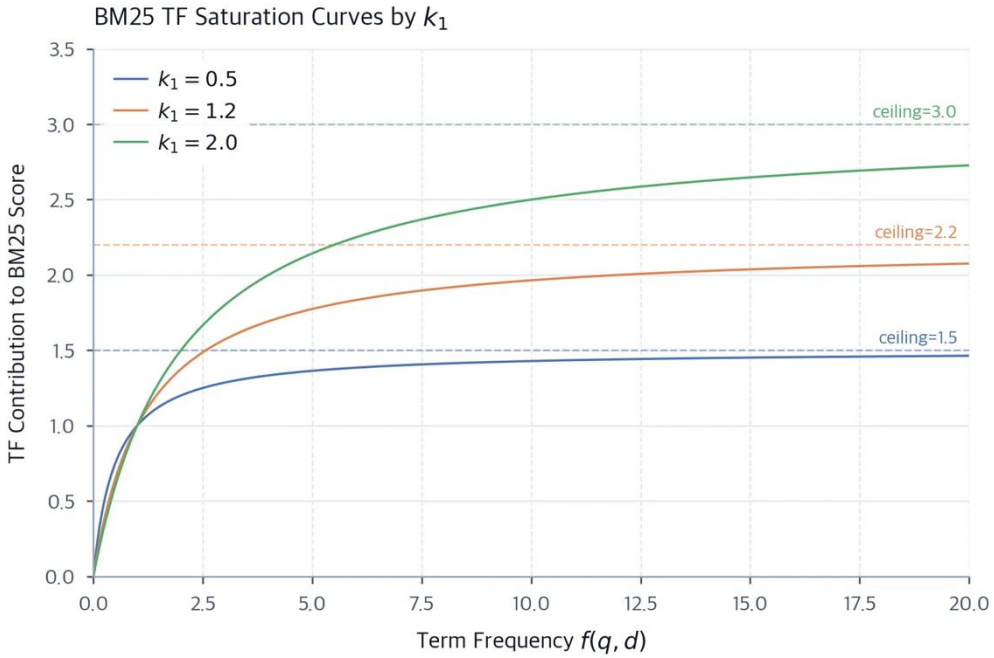
- **Pipeline:** Score every document -> sort by score -> return top *k*.
- ***Key advantage****:*
  - *No more hard Boolean matching*
  - *Every document receives a continuous real-valued score.*

# Limites of TF-IDF

- TF-IDF grows linearly with term frequency.
- In practice, repetition matters less and less after the first few occurrences.

# BM25: Term saturation

- **Concept**: Score grows with *tf* but LEVEL OFF (diminishing returns).
- **Parameter $k_1$**:
  - Controls saturation speed (typical value: 1.2 – 2.0).
  - Higher $k_1$ = slower saturation (repetion count longer).




Source: https://mbrenndoerfer.com/writing/hybrid-search-bm25-dense-retrieval-fusion

# BM25: Length normalization

- **Principle:** A term match in a **SHORT** document is stronger evidence than in a **LONG** one.
- **Parameter** $b$:
  - Control length normaization (0 = no norm, 1 = full norm; default 0.75).
- **Mechanism**: Evaluate document length $|d|$ realtive to average length $avgdl$ across the corpus.

# BM25: The BM25 formula

$$Score(q,d) = \sum_{t \in q} idf_{BM25}(t) \times \frac{tf(t,d) \cdot (k_1 + 1)}{tf(t,d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{avgdl}\right)}$$

**Wikipedia:** https://en.wikipedia.org/wiki/Okapi_BM25

# BM25: The BM25 formula

- **Components**:

1. $idf_{BM25}(t)$: Inverse document frequency variant.

$$idf_{BM25}(t) = log\left(\frac{N\ - df(t) + 0.5}{df(t) + 0.5} + 1\right)$$

2. $tf\,(t, d)$: Raw term frequency.

3. $\frac{|d|}{avgdl}$: Ratio of document length to collection average.

**Wikipedia**: https://en.wikipedia.org/wiki/Okapi_BM25

# Short documents, Zipf's law

- A few terms are extremely frequent, most are rare (**Zipf's law**) -> This is why $idf$ works at all
- Very short documents/queries make length normalization (b) especially important.

# Recap

1. TF-IDF $= tf \times idf$, compared by consine similarity
2. BM25 saturates term frequency and normalize for document length
3. Both beat Boolean retrieval by producing a RANKED list
4. Neither model fixes vocabulary mismatch