



CSCE 670 - Information Storage and Retrieval

Week 2: TF-IDF; Vector Space Model; BM25

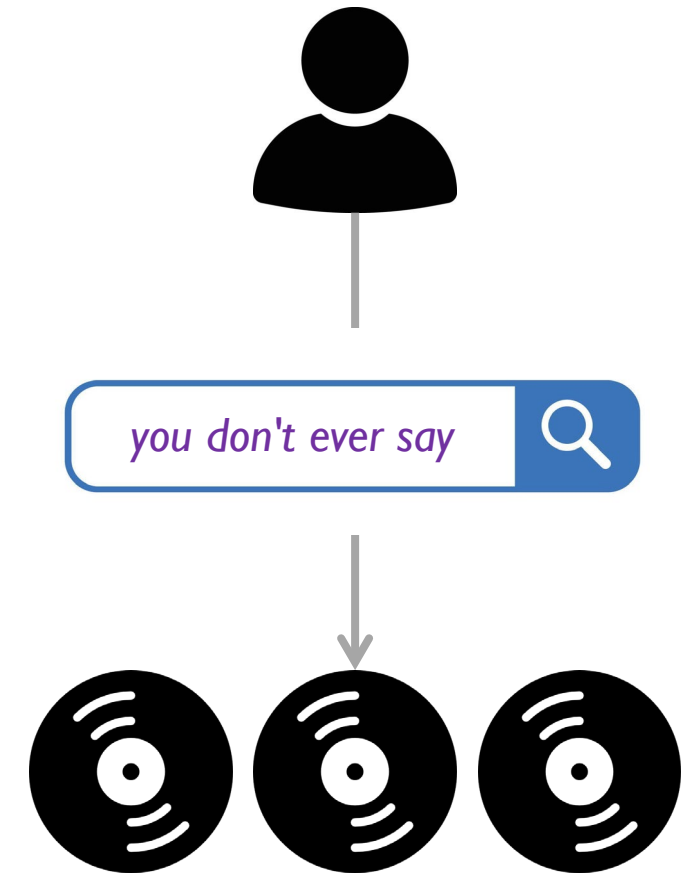
Yu Zhang

yuzhang@tamu.edu

Course Website: <https://yuzhang-teaching.github.io/CSCE670-S26.html>

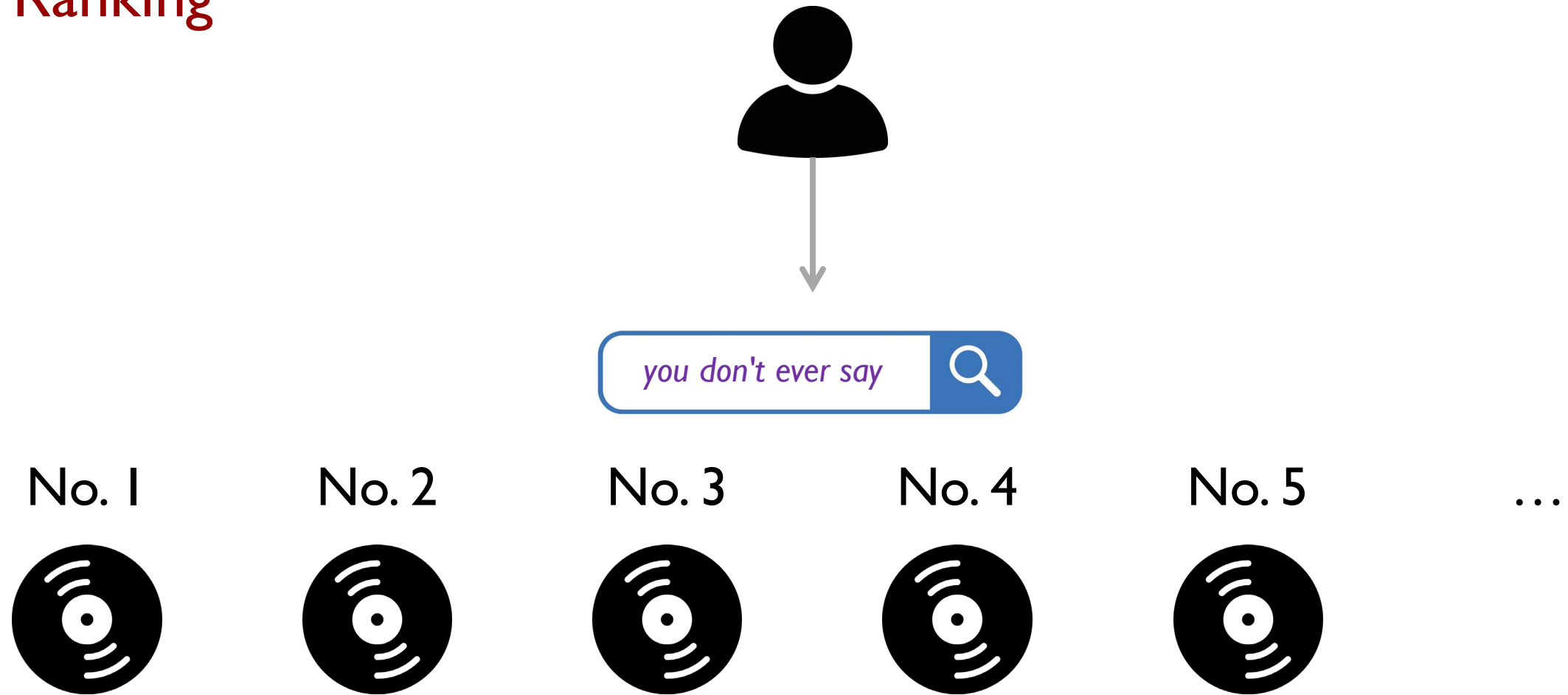
Recap: Boolean Retrieval

- Our capabilities so far:
 - Boolean keyword queries (AND, OR, NOT)
 - Inverted index
 - Phrase queries (“x y”)
 - Positional index
 - Proximity queries (“x NEAR:3 y”)
 - Positional index
 - Wildcard queries (“x*”)
 - Permuterm Index



We return a set of matching albums. (100s or 1000s of matches in some cases!)

Ranking



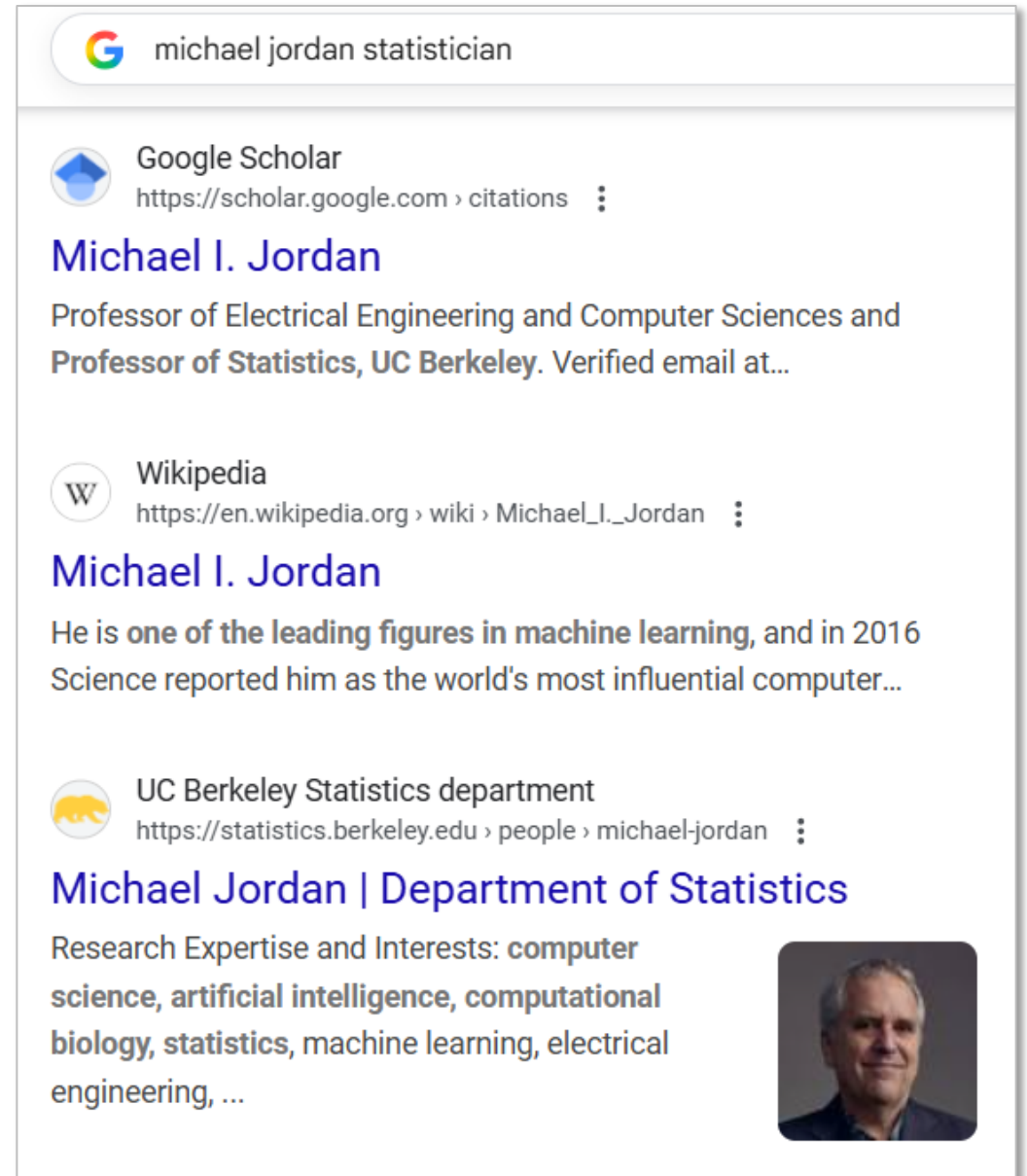
We return a **ranked** list of albums.

Our Plan: Ranking

- Why is ranking important?
- What factors impact ranking?
- Two foundational text-based approaches
 - TF-IDF
 - BM25
- Two foundational link-based approaches
 - PageRank
 - HITS
- Machine-learned ranking (“learning to rank”)

Why is ranking important?

- **User Study:** eye-tracking and relevance
- **Scenario:** Participants were asked to answer 10 questions using Google.
 - E.g., “*Find the homepage of Michael Jordan, the statistician.*”
- **Eye-Tracking:**
 - Record the sequence of eye movements
 - Analyze how users scan the results page of Google




Google search results for "michael jordan statistician".

Google Scholar
https://scholar.google.com › citations
Michael I. Jordan
Professor of Electrical Engineering and Computer Sciences and Professor of Statistics, UC Berkeley. Verified email at...

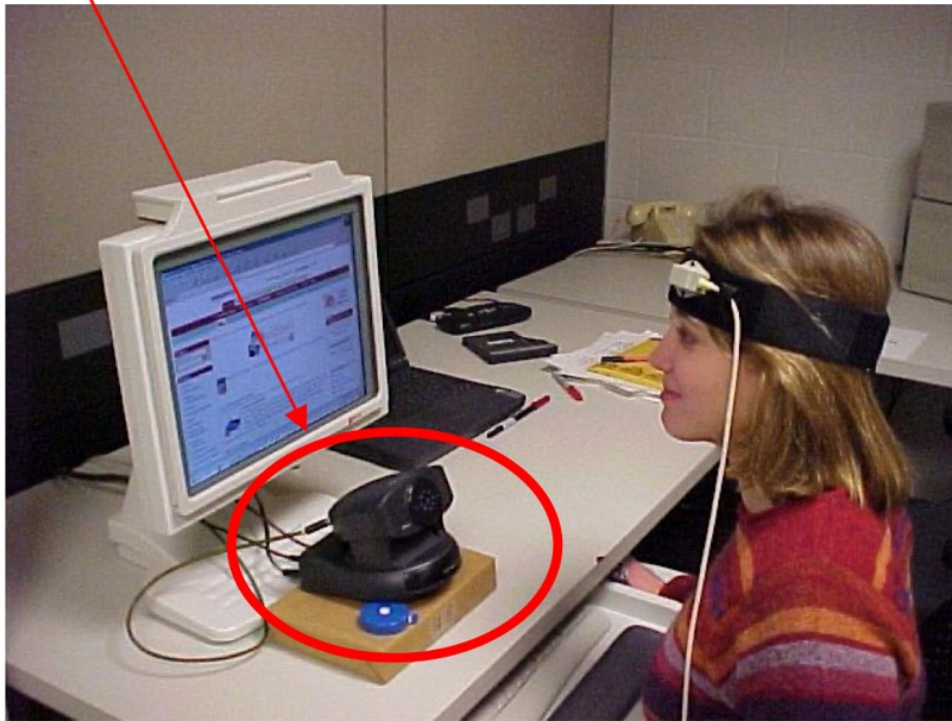
Wikipedia
https://en.wikipedia.org › wiki › Michael_I._Jordan
Michael I. Jordan
He is **one of the leading figures in machine learning**, and in 2016 Science reported him as the world's most influential computer...

UC Berkeley Statistics department
https://statistics.berkeley.edu › people › michael-jordan
Michael Jordan | Department of Statistics
Research Expertise and Interests: **computer science, artificial intelligence, computational biology, statistics**, machine learning, electrical engineering, ...



What is Eye-Tracking?

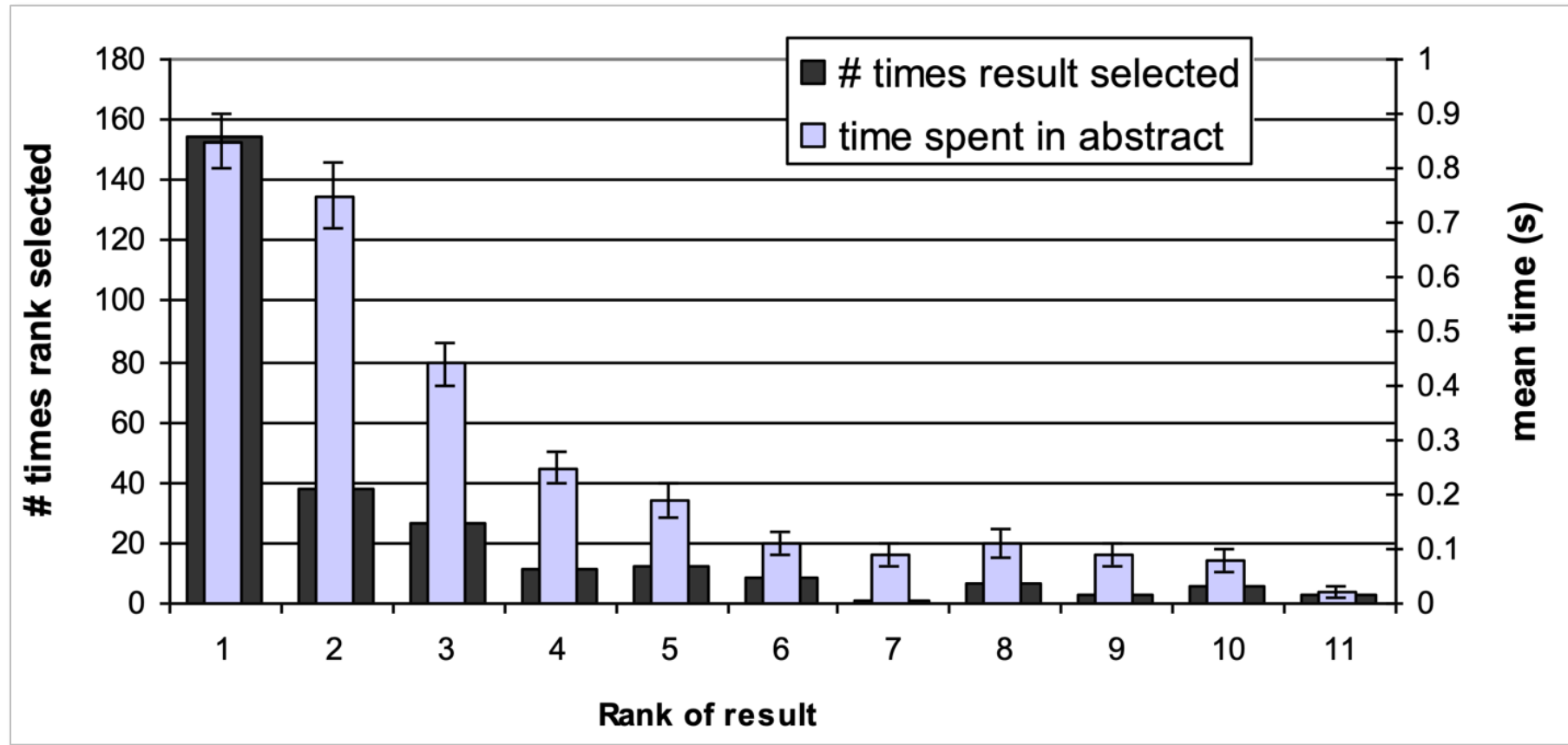
Eye tracking device



- Device to detect and record where and what people look at
 - **Fixations**: ~200-300ms, information is acquired
 - **Saccades**: extremely rapid movements between fixations

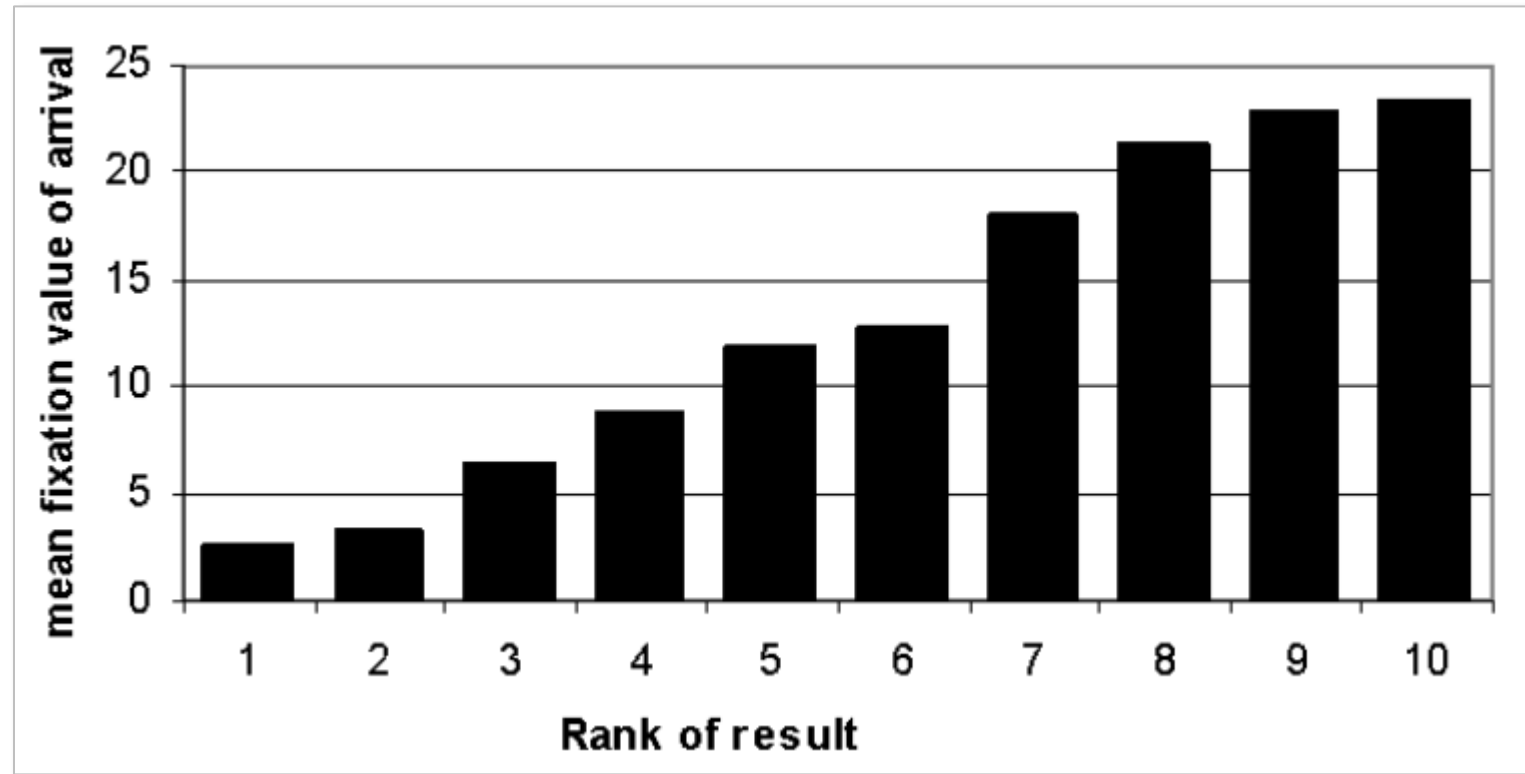


Viewing vs. Clicking



- Users view Documents 1 and 2 more thoroughly / often.
- Users click most frequently on Document 1.

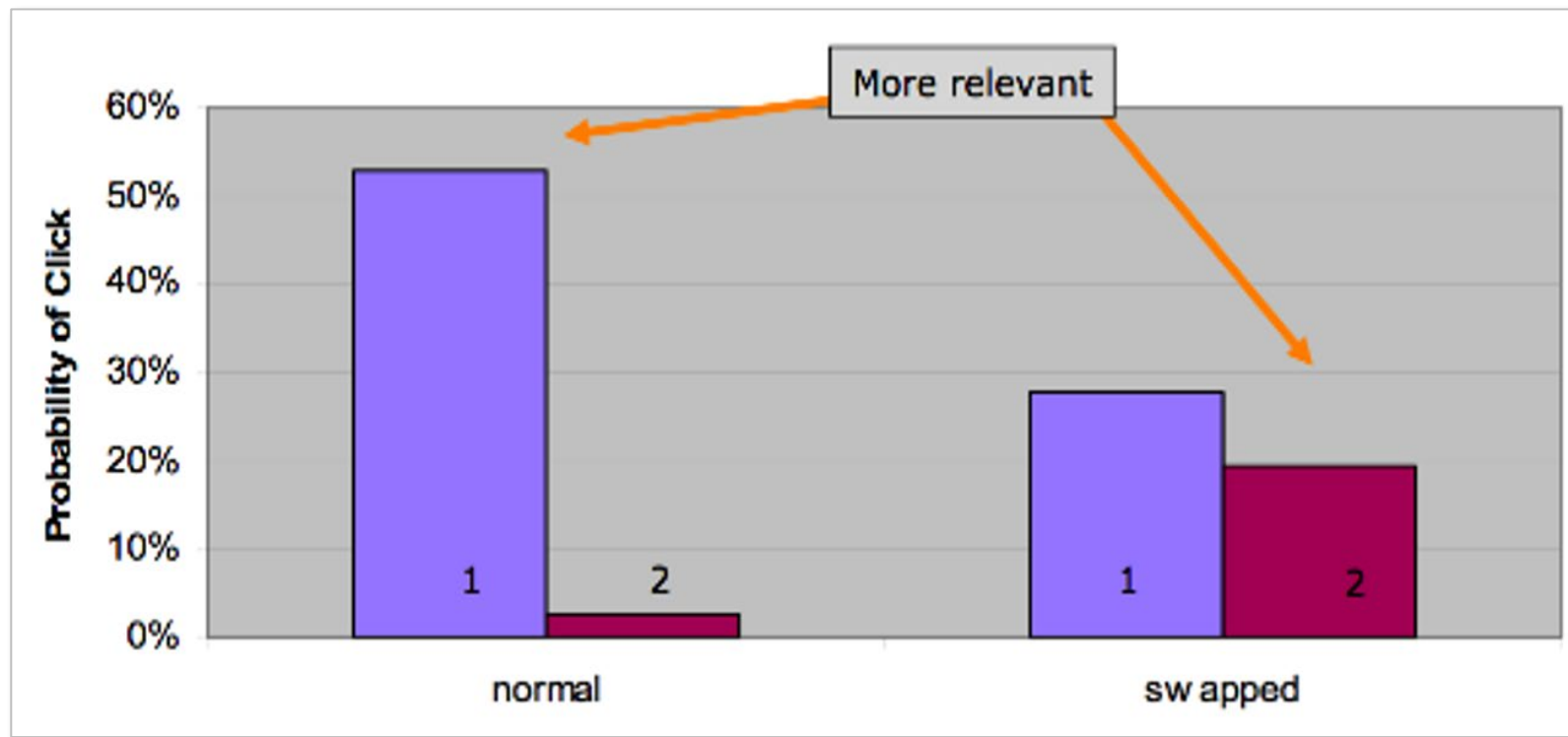
In which order are the results viewed?



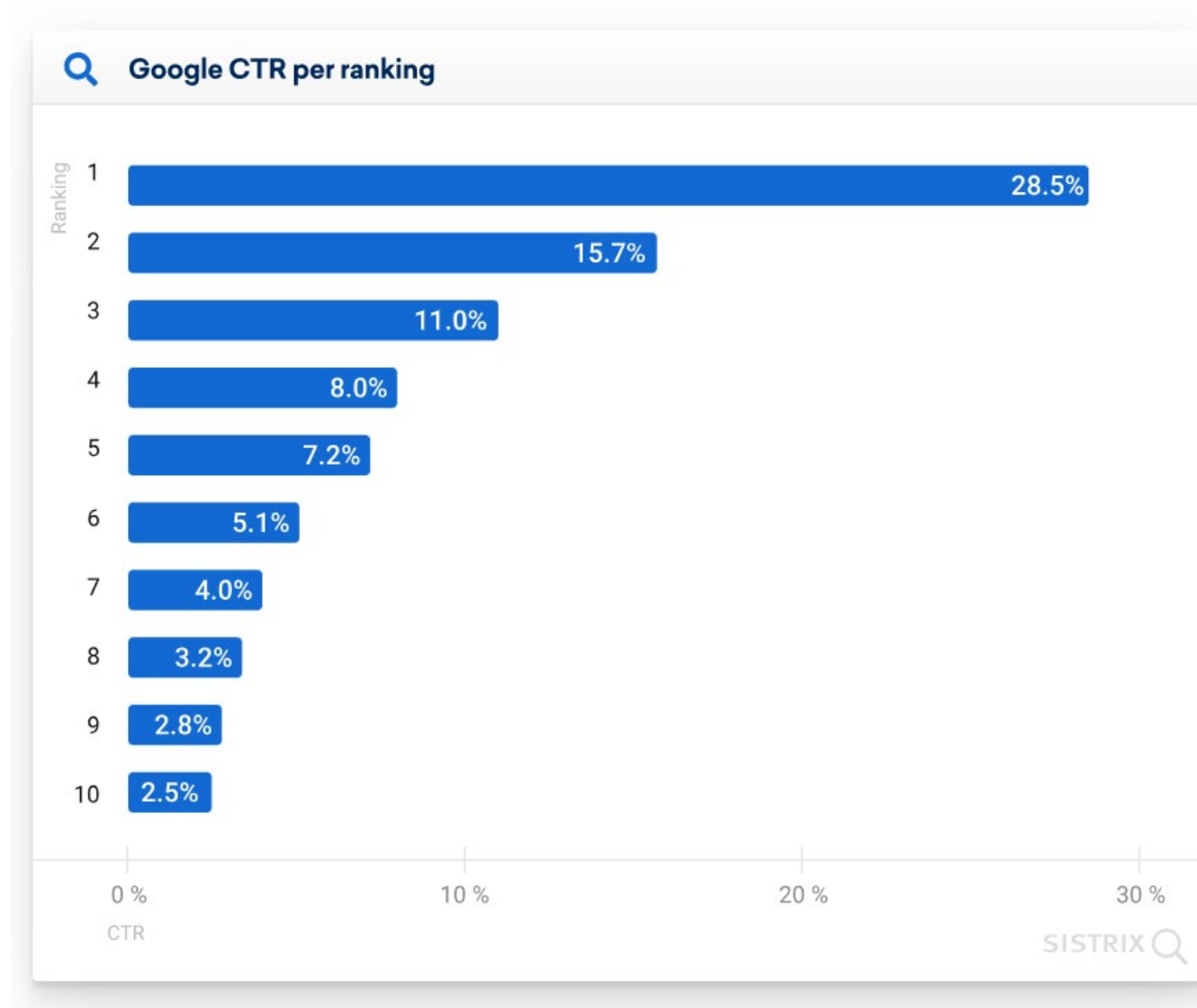
- Users tend to read the results in order.

Presentation Bias

- The top two results returned by Google were switched in order and then presented to the participants.
- Order of presentation influences where users look AND where they click.



Click-Through Rate (CTR) Per Ranking



<https://www.sistrix.com/blog/why-almost-everything-you-knew-about-google-ctr-is-no-longer-valid/>

- Accurate ranking increases the likelihood that the documents users click on are actually relevant.

Scoring as the Basis of Ranked Retrieval

- Let's try to build a scoring function:

$$\text{Score}(q, d) \in \mathbb{R}$$

to score every document d for a particular query q .

- Our hope is to design a scoring function so that the “best” documents (the most relevant, the best at satisfying the user) are scored highest.

TF-IDF

What factors impact scoring?

- Query: “*information retrieval*”
- Document 1: “*Information retrieval is a core area of computer science. Modern *information retrieval* systems use ranking algorithms to improve search results. Deep learning has also been applied to *information retrieval* tasks. Classic *information retrieval* models include TF-IDF and BM25. Evaluation in *information retrieval* typically involves metrics like MAP and NDCG.*”
- Document 2: “*Natural language processing (NLP) has many applications, such as text classification, machine translation, and *information retrieval*. These tasks often require a deep understanding of both syntax and semantics. Recent advancements in large language models have significantly improved the performance of NLP systems across many benchmarks.*”
- Which document should be ranked higher? Why?

Term Frequency (TF)

- Score each term t in a document d by the number of times it occurs in the document, denoted as $tf_{t,d}$
 - **Intuition:** The more frequently a term appears in a document, the more important it is considered within that document.
- Example
 - Document d : “zebra any love any zebra”
 - $tf_{\text{love},d} = 1$
 - $tf_{\text{zebra},d} = 2$
 - $tf_{\text{dream},d} = 0$

Variants of TF

Term frequency	
n (natural)	$\text{tf}_{t,d}$
l (logarithm)	$1 + \log(\text{tf}_{t,d})$
a (augmented)	$0.5 + \frac{0.5 \times \text{tf}_{t,d}}{\max_t(\text{tf}_{t,d})}$
b (boolean)	$\begin{cases} 1 & \text{if } \text{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$
L (log ave)	$\frac{1 + \log(\text{tf}_{t,d})}{1 + \log(\text{ave}_{t \in d}(\text{tf}_{t,d}))}$

- Why do we need these variants?
- If our scoring function is based **solely** on TF
 - Any monotonic transformation of TF is equivalent to TF in terms of ranking.
- If TF is **only one component** of our scoring function, we may want to control its “weight” in the function.
 - Do we want the growth of TF's weight to be equal in the following two cases?
 - $\text{tf}_{t,d}$ increases from 1 to 2
 - $\text{tf}_{t,d}$ increases from 100 to 101
 - “diminishing marginal gain”

What factors impact scoring?

- Query: “*mitochondria cell*”
- Document 1: “The *cell* structure of an organism varies depending on the type of *cell*. In multicellular organisms, each *cell* has a specific function. *Cell* division plays an important role in growth and repair.”
 - $TF(\text{“cell”}, \text{Document 1}) = 4$; $TF(\text{“mitochondria”}, \text{Document 1}) = 0$
- Document 2: “*Mitochondria* are known as the powerhouse of the *cell*. They play a critical role in ATP production and cellular respiration. Damage to *mitochondria* can lead to metabolic disorders.”
 - $TF(\text{“cell”}, \text{Document 2}) = 1$; $TF(\text{“mitochondria”}, \text{Document 2}) = 2$
- Which document should be ranked higher? Why?

Inverse Document Frequency (IDF)

- Score each term in a document by how rare it is across all documents
 - **Intuition:** The rarer a term is across a collection, the more valuable it is.
 - “*mitochondria*” is more valuable than “*cell*”.
 - If a document matches the term “*mitochondria*” once, it should receive a greater reward than matching the term “*cell*” once.

$$\text{idf}_t = \log \frac{|\mathcal{D}|}{|d \in \mathcal{D}: t \in d|}$$

- $|\mathcal{D}|$: number of documents in the collection
- $|d \in \mathcal{D}: t \in d|$: number of documents in the collection that contain t

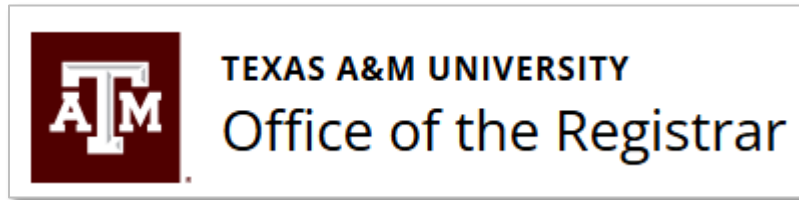
Inverse Document Frequency (IDF)

$$\text{idf}_t = \log \frac{|\mathcal{D}|}{|d \in \mathcal{D}: t \in d|}$$

- Example (We use 10 as the base of the logarithm)
 - Suppose you want to rank 100,000 documents.
 - “*cell*” appears in 1,000 of them
 - $\text{idf}_{\text{cell}} = \log \frac{100000}{1000} = \log 100 = 2$
 - “*mitochondria*” appears in 10 of them
 - $\text{idf}_{\text{mitochondria}} = \log \frac{100000}{10} = \log 10000 = 4$
- Can you roughly estimate the IDF value of “*a*”?

What factors impact scoring?

- Query: “TAMU 2026 Spring Break”
- Document 1: <https://registrar.tamu.edu/academic-calendar/spring-2026>



- Document 2: A social media post written by an account with 10 followers mentioning the time of TAMU 2026 Spring Break
- Which document should be ranked higher? Why?
- How can we know the “reputation” of a website?
 - Next week!

Let's first consider the textual factors (TF and IDF) together.

- Given a query q and a document d , we want to calculate $\text{Score}(q, d)$.
- The query q may have one or more terms. For each term t ,

$$\text{tfidf}_{t,d} = \text{tf}_{t,d} \times \text{idf}_t$$

- The overall score is the sum of the TF-IDF scores of all terms.

$$\text{Score}(q, d) = \sum_{t \in q} \text{tfidf}_{t,d}$$

- How should we interpret this formula?
 - A weighted sum of TF
 - The weight is IDF

Example

- Suppose you want to rank 100,000 documents. “*cell*” appears in 1,000 of them; “*mitochondria*” appears in 10 of them. (We use 10 as the base of the logarithm in IDF.)
- Query: “*mitochondria cell*”
- Document 1: “*The cell structure of an organism varies depending on the type of cell. In multicellular organisms, each cell has a specific function. Cell division plays an important role in growth and repair.*”
 - $\text{TF}(\text{“cell”}, \text{Document 1}) = 4; \text{IDF}(\text{“cell”}) = 2$
 - $\text{TF-IDF}(\text{“cell”}, \text{Document 1}) = 4 \times 2 = 8$
 - $\text{TF}(\text{“mitochondria”}, \text{Document 1}) = 0; \text{IDF}(\text{“mitochondria”}) = 4$
 - $\text{TF-IDF}(\text{“mitochondria”}, \text{Document 1}) = 0 \times 4 = 0$
 - $\text{TF-IDF}(\text{Query}, \text{Document 1}) = 8 + 0 = 8$

Example

- Suppose you want to rank 100,000 documents. “*cell*” appears in 1,000 of them; “*mitochondria*” appears in 10 of them. (We use 10 as the base of the logarithm in IDF.)
- Query: “*mitochondria cell*”
- Document 2: “*Mitochondria* are known as the powerhouse of the *cell*. They play a critical role in ATP production and cellular respiration. Damage to *mitochondria* can lead to metabolic disorders.”
 - $\text{TF}(\text{“cell”}, \text{Document 2}) = 1; \text{IDF}(\text{“cell”}) = 2$
 - $\text{TF-IDF}(\text{“cell”}, \text{Document 2}) = 1 \times 2 = 2$
 - $\text{TF}(\text{“mitochondria”}, \text{Document 2}) = 2; \text{IDF}(\text{“mitochondria”}) = 4$
 - $\text{TF-IDF}(\text{“mitochondria”}, \text{Document 2}) = 2 \times 4 = 8$
 - $\text{TF-IDF}(\text{Query}, \text{Document 2}) = 2 + 8 = 10$

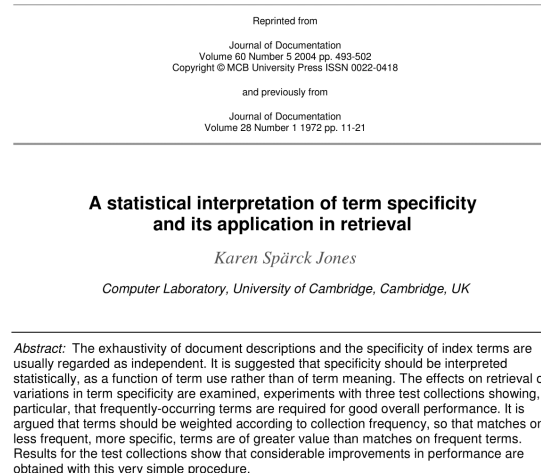
Vector Space Model (VSM)

Early History of IR

TF (1957; by Hans Peter Luhn)



IDF (1972; by Karen Spärck Jones)



Vector Space Model (1975; by Gerard Salton)

Information Retrieval and Language Processing C.A. Montgomery Editor

A Vector Space Model for Automatic Indexing

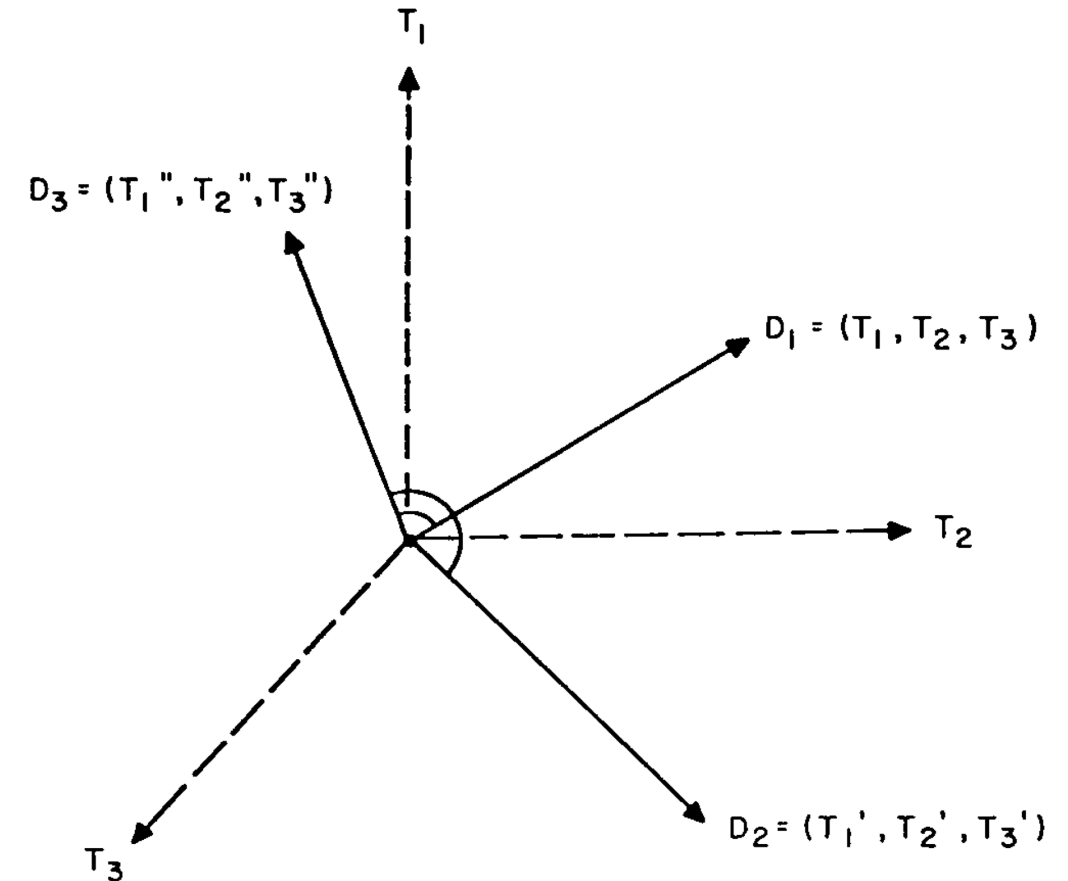
G. Salton, A. Wong and C. S. Yang
Cornell University

In a document retrieval, or other pattern matching environment where stored entities (documents) are compared with each other or with incoming patterns (search requests), it appears that the best indexing (property) space is one where each entity lies as far away from the others as possible; in these circumstances the value of an indexing system may be expressible as a function of the density of the object space; in particular, retrieval performance may correlate inversely with space density. An approach based on space density computations is used to choose an optimum indexing vocabulary for a collection of documents. Typical evaluation results are shown, demonstrating the usefulness of the model.

General Idea of VSM

- Represent documents and queries as vectors in a high-dimensional space
- A document is more **relevant** to a query if they are more “**similar**” in the vector space.
 - We will define “**similar**” later.
- What are the axes?
- How about each dimension representing **a word in the vocabulary**?

Fig. 1. Vector representation of document space.



VSM: Example 1

- If the word appears in a query/document, the corresponding entry is 1.
- Otherwise, the corresponding entry is 0.
- **Query q :** “any AND zebra”
- **Document d :** “zebra any love any zebra”

Vocabulary	Vector(q)	Vector(d)
any	1	1
believe	0	0
choose	0	0
...
love	0	1
starring	0	0
zebra	1	1

- What is the inner product of **Vector(q)** and **Vector(d)**?
 - $\text{Vector}(q) \cdot \text{Vector}(d) = 2$
- In the setting of **Boolean retrieval**, d is relevant to this q **if and only if** $\text{Vector}(q) \cdot \text{Vector}(d) = 2$.

VSM: Example 1

- If the word appears in a query/document, the corresponding entry is 1.
- Otherwise, the corresponding entry is 0.
- **Query** q : “ t_1 AND t_2 AND ... AND t_N ” (There are N words that must appear in d .)
- In the setting of **Boolean retrieval**, d is relevant to this q **if and only if**
$$\text{Vector}(q) \cdot \text{Vector}(d) = N$$

VSM: Example 2

- Given a query q , the corresponding entry of word t is $tf_{t,q}$.
- Given a document d , the corresponding entry of word t is $tf_{t,d} \times idf_t$.
- **Query q :** “any any zebra”
- **Document d :** “zebra any love any zebra”

Vocabulary
any
believe
choose
...
love
starring
zebra

Vector(q)
2
0
0
...
0
0
1

Vector(d)
$tf_{any,d} \times idf_{any}$
0
0
...
$tf_{love,d} \times idf_{love}$
0
$tf_{zebra,d} \times idf_{zebra}$

- What is the inner product of **Vector(q)** and **Vector(d)**?

VSM: Example 2

- Query q : “any any zebra”
- Document d : “zebra any love any zebra”
- What is the inner product of $\text{Vector}(q)$ and $\text{Vector}(d)$?

Vocabulary
any
believe
choose
...
love
starring
zebra

Vector(q)
2
0
0
...
0
0
1

Vector(d)
$\text{tf}_{\text{any},d} \times \text{idf}_{\text{any}}$
0
0
...
$\text{tf}_{\text{love},d} \times \text{idf}_{\text{love}}$
0
$\text{tf}_{\text{zebra},d} \times \text{idf}_{\text{zebra}}$

$$\begin{aligned}
 \text{Vector}(q) \cdot \text{Vector}(d) &= 2 \times \text{tf}_{\text{any},d} \times \text{idf}_{\text{any}} + 1 \times \text{tf}_{\text{zebra},d} \times \text{idf}_{\text{zebra}} \\
 &= \text{tf}_{\text{any},d} \times \text{idf}_{\text{any}} + \text{tf}_{\text{any},d} \times \text{idf}_{\text{any}} + \text{tf}_{\text{zebra},d} \times \text{idf}_{\text{zebra}} \\
 &= \sum_{t \in q} (\text{tf}_{t,d} \times \text{idf}_t) \\
 &= \text{TF-IDF}(q, d)
 \end{aligned}$$

- TF-IDF is a special case of VSM.

Problems with Inner Product?

- Query q : “any any zebra”
- Document d_1 : “zebra any love any zebra”
- Document d_2 : “zebra any love any zebra zebra any love any zebra”

Vocabulary
any
believe
choose
...
love
starring
zebra

Vector(q)
2
0
0
...
0
0
1

Vector(d_1)
$\text{tf}_{\text{any},d_1} \times \text{idf}_{\text{any}}$
0
0
...
$\text{tf}_{\text{love},d_1} \times \text{idf}_{\text{love}}$
0
$\text{tf}_{\text{zebra},d_1} \times \text{idf}_{\text{zebra}}$

Vector(d_2)
$2 \times \text{tf}_{\text{any},d_1} \times \text{idf}_{\text{any}}$
0
0
...
$2 \times \text{tf}_{\text{love},d_1} \times \text{idf}_{\text{love}}$
0
$2 \times \text{tf}_{\text{zebra},d_1} \times \text{idf}_{\text{zebra}}$

Problems with Inner Product?

- Query q : “any any zebra”
- Document d_1 : “zebra any love any zebra”
- Document d_{100} : “zebra any love any zebra zebra any love any zebra ... (repeat 100 times)”

Vocabulary
any
believe
choose
...
love
starring
zebra

Vector(q)
2
0
0
...
0
0
1

Vector(d_1)
$\text{tf}_{\text{any},d_1} \times \text{idf}_{\text{any}}$
0
0
...
$\text{tf}_{\text{love},d_1} \times \text{idf}_{\text{love}}$
0
$\text{tf}_{\text{zebra},d_1} \times \text{idf}_{\text{zebra}}$

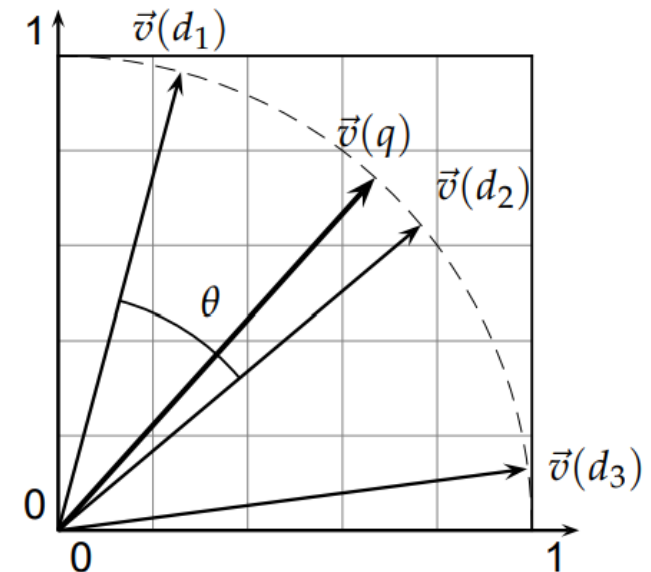
Vector(d_{100})
$100 \times \text{tf}_{\text{any},d_1} \times \text{idf}_{\text{any}}$
0
0
...
$100 \times \text{tf}_{\text{love},d_1} \times \text{idf}_{\text{love}}$
0
$100 \times \text{tf}_{\text{zebra},d_1} \times \text{idf}_{\text{zebra}}$

Problems with Inner Product?

- $\text{Vector}(d_2) = 2 \text{Vector}(d_1)$
- $\text{Vector}(q) \cdot \text{Vector}(d_2) = 2 \text{Vector}(q) \cdot \text{Vector}(d_1)$
- $\text{Vector}(d_{100}) = 100 \text{Vector}(d_1)$
- $\text{Vector}(q) \cdot \text{Vector}(d_{100}) = 100 \text{Vector}(q) \cdot \text{Vector}(d_1)$
- ...
- We can make the inner product of q and d in the vector space as large as possible by just making d longer!
- How to take the document length factor into account?

Cosine Similarity

- $\mathbf{x} = [x_1, x_2, \dots, x_N]$
- $\mathbf{y} = [y_1, y_2, \dots, y_N]$
- $$\cos(\mathbf{x}, \mathbf{y}) = \frac{x_1y_1 + x_2y_2 + \dots + x_Ny_N}{\sqrt{x_1^2 + x_2^2 + \dots + x_N^2} \times \sqrt{y_1^2 + y_2^2 + \dots + y_N^2}} = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|} = \left(\frac{\mathbf{x}}{\|\mathbf{x}\|} \right) \cdot \left(\frac{\mathbf{y}}{\|\mathbf{y}\|} \right)$$
- Equivalent to first **normalizing the vectors to unit length**, and then computing the dot product
 - \mathbf{x} , $2\mathbf{x}$, and $100\mathbf{x}$ will be the same vector after length normalization.
- The larger the cosine similarity, the smaller the **angle** between the two unit vectors (i.e., the more “similar” they are).



Example

- Query q : “any any zebra”
- Document d_1 : “zebra any love any zebra”
- Let's assume $\text{idf}_{\text{any}} = \text{idf}_{\text{love}} = 2$ and $\text{idf}_{\text{zebra}} = 4$.

Vocabulary
any
believe
choose
...
love
starring
zebra

Vector(q)
2
0
0
...
0
0
1

Vector(d_1)
$\text{tf}_{\text{any},d_1} \times \text{idf}_{\text{any}}$
0
0
...
$\text{tf}_{\text{love},d_1} \times \text{idf}_{\text{love}}$
0
$\text{tf}_{\text{zebra},d_1} \times \text{idf}_{\text{zebra}}$

Vector(d_1)
$2 \times 2 = 4$
0
0
...
$1 \times 2 = 2$
0
$2 \times 4 = 8$

Example

- $||\text{Vector}(q)|| = \sqrt{2^2 + 1^2} = \sqrt{5}$
- $||\text{Vector}(d_1)|| = \sqrt{4^2 + 2^2 + 8^2} = \sqrt{84}$
- $\cos(\text{Vector}(q), \text{Vector}(d_1)) = \frac{2 \times 4 + 0 \times 2 + 1 \times 8}{\sqrt{5} \times \sqrt{84}} \approx 0.781$




Vocabulary
any
believe
choose
...
love
starring
zebra

Vector(q)
2
0
0
...
0
0
1

Vector(d_1)
4
0
0
...
2
0
8

What's the range of the cosine similarity in VSM?

Our Plan: Ranking

-  Why is ranking important?
-  What factors impact ranking?
- Two foundational text-based approaches
 -  TF-IDF
 - BM25
- Two foundational link-based approaches
 - PageRank
 - HITS
- Machine-learned ranking (“learning to rank”)

BM25 (or Okapi BM25)

- **BM** = Best Match
- **25** = the 25th version of the scoring function
- Over time, BM25 has become a default scoring function used broadly across many real-world systems.
- Goal: be sensitive to term frequency and document length while not adding too many parameters

History of BM25: The 1994 Text REtrieval Conference (TREC-3)

Overview of the Third Text REtrieval Conference (TREC-3)

Donna Harman

**National Institute of Standards and Technology
Gaithersburg, MD. 20899**

1. Introduction

In November of 1992 the first Text REtrieval Conference (TREC-1) was held at NIST [Harman 1993]. The conference, co-sponsored by ARPA and NIST, brought together information retrieval researchers to discuss their system results on a new large test collection (the TIPSTER collection). This conference became the first in a series of ongoing conferences dedicated to encouraging research in retrieval from large-scale test collections, and to encouraging increased interaction among research groups in industry and academia. From the beginning there has been an almost equal number of universities and companies

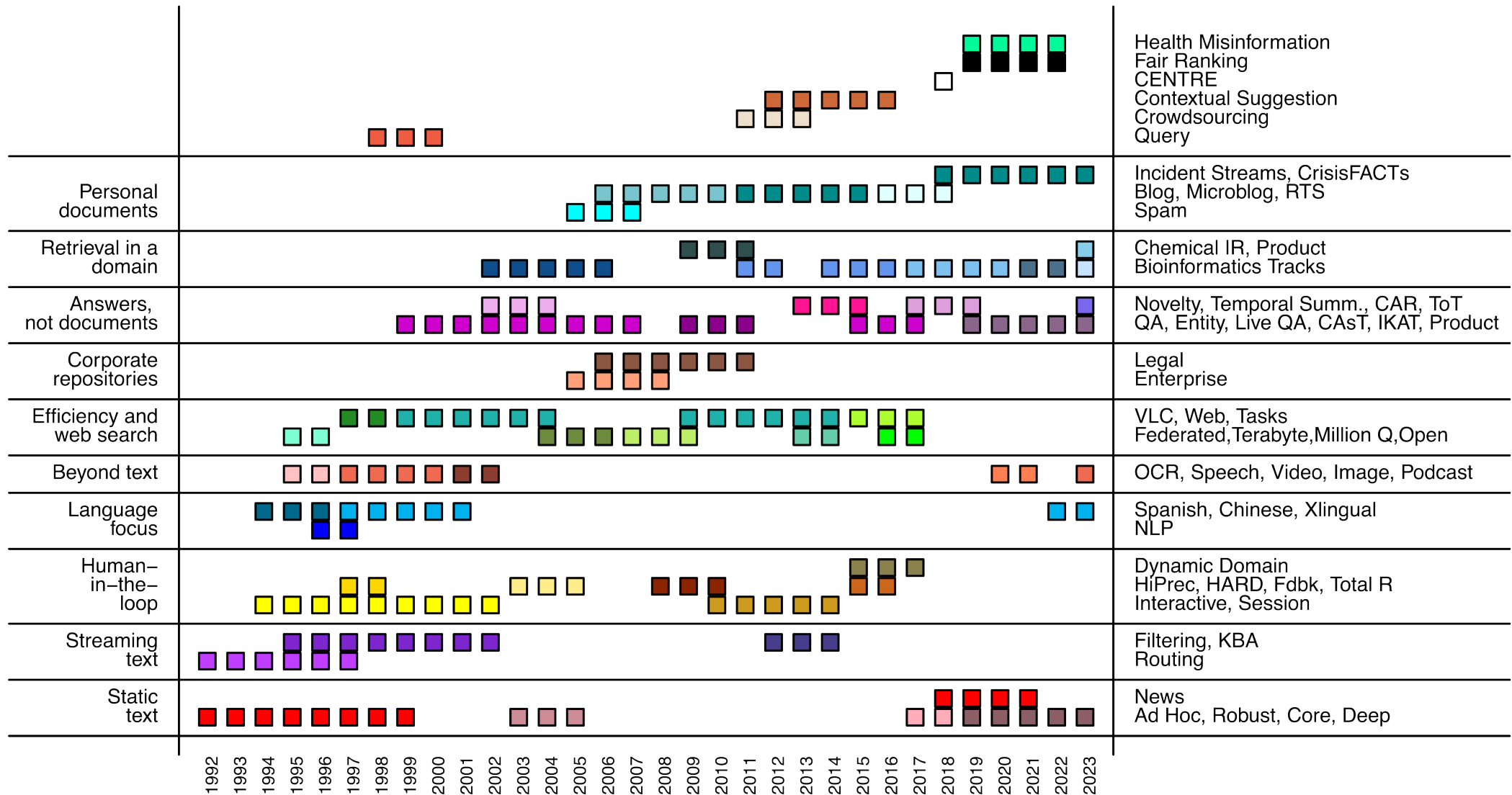
description of the test collection being used, and an overview of the results. The papers from the individual groups should be referred to for more details on specific system approaches.

2. The Task and the Participants

The three TREC conferences have all centered around two tasks based on traditional information retrieval modes: a "routing" task and an "ad hoc" task. In the routing task it is assumed that the same questions are always being asked, but that new data is being searched. This task is similar to that done by news clipping services or by li-

TREC: <https://trec.nist.gov/>

IR challenges held each year, featuring expert-annotated data (e.g., relevant query-document pairs) from various domains



TREC-3

Table 1: TREC-3 Participants (14 companies, 19 universities)

Australian National University	Bellcore
Carnegie Mellon University/CLARITECH	CITRI, Australia
City University, London	Cornell University
Dublin City University	Environment Research Institute of Michigan
Fulcrum	George Mason University
Logicon Operating Systems	Mayo Clinic/Foundation
Mead Data Central	National Security Agency
New York University	NEC Corporation
Queens College	Rutgers University (two groups)
Siemens Corporate Research Inc.	Swiss Federal Institute of Technology (ETH)
TRW/Paracel	Universitaet Dortmund, Germany
University of California - Berkeley	University of Central Florida
University of Massachusetts at Amherst	VPI&SU (Virginia Tech)
University of Minnesota	University of Toronto
Universite de Neuchatel, Switzerland	Verity Inc.
West Publishing Co.	Xerox Palo Alto Research Center

TREC-3

Okapi at TREC-3

S E Robertson

S Walker

S Jones

M M Hancock-Beaulieu

M Gatford

Centre for Interactive Systems Research

Department of Information Science

City University

Northampton Square

London EC1V 0HB

UK

Advisers: E Michael Keen (University of Wales, Aberystwyth), Karen Sparck Jones (Cambridge University), Peter Willett (University of Sheffield)

1 Introduction

The sequence of TREC conferences has seen the City University Okapi IR system evolve in several ways. Be-

City at TREC-2

For TREC-2 the simple inverse collection frequency (ICF) term-weighting scheme was elaborated to embody within-document frequency and document length components, as well as within-query frequency, and a large number of weighting functions were investigated. Because of hardware failures few of the runs were ready in time, and City's official results were very poor. However, later automatic ad hoc and routing results are

What is BM25?
Why are we talking about a method from 1994?

BM25 in (Almost) One Slide

$$\text{BM25}(q, d) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{\text{TF}(t, d) \cdot (k_1 + 1)}{\text{TF}(t, d) + k_1(1 - b + b \cdot \frac{|d|}{\text{avgdl}})}$$

- k_1 controls term frequency scaling
 - $k_1 = 0$: binary model
 - k_1 very large: raw term frequency
- b controls document length normalization
 - $b = 0$: no document length normalization
 - $b = 1$: relative frequency (full document length normalization)
- Typically, k_1 is set between 1.2 and 2; b is set around 0.75
- $|d|$ is the length of d (in words); avgdl = average document length (in words)

The IDF Component in BM25

$$\text{IDF}(t) = \log \left(\frac{N - n(t) + 0.5}{n(t) + 0.5} + 1 \right)$$

- $N = |\mathcal{D}|$, number of documents
- $n(t) = |\{d \in \mathcal{D} : t \in d\}|$, number of documents containing the term t

Why are we talking about a method from 1994?

- Even compared to today's large language models, BM25 still delivers strong performance on text retrieval tasks. It is also **easy to implement**, requires **no machine learning training**, and does **not rely on GPU support**.

OPINION

The Neural Hype and Comparisons Against Weak Baselines

Jimmy Lin

David R. Cheriton School of Computer Science, University of Waterloo

Setup

- Two recent IR papers: Paper 1 and Paper 2
 - Several “neural” models proposed/compared in the papers
- “Traditional” (non-neural) ranking models:
 - BM25
 - QL: query likelihood with Dirichlet smoothing
 - RM3 variant of relevance models (pseudo-relevance feedback where top results returned by a QL model are treated as relevant)
- Experiments in Anserini (open-source engine built on Lucene)

Results

Condition	AP	P20
QL	0.2499	0.3556
QL + RM3	0.2865	0.3773
Neural ₁	0.2815	0.3752
Neural ₂	0.2801	0.3764
Neural ₃	0.2856	0.3766
Neural ₃ '	0.2971	0.3948
Anserini: QL	0.2496	0.3543
Anserini: BM25	0.2526	0.3604
Anserini: BM25 + RM3 (independent)	0.2954	0.3885
Anserini: BM25 + RM3 (joint)	0.2973	0.3871

Table 1: Comparison of Anserini results to Paper 1.

Condition	AP	P20
BM25	0.238	0.354
BM25 + Features	0.250	0.367
Neural _x	0.258	0.372
Neural _y	0.256	0.370
Neural _x + Neural _y	0.259	0.373
A + Neural _y	0.263	0.380
A + Neural _y + M	0.265	0.380
B + Neural _y	0.270	0.383
B + Neural _y + M	0.272	0.386
Anserini: QL	0.2481	0.3517
Anserini: BM25	0.2528	0.3598
Anserini: BM25 + RM3 (independent)	0.2991	0.3901
Anserini: BM25 + RM3 (joint)	0.2956	0.3931

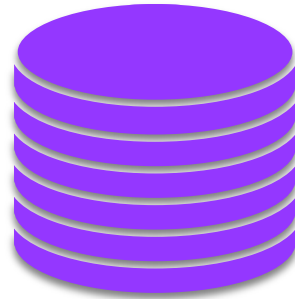
Table 2: Comparison of Anserini results to Paper 2.

How to interpret the BM25 scoring function?
Why is it so complicated?

Generative Model for Documents

- Let's consider a **probabilistic** interpretation.
- Idea: Words are drawn **independently** from the vocabulary using a multinomial distribution.

$p(\text{meet}) = 0.002$
 $p(\text{midnight}) = 0.001$
 $p(\text{the}) = 0.015$
...



... meet me at midnight ...

- If we use this model to generate a sentence
 - What is the probability that the first word is “*midnight*”?
 - What is the probability that the second word is “*midnight*”, given the first word is “*midnight*”?

Generative Model for Documents

- Given a specific term (e.g., “*midnight*”)
 - The distribution of its term frequency (TF) follows a **binomial** distribution.



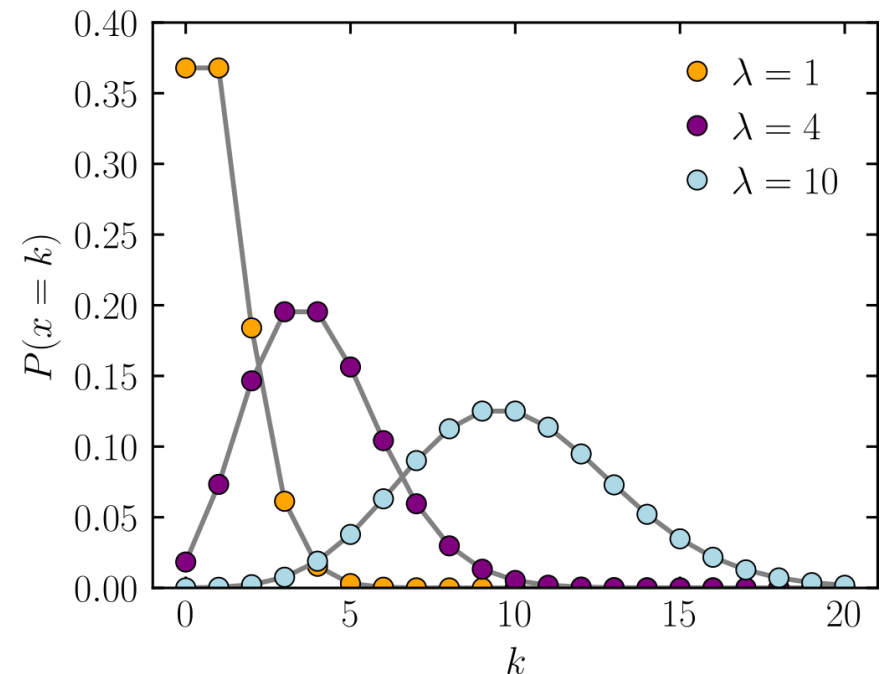
- Binomial distribution:** There is a coin for which the probability of landing heads up in a single toss is **0.001**, and tails up is **0.999**. What is the probability of getting heads exactly k times in **100** tosses? ($k = 0, 1, 2, \dots, 100$)

$$p_k = \binom{100}{k} \times 0.001^k \times 0.999^{100-k}$$

Generative Model for Documents

- When the number of tosses is large (e.g., ≥ 100) and the probability of getting heads is small, a **binomial** distribution can be approximated by a **Poisson** distribution.
- **Poisson distribution**: A call center receives incoming calls at an average rate of λ (i.e., λ calls per hour). The time interval between successive calls is independent of the time of the previous call. Under these conditions, the probability that the call center receives k calls ($k = 0, 1, 2, \dots$) in one hour is given by:

$$p_k = \frac{\lambda^k e^{-\lambda}}{k!}$$



Generative Model for Documents

- Why can we also approximate TF as following a Poisson distribution?
- Assume all documents have the same length (e.g., 100 words). *We will fix this later!*
- Given a word (e.g., “*midnight*”)
 - $p(\textit{midnight}) = 0.001$
 - “*midnight*” should “arrive” at an average rate of $0.001 \times 100 = 0.1$ per document.
 - The number of words between two occurrences of “*midnight*” is independent of the position of the previous “*midnight*”.
 - TF = the number of occurrences of “*midnight*” in a document

“One” Poisson Model

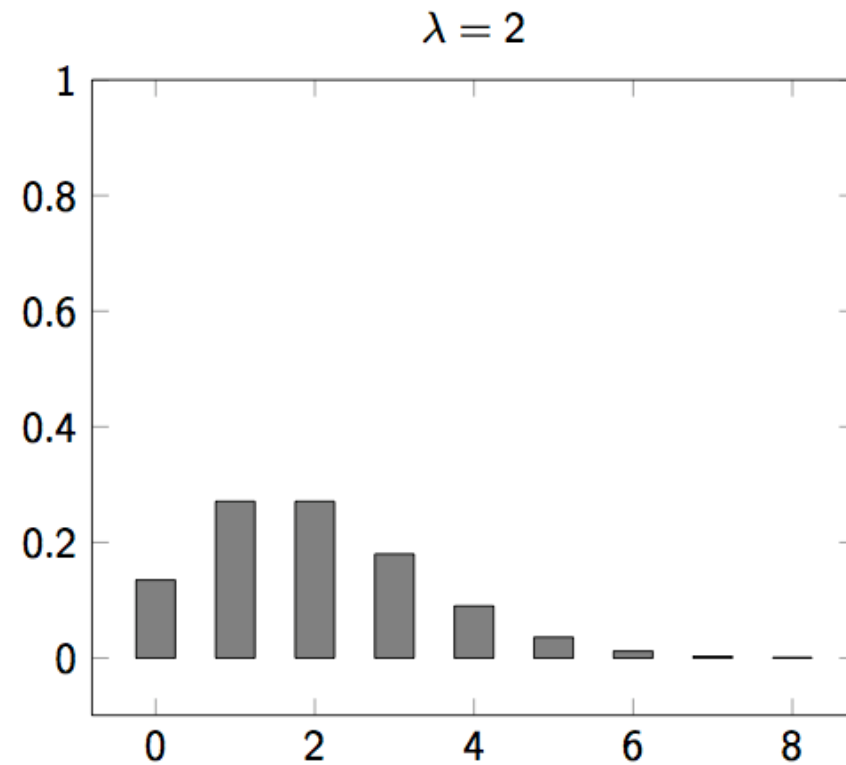
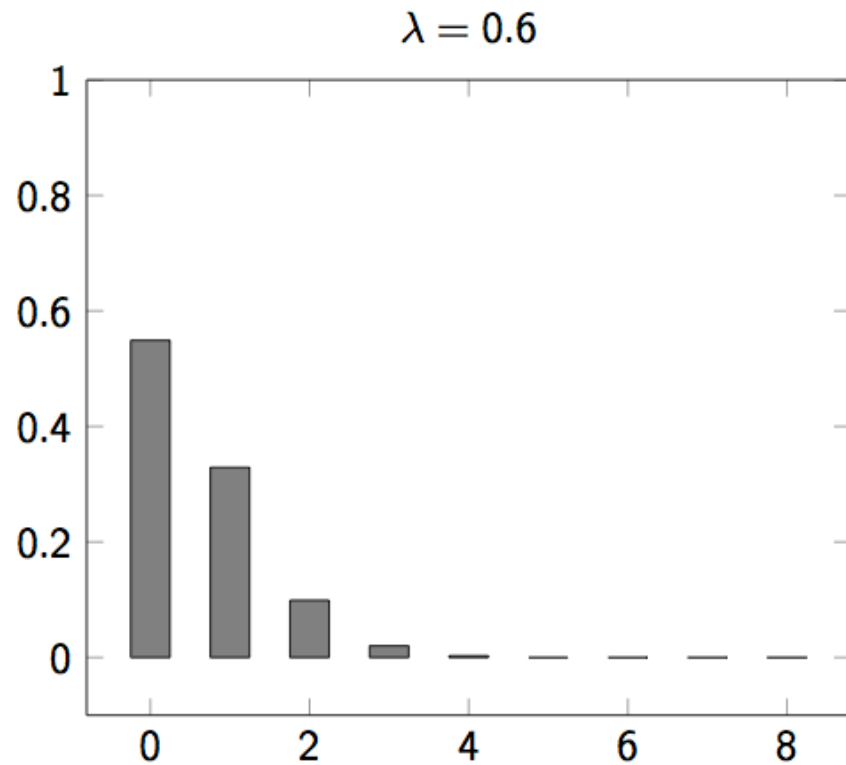
- Given 10,000 documents and a word “*information*”
 - For each document, “*information*” has a term frequency.
 - $tf_1, tf_2, \dots, tf_{10000}$
 - How should these number be distributed?
- Assume “*information*” appears M times in total in these 10,000 documents, then

$$\lambda = \frac{M}{10000}$$

(Definition of λ : the average number of occurrences in a document)

“One” Poisson Model

- $tf_1, tf_2, \dots, tf_{10000}$ should follow a Poisson distribution with $\lambda = \frac{M}{10000}$.



Theory vs. Practice

Table 1. Frequency Distributions for 19 Word Types and Expected Frequencies Assuming a Poisson Distribution with $\lambda = 53/650$

Frequency	Word Type	Number of Documents Containing k Tokens													
		k	0	1	2	3	4	5	6	7	8	9	10	11	12
51	act	608	35	5	2										
51	actions	617	27	2	0	2	0	2							
54	attitude	610	30	7	2	1									
52	based	600	48	2											
53	body	605	39	4	2										
52	castration	617	22	6	3	1	1								
55	cathexis	619	22	3	2	1	2	0	1						
51	comic	642	3	0	1	0	0	0	0	0	0	1	1	2	
53	concerned	601	45	4											
53	conditions	604	39	7											
55	consists	602	41	7											
53	factor	609	32	7	1	1									
52	factors	611	27	11	1										
55	feeling	613	26	7	3	0	0	1							
52	find	602	45	2	1										
54	following	604	39	6	1										
51	force	603	43	4											
51	forces	609	33	6	2										
52	forgetting	629	11	3	2	2	1	1	0	0	0	1			
53	expected, assuming Poisson distribution	599	49	2											

Limitations of the “One” Poisson Model

- In the table on the previous slide
 - According to the theory, it is almost impossible for a word to appear ≥ 5 times in a document.
 - In practice, a word can appear 10, 11, or 12 times in a document.
 - Why?
- Each word can be a background word or a topic-specific word in a document.
 - “information” may be a topic-specific word in information retrieval papers. (*Discussions revolve around the term “information retrieval”.*)
 - “information” may be a background word in sports news. (*Discussions typically do not revolve around this word; it is only used when generally needed.*)
- “One” Poisson model gives a reasonable fit for background words but a poor fit for topic-specific words.

“Two” Poisson Model

- $\pi \in [0,1]$: how “relevant” a word t is to a document d
 - $\pi = 0$: totally irrelevant $\rightarrow t$ is a **background** word in d
 - $\pi = 1$: totally relevant $\rightarrow t$ is a **topic-specific** word in d
 - $\pi \in (0,1)$: somewhere in between
- When t is a **background** word, the distribution of its TF follows a Poisson distribution:

$$P(\text{tf} = k | \pi = 0) = \frac{\mu^k e^{-\mu}}{k!}$$

- When t is a **topic-specific** word, the distribution of its TF follows another Poisson distribution:

$$P(\text{tf} = k | \pi = 1) = \frac{\lambda^k e^{-\lambda}}{k!}$$

“Two” Poisson Model

- $\pi \in (0,1)$: somewhere in between

$$P(\text{tf} = k | \pi) = \pi \frac{\lambda^k e^{-\lambda}}{k!} + (1 - \pi) \frac{\mu^k e^{-\mu}}{k!}$$

- Given 10,000 documents and a word “*information*”
 - How can we know the distribution?
 - Hard! Because π is a **hidden** variable for each document.
 - Unlike **observed** variables (e.g, TF) that can be directly calculated

“Two” Poisson Model

- Learning latent variable models from large-scale text data used to be a central problem in IR, NLP, and even machine learning in the broader sense.

[\[PDF\] Probabilistic latent semantic indexing](#)

[T Hofmann](#) - Proceedings of the 22nd annual international ACM ..., 1999 - [dl.acm.org](#)

... success in different domains including automatic **indexing** (**Latent Semantic Indexing**, LSI)

[1... approach to LSA and factor analysis { called **Probabilistic Latent** Semantic Analysis (PLSA) { ...

☆ Save [Cite](#) Cited by 7913 Related articles All 29 versions [↗](#)

[\[PDF\] acm.org](#)

[Full View](#)

[\[PDF\] Latent dirichlet allocation](#)

[DM Blei](#), [AY Ng](#), [MI Jordan](#) - Journal of machine Learning research, 2003 - [jmlr.org](#)

We describe **latent Dirichlet allocation** (LDA), a generative probabilistic model for collections of discrete data such as text corpora. LDA is a three-level hierarchical Bayesian model, in ...

☆ Save [Cite](#) Cited by 57586 Related articles All 92 versions Web of Science: 23713 [↗](#)

[\[PDF\] jmlr.org](#)

[Discover Full](#)

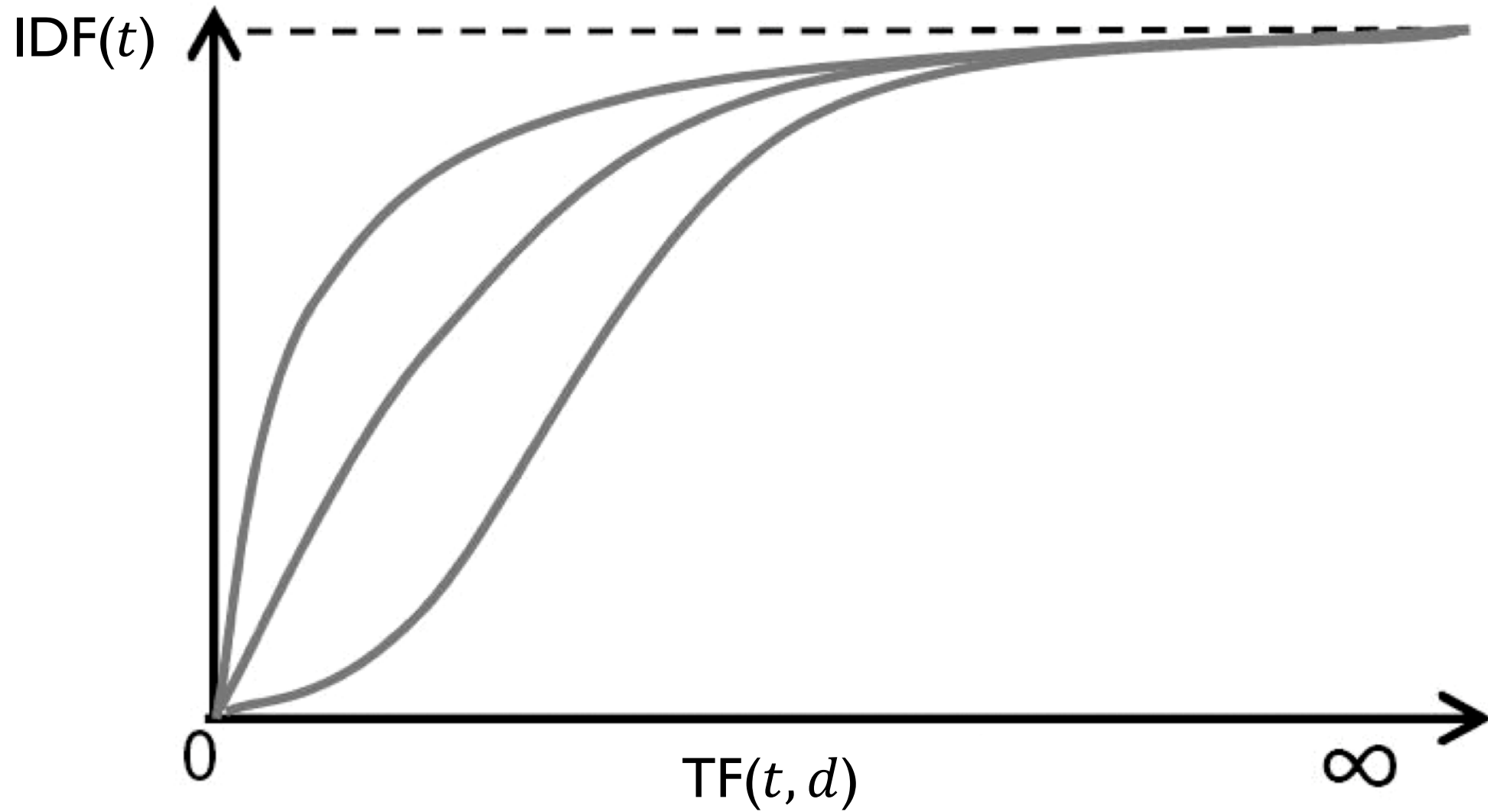
Let's make this simpler (?)

- Robertson-Spärck-Jones (**RSJ**) model:

$$\text{score}(q, d) = \sum_{t \in q, \text{TF}(t, d) > 0} \log \frac{P(\text{tf} = \text{TF}(t, d) | \pi = 1) \times P(\text{tf} = 0)}{P(\text{tf} = \text{TF}(t, d)) \times P(\text{tf} = 0 | \pi = 1)}$$

- If we plug the “**One**” **Poisson model** into RSJ and add some “okay” assumptions, we get TF-IDF.
- If we plug the “**Two**” **Poisson model** into RSJ, the formula will become complicated and ugly.
 - BUT it is monotonically increasing to $\text{IDF}(t)$ when $\text{TF}(t, d) \rightarrow +\infty$!

RSJ + “Two” Poisson Model



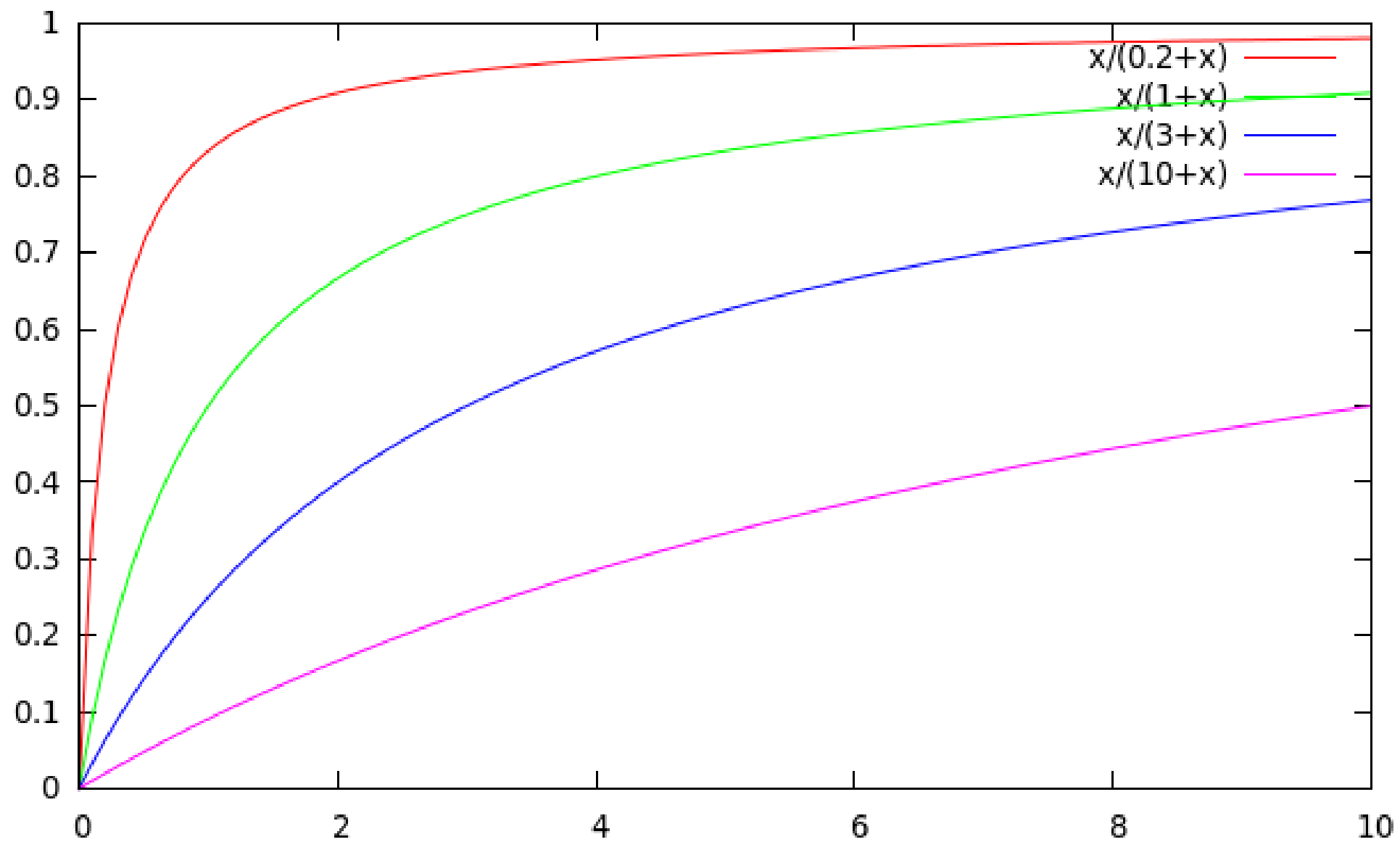
Let's make this simpler!

$$\text{score}(q, d) = g(\text{TF}(t, d)) \times \text{IDF}(t)$$

- g saturates toward a maximum value of 1, which is not true for simple TF-IDF scoring.
 - Think of raw TF or even log-based TF
- Let's approximate g with a simple parametric curve that has the same qualitative properties.

$$\frac{\text{TF}(t, d)}{k_1 + \text{TF}(t, d)}$$

- $k_1 > 0$ is a hyperparameter
- $\frac{\text{TF}(t, d)}{k_1 + \text{TF}(t, d)} = 0$ when $\text{TF}(t, d)$ is 0
- $\frac{\text{TF}(t, d)}{k_1 + \text{TF}(t, d)} = 1$ when $\text{TF}(t, d)$ is ∞



An Early Version of BM25

$$\text{score}(q, d) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{\text{TF}(t, d) \cdot (k_1 + 1)}{\text{TF}(t, d) + k_1}$$

- The $(k_1 + 1)$ factor does not change ranking, but makes $\frac{\text{TF}(t, d) \cdot (k_1 + 1)}{\text{TF}(t, d) + k_1} = 1$ when $\text{TF}(t, d) = 1$.
- The model is similar to TF-IDF, but the term frequency part is bounded.

Wait! There is also the factor of document length.

- Longer documents are likely to have larger TF values
- Why might documents be longer?
 - **Verbosity**: suggests observed TF too high
 - **Larger Scope**: suggests observed TF may be accurate
- A real document collection probably has both effects, so we should apply some kind of “partial” normalization.
- **Length normalization factor**: $B = 1 - b + b \cdot \frac{|d|}{\text{avgdl}}$
 - $|d|$ is the length of d (in words); avgdl = average document length (in words)
 - $b \in [0,1]$ is a hyperparameter

Document Length Normalization

- $B = 1 - b + b \cdot \frac{|d|}{\text{avgdl}}$
- What if the length of d is exactly the average document length?
 - $B = 1 - b + b = 1$, no need to normalize
- What if d is longer than average (e.g., $|d| = 2 \times \text{avgdl}$)?
 - $B = 1 - b + 2b = 1 + b > 1$
- What if d is shorter than average (e.g., $|d| = 0.5 \times \text{avgdl}$)?
 - $B = 1 - b + 0.5b = 1 - 0.5b < 1$
- $b = 1$: full document length normalization
- $b = 0$: no document length normalization

Putting it all together ...

$$\text{BM25}(q, d) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{\text{TF}(t, d) \cdot (k_1 + 1)}{\text{TF}(t, d) + k_1 (1 - b + b \cdot \frac{|d|}{\text{avgdl}})}$$

$= B$

- k_1 controls term frequency scaling
 - $k_1 = 0$: binary model
 - k_1 very large: raw term frequency
- b controls document length normalization
 - $b = 0$: no document length normalization
 - $b = 1$: relative frequency (full document length normalization)
- Typically, k_1 is set between 1.2 and 2; b is set around 0.75
- $|d|$ is the length of d (in words); avgdl = average document length (in words)

BM25 in Homework 2, Quiz 1, and Final

- Homework 2
 - You need to implement the BM25 scoring function.
- Quiz 1 and Final Exam
 - You should be familiar with the binomial generative model and the “One” Poisson model.
 - You should know document length normalization in the BM25 scoring function.
 - You do NOT need to master the derivation of the “Two” Poisson model and the Robertson-Spärck-Jones model.
 - You will NOT be required to manually calculate the BM25 score.



Thank You!

Course Website: <https://yuzhang-teaching.github.io/CSCE670-S26.html>