Information Retrieval

By

Dr Syed Khaldoon Khurshid

Focus of the Week:

- Information Retrieval and Its Components
- Retrieval: Adhoc and Filtering
- Information Retrieval Models
  - A taxonomy of information retrieval models
  - A formal characterization of IR models
  - Classic Information Retrieval Models:
    - Boolean Model
    - Space Vector Model
      - Clustering and Classification in Information Retrieval

What is Information Retrieval?

Information Retrieval is an activity of obtaining information resources relevant to an information need from a collection of information resources.

Searches can be based on metadata or on full-text (or other content-based) indexing.

GOAL: Find the documents Most Relevant to a certain Query

Three components of IR:

- Collection of Documents
- Query
- Notion of Relevancy

An Information Retrieval System comprises of the following components:

1. A model for representing documents and query statement

D+Q

2. A Matching function which evaluates the relevancy of the documents with respect to the user query

MF

What are Models?

A model is a construct designed help us **understand a complex system.**

Models always rely on **simplifying assumptions.**

Information Retrieval Models

Information Retrieval Models are **specific methods or systems designed to efficiently search and organize large amounts of information.** **Example**

They are tailored to the task of retrieving information from databases or the internet, making them well-suited for this particular purpose.

These models have **features and algorithms** that distinguish them from general-purpose software or tools used for other tasks.

Different Types of General Models:

- *Conceptual Model*
- *Mathematical Model*
- *Physical Analog Models*

Information Retrieval Models

IR models are different based on how they **show documents and queries**, how they **find matching documents** in a collection, and how they **rank those documents**.

Meaning...

An IR model defines the following aspects of retrieval procedure of a search engine:

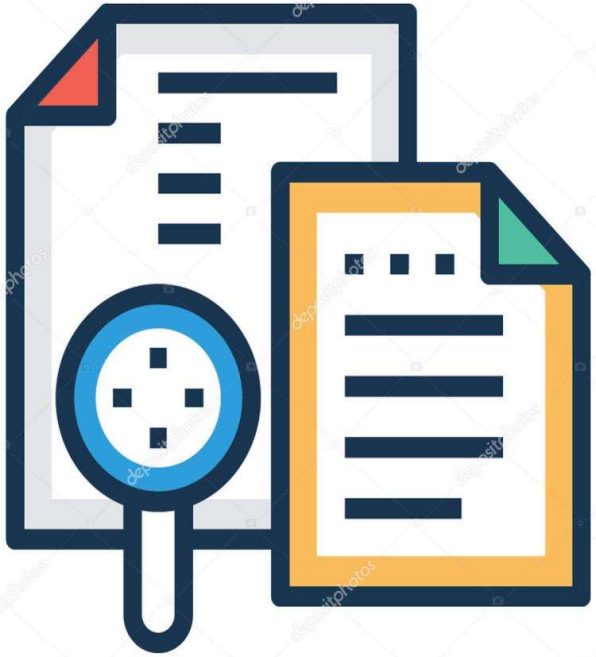
a. How the documents in the collection and user's queries are transformed?

b. How system identifies the relevancy of the documents based on the query word/phrase given by the user?

c. How system ranks the retrieved documents based on the relevancy?

Traditional Information Retrieval

- Traditional information retrieval systems usually adopt index terms to index and retrieve documents.
- An index term is a keyword(or group of related words) which has some meaning of its own (usually a noun).
  - In computer science, think of an index term like a variable name in a programming language. Just as a variable name represents something meaningful in code (like "count" to store the number of items), an index term represents a meaningful concept or keyword in information retrieval, making it easier to search for specific information.



The advantage of using Index Terms

Simple

- The semantic of the documents and of the user information need can be naturally expressed through sets of index terms.

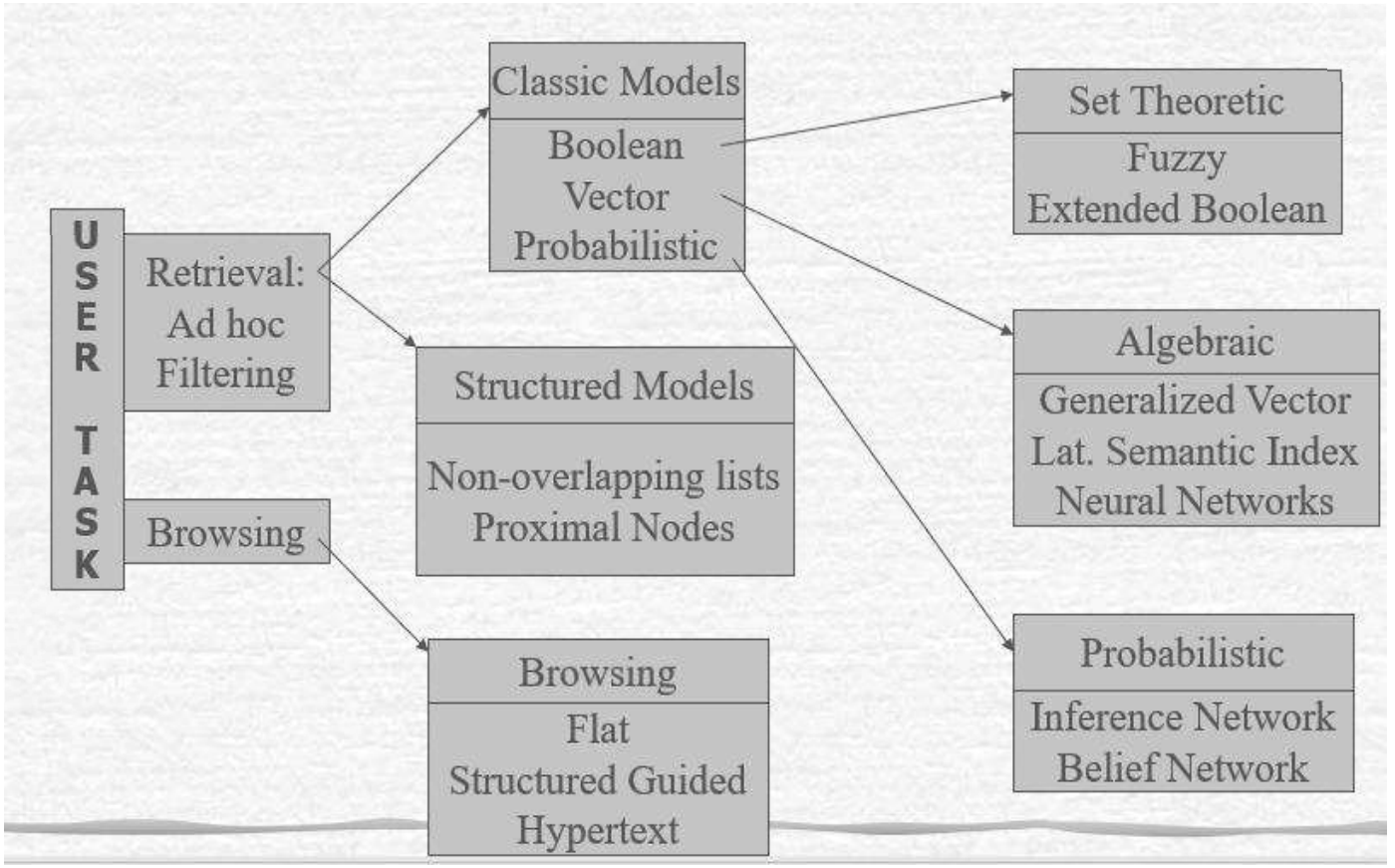
Mathematically:

- D = {d1, d2, d3, ...} (Set of documents)
- Q = {q1, q2, q3, ...} (Set of user information needs)
- T = {t1, t2, t3, ...} (Set of index terms)

Ranking Algorithm

- Ranking algorithms are at the core of information retrieval systems (predicting which documents are relevant and which are not).

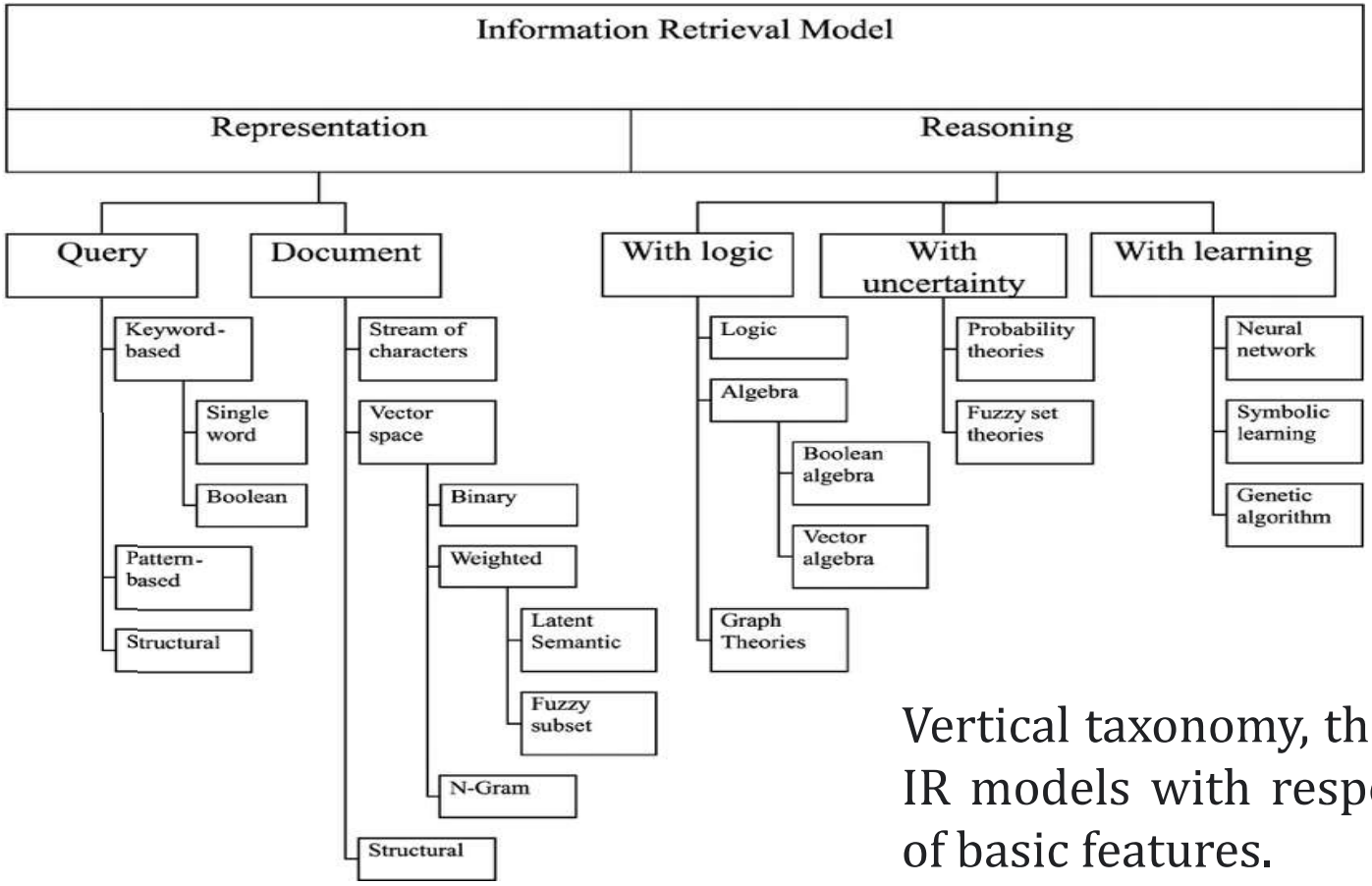
A taxonomy of information retrieval models



A taxonomy of information retrieval models

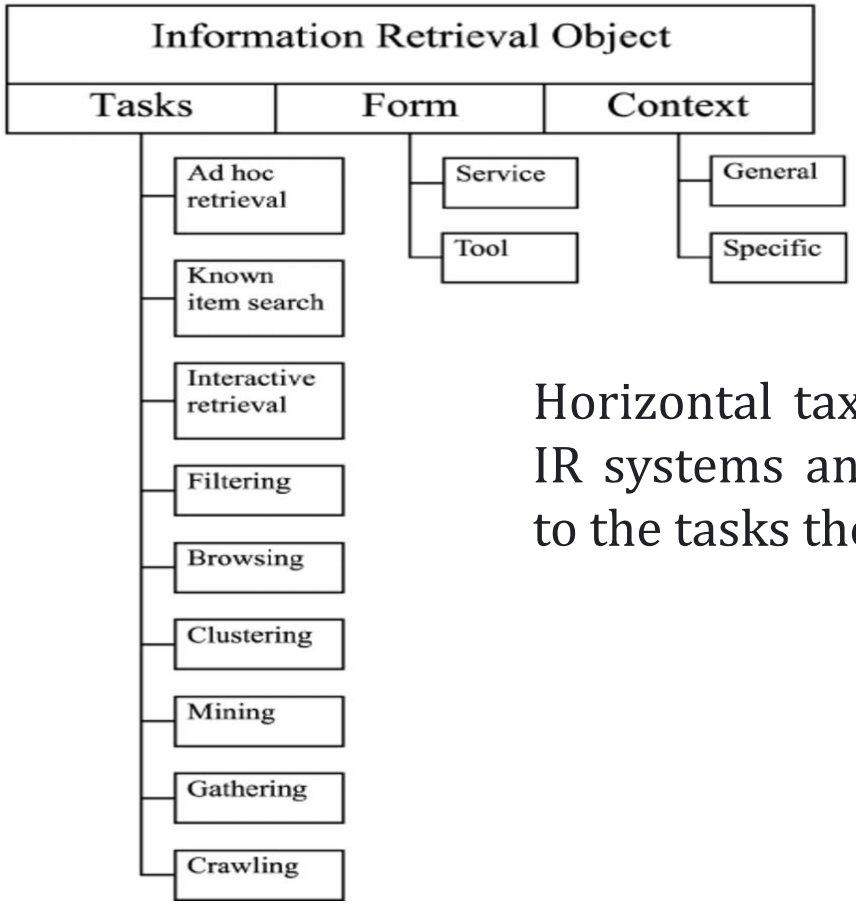
- Taxonomy -- meaning arrangement or division, meaning law -- is **the science of classification according to a predetermined system, with the resulting catalog used to provide a conceptual framework for discussion, analysis or information retrieval**
- Ad hoc retrieval is a standard retrieval task in which the user specifies **his information need through a query which initiates a search (executed by the information system) for documents which are likely to be relevant to the user.**

Vertical Taxonomy



Vertical taxonomy, that classifies IR models with respect to a set of basic features.

Horizontal Taxonomy



Horizontal taxonomy, which classifies IR systems and services with respect to the tasks they support.

Logical View of Documents

	Index Terms	Full Text	Full Text+ Structure
Retrieval	Classic Set Theoretic Algebraic Probabilistic	Classic Set Theoretic Algebraic Probabilistic	Structured
Browsing	Flat	Flat Hypertext	Structure Guided Hypertext

Retrieval : Ad hoc and Filtering

Ad-Hoc Retrieval:

- Imagine you have a big library of books, and you want to find a specific book on a topic, like "Dinosaurs."
- Ad-hoc retrieval is like going to the library, asking the librarian for books about dinosaurs, and getting a list of books that match your request.
- In IR, it means you're searching for information right now, and you want relevant documents based on your current query or need.

Retrieval : Ad hoc and Filtering

Filtering:

- Think of filtering like having a big bowl of mixed fruit, including apples, bananas, and oranges.
- If you only want apples, you use a filter to separate them from the rest. So, you're left with just what you want.
- In IR, filtering means you're narrowing down or refining a set of documents to only get the ones that meet specific criteria or match certain requirements. It's like using a filter to get the most relevant information.

Retrieval : Ad hoc and Filtering

Ad hoc

- The documents in the collection remain relatively static while new queries are submitted to the system.

Filtering

- The queries remain relatively static while new documents come into the system.

Filtering

...

Typically, the filtering task simply indicates to the user the documents which might be of interest to him.

...

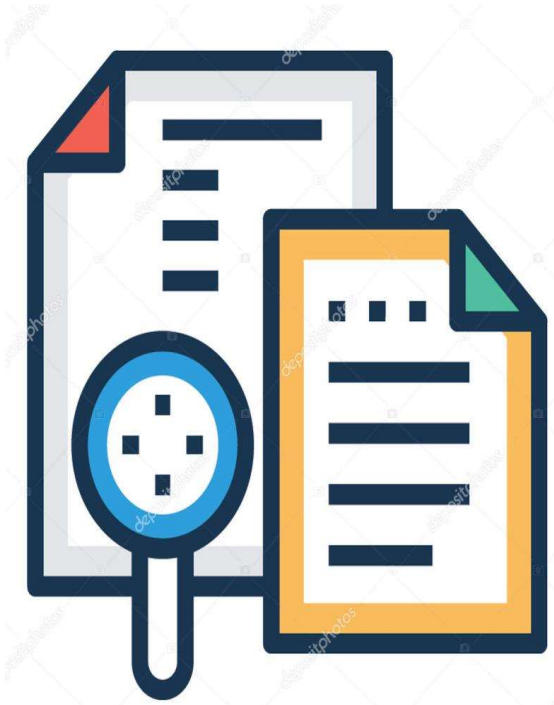
Routing : Rank the filtering documents and show this ranking to the user.

...

Constructing user profiles in two ways.

A formal characterization of IR models

- **D** : A set composed of logical views(or representation) for the **documents** in the collection.
- **Q** : A set composed of logical views(or representation) for the user information needs(**queries**).
- **F** : A **framework** for modeling document representations, queries, and their relationships.
- **$R(q_i, d_j)$** : A **ranking function** which defines an ordering among the documents with regard to the query.



Formulation of the Information Retrieval (IR) Model

- The Information Retrieval (IR) Model can be expressed as:
- **$R(q_i, d_j) = F(Q, D)$**
- In this formulation:
- **$R(q_i, d_j)$** represents the ranking function that orders documents (d_j) based on a specific query (q_i).
- **$F(Q, D)$** denotes the framework that models the relationship between the query and document representations, determining the ranking of documents.
- This formula illustrates that the ranking function is determined by the framework (**F**) applied to the logical views of queries (**Q**) and documents (**D**) in the collection.

Mathematical Formulations for Indexing and Weighting in Information Retrieval

- **k_i**: This represents a generic index term, which is a keyword or term used for indexing and retrieving information.
- **K**: This denotes the set of all index terms, represented as $\{k_1, k_2, ..., k_t\}$, where each k represents a unique index term.
- **$w_{i,j}$**: This represents a weight associated with an index term k_i in a document d_j. It signifies the importance or relevance of the index term to the specific document.

Mathematical Formulations for Indexing and Weighting in Information Retrieval

- **g_i**: A function returns the weight associated with k_i in any t-dimensional vector **($g_i(d_j)=w_{i,j}$).** This is a function that returns the weight (**$w_{i,j}$**) associated with the index term k_i in a document **d_j**. It maps the index term to its corresponding weight in a t-dimensional vector, where **t** represents the total number of unique index terms in the collection.
- **These expressions are related to the representation and weighting of index terms within documents, allowing us to assess the significance of individual terms when searching for information in an Information Retrieval (IR) context.**

Simple Example:

- Suppose we have the following information:
- **k_i**: An index term, let's say "apple."
- **K**: The set of all index terms, {apple, banana, cherry}.
- **$w_{i,j}$**: The weight associated with the index term k_i (apple) in a specific document d_j.

Now, let's consider a specific document:

- **d_j**: Document representing a fruit basket with some text. We want to calculate the weight of the index term "apple" (k_i) within this document.

Simple Example:

- We'll assume that for this document d_j, the weight of "apple" ($w_{i,j}$) is 0.8, which indicates a relatively high relevance of the term "apple" within the document.

So, in this example:

- $\mathbf{k_i}$: "apple" (an index term).
- **K**: {apple, banana, cherry} (the set of all index terms).
- $\mathbf{w_{i,j}}$: 0.8 (the weight associated with "apple" in document d_j).
- $\mathbf{g_i(d_j)}$: When we apply the function g_i to document d_j, it returns the weight of "apple," which is 0.8. Therefore, $g_i(d_j) = 0.8$.
- This simplified example demonstrates how the formulation works. In practice, there would be more index terms and documents, and the weights would be calculated based on various factors, but this illustrates the basic concept of indexing and weighting in Information Retrieval.

Classical Information Retrieval Models:

These are the simplest models to build an Information Retrieval system. These models are **based on the well-recognized and easy to understand the knowledge of mathematics**, for example Probability concepts.

Classical models are easy to implement and are very efficient.

The three classical models of information retrieval are:

- Boolean Model (Set Theoretic)
- Vector Space Model (Statistical/ Algebraic)
- Probabilistic Model

Classic information retrieval model

Basic Concepts:

Each document is described by a set of representative keywords called index terms.

...

Assign a numerical weights to distinct relevance between index terms.

Boolean model

This is the simplest retrieval model which retrieves the information on the basis of the query given in Boolean expression. Boolean queries are queries that uses And, OR and Not Boolean operations to join the query terms.

Based on a binary decision criterion without any notion of a grading scale.

Boolean expressions have precise semantics. It is not simple to translate an information need into a Boolean expression.

you can simplify complex queries or conditions by breaking them down into simpler conditions connected by "OR" (disjunction) and "AND" (conjunction) operators. These conditions are typically represented as vectors within a vector space model, and when expressed this way, it's referred to as being in Disjunctive Normal Form (DNF).

Simple Example:

- Imagine you have a collection of documents, and you want to retrieve documents that contain specific words. Let's say you have three documents:
- **Document A:** "The quick red fox"
- **Document B:** "jumps over the brown dog"
- **Document C**: "A brown dog is quick too"
- Now, you want to find documents that contain both the words "quick" and "brown." In the Boolean Model, you can perform a Boolean **AND** operation to retrieve such documents. Here's how it works:
- **Document A** contains "quick" but not "brown," so it's not a match.
- **Document B** contains "brown" but not "quick," so it's not a match.
- **Document C** contains both "quick" and "brown," so it's a match.
- So, in this simple example, using the Boolean Model with the query "quick AND brown" would retrieve only Document C because it's the only document that satisfies both conditions (contains **"quick" AND "brown"**).

Another Example:

- Ql: Javaid AND Inzimam

Document collection:

- Dl = "Inzimam scores hundred."
- D2 = "Javaid is the most technical batsman of the era."
- D3 = "Inzimam, Javaid duo is the best to watch."
- D4 = "Pakistan won courtesy to Inzimam, Javaid partnership."

Result set:

{Dl, D3, D4} AND {D2, D3, D4} = {D3, D4}

Shortcomings

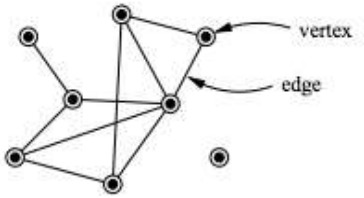
1. The main shortcoming of this model is that it requires **Boolean query instead of free text.**
2. The other disadvantage is that Boolean information retrieval model **cannot rank the documents on the basis of relevance with the user query.** It just gives the document if it contains the query word, regardless the term count in the document or the actual importance of that query word in the document.

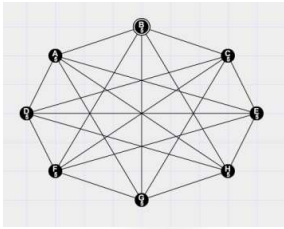
Vector Space Model

- **In vector space model documents and queries are represented as vectors of features representing terms.** Features are assigned some numerical value that is usually some function of frequency of terms.
- **Vector Space Model of Information Retrieval provides rankings of the resulted documents based on the similarity of the query vector with the documents vector**. That is, it provides documents in the order of relevance with the user query.

Illustration...

- In the Vector Space Model (VSM), we treat each document like a special list of numbers. Imagine you have a list of words, and for each word, you have a number that tells you how important that word is in the document.
- Now, think of this like a map.
- **The words are like the lines on the map (up, down, left, right).**
- **The documents are like points on this map.**
- So, in VSM, we put words on the map as lines (terms are the axes), and documents are like points on this map (they live in this space). This helps us find documents when we search for specific words or topics.





VECTOR SPACE MODEL

…

Assign non-binary weights to index terms in queries and in documents.

…

Compute the similarity between documents and query.

…

More precise than Boolean model.

These models represent documents and queries as vectors in a high-dimensional space. The cosine similarity between these vectors is used to determine relevance.

VSM and Ranking Algorithms

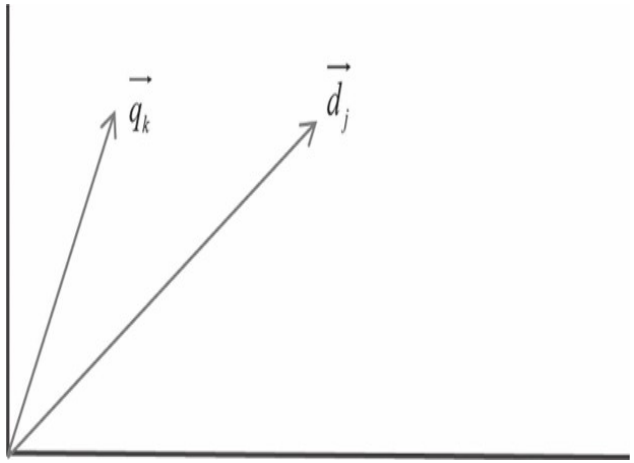
- The Vector Space Model is a fundamental framework for representing and comparing documents and queries in IR, and ranking algorithms are used to order documents based on their relevance to a query within this model.
- Here's how the connection works:
- **Vector Space Model (VSM)**:
  - In VSM, **documents and queries are represented as vectors in a multi-dimensional space.**
  - Each dimension (or axis) corresponds to **a unique term or keyword.**
  - The values in these vectors are typically weighted using techniques like TF-IDF **(Term Frequency-Inverse Document Frequency)** to reflect the importance of terms.
  - VSM provides a way to measure the similarity between a query vector and document vectors, often using **cosine similarity.**

VSM and Ranking Algorithms

- **Ranking Algorithms:**
  - Ranking algorithms in IR are responsible for **determining the order in which documents are presented to users in response to a query**.
  - These algorithms calculate scores that represent **the degree of relevance between a query and each document in the collection**.
  - The higher the score, the more relevant the document is considered to be to the query.
- The connection occurs when ranking algorithms use the VSM framework to compute these relevance scores. Specifically, they calculate the similarity between the query vector and the document vectors within the VSM. Documents with higher similarity scores are ranked higher and presented to the user as potentially more relevant.
- In summary, **the Vector Space Model serves as the foundation for measuring the similarity between queries and documents in IR, and ranking algorithms leverage this model to determine the ranking order of documents for retrieval.**

Relevance Scoring and Document Proximity in Vector Space

- Ranking algorithm compute **similarity between document and query vectors to yield a retrieval score to each document.** The Postulate is: Documents related to the same information are close together in the vector space.



$$\vec{d}_j = (t_{1,j}, t_{2,j}, \ldots, t_{n,j})$$
$$\vec{q}_k = (t_{1,k}, t_{2,k}, \ldots, t_{n,k})$$

Cosine Similarity Formula for Measuring Vector Similarity

- The general formula for calculating the cosine similarity between two vectors is as follows:
- Cosine Similarity ($\cos(\theta)$) = $(A \cdot B) / (||A|| * ||B||)$
- Where:
- A and B are the two vectors being compared.
- $(A \cdot B)$ represents the dot product of vectors A and B.
- $||A||$ represents the Euclidean norm (magnitude) of vector A.
- $||B||$ represents the Euclidean norm (magnitude) of vector B.
- In the context of Information Retrieval (IR), A and B often represent the document vector and query vector, respectively. The cosine similarity score measures the similarity or relevance between the document and the query in a multi-dimensional vector space. Higher cosine similarity scores indicate greater similarity between the two vectors.

Simple Example:

- Sure, here's a simple numerical example based on the provided text:
- Imagine we have a small collection of three documents (D1, D2, and D3) and a query (Q). We'll represent them in a simplified vector space where each dimension corresponds to a single word, and we'll use a simple similarity measure (cosine similarity) to compute retrieval scores.

Document Vectors (D1, D2, D3):

- D1: [1, 1, 2] (The words "cat" and "dog" appear once and "house" twice)
- D2: [2, 1, 1] (The words "cat" appears twice, "dog" once, and "house" once)
- D3: [2, 1, 0] (The words "cat" appears twice, "dog" once, and "house" not at all)

Query Vector (Q):

- Q: [1, 1, 1] (The words "cat," "dog," and "house" each appear once)

Simple Example:

- Now, let's calculate the cosine similarity between the query vector and each document vector:
- Certainly, let's calculate the cosine similarity between the query vector (Q) and each document vector (D1, D2, D3) using the provided vectors:
- **Document Vectors (D1, D2, D3):**
- - D1: [1, 1, 2]
- - D2: [2, 1, 1]
- - D3: [2, 1, 0]
- **Query Vector (Q):**
- - Q: [1, 1, 1]

Cosine similarity calculates similarity based on the cosine of the angle between two vectors. When vectors are similar, the angle between them is small, resulting in a cosine similarity value close to 1. When they are dissimilar, the angle is large, resulting in a cosine similarity value close to -1. This intuitive concept makes it easy to interpret the results.

- The formula for calculating cosine similarity is:
- Cosine Similarity $(\cos(\theta)) = (A \cdot B) / (||A|| * ||B||)$
- Now, let's calculate the similarity for each document:
- 1. $\text{Similarity}(Q, D1) = (Q \cdot D1) / (||Q|| * ||D1||)$
- $= [(1*1) + (1*1) + (1*2)] / [\sqrt{(1^2 + 1^2 + 1^2)} * \sqrt{(1^2 + 1^2 + 2^2)}]$
- $= (1 + 1 + 2) / (\sqrt{3} * \sqrt{6})$
- $= 4 / (\sqrt{18})$

Simple Example:

- 2. Similarity(Q, D2) = (Q · D2) / (||Q|| * ||D2||)
- = [(1*2) + (1*1) + (1*1)] / [√(1^2 + 1^2 + 1^2) * √(2^2 + 1^2 + 1^2)]
- = (2 + 1 + 1) / (√3 * √6)
- = 4 / (√18)

- 3. Similarity(Q, D3) = (Q · D3) / (||Q|| * ||D3||)
- = [(1*2) + (1*1) + (1*0)] / [√(1^2 + 1^2 + 1^2) * √(2^2 + 1^2 + 0^2)]
- = (2 + 1 + 0) / (√3 * √5)
- = 3 / (√15)

- So, the cosine similarity values are as follows:
- - Similarity(Q, D1) ≈ 0.7454 (rounded to four decimal places)
- - Similarity(Q, D2) ≈ 0.7454 (rounded to four decimal places)
- - Similarity(Q, D3) ≈ 0.6708 (rounded to four decimal places)
- **These values represent the cosine similarity between the query vector and each document vector. The higher the value, the more similar the document is to the query.**

Pros and Cons

Vector model is simple and fast. It's a popular retrieval model.

Index terms are assumed to be mutually independent.

It doesn't account for index term dependencies.

Clustering and Classification in Information Retrieval: Organizing Documents for Enhanced Retrieval

- **These topics are important in Information Retrieval because they help in organizing and categorizing documents to improve retrieval efficiency and accuracy.**
- In practice, clustering and classification techniques are often used **in conjunction with retrieval models like the Vector Space Model (VSM)** to enhance the retrieval process.

Simple concepts of Inter and Intra Clustering:

Intra Clustering (Intra-Class):

- Imagine you have a collection of documents about animals.
- Intra clustering is like sorting these documents into smaller groups based on similarities within each group.
- For example, you create one group for documents about "mammals" and another for "birds." Each group contains documents that are similar to each other based on their content.

Inter Clustering (Inter-Class):

- Now, think about having multiple collections of documents, each about different topics.
- Inter clustering is like finding similarities between these different collections.
- For instance, you notice that the "animal" collection and the "science" collection share some common terms like "biology." **Inter clustering helps identify these shared terms and connections between different document sets.**

Another Example:

Intra Clustering (Intra-Class):

- Consider a database of computer science research papers.
- Intra clustering involves organizing these papers into smaller groups based on their topics or keywords.
- For instance, you create one group for papers on "machine learning" and another for "network security." Each group contains papers that are similar to each other in terms of their computer science subject matter.

Inter Clustering (Inter-Class):

- Now, think about having multiple databases of research papers, each from a different subfield of computer science, like "artificial intelligence," "database management," and "software engineering.“
- Inter clustering helps find connections and common themes between these different subfields. For example, you discover that the "machine learning" papers from the AI database share some concepts with the "data modeling" papers from the database management field. Inter clustering helps identify these cross-disciplinary connections in computer science research.

Clustering and Classification in Information Retrieval: Organizing Documents for Enhanced Retrieval

In the field of Information Retrieval (IR), the topics of intra-clustering and inter-clustering are typically discussed within the broader context of **"Text Clustering" or "Text Classification."**

1. **Text Clustering (Intra-Clustering): This topic involves grouping similar documents together into clusters or categories based on their content.** Intra-clustering focuses on the relationships and similarities among documents within the same cluster. **Techniques like hierarchical clustering, k-means clustering, or DBSCAN (Density-Based Spatial Clustering of Applications with Noise) are often discussed in this context.**

Clustering and Classification in Information Retrieval: Organizing Documents for Enhanced Retrieval

2. **Text Classification (Inter-Clustering): Text classification, also known as document classification or categorization, deals with assigning predefined labels or categories to documents based on their content.** It's concerned with the inter-clustering aspect, where documents are categorized into distinct classes or topics. **Techniques like machine learning algorithms (e.g., Naive Bayes, Support Vector Machines) and Natural Language Processing methods are commonly used in text classification.**

Continued...

We think of the documents as a collection C of objects and think of the user query as a specification of a set A of objects. In this scenario, the IR problem can be reduced to the problem of determine which documents are in the set A and which ones are not(i.e., the IR problem can be viewed as a clustering problem).

Intra-cluster

- One needs to determine what are the features which better describe the objects in the set A.

Inter-cluster

- One needs to determine what are the features which better distinguish the objects in the set A.

Term Frequency Vs Inverse Document Frequency

Continued...

Term Frequency

- Inter-clustering similarity is quantified by measuring the raw frequency of a term k_i inside a document d_j, such term frequency is usually referred to as the tf factor and provides one measure of how well that term describes the document contents.

Inverse Document Frequency

- Inter-clustering similarity is quantified by measuring the inverse of the frequency of a term k_i among the documents in the collection. This frequency is often referred to as the inverse document frequency.

Term Frequency (TF):

- Think of it as counting how many times a word (or term) appears in a document.
- It tells you how often a specific word occurs in a piece of text.
- For example, if you have a document about cats, and the word "cat" appears 5 times in that document, the term frequency of "cat" in that document is 5.
- TF is one of the basic building blocks in IR and is often used to measure the importance or relevance of a term within a document or a collection of documents.

Inverse Document Frequency (IDF):

- Inverse Document Frequency (IDF) is a simple concept that helps improve search in Information Retrieval (IR) systems:
- **Imagine you're looking for a rare gem in a treasure chest**.
- If everyone has that gem, it's not very valuable for your search.
- But if only a few people have it, it's much more precious for your quest.
- **IDF is like a measure of how rare or unique a word is in a collection of documents.**
- It helps give more weight to words that are less common because they can be more important for finding relevant documents.
- In practice, **IDF is calculated to boost the importance of words that are less frequent across a collection of documents. This helps IR systems find more relevant documents by emphasizing unique or less common terms.**

Another Example:

- Imagine you have a large collection of computer science research papers. In this collection, the term "**algorithm**" appears in many documents because it's a common word in this field. However, the term "**quantum computing**" is relatively rare and only appears in a few documents.
- **Term Frequency (TF):** If you calculate the TF for "algorithm" in a specific document, you might find that it appears 10 times. For "quantum computing," it appears only once in that document.
- **Inverse Document Frequency (IDF):** Now, you calculate the IDF for both terms across the entire collection. You realize that "algorithm" appears in 90% of the documents, so its IDF is low because it's common. On the other hand, "quantum computing" appears in only 5% of the documents, so its IDF is high because it's rare.
- In this example, IDF helps the Information Retrieval system recognize that "quantum computing" is a more unique and potentially valuable term for retrieving relevant documents. It gives more weight to "quantum computing" when ranking documents, making it easier to find research papers specifically related to that topic in the vast collection of computer science papers.

Formulation:

- The Inverse Document Frequency (IDF) can be simply formulated as:
- **IDF = log(N / (n + 1))**
- Where:

the use of the logarithm in the IDF formula is a standard practice in Information Retrieval to normalize and balance term importance scores, prevent division by zero, and provide a more effective and intuitive measure of a term's rarity or importance within a document collection.

- **N** is the total number of documents in the collection.
- **n** is the number of documents that contain the term you're interested in.
- This formula helps you calculate the IDF score for a specific term in a collection of documents. It quantifies how rare or common a term is in the collection, with higher values indicating that the term is rarer and potentially more valuable for Information Retrieval.

Solved Example:

- **Scenario**: Imagine you have a collection of 1000 documents in a library, and you want to calculate the IDF for the term "computer science" within this collection.
- Total number of documents in the collection (N) = 1000
- Number of documents containing the term "computer science" (n) = 50
- **IDF Calculation**:
- Using the IDF formula: IDF = log(N / (n + 1))
- IDF = log(1000 / (50 + 1))
- IDF = log(1000 / 51)
- IDF ≈ log(19.61)
- IDF ≈ 1.29 (rounded to two decimal places)
- **Interpretation**: The IDF score for the term "computer science" in this collection is approximately 1.29. This means that "computer science" is relatively common in the collection because the IDF score is not exceptionally high. In practical terms, it's a moderate rarity factor.
- This IDF score helps you understand that while "computer science" is not as common as everyday words, it's also not extremely rare within the context of this document collection. It provides a balanced assessment of the term's importance in information retrieval, considering its frequency in the collection.

REFERENCES:

[1] “INFORMATION RETRIEVAL MODELS,” *Ebrary*, 2013. https://ebrary.net/201793/psychology/retrieval_models (accessed Feb. 22, 2023).

[2] “Ch_2 Information Retrieval Models,” *Rutgers.edu*, 2023. https://aspoerri.comminfo.rutgers.edu/InfoCrystal/Ch_2.html (accessed Feb. 22, 2023).

[3] “Fig. 2. Horizontal taxonomy.,” *ResearchGate*, 2021. https://www.researchgate.net/figure/Horizontal-taxonomy_fig2_47397195 (accessed Feb. 22, 2023).

[4] “4.1. IR MODELS – BASIC CONCEPTS – Wachemo University e-Learning Platform,” *Wachemo-elearning.net*, 2023. https://wachemo-elearning.net/courses/information-storage-and-retirievalitec3081/lessons/chapter-four-ir-model/topic/4-1-ir-models-basic-concepts/ (accessed Feb. 22, 2023).

Case Study Assignment-II for IR:

Implement a Basic Document Ranking System


Any
Questions