



Information Retrieval

CSC401/2511 – Natural Language Computing – Winter 2024

Lecture 11

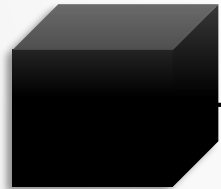
University of Toronto

What is Information Retrieval?

Given a **query**, **search for** the most relevant **document** among a **knowledge base**.

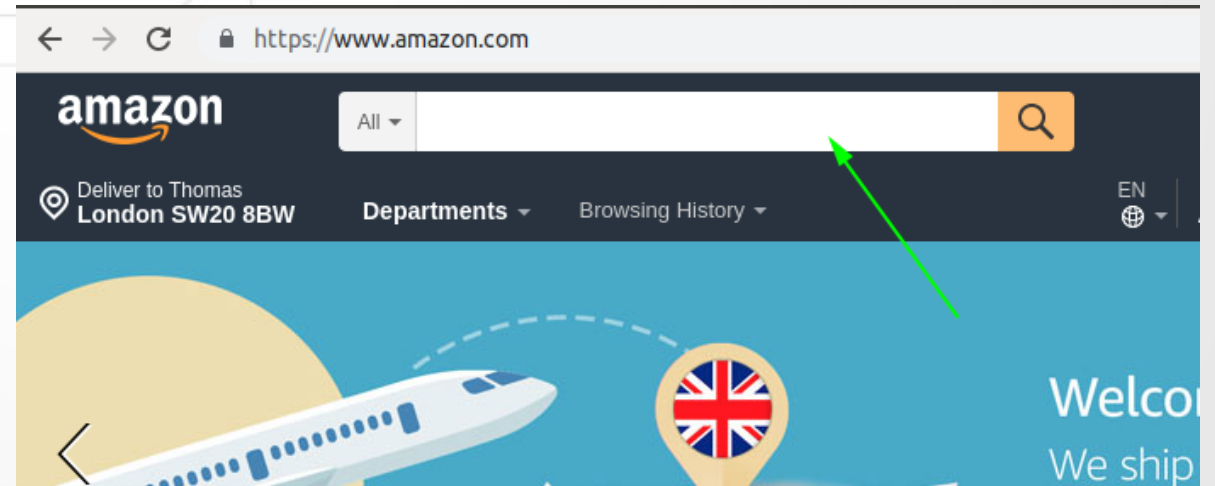


Which woman has won more than 1 Nobel prize?



(Marie Curie)

Search Engines are (mainly) IR systems



Information Retrieval ↔ Information Extraction



Question Answering ↔ Text Summarization

Information Retrieval system

Given a **query**, **search for** the most relevant **document** among a **knowledge base**.

- Three key problems here:
 - **How to represent the query?**
 - **How to store a knowledge base?**
 - **How to search efficiently and accurately?**
- The problems are closely related. We will look at some popular approaches.

Scenario 1: SQL

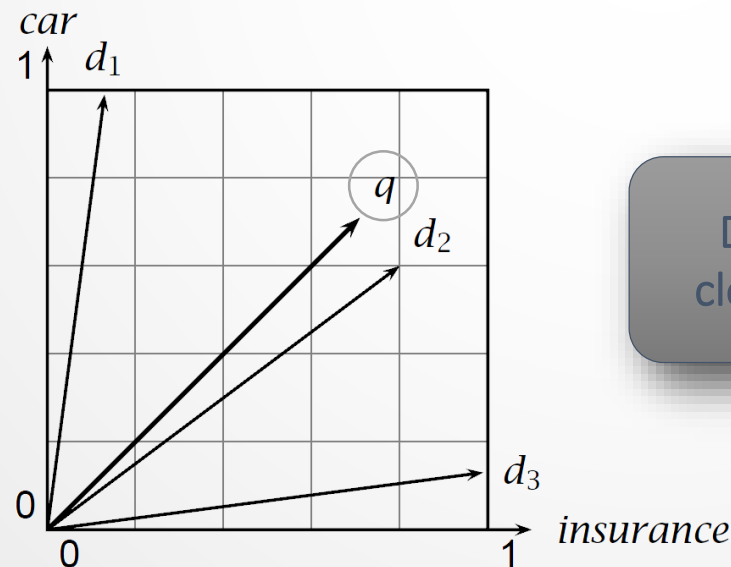
- Structured Query Language (SQL) query
- How to represent the query?
SQL queries.
- How to store a knowledge base?
Tabular entries with predefined schemas.
- How to search efficiently and accurately?
Compile and execute the SQL queries.

Scenario 2: Max-similarity search

- Find the document that is the most similar to the query.
- How to represent the query?
Query is just another text-based document.
- How to store a knowledge base?
Vectorized documents.
- How to search efficiently and accurately?
Compute the **similarity score** between the query and each document. Return the document with the highest similarity score.

Similarity score

- If the query and the available documents can be represented by vectors, we can determine **similarity** according to their **cosine distance**.
 - Vectors that are **near** each other (within a certain **angular radius**) are considered relevant.



Document d_2 is closest to query q .

Vectorization: *tf.idf*

- *tf.idf* is a traditional method to vectorize the documents.
- It starts by weighting *words* in the *documents*.
 - **Term frequency, tf_{ij} :** number of occurrences of word w_i in document d_j .
 - **Document frequency, df_i :** number of documents in which w_i appears.
 - **Collection frequency, cf_i :** total occurrences of w_i in the collection.

Term frequency

- **Higher** values of tf_{ij} (for contentful words) suggest that word w_i is a **good** indicator of the content of document d_j .
 - When considering the relevance of a document d_j to a keyword w_i , tf_{ij} should be **maximized**.
- We often **dampen** tf_{ij} to temper these comparisons.
 - $tf_{dampen} = 1 + \log(tf)$, if $tf > 0$.

Document frequency

- The **document frequency**, df_i , is the number of documents in which w_i appears.
 - **Meaningful** words may occur repeatedly in a related document, but **functional** (or less meaningful) words may be distributed evenly over all documents.

Word	Collection frequency	Document frequency
<i>kernel</i>	10,440	3997
<i>try</i>	10,422	8760

- E.g., *kernel* occurs about as often as *try* in total, but it occurs in fewer documents – it is a more **specific** concept.

Inverse document frequency

- Very specific words, w_i , would give **smaller** values of df_i .
- To maximize specificity, the **inverse document frequency** is

$$idf_i = \log \left(\frac{D}{df_i} \right)$$

where D is the total number of documents
and we scale with log (why? next slide)

- This measure gives **full** weight to words that occur in 1 document, and **zero** weight to words that occur in all documents.

Inverse document frequency

- The probability of a document containing word i is:

$$\frac{df_i}{D}$$

“A document containing word i ” is an event.

Small p : this event is more surprising.

Therefore, more information

- idf_i is the amount of information provided by observing the event.

tf.idf vectorization of a document

- We combine the **term frequency** and the **inverse document frequency** to give us a joint measure of **relatedness** between words and documents:

$$tf.idf(w_i, d_j) = \begin{cases} (1 + \log(tf_{ij})) \log \frac{D}{df_i} & \text{if } tf_{ij} \geq 1 \\ 0 & \text{if } tf_{ij} = 0 \end{cases}$$

- The j^{th} document is therefore represented by a vector:

$$\begin{bmatrix} tf.idf(w_1, d_j), \\ tf.idf(w_2, d_j), \\ \dots, \\ tf.idf(w_{|W|}, d_j) \end{bmatrix}$$

Aside: BM25

- BM25 is a baseline algorithm of IR.
- Given query $Q = [q_1, q_2, \dots, q_n]$, BM25 computes a similarity score for document d_j as:

$$\text{Score}(Q) = \sum_{i=1}^n \log \frac{D}{df_i} \times g(\text{tf}(q_i, d_j); k_1, b)$$

$g(\cdot)$ is an engineered function that has hyperparameters k_1 and b
The details of $g(\cdot)$ are unimportant to our discussion.

Scenario 3: Semantic Doc2Vec

- IR setting: Also using max-similarity search.
- The idea of word2vec can be applied as well.
- Goal: train a document encoder E .
- Design optimization goals for E so that:
 - If d_A and d_B are **close** to each other, then $\text{sim}\langle E(d_A), E(d_B) \rangle$ should be large.
 - If d_A and d_B are **far** from each other, then $\text{sim}\langle E(d_A), E(d_B) \rangle$ should be small.
- The definitions of **closeness** vary from algorithm to algorithm.

Semantic Doc2Vec

- Example: How does the [Contriever paper](#) define the closeness?
 - Positive samples d_+ for a document are **augmented** following some heuristics.
 - Negative samples d_- are **randomly sampled** from within the batches.
- A contrastive loss objective is:

$$L(q, d_+, d_-) = \frac{e^{\text{sim}\langle E(q), E(d_+) \rangle / \tau}}{e^{\text{sim}\langle E(q), E(d_+) \rangle / \tau} + \sum_i e^{\text{sim}\langle E(q), E(d_-) \rangle / \tau}}$$

where τ is the temperature of the softmax.

Evaluating the retrieval systems

- Some commonly used metrics include:
 - Precision
 - Recall
 - F-score
 - Precision @ k

Precision and Recall

- **Precision:** $\frac{N_{\text{relevant \& retrieved}}}{N_{\text{retrieved}}}$
 - Among all **retrieved** documents, how many are relevant?
 - Precision in machine learning: $\frac{TP}{P}$
- **Recall:** $\frac{N_{\text{relevant \& retrieved}}}{N_{\text{relevant}}}$
 - Among all **relevant** documents, how many are retrieved?
 - Recall in machine learning: $\frac{TP}{T}$
- Note: Precision and recall has some tradeoff.

F-measure

- **F-measure** is the weighted harmonic mean of precision and recall:

- $$F = \frac{1}{\alpha \frac{1}{p} + (1-\alpha) \frac{1}{r}}$$

- Where p is precision, r is recall, and $\alpha \in [0,1]$.

- Notes:

- When $\alpha = \frac{1}{2}$, we have $F_1 = \frac{2pr}{p+r}$

- If either of precision or recall is 0 (i.e., true positive count $TP = 0$), then F is arbitrarily set to 0.

Precision at k

- Modern IR systems usually do not just give one result.
 - Even if the 1st result is not relevant, the 2nd, etc. results could be relevant too.
- People sometimes measure the **precision at k ($P@k$)**:
 - Among the top k results, how many of them are relevant?
- **$P@k$** has some potential problems:
 - The 1st, 2nd, ..., k^{th} locations have no differences.
 - If there are less than k relevant results, then even the best system can't get **$P@k=1$** .

Lecture review questions

By the end of this lecture, you should be able to:

- Describe the procedure of max-similarity search.
- Describe the tf.idf vectorization.
- Describe a contrastive objective function of a semantic doc2vec method.
- Identify some evaluation metrics for IR systems and describe the trade-offs between these metrics.

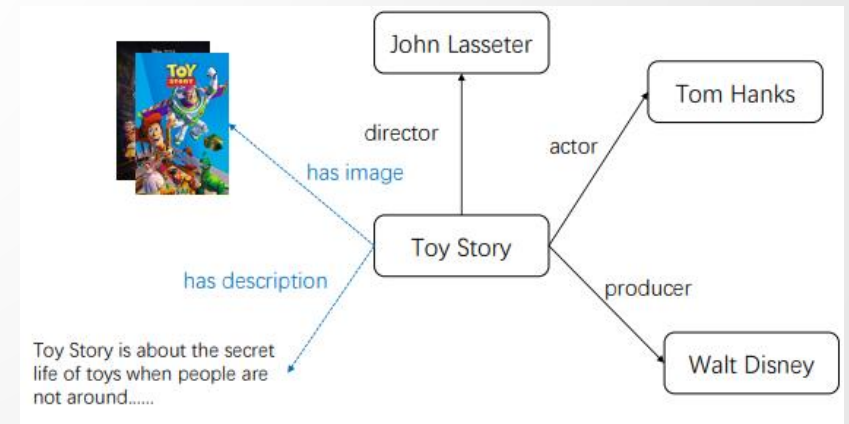
Appendix: Recent challenges of IR

- Structured, relational data
- Multi-modal data

Structured relational data

- Plain texts are **unstructured**.
- Many modern IR systems use **structured** data.
 - E.g., docs vectorized to the same dimensions.
 - E.g., relational data.
- Benefits & challenges of structured data.

```
{  
  "name": "Toy Story",  
  "director": "John Lasseter",  
  ...  
}
```



Storing structured data

- Saving each complex object as a database entry is one option.
- We can also store (or embed) the $\{R, S, T\}$ triplets.
 - R is the **relation** (e.g., “has-director”) between:
 - the **source** S (e.g., “Toy Story”) and
 - the **target** T (e.g., “John Lasseter”)

Multimodal data

- Most modern IR systems are **multimodal**.
- The objects contain more than texts.
 - Images, sounds, even videos are stored too.
 - Choosing the right schemas is very important!

