Week 3 - Evaluation of IR Systems

Introduction

When evaluating a computer system or anything we develop, we're interested in two main aspects: **correctness** and **efficiency**. We want our algorithms to behave as they should and ensure that the results posses the desired features

Standard Testing

Correctness Testing: We can test individual functions to improve correctness. Standard testing requirements involve checking if the system behaves as it should. Essentially, a system should take in a query and return the appropriate answer.

Performance (Efficiency)

- **Response Time (Speed):** Is the system fast enough? we consider many queries, numerous users, different documents, and varying collection sizes. Empirically, we can measure response times in seconds or milliseconds. We assess worst case and average case performance.
- **Space Requirements:** We're also concerned about space, such as index size. How can we efficiently index a large collection?
- **Formal Algorithm Analysis:** We can perform a detailed analysis of the algorithms. For example, using a certain type of index, our search time might be bound to a logarithmic function (e.g., log n). We might develop parallel algorithms and examine the efficiency when partitioning the dataset. These are all standard practices in databases etc, focusing on ensuring the system is both correct and fast

Retrieval Performance

- **Usefulness of Retrieval:** In IR we also need to measure retrieval performance - how useful is the answer set
- **Relevance of Results:** Suppose a user provides a query. We run it against our collection and return a **set *A*** (a set of documents we believe are relevant to the query). We need a way to evaluate how good **set *A*** is.
  - An IR system might be very efficient and fast, but if it doesn't return the correct documents, it's useless
- **Measuring Correctness:** We aim to develop measures of **correctness** - given the query, how accurate was the returned dataset?
  - For example, you might search for a term and receive documents that are not relevant. They contain the query terms but have different meanings or context

Test Collections

A test collection typically consists of:

- **A collection of documents (D):** This is the dataset that the IR system searches through
- **A set of info needs or Queries (Q):** Sometimes, an info need can be directly expressed as a query. In other cases, the info need may be paragraphs long/ a detailed description, from which a query is derived.
- **A list of relevance judgments for each query-document pair:** For every, there is a list of documents that are deemed relevant. This results in a large matrix of queries and documents, typically with binary values indicating relevance (relevant or not)
  - **Example:** Fr query **q1**, documents **doc7** and **doc8** are relevant. If our system is performing correctly it should retrieve both **doc7** and **doc8** when processing **q1**

Evaluating IR Systems Using Test Collections

- **Correctness Measurement:** We assess the **correctness** of our IR system by comparing its outputs to the **gold standard** provided by the tests collection (the **gold standard** in IR refers to a test set created through manual assessments by experts or professionals. These assessors review the results of certain queries an determine which documents are truly relevant to each query based on the info need. This process produces the correct answer set for each query, which is then used as the benchmark for evaluating the performance of an IR System). The notion of correctness is based on how closely the system's retrieved documents match the relevance judgments.
- **Relevance Defined by Information Need:** The idea of relevance is tied to the user's information need, not just the query terms.
  - **Information Need vs Query:** While the information need is what the user actually wants to find out, the query is often a simplified representation of that need.

Understanding Relevance in Retrieval

- **Potential Mismatches:** A document might contain all the query terms and appear relevant, but it may not satisfy the actual information need due to context differences.
  - **Example:** A document includes all the query terms in the correct context, but if it doesn't address the user's specific information need, its not truly relevant
- **Context and Temporal Aspects:** Relevance can be affected by context, temporal factors, or nuances in the information need that aren't captured by the query alone.
  - **Topical Vs Relevant:** A document may be topically related but not relevant if it doesn't align with the user's specific need or if it's outdated.

Tuning and Optimization

- **Weighting Schemes:** In IR systems, various weighting schemes are used to determine the importance of terms and documents.
- **Tuning Parameters:** There may be many ways to tune these weighting schemes. However, over tuning to a specific document collection can make the system too specialized, reducing its effectiveness on other collections.
  - **Avoid Overfitting:** To prevent overfitting a standard approach is to tune the system on a training set and then test it on a separate test set.
- **Generalization:** The goal is to optimize the system so that it performs well across different collections, not just the one it was tuned on.

Challenges in Obtaining Relevance Judgments

- **Cost of Relevance Judgments:** Obtaining relevance judgments can be very costly because it often requires human evaluators to assess whether each document is relevant to a particular query.
- **Crowdsourcing:** To mitigate costs, crowdsourcing is used to distribute the task of relevance judgment to a large pool of people over the internet. This approach can reduce expenses and speed up the process.
- **Pooling Approaches:** Pooling involves combining the top results from multiple IR systems to create a smaller set of documents that are more likely to be relevant. This reduces the number of documents that need to be evaluated.
- **Non-Binary Relevance Judgments:** Relevance judgments don't have to be just binary (relevant or not relevant). They can be graded on a scale to indicate different levels of relevance, providing a more nuanced evaluation.
- **Agreement Among Judges:** Achieving consensus among different judges can be challenging due to subjective interpretations of what is relevant. Disagreements can affect the reliability of the evaluation.

Ad-hoc Retrieval and User Interaction

- **One-Off Queries (Ad-hoc Retrieval):**
  - **Simplistic Evaluation Model:** Traditionally, a user submits a single query, the system retrieves documents, and we evaluate the accuracy of the results.
- **Complexity of Real User Interactions:**
  - **Iterative Process:** In reality, users often engage in a more complex, iterative search process. They may:
    - **Follow Links:** Navigate through hyperlinks to find information.
    - **Modify Queries:** Refine or reformulate their queries based on initial results.
    - **Evolve Information Needs:** Adjust what they're looking for as they learn more
  - **Capturing Additional Aspects:** Evaluations aim to capture these dynamic behaviors to better understand user interactions with the system.

Measuring User Effort and Satisfaction

- **User Effort Metrics:**
  - **Session Duration:** Total time spent during a search session
  - **Number of Queries:** How many times the user had to submit or modify queries.
  - **Document Interactions:** The number of documents the user examined before finding relevant information.
  - **Time to Find Relevant Documents:** How long it took to satisfy the information need.
- **Ask a User For Subjective Measures:**
  - **User Satisfaction:** Feedback on how satisfied users were with the search experience.
  - **Ease of Use:** Whether the system was easy or difficult to use.
  - **Perceived Effectiveness:** Users' opinions on the relevance and quality of the results.

This requires a need for real users - such evaluation require recruiting actual users, which can be time-consuming and costly. This requires more effort - evaluating real user interactions is significantly more complex and requires more resources than algorithmic testing

Precision and Recall

When evaluating an answer set, we aim to measure its quality. This is commonly done using **precision** and **recall**.

- **The Query and Answer Set**
  - A user submits a query **Q**, to a document collection **D**, and receives an answer set **A** (a set of documents returned by the system).
  - In a **Boolean model**, **A** is a distinct set of relevant documents. In more sophisticated models, such as the **vector space model** with a weighting scheme, **A** could be a ranked list, but for simplicity, we can treat it as a set of the top N documents (e.g., the top 50).
- **Relevance and the Ideal Answer Set**
  - For any query, there exists a set **R**, which represents all the documents that are truly relevant to the query, as identified by human judges or evaluators.
  - The goal of a good system is for **A** (the system's returned answer set) to be as close as possible to **R** (the ideal, human-identified relevant set). However, it's challenging since relevant documents may not always contain the exact query terms (e.g., a user searches for "**soccer**" but the relevant document uses "**football**").

$$\text{Precision} = \frac{|R \cap A|}{|A|}$$

$$\text{Recall} = \frac{|R \cap A|}{|R|}$$

Scenario: We have 5 relevant documents **R** = {doc1, doc2, doc3, doc4, doc5}, and the system retrieves 7 documents **A** = {doc1, doc2, doc6, doc7, doc8, doc9, doc10}.

$$\text{Precision} = \frac{|R \cap A|}{|A|} = \frac{2}{7} \approx 0.29$$

Out of 7 retrieved documents, only 2 (doc1, doc2) are relevant, so precision is 29%.

$$\text{Recall} = \frac{|R \cap A|}{|R|} = \frac{2}{5} = 0.4$$

The system found 2 out of 5 relevant documents, so recall is 40%.
Summary: Precision is low (29%) due to many irrelevant documents retrieved, while recall is moderate (40%) as only 2 out of 5 relevant documents were found.

Balancing Precision and Recall

There is often a trade-off between precision and recall

- **Web Search:** users typically care more about precision — they want the first few documents to be highly relevant (e.g., top 10 results for a local restaurant query).
- **legal field:** recall may be prioritized. Users are willing to review many irrelevant documents as long as they retrieve all the relevant ones (e.g., finding every case related to a legal query).

F-Measure

To combine precision and recall into a single value, we use the **F-measure**, which is a weighted harmonic mean of precision **P** and recall **R**

- The F-measure can be tuned by adjusting the weighting depending on whether precision or recall is more important for a particular task

Calculating & Evaluation

Precision-Recall plots

- Returned documents are usually ranked.
- Precision-Recall plots are used to evaluate system performance.
- Typically, precision is plotted against recall.
- In an ideal system, for a recall value of 1, precision would also be 1.
- This means all relevant documents are returned with no irrelevant documents.

When calculating precision and recall, we often deal with ranked lists rather than simple sets. In these cases, we evaluate precision and recall progressively as we go down the ranked list.

- **Ranked Lists Vs Sets:**
  - In theory, calculating precision and recall is straightforward when the answer set is a strict set. However, in practice, search systems typically return ranked lists.
  - As we move down the ranked list, we calculate precision and recall at different points.

- **Progressive Calculation:**
  - Since the list can be long (e.g., thousands of documents), systems tend to calculate precision at specific recall thresholds. For example, as the system retrieves documents
  - As we move down the ranked list, we calculate precision and recall at different points.
    - Once **10%** of the relevant documents are found **(0.1 recall)**, we calculate precision at that point.
    - Then, we continue until **20%** of the relevant documents are found **(0.2 recall)** and calculate precision again.

Example:

- Let $|D| = 20$ (total documents) and $|R| = 10$ (relevant documents).
- A ranked list of length 10 is returned: **d1**, **d2**, $d3$, **d4**, $d5$, $d6$, **d7**, $d8$, $d9$, $d10$, where the relevant documents are in bold

Calculating precision and recall:

- As far as the first document: Precision = 1, Recall = 0.1
- As far as the first 2 documents: Precision = 1, Recall = 0.2
- As far as the first 3 documents: Precision = $\frac{2}{3} \approx 0.67$, Recall = 0.2

Usually, precision is calculated for recall values = **10% to 90%**.

Averaging Precision Across Multiple Queries

- To account for query variability, we calculate precision at different recall points for a set of queries (e.g., Q1, Q2, Q3).
- For each recall point (e.g., 0.1 recall), we average the precision across all queries
- By averaging over multiple queries, we obtain a more reliable measure of the system's overall performance.

Typically calculate precision for these recall values over a set of queries to get a truer measure of a system's performance. The average precision at a given recall r across N queries is given by:

$$P(r) = \frac{1}{N} \sum_{i=1}^{N} P_i(r)$$

Single Value Measures:

- **1. Precision when every relevant document is retrieved:** Calculate precision at each relevant document and average these values.
- **2. Precision at first relevant document:** Evaluate precision when the first relevant document is retrieved to measure early system performance.
- **3. R-precision:** Precision at the point when all relevant documents have been retrieved.
- **4. Precision at k (P@k):** Precision for the top k retrieved documents.
- **5. Mean Average Precision (MAP):** is a widely used metric for evaluating IR systems. It allows us to calculate precision at multiple recall levels for a ranked list of documents, then average that across multiple queries. By doing this, we can compare different systems based on average precision across, say, 1000 queries.

Precision histograms provide a way to visualize and compare the precision of two systems at different points in the ranked list.

Advantages

- They are widely used and provide a **definable measure** that allows different systems to be tested and compared.
- They **summarize system behavior**, providing insight into overall performance and potential anomalies.

Disadvantages

- It's not always possible to calculate recall, particularly in batch mode, and precision/recall doesn't account for the iterative nature of real-world querying, where users frequently modify their queries.
- Precision/recall graphs require ranked documents and may not fully reflect the user's interests or needs.

User-Oriented Measures

In user-centered IR systems (like Google, Netflix, Spotify), how do we capture user satisfaction beyond precision and recall

- **User Knowledge:**
  - Suppose we have a document collection **(D)**, a set of relevant documents **(R)**, an answer set **(A)** and a set of relevant documents already known to the user **(U)**
- How does this influence the user's view of the system? It depends on the context for example:
  - If I use Google and search for query expansion mechanisms, I want to see **novel** information – don't give me documents I already know about.
  - If Netflix recommends shows I've already seen, that's not a good recommendation. I'm looking for novelty in this context.
- **Novelty vs. Coverage:**
  - Let $\boldsymbol{AU}$ be the set of returned documents previously known to the user.

$$\text{Coverage} = \frac{|AU|}{|U|}$$

  - Let $\boldsymbol{New}$ be the set of relevant documents returned that were previously unknown to the user. Novelty can be defined as:

$$\text{Novelty} = \frac{|New|}{|New|+|AU|}$$

  - **Novelty:** The user seeks new, unknown relevant documents.
  - **Coverage:** The user wants to retrieve all relevant documents, especially ones they do not already know.

Disadvantages

Evaluating user satisfaction in interactive search sessions is much more difficult compared to traditional IR metrics. Much of this work comes from the field of **Human-Computer Interaction (HCI)**, where systems are evaluated through:

- **User Behavior Monitoring:** Observing how users interact with the system.
- **Surveys:** Gathering feedback on the user experience.

Information visualization is another important area, focusing on how to best represent retrieved data for users.