Exercise 3.1
What does the Probabilistic Ranking Principle state?

Exercise 3.2
Both Probabilistic Indexing (PI) and Binary Independence Retrieval (BIR) try to estimate \( \Pr(d \in R_q) \) using Bayes’ Theorem. What are the two different approaches used here? Do not use any formulas!

Exercise 3.3
Both PI and BIR make an independence assumption. Explain each assumption in your own words (do not use any formulas!) and discuss whether it is reasonable or not.

Exercise 3.4
What do Zipf’s Law and Heaps’ Law state? Use your own words. What are the consequences of these laws for information retrieval systems?

Exercise 3.5
What are the advantages and disadvantages of stemming?
Exercise 3.6
Why do we need inverted indexes in information retrieval?

Exercise 3.7
Hard disks are extremely cheap. Then, why waste CPU time by compressing and decompressing inverted indexes?

Exercise 3.8
LSI can be used to recognize synonyms, antonyms, and semantically related terms. Briefly explain how this works. Use your own words, no not use any formulas.

Exercise 3.9
LSI also has interesting applications in automated document translation. Can you imagine how this works?

The oral exams will be held at the following dates:
• Monday, August 15
• Monday, September 26
• Tuesday, September 27
• Wednesday, September 28
• Thursday, September 29
• Friday, September 30

Please make an appointment with our secretary Regine Dalkran.

Lecture 6: Language Models and Evaluation
1. Language Models
2. Evaluation of IR Systems
Formal Grammars

• Why formal grammars will not help us:
  – Grammars capture syntactical correctness but not style
  – Natural language does not strictly obey grammar rules
  – The writing style or topic of a document largely depends on how typical words, phrases, or sentences look like
  – Formal grammars fail to capture statistical properties of text, they just describe the set of “correct” documents

Statistical Language Models

• A statistical language model consists of probability distributions:
  – For any given n, there is a probability distribution such that every document \( w_1, \ldots, w_n \) of length n (word count) gets assigned its probability of generation \( \Pr(w_1, \ldots, w_n) \)

• Example:
  – Assume that only the words “cat” and “dog” are generated

<table>
<thead>
<tr>
<th>n=1</th>
<th>( \Pr(\text{cat}) )</th>
<th>( \Pr(\text{dog}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>1</td>
<td>0.7</td>
<td>0.3</td>
</tr>
<tr>
<td>2</td>
<td>0.2</td>
<td>0.8</td>
</tr>
</tbody>
</table>

• How to describe “language” within a formal model?
  – Well-known from theoretical computer science: Formal grammars
  – A way to describe correct syntax
    – Example:
      - sentence \( \rightarrow \) noun_phrase, verb_phrase
      - verb_phrase \( \rightarrow \) verb, noun_phrase
      - verb \( \rightarrow \) noun_phrase, the man

• Observation: There are many different styles of writing, especially depending on topics
  – For example, political news articles use a completely different vocabulary than personal blog entries
  – There are models available to describe such “languages”
    – Each topic corresponds to a specific language
    – Represent each document by its corresponding language model (different parameters)
  – Querying then becomes:
    – To which document’s language model the query fits best?

• Usually, some structure is assumed
  – Unigram model (assume independence, ignore context):
    \[ \Pr(w_1, \ldots, w_n) = \Pr(w_1) \cdot \Pr(w_2) \cdot \ldots \cdot \Pr(w_n) \]
  – Bigram model (assume dependence on the previous word only):
    \[ \Pr(w_1, w_2) = \Pr(w_1) \cdot \Pr(w_2 | w_1) = \Pr(w_1, w_2) \cdot \Pr(w_3 | w_2) \cdot \ldots \cdot \Pr(w_n | w_{n-1}) \]
  – Trigram model (assume dependence on the previous two words):
    \[ \Pr(w_1, w_2, w_3) = \Pr(w_1) \cdot \Pr(w_2 | w_1) \cdot \Pr(w_3 | w_2, w_1) \cdot \Pr(w_4 | w_3, w_2, w_1) \cdots \]
Statistical Language Models (3)

Example of a three-word bigram model:

<table>
<thead>
<tr>
<th>word</th>
<th>P(word)</th>
<th>Pr(new</th>
<th>column)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>0.4</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>dog</td>
<td>0.5</td>
<td>0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>mouse</td>
<td>0.1</td>
<td>0.8</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Some randomly generated 6-word sentences:
- dog mouse cat mouse cat mouse
dog dog dog mouse cat mouse
dog mouse cat mouse cat mouse
cat mouse cat dog mouse cat
cat mouse cat mouse cat mouse

Statistical Language Models (4)

Observation: Generative models can be used to:
- generate documents, or
- recognize documents

Document recognition:
- “Which document fits a given model best?”
- Usually based on probabilities of generation
- Popular applications: OCR, speech recognition, …

Language Models in IR

- How to apply language models in information retrieval?
- Assumptions:
  - For each document, there is a “true” (but unknown) statistical document model
  - Each document was generated from its corresponding model by a random generation process, i.e., it is a random sample
  - The query also is a sample or a description of an underlying language model describing the user’s information need

Typical application of language models in IR:
1. Estimate a model for each document
2. For each estimated model, compute the probability of generating the query
3. Rank documents by these probabilities

Language Models in IR (2)

Typical application of language models in IR:
1. Estimate a model for each document
2. For each estimated model, compute the probability of generating the query
3. Rank documents by these probabilities

Language Models in IR (3)

- Open problems:
  - How to estimate the “true” language models from the observations (i.e., documents) we have?
  - Which language model should we use (unigram, bigram, …)?
- For practical reasons, unigram models are used in IR (sometimes bigram models)
- “Practical reasons” refers to:
  - Reduced computational complexity
  - Problem of sparse data:
    Documents usually are short and its size and content are fixed
    Losses from data sparseness (i.e., bad estimations) tend to outweigh any gains from richer models

Language Models in IR (4)

- We will deal with unigram models only
- Now, how to estimate the “true” models?
- Straightforward approach:
  - Given: Document \( d = (w_1, \ldots, w_n) \)
  - Estimate \( \Pr(w) \) by \( \frac{\# w}{\# d} \)
  - This is the maximum likelihood estimator (MLE)
- Example:
  \( d = (\text{the, big, dog, jumps, over, the, small, dog}) \)
  - Estimate \( \Pr(\text{dog}) \) by \( \frac{2}{8} = 0.25 \)
  - Estimate \( \Pr(\text{cat}) \) by \( 0 \)
Language Models in IR (5)

• Problem of the MLE approach:
  Document size often is too small
  – Many terms would be missing in a doc, which implies a zero probability estimate
  – Probability of terms occurring once in the document normally is overestimated, because this occurrence was partly by chance

• Solution: Smoothing
  – Allocate some probability mass to missing terms
  – Pull all estimates in the direction of the collection mean
  – There are many ways to do this

Simple smoothing (as used in TF-IDF):
– Add some small number $\alpha$ (e.g. 1 or 0.5) to all observed counts
– Renormalize to give a probability distribution

Example (use $\alpha = 1$):

<table>
<thead>
<tr>
<th>Term</th>
<th>Initial Estimate</th>
<th>Smoother Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>3/8</td>
<td>3/15</td>
</tr>
<tr>
<td>big</td>
<td>2/8</td>
<td>2/15</td>
</tr>
<tr>
<td>dog</td>
<td>3/8</td>
<td>3/15</td>
</tr>
<tr>
<td>jumps</td>
<td>2/8</td>
<td>2/15</td>
</tr>
<tr>
<td>over</td>
<td>2/8</td>
<td>2/15</td>
</tr>
<tr>
<td>small</td>
<td>2/8</td>
<td>2/15</td>
</tr>
<tr>
<td>cat</td>
<td>1/8</td>
<td>1/15</td>
</tr>
</tbody>
</table>

normalize (divide by 15/8)

Linear smoothing:
– Estimate $Pr(w_i)$ by
  \[ \frac{\lambda \cdot df(w_i)}{n} + (1 - \lambda) \cdot \frac{cf(w_i)}{N} \]
– $n$: document size
– $cf(w_i)$: collection frequency of $w_i$, i.e., the number of occurrences of $w_i$ in the whole collection
– $N$: collection size, i.e., number of words in the whole collection
– $\lambda$: some parameter between 0 and 1

Approach by Ponte and Croft (1998):
– Use corpus data to stabilize document model
– If a term is missing in the document: Estimate its probability by its corpus probability (i.e., use an MLE over the whole collection)
– If a term appears in the document: Smooth MLE using average MLE (over all documents containing the term)

There are many more advanced smoothing methods...

Pros and Cons of Language Models

Pros:
– Clear statistical foundation, no ad hoc weightings
– Collection statistics are an integral part of the model, rather than being used heuristically
– Works good, comparable to the vector space model

Cons:
– Independence assumption in unigram model
– No explicit notion of relevance, integration of user feedback is difficult
Deepdyve:

- Some old marketing claims:
  - “The content is your query”
  - “Queries can be words, phrases, paragraphs, formulas, whole documents or even sets of documents”
  - “It indexes every word, as well as every phrase in each document, and weighs their informational impact using advanced statistical computation”
  - “It is language independent”
  - “Results are arranged by topic”
- Claims are gone, but Deepdyve still encourages users to use whole paragraphs as queries

What to Evaluate?

What should be evaluated in IR?

- Efficiency:
  - Use of system resources
  - Scalability
- Effectiveness:
  - Result quality
  - Usability

Efficiency

- Efficiency:
  - Storage space
  - CPU time
  - Number of I/O operations
  - Response time
- Depends on hardware and software
- Goal in IR: “be efficient enough”
- Efficiency usually is easy to evaluate, therefore it will not be discussed here any further

Effectiveness

- Effectiveness: How to measure result quality?
- Key concept is relevance
- There is no fully satisfactory definition of relevance
- The same problem as with “information” and “intelligence”…
- What we will do next?
  - Point out some important aspects of relevance
  - Give a pragmatic approach from the system builder’s point of view
- Fortunately, often we don’t need a precise definition (think of probabilistic retrieval)

Relevance is Multidimensional

- Saracevic (2007) identifies five manifestations of relevance:
  - System or algorithmic relevance
  - Topical or subject relevance
  - Cognitive relevance or pertinence
  - Situational relevance or utility
  - Affective relevance
System or Algorithmic Relevance

- **System or algorithmic relevance:**
  - Relevance as a static and objective concept
  - Relevance can be judged by some algorithm:
    - “How close is the fit between the retrieved set of documents and the user’s query?”
    - Not influenced by users
  - The most common and clearest definition of relevance
  - “How well does the topic of the retrieved information match the topic of the request?” (Problem: “topic” is undefined)
  - Example: “Vector space model relevance”

Topical or Subjective Relevance

- **Topical or subject relevance:**
  - Relevance as a subjective or user-based concept
  - Still a static concept
  - The concept of topic is understood as aboutness, not contents, i.e., an intellectual assessment of how a document corresponds to the topical area required and described by the query
  - “How close is the semantic fit between the query and the topics of the document retrieved?”
  - Consequently, based on judgments
  - Documents may be assessed for aboutness independent of the query

Cognitive Relevance or Pertinence

- **Cognitive relevance or pertinence:**
  - Again, subjective
  - Relevance as relation between documents and the cognitive state of knowledge and information need of a user
  - “What is the user’s judgment about the applicability of the retrieved documents to the matter at hand?”
  - Relevance may be dynamic, i.e., change over session time

Situational Relevance or Utility

- **Situational relevance or utility:**
  - Again, subjective and dynamic
  - Relevance as the relation between the situation, task, or problem at hand, and documents
  - “Do the retrieved items allow the user to complete the task at hand?”
  - Involves serendipity: Information may be useful although you did not expect this in advance

Affective Relevance

- **Affective relevance:**
  - Again, subjective and dynamic
  - Relevance as the relation between documents and the intents, goals, emotions, and motivations of a user
  - Represents the human drive for information

Manifestations of Relevance

<table>
<thead>
<tr>
<th>Type of Relevance</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>System or algorithmic relevance</td>
<td>Rules for comparative judgments</td>
</tr>
<tr>
<td>Topical or subject relevance</td>
<td>Aboutness</td>
</tr>
<tr>
<td>Cognitive relevance or pertinence</td>
<td>Informativeness, novelty, information quality,</td>
</tr>
<tr>
<td>Situational relevance or utility</td>
<td>Usefulness in decision making, appropriateness of information in resolution of a problem, reduction of uncertainty, …</td>
</tr>
<tr>
<td>Affective relevance</td>
<td>Satisfaction, success, accomplishment, …</td>
</tr>
</tbody>
</table>
What's Our Notion of Relevance?

• Our notion of relevance: Topical or subject relevance
• Current goal of IR:
  – Build an algorithm resembling topical relevance for “most” users
• Future goals (current research):
  – Address the other subjective manifestations of relevance

Evaluating Relevance

• Back to our initial question:
  How to evaluate a system’s result quality?
• Traditional approach: Evaluation benchmarks
  – A benchmark document collection
  – A benchmark suite of information needs, expressible as queries
  – An assessment of the relevance of each query–document pair, called “gold standard” or “ground truth”
    • Usually, relevance is assessed in binary fashion
  – Example of an information need:
    – “What are the prospects of the Quebec separatists achieving independence from the rest of Canada?”

Cranfield Collection

• The Cranfield collection:
  – Pioneering test collection
  – Cranfield University (UK)
  – 1960s
  – Total size: 1.6 Mbytes
  – 1400 abstracts of aerodynamics (aircraft design) journal articles
  – 225 queries generated by some of the documents’ authors
  – Exhaustive relevance judgments for all query–document pairs (done by students and “experts”)

Queries and Information Needs

• Relevance vs. pertinence:
  • Be careful:
    – Often “relevant to a query” means “relevant to a ‘typical’ information need that fits the query”

Evaluating Relevance (2)

• How to completely assess very large collections?
• The pooling method is widely used:
  – Run each query on a set of very different IR systems
  – “Pool” their results to form a set of documents, which have at least this recommendation of potential relevance
    (usually, take top k results from each system)
  – The union of these retrieved sets is presented to human judges for relevance assessment
  – Assumption: Unassessed documents are irrelevant!

Cranfield Collection (2)

• Rating scale used for relevance judgments:
  1. References which are a complete answer to the question
  2. References of a high degree of relevance, the lack of which either would have made the research impracticable or would have resulted in a considerable amount of extra work
  3. References which were useful, either as general background to the work or as suggesting methods of tackling certain aspects of the work
  4. References of minimum interest, for example, those that have been included from an historical viewpoint
  5. References of no interest
Example document:
- “viscous flow along a flat plate moving at high speeds. by the distortion of coordinates, it is shown that, in the case of supersonic viscous flow past a flat plate, the boundary-layer and simple wave theories can be combined to give a complete representation of the velocity and pressure fields. [...]”

Example query:
- “why does the compressibility transformation fail to correlate the high speed data for helium and air”
First, we deal with the evaluation of IR systems that return result sets, i.e., they do not provide any ranking.

**Idea:** Compare result set with ground truth result set.

What sets are involved here?

- **False positives:**
  - Irrelevant documents returned by the system
  - Extend the result set unnecessarily
  - Often inevitable
  - Usually can be filtered out by the user quite easily

- **False negatives:**
  - Relevant documents not returned by the system
  - Problematic, since the user usually is not aware of them
  - Often worse than false positives

Remaining sets: True positives and true negatives

**Precision, recall, and fallout are important measures of (unranked) answer sets.**

**Precision:**
- Uses the number of true positives as measure of result quality
- How many of the returned documents are relevant?
- Definition:
  \[
  \text{Precision} = \frac{\#(\text{relevant items retrieved})}{\#(\text{items retrieved})}
  \]
- Value in [0, 1], where 1 is best
- High precision usually is important in Web search (result set = first page of results)

**Recall:**
- Also uses the number of true positives as measure of quality
- How many of all relevant documents have been returned?
- Definition:
  \[
  \text{Recall} = \frac{\#(\text{relevant items retrieved})}{\#(\text{relevant items})}
  \]
- Value in [0, 1], where 1 is best
- High recall usually is important for professional searchers such as paralegals and intelligence analysts; it is also important for desktop search
Precision and recall clearly trade off against one another:

- Achieve perfect recall (but awful precision) by always returning all documents in the collection.
- Achieve very good precision (but awful recall) by always returning only the single result that seems to match best.

Normally, this leads to tradeoffs in system tuning:

- Small result sets usually lead to better precision but worse recall.

What about measurement?

- Precision is easy to measure.
- Measuring recall is at least very difficult, and often impossible.

Fallout:

- Uses false positives to measure retrieval quality.
- How many returned documents have been nonrelevant?

Definition:

- Value in $[0, 1]$, where 0 is best.
- Zero fallout can be achieved by returning empty result sets.
- Fallout usually only makes sense for large result sets.
- For typical queries, most documents in the collection are nonrelevant.

The F measure combines precision and recall:

- It’s a weighted harmonic mean of precision and recall.
- Definition:

$$F = \frac{1}{\frac{\alpha}{\text{precision}} + \frac{1 - \alpha}{\text{recall}}}$$

- Parameterized by weighting factor $\alpha \in [0, 1]$.
- Balanced F measure: $\alpha = \frac{1}{2}$.
- Value in $[0, 1]$, where 1 is best.
- Why do we use the harmonic mean?

With the arithmetic mean, an F measure of 0.5 could easily be achieved e.g. by returning all documents.

Ordered Result Lists

- Now, how to evaluate ordered result lists?
  - Idea: Compute precision and recall for the set of the top $k$ retrieved documents; repeat this for many different $k$.
  - We then get the precision at $k$ and the recall at $k$.
- Example result list (assume there are 5 relevant docs):

<table>
<thead>
<tr>
<th># Relevant?</th>
<th># Relevant</th>
<th>Recall at k</th>
<th>Precision at k</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>1/5 = 0.2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Yes</td>
<td>2/5 = 0.4</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>3/5 = 0.6</td>
<td>3/5 = 0.67</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>3/5 = 0.6</td>
<td>3/4 = 0.75</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>3/5 = 0.6</td>
<td>3/5 = 0.6</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>3/5 = 0.6</td>
<td>3/6 = 0.5</td>
</tr>
</tbody>
</table>

- Plotting precision at $k$ and recall at $k$, for many $k$, again gives us a precision-recall curve.
- Example from (Manning et al., 2008):
To get rid of the sawtooth shape, we can use the interpolated precision at a certain recall level instead.

**Definition:**
The interpolated precision at recall level \( r \) is the highest precision found for any recall level \( r' > r \).

TREC uses eleven-point interpolated average precision:
- Recall levels used are 0.0, 0.1, ..., 1.0
- Precision values are averaged over many different queries

Averaged eleven-point interpolated precision/recall; example from (Manning et al., 2008):

Some people like single aggregate values instead of curves.

A popular one is the mean average precision (MAP).

**Definition:**
1. Compute precision at \( k \), for any \( k \) such that there is a relevant document at position \( k \) in the result list
2. Then compute the arithmetic mean of all these precision values
3. Compute the mean over many different queries; this value is the mean average precision of the IR system

MAP has been shown to have especially good discrimination and stability.

Broadly spoken: MAP is the average area under the precision–recall curve for a set of queries.