IR Evaluation and re-ranking

COMP90042 Lecture 6

Overview

- Evaluation
- Re-ranking documents
- Learning-to-Rank

Efficiency vs effectiveness

- Up until now we looked at how to make an inverted index
  - Space efficient (Compression)
  - Fast (Top-K query processing)
- Today we will investigate the **quality** of returned results for a search query:
  - How do you measure quality?
  - Ways to improve result quality.

Evaluating effectiveness

- Hard to characterise the quality of a system's results
  - a **subjective** problem, depends on the user's information need and how well the results meet that need
  - query is **not** the **information need** itself, but an expression thereof
- Obvious evaluation method: human judgements
  - directly measure effectiveness in user studies; for reported satisfaction, completion of tasks, …
  - but too expensive and slow, especially when tuning parameters of the system (e.g., flavour of TF*IDF, use of stopwords, etc…)

Automatic evaluation

- Make simplifying assumptions
  - retrieval is ad-hoc
    - query performed only once, and with no prior knowledge of the user or their behavior
  - effectiveness based on relevance
    - each document is either relevant or irrelevant to information need (often binary, sometimes also multiple grades of relevance)
    - relevance of documents are independent from others (no consideration of redundancy)
- Effectiveness is a function of the relevance of documents returned by the system

Test collections

- Several reusable test collections constructed for IR evaluation, e.g., for TREC competitions; comprising
  - ***corpus*** of documents
  - set of ***queries***, sometimes including long-form elaboration of information need
  - relevance judgements (***qrels***), a human judgement of whether the document is relevant to the information need in the given query.
- Typically not all documents have *qrels,* collection is simply too big and most are likely to be irrelevant.

Example from TREC 5

⟨num⟩ Number: 252
⟨title⟩ Topic: Combating Alien Smuggling
⟨desc⟩ Description: What steps are being taken by governmental or even private entities world-wide to stop the smuggling of aliens.
⟨narr⟩ Narrative: To be relevant, a document must describe an effort being made (other than routine border patrols) in any country of the world to prevent the illegal penetration of aliens across borders.

Qrels

Topic	Docid	Rel
252	AP881226-0140	1
252	AP881227-0083	0
252	CR93E-10038	0
252	CR93E-1004	0
252	CR93E-10211	0
252	CR93E-10529	1
...		

Runfile

Topic	Docid	Score
252	CR93H-9548	0.5436
252	CR93H-12789	0.4958
252	CR93H-10580	0.4633
252	CR93H-14389	0.4616
252	AP880828-0030	0.4523
252	CR93H-10986	0.4383
...		

Example relevance vector

- Based on retrieval run, calculate binary vector indicating relevance for each ranked document

Retrieval run

Docid	Score
CR93H-9548	0.5436
CR93H-12789	0.4958
CR93H-10580	0.4633
CR93H-14389	0.4616
AP880828-0030	0.4523
CR93H-10986	0.4383
...	

Qrels

Docid	Rel
AP880828-0030	0
AP881226-0140	1
AP881227-0083	0
CR93H-14389	0
CR93H-9548	1
CR93H-10580	0
CR93H-10986	1
CR93H-12789	0
...	
...	

Relevance vector

$$\langle 1, 0, 0, 0, 0, 1, \ldots \rangle$$

Relevance measures

- How to map relevance vector to a number?
- Natural candidates are precision & recall
  - but recall is hard to calculate (why?); and
  - how to deal with ranked outputs?
- Mainly use precision oriented metrics:
  - **precision @ k**: compute precision using only ranks 1 .. k
  - **average precision**: take average over prec@k for each k where rank k item is relevant; measure becomes *rank sensitive*
  - ***Mean Average Precision (MAP):*** *AP averaged across multiple queries*

Relevance example

- Relevance vector

  < 1, 0, 0, 0, 0, 1, 0, 1, 0, 0 >

- Precision
  - P@1 = 1/1P@2 = ½ P@3 = 1/3 P@4 = ¼
    P@5 = 1/5P@6 = 2/6 P@7 = 2/7 P@8 = 3/8
    P@9 = 3/9P@10 = 3/10

- AP (average precision) = 1/3 *(P@1 + P@6 + P@8) = 0.57 (assuming only 3 docs are relevant, giving 1/3 scale)

- Results then averaged over all queries in test collection (MAP).

- Many more measures exist!

Mean reciprocal rank

- Reciprocal rank = 1 / rank of first correct answer
- Examples:
  - relevance `< 1, 0, 0, 0, 0, 1, 0, 1, 0 >`
    RR = 1 / 1 = 1
  - relevance `< 0, 0, 1, 0, 1, 0, 0, 0, 1 >`
    RR = 1 / 3 = 0.33
- Take mean over collection
  - e.g., for above two queries, mean(1, 0.33) = 0.67
- Insensitive to results after first correct answer

Utility based metrics

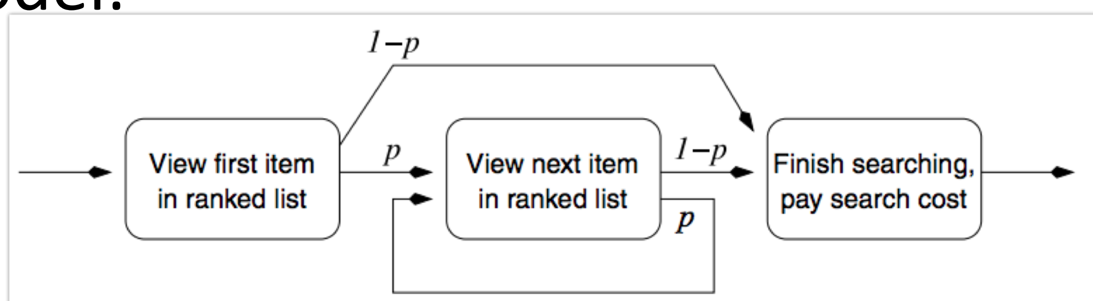
- Example: Rank-biased precision (Moffat & Zobel 2008)
- Idea: User will pay $1 for each relevant answer but nothing for irrelevant answers. Models **utility** gained by searcher.
- User processes list top-to-bottom with persistence (probability) P
- User always looks at first result. User looks at second result with probability P. Third result: P^2,P^3,P^4...
- Search engine gets paid based on how much relevant documents it provides until the user stops

Rank-biased precision

- RBP Formula (r_i is the ith element of the relevance vector of length d)

$$RBP = (1-p) \times \sum_{i=1}^{d} r_i \times p^{i-1}$$

- User Model:



- Patient user: p = 0.95, Impatient user: p=0.50

Taken from: Rank-Biased Precision for Measurement of Retrieval Effectiveness, Moffat , Zobel, TOIS 2008

RBP EXAMPLE

- Relevance vector:
  <1,1,0,0,0,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0>

Document	$p = 0.50$	$p = 0.80$	$p = 0.95$
1	1.0000	1.0000	1.0000
2	0.5000	0.8000	0.9500
6	0.0313	0.3277	0.7738
11	0.0010	0.1074	0.5987
17	0.0000	0.0281	0.4401
Total	1.5322	2.2632	3.7626
$\times(1-p)$	0.7661	0.4526	0.1881

Taken from: Rank-Biased Precision for Measurement of Retrieval Effectiveness, Moffat , Zobel, TOIS 2008

Effectiveness in practice

- In addition to explicit human judgements we also look at query logs and click logs
- For a given query and a specific result page, which result did users click on?
- After clicking, did they come back and click on other results?
- Indirect relevance feedback! Why?

Improving effectiveness

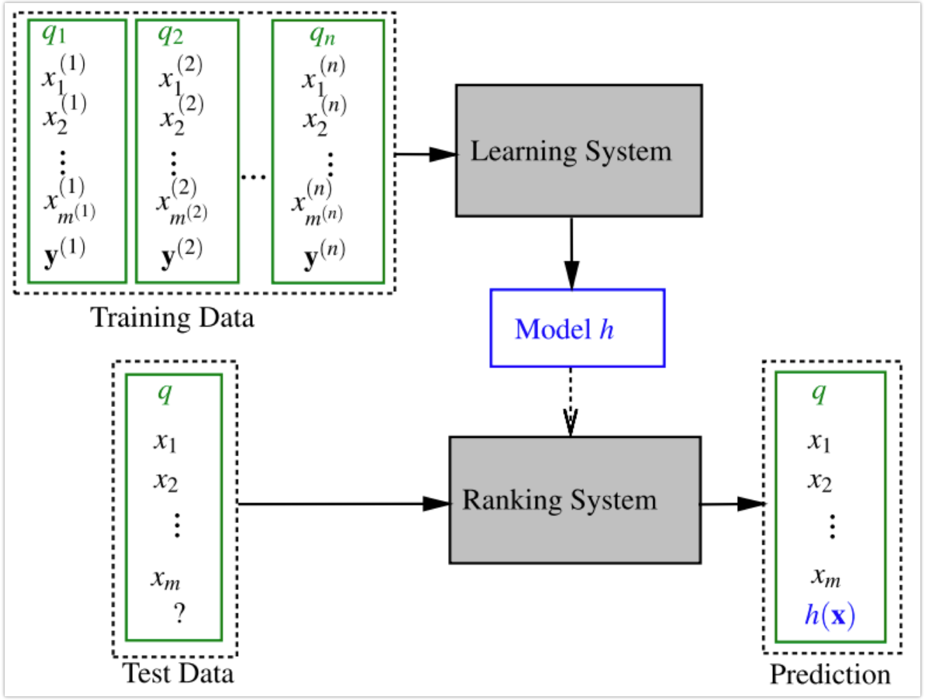
- Suppose, we find that for some queries, users click on the second result instead of the first result
- How do we incorporate this information into our similarity metric (BM25?) to rank these results higher?
- Construct (learn!) a similarity metric **automatically** from training data (queries, click data, documents) to better rank documents by relevance

Multi-stage retrieval

- Use a cheap, fast, simple similarity metric (such as BM25) to retrieve an initial set of relevant documents (top-*k* retrieval)
- For those *k* documents, apply a Machine Learning algorithm which uses more features to **re-rank** the initial set of *k* documents
- Why not apply Machine Learning to rank all documents? Expensive!

Learning to Rank

- Given queries, *m* (*k* before) documents documents for each query, click data (or human judgements) use Machine Learning techniques to rank documents



Taken from: Tie-Yan Liu: Learning to Rank for Information Retrieval

Learning to Rank II

- Learn a ranking model that can rank the list of *k* documents for an unknown query
- Use training data consisting of tuples $<q,d_i,u,r_i>$ which represent the query *q*, the *k* documents ($d_1,\ldots,d_k$), user *u* and Relevance Vector R ($r_1,\ldots,r_k$),
- Learn to combine features representing $x = <q,d_i,u>$ to to predict r_i
- Challenges:
  - Finding the right features representing $x = <q,d_i,u>$
  - Defining the objective that we want to optimize that corresponds to ranking documents

User features

- What kind of documents has the user been looking for?
- What kind of links is the user clicking on?
- How long does the user stay on a URL before returning?
- What are your friends searching/clicking on?
- Location
- Native Language
- Age
- …

Document features

- Various tf/idf features (for example document lengths)
- Number of slashes in URL
- Main topics (see Topic Models!)
- Length of URL
- Pagerank / Number of Inlinks or Outlinks
- How long do users stay on the URL before returning to search engine (dwell time)
- Quality score (spam or no spam?)
- Navigational vs Informational
- For a given query Q, how often was document D first click, last click, only click?
- Users that come view are documents come from the same location?

Query Features

- Number of queries terms
- Popularity of the query (query log)
- Time sensitive? ”World Cup”
- Number of matching documents
- BM25 score distribution
- …

Learning to rank Objectives

- Point-wise objective
  - Given a query q, a document d_i, and a user u, find a function $f(q,d_i,u)$ that predicts r_i for document d_i.
  - Ask the user: **How relevant is** d_i?
  - Relevance judgement might be binary (yes or no) or muli-graded relevance (very relevant, relevant, not relevant)

Point-wise objective

- Input: feature vectors x_i for each $<q,d_i,u>$ tuple
- Learn model $y = f(x_i)$ that outputs real numbers
- Rank documents by sorting based on $y = f(x_i)$
- To "learn a model" we define an objective that we try to minimize. This is usually referred to as a loss function
- Here: the output y should correspond to relevance!
- How do we do this?

Point-wise – Algorithm Sketch

- Train classifier that can predict r_i
- Train model that can compute:

$$P(r_i = \text{relevant} | x_i)$$

- Sort documents by the probability of being relevant
- Multiple classes: Assign classes a value and compute expectation (e.g. -2 highly non relevant, 2 highly relevant)

Other Learning-2-Rank Objectives

- Pair-wise objective
  - Given a query *q*, user *u*, and two documents d_1 and d_2 predict the correct relative order of d_1 and d_2
  - Ask the user: Which of these **two** documents is more relevant?
- List-wise objective
  - Output is a ranked list. Modelled based on whole ordered list, which cannot be decomposed into scores of individual documents or pairs.
  - Ask the user: Rearrange this list of documents based on relevance

Learning to Rank in Practice

- The secret sauce behind many search engines (and other websites such as Amazon)
- Rank high and make lots of money
- Use many features to create complex personalized, localized ranking models
- Use A/B testing to test new ranking models
- SEO – Reverse engineer the features used to rank higher

Summary

- Evaluation using relevance judgements
- Precision@k, (M)AP, (M)RR, RBP evaluation metrics
- Use BM25 as a first step in multi-stage retrieval system
- Use complex trained ranking models to re-rank the original BM25 ranking
- Many features and training methods exists

Reading

- Reading
  - MRS Chapter 8
  - Tie-Yan Liu: Learning to Rank for Information Retrieval, Section 1.3, 2011, ISBN 978-3-642-14266-6 (ebook)
- Optional extras
  - Hang Li: Learning to Rank for Information Retrieval and Natural Language Processing, Morgan & Claypool, 2015
  - Alistair Moffat, Justin Zobel: Rank-Biased Precision for Measurement of Retrieval Effectiveness. TOIS 2008