Text Analysis and Retrieval

3. Basics of Information Retrieval

Assoc. Prof. Jan Šnajder

With contributions from
dr. sc. Goran Glavaš
dr. sc. Mladen Karan

University of Zagreb
Faculty of Electrical Engineering and Computing (FER)

Academic Year 2019/2020

v2.1

Outline

Learning outcomes 1

1. List three main components of an IR model
2. Describe the vector space model and the TF-IDF weighting scheme
3. Explain the probability ranking principle and BM25
4. Describe the LM information retrieval model
5. List the main IR tools available

Retrieval and search

- What is your first association with "information retrieval"?
- What is your first association with "search" or "search engine"?

Retrieval and search

Reminder: What is IR?

Information retrieval *(Wikipedia)*

The activity of obtaining information resources relevant to a user's information need from a collection of information resources.

- **Information needs** (expressed by users in the form of queries)
- **Information resources** (typically unstructured – text, images, video, audio, etc.)

Information needs

Information need

Information need is an individual or group's desire to locate and obtain information to satisfy a conscious or unconscious need. Needs and interests call forth information.
(Robert S. Taylor: "The process of asking questions", 2007)

- (Un)conscious needs for information are expressed via queries
  - In text retrieval: words and phrases
    (e.g., "ISIS attacks", "coronavirus pandemic")
  - In image content retrieval: images

Information retrieval problem

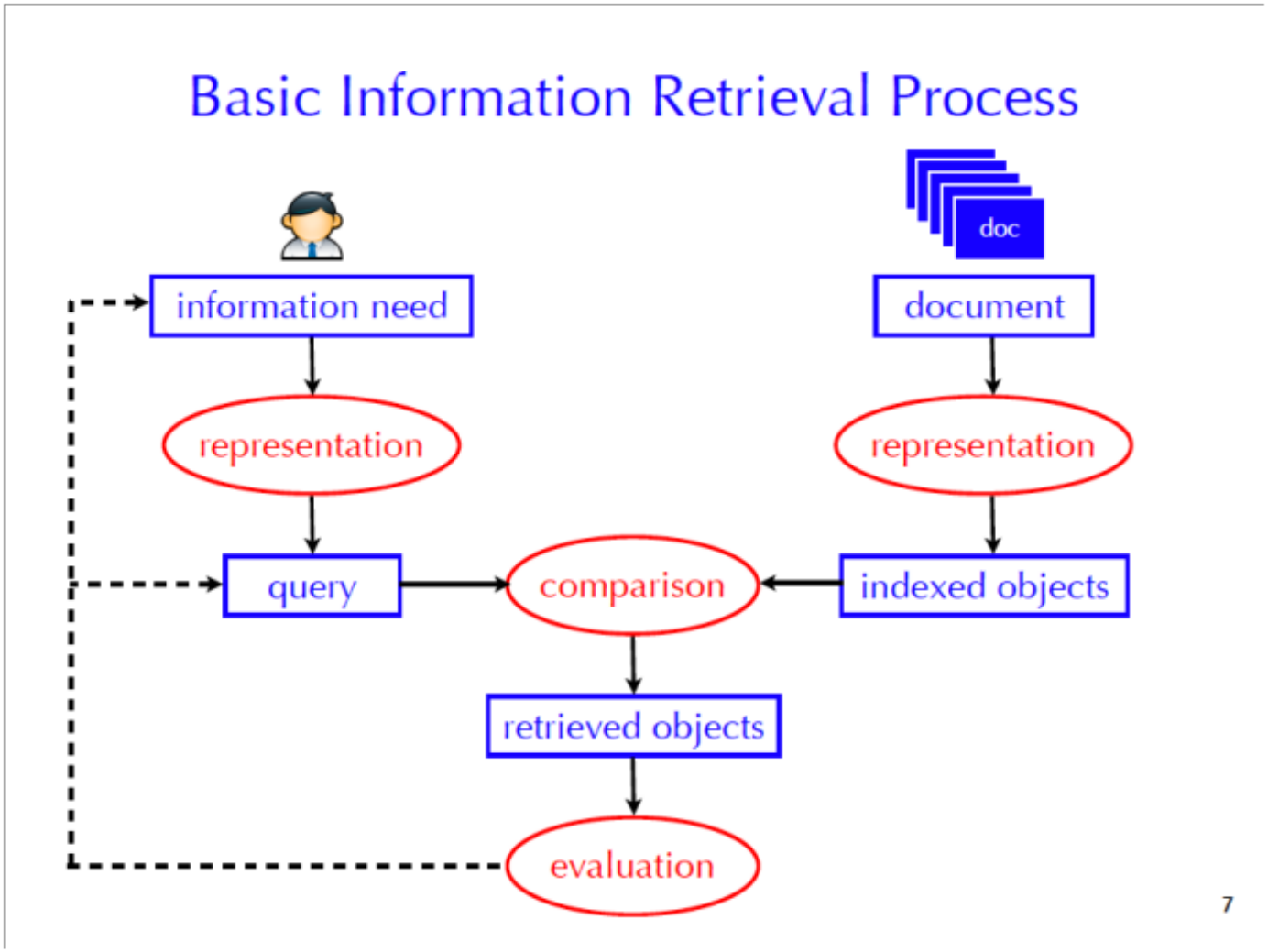


Diagram from Jaime Arguello, UNC-Chapel Hill

Representations and ranking

Search engines differ in:

1. The **representation** of documents and queries
2. How they determine the **relevance** of a document given a query
   - Relevance of a document is typically given as a real-valued **score**
   - Given a query, documents are **ranked** according to their scores
   - Relevance scores usually incorporate an element of **uncertainty**

Document/query representation

- **Unstructured representation**
  - Text represented as an unordered set of terms (the so-called *bag-of-words* representation)
  - Considerable oversimplification: ignoring syntax and semantics
  - Despite oversimplifying, satisfiable retrieval performance
- **Weakly-structured representations**
  - Certain groups of terms given more importance (other terms' contribution downscaled or ignored)
  - Noun phrases (NP), named entities, etc.
- **Structured representations**
  - Use of information extraction (IE) techniques
  - IE techniques not sufficiently accurate and time-costly
  - Rarely used in IR

Unstructured document representation

Document snippet

One evening Frodo and Sam were walking together in the cool twilight. Both of them felt restless again. On Frodo suddenly the shadow of parting had falling: the time to leave Lothlorien was near.

Bag of words

{(One, 1), (evening, 1), (Frodo, 2), (and, 2), (Sam, 1) (were, 1), (walking, 1), (together, 1), (in, 1), (the, 3), (cool, 1), (twilight, 1), (Both, 1), (of, 2), (them, 1), (felt, 1), (restless, 1), (again, 1), (On, 1), (suddenly, 1), (shadow, 1), (parting, 1), (had, 1), (falling, 1), (time, 1), (to, 1), (leave, 1), (Lothlorien, 1), (was, 1), (near,1)}

Weakly-structured document representation

Document snippet

One evening Frodo and Sam were walking together in the cool twilight. Both of them felt restless again. On Frodo suddenly the shadow of parting had falling: the time to leave Lothlorien was near.

Bag of nouns

{(evening, 1), (Frodo, 2), (Sam, 1), (twilight, 1), (shadow, 1), (parting, 1), (time, 1), (Lothlorien, 1)}

Bag of named entity terms

{(Frodo, 2), (Sam, 1), (Lothlorien, 1)}

Preprocessing

1. **Morphological normalization (stemming/lemmatization)**
   - Conflating various forms of the same word to a common form
   - Important for morphologically rich languages such as Croatian
   - Stemming (e.g., kućom → kuć) more often used than lemmatization (e.g., kućom → kuća)
2. **Stop words removal**
   - Stop words: semantically void terms such as determiners, prepositions, conjunctions, pronouns, etc.
   - Content words: nouns, verbs, adjectives, adverbs
   - Stop words removal: removes stop words and retains only the content words
   - English stop words: `https://www.ranks.nl/stopwords`

⇒ both methods reduce the size of the bag-of-words representation and generally improve retrieval performance

Three components of an IR model

The **basic retrieval model** is a triple (f_d, f_q, r):

1. f_d is a function that maps documents to retrieval representations

$$f_d(d) = x_d$$

2. f_q is a function that maps queries to retrieval representations

$$f_q(q) = x_q$$

3. r is a ranking function that produces a relevance score: a real-valued number that indicates the potential relevance of the document d for query q based on x_d and x_q

$$relevance(d, q) = r\big(f_d(d), f_q(q)\big) = r(x_d, x_q)$$

IR paradigms

- Information retrieval models roughly fall into three paradigms:
  1. **Set-theoretic models**
     - Boolean model
     - Extended Boolean model
  2. **Algebraic models**
     - Vector space model
     - Latent semantic indexing
  3. **Probabilistic models**
     - BM25
     - Language model
- Additionally, there are IR models that utilize **link analysis algorithms** (e.g., PageRank, HITS)

Vector space model

- Documents and queries are represented as vectors of index terms
- Weights are non-negative real numbers

$$\mathbf{d_j} = [w_{1j}, w_{2j}, \ldots, w_{tj}]$$
$$\mathbf{q} = [w_{1q}, w_{2q}, \ldots, w_{tq}]$$

- The relevance of the document for the query is estimated by computing some **distance or similarity metric** between the two vectors
  - Distance metrics – Euclidean, Manhattan, etc.
    - More relevant when distance is lower
  - Similarity metrics – Cosine, Dice, etc.
    - More relevant when similarity is larger

Vector space model – distance metrics

- Euclidean distance

$$dis_E(\mathbf{d_j}, \mathbf{q}) = \sqrt{\sum_{i=1}^{t} (w_{ij} - w_{iq})^2}$$

- Manhattan distance

$$dis_M(\mathbf{d_j}, \mathbf{q}) = \sum_{i=1}^{t} |w_{ij} - w_{iq}|$$

Vector space model – similarity metrics

- Cosine similarity

$$Cosine(\mathbf{d_j}, \mathbf{q}) = \frac{\mathbf{d_j} \cdot \mathbf{q}}{\|\mathbf{d_j}\|\|\mathbf{q}\|} = \frac{\sum_{i=1}^{t} w_{ij} w_{iq}}{\sqrt{\sum_{i=1}^{t} w_{ij}^2} \sqrt{\sum_{i=1}^{t} w_{iq}^2}}$$

- Dice similarity

$$Dice(\mathbf{d_j}, \mathbf{q}) = \frac{2 \sum_{i=1}^{t} w_{ij} w_{iq}}{\sum_{i=1}^{t} w_{ij} + \sum_{i=1}^{t} w_{iq}}$$

Cosine similarity

Salton, 1983

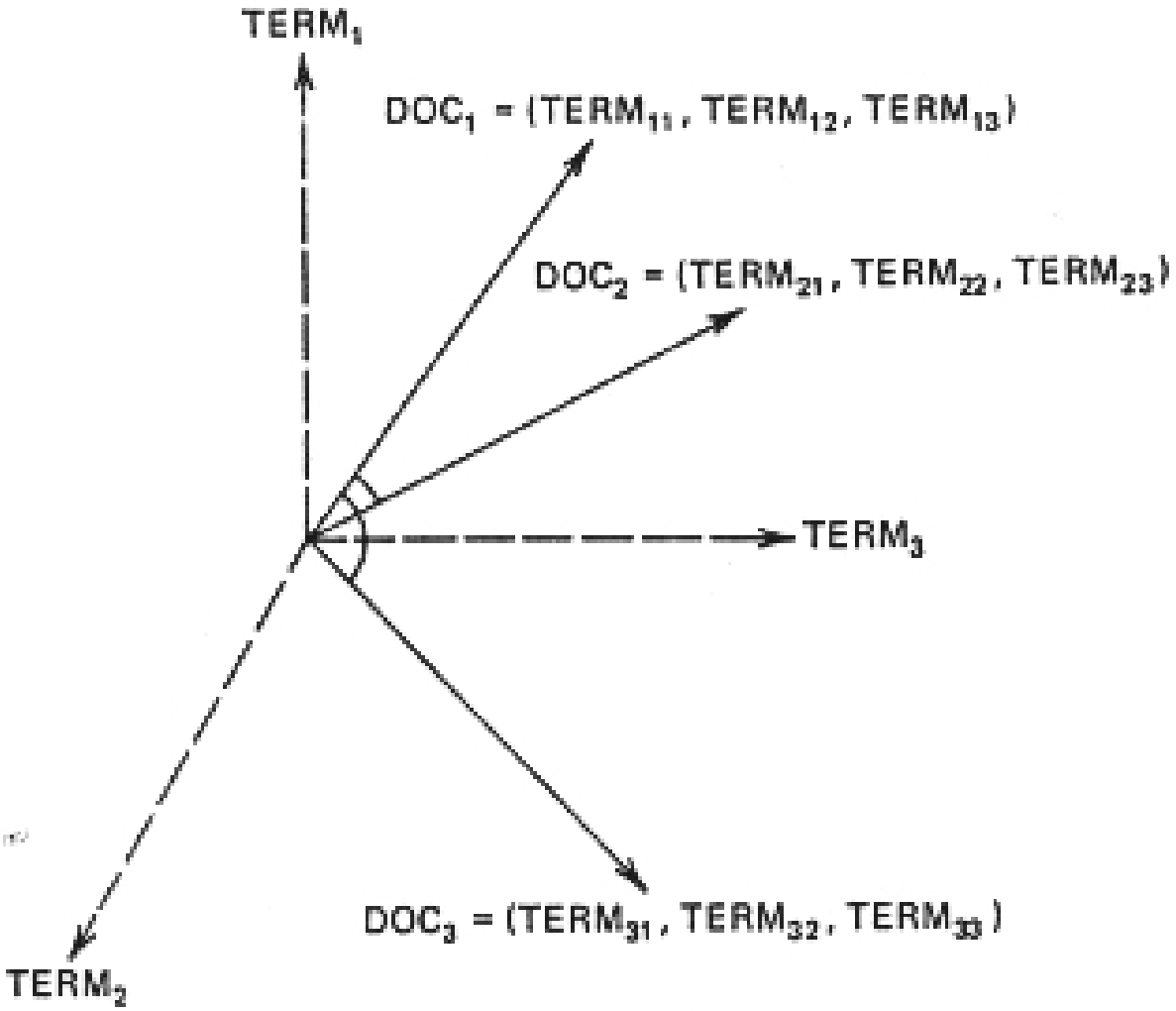


Figure 4-2 Vector representation of document space.

Vector space model – term weighting

- How are the weights w_{ij} of index terms for documents computed?
- Two intuitive assumptions:
  1. The relevance of an index term for the document is proportional to its frequency in the document (**term frequency** component)
     - i.e., more frequent $\Rightarrow$ more relevant
  2. The relevance of an index term for any document is inversely proportional to the number of documents in the collection in which it occurs (**inverse document frequency** component)
     - i.e., more common across documents $\Rightarrow$ less relevant (e.g., stopwords such as "*the*")

TF-IDF weighting scheme

- The weight computed as the product of the term frequency component and the inverse document frequency component

$$w_{ij} = tf(t_i, d_j) \cdot idf(t_i, D)$$

  where t_i is the i-th term from the index
- The most popular local and global schemes:

$$tf(t_i, d_j) = 0.5 + \frac{0.5 \cdot freq(t_i, d_j)}{\max_{t \in d_j} freq(t, d_j)}$$

$$idf(t_i, D) = \log \frac{|D|}{|\{d_j \in D \mid t_i \in d_j\}|}$$

Probabilistic retrieval models

- **Probabilistic retrieval models**
  - View retrieval as a problem of **estimating the probability of relevance** given a query, document, collection, etc.
  - Documents are ranked in decreasing order of this probability
- Rely on probability ranking principle

Probability ranking principle

Probability ranking principle *(Robertson, 1977)*

If an IR system's response to each query is a ranking of the documents in the collection in order of decreasing probability of relevance, the overall effectiveness of the system to its user will be maximal.

$\Rightarrow$ relevance score = probability of relevance

- Probabilistic IR models need to answer the *basic question*: What is the probability that a user will judge *this* document as relevant for *this* query? (Sparck Jones et al., 2000)
- How do we formalize this question?

Probability ranking principle (2)

- Random variables:
  - $D = \{D_1, \ldots D_t, \ldots D_N\}$ – document (set of terms)
  - $Q = \{Q_1, \ldots Q_t, \ldots Q_L\}$ – query (set of terms)
  - $R \in \{0, 1\}$ – relevance judgement, where $R = 1$ if D is relevant for Q, $R = 0$ otherwise
- The *basic question* now translates to estimating:

$$P(R = 1|D = d, Q = q)$$

- Let r be a shorthand for $R = 1$, and $\bar{r}$ for $R = 0$, so we'll have:
  - $p(r|D, Q)$ – probability of D being relevant for Q
  - $p(\bar{r}|D, Q) = 1 - p(r|D, Q)$ – prob. of D not being relevant for Q

Probability ranking principle (3)

- Let's apply the logit function to the posterior probability:

$$logit(p) = \log\left(\frac{p}{1-p}\right)$$

- This is a **rank-preserving transformation**, giving:

$$\log\frac{p(r|D,Q)}{1-p(r|D,Q)} = \log\frac{p(r|D,Q)}{p(\bar{r}|D,Q)}$$

- By applying the Bayes rule and the chain rule we can derive:

$$\log\frac{p(r|D,Q)}{p(\bar{r}|D,Q)} \propto \log\frac{p(D|Q,r)}{p(D|Q,\bar{r})}$$

- Probablistic models differ in how they model this quantity

Two-Poisson model

- Models the document as a vector of word frequencies
- Uses the **Poisson distribution** to model frequencies
  - Assumption: all documents are of **equal length**
- Can be approximated by the following expression:

$$\log \frac{p(D|Q,r)}{p(D|Q,\bar{r})} \approx \sum_{t \in q} \frac{freq(t,d) \cdot (k_1+1)}{k_1 + freq(t,d)} \cdot idf(t,D)$$

where k_1 a constant (typically $1 \leq k_1 < 2$)
$\Rightarrow$ higher frequency words get their weights boosted

BM11

- Removes document length assumptions of the two-Poisson model: matches in longer documents should be less important
- We can correct the frequency $freq'(t,d) = freq(t,d)cdot(l_{avg}/l_d)$
  - l_{avg} – the average length of a document
  - l_d – the length of document d
- After plugging in the corrected frequency, the relevance score becomes:

$$\sum_{t \in q} \frac{freq(t,d)(k_1+1)}{k_1(l_d/l_{avg}) + freq(t,d)} \cdot idf(t,d)$$

BM25

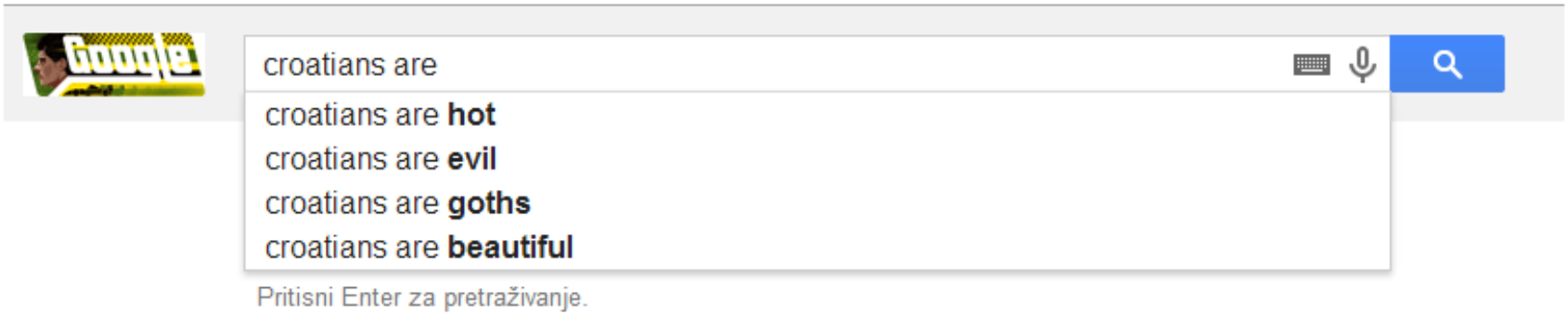
- BM11 still has weaknesses:
  - Long relevant documents are getting too much dampening
  - Short irrelevant documents are getting too much boosting
- To control the amount of correction, we introduce b (often set to 0.75)

$$\sum_{t \in q} \frac{freq(t,d)(k_1+1)}{k_1(1-b)+k_1(l_d/l_{avg})b+freq(t,d)} \cdot idf(t,d)$$

- This expression represents the famous BM25 ranking function, which gives state-of-the-art results

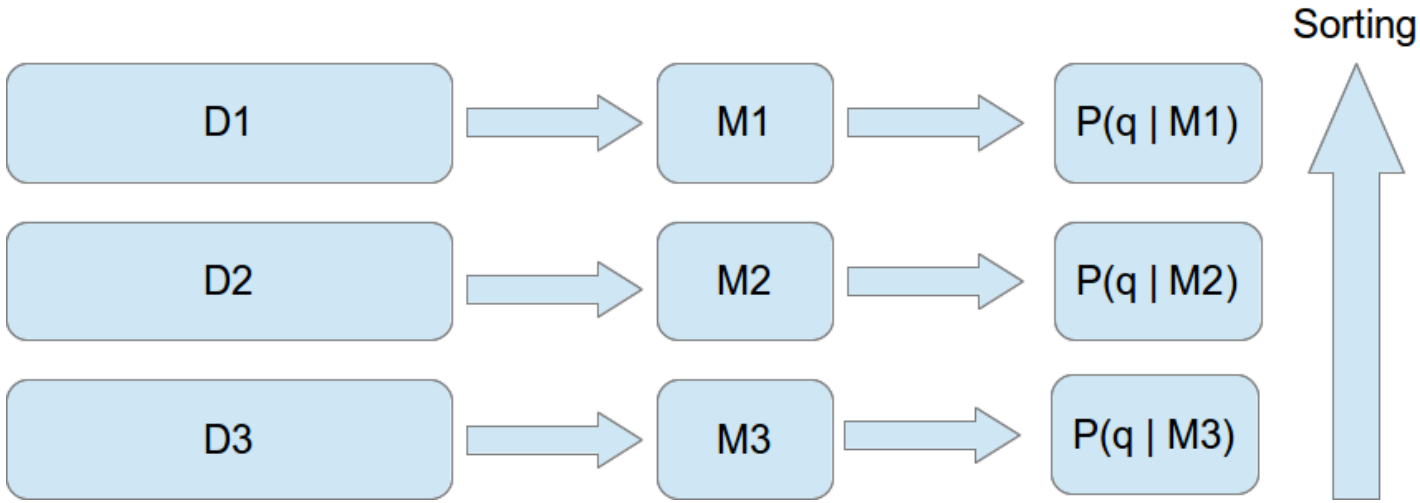
Language modeling for IR

- Approaching the probabilistic information retrieval problem from a different perspective
- Instead of modeling document probability given the query, we model the **query probability given the document**



Query likelihood model

- Given a document collection D and a query q
- A language model M_d is built for **each document**
- Documents are scored according to the probability $P(q|M_d)$



- **Intuition:** language models corresponding to relevant documents should assign higher probability to the query

Smoothing language models

- Typically, we use unigram models, because bigram models at the document level will suffer from sparsity issues
- However, we'll still get a probability of 0 for queries which contain terms that *do not occur* in the document
- We can prevent this by using **smoothing techniques**
- Smoothing adds a small probability under the model even to unseen words

Smoothing techniques

- Laplace smoothing – Adding a fixed small count (e.g., 1) to all word counts (even the unobserved ones) and renormalizing to get a probability distribution

$$p'(t_i|M_d) = \frac{n_{i,d} + \alpha}{n_d + |V|\alpha}$$

- Adds an artificial count α for each possible word in our vocabulary V

Smoothing techniques

- Laplace smoothing assumes all unseen words are *equally* likely!
- Jelinek-Mercer smoothing rather builds a language model M_D of the entire document collection, and interpolates:

$$p'(t_i|M_d) = \lambda p(t_i|M_d) + (1 - \lambda)p(t_i|M_D)$$

- Words absent from a document will still get some probability mass from the right term, but the amount each word gets will vary depending on their likelihood in the collection as a whole

IR Tools

- **Terrier IR platform**
  - Open command-line IR platform
  - Popular in academia – used for IR evaluations in research
  - Stepwise usage – first indexing, then retrieval, and then evaluation
- **Lucene**
  - Open source information retrieval library written in Java (ports to many languages exist)
  - Describes a document as a set of (user definable) fields
  - Very flexible solution for applications requiring full text indexing and search
- **Elasticsearch** – RESTful search engine on top of Lucene
  - Docs: https://www.elastic.co/guide/index.html
  - Very limited NLP functionality:
    https://www.elastic.co/blog/text-classification-made-easy-with-elasticsearch

Terrier (University of Glasgow)

`http://terrier.org`

- Implements a wide variety of IR models – VSMs, LMs, PMs
- Stepwise usage
  - Loading and indexing the collection of documents (which need to be in the TREC format)

    ```
    >> trec_setup.sh <coll-path>
    >> trec_terrier.sh -i
    ```

  - Performing retrieval (must provide the collection of queries and define the retrieval model)

    ```
    >> trec_terrier.sh -r -Dtrec.topics=<q-path>
                          -Dtrec.model=<ir-model>
    ```

  - Evaluating the retrieval performance (must provide relevance judgements)

    ```
    >> trec_terrier.sh -e Dtrec.qrels=<rj-path>
    ```

Learning outcomes 1 – CHECK!

1. List three main components of an IR model
2. Describe the vector space model and the TF-IDF weighting scheme
3. Explain the probability ranking principle and BM25
4. Describe the LM information retrieval model
5. List the main IR tools available

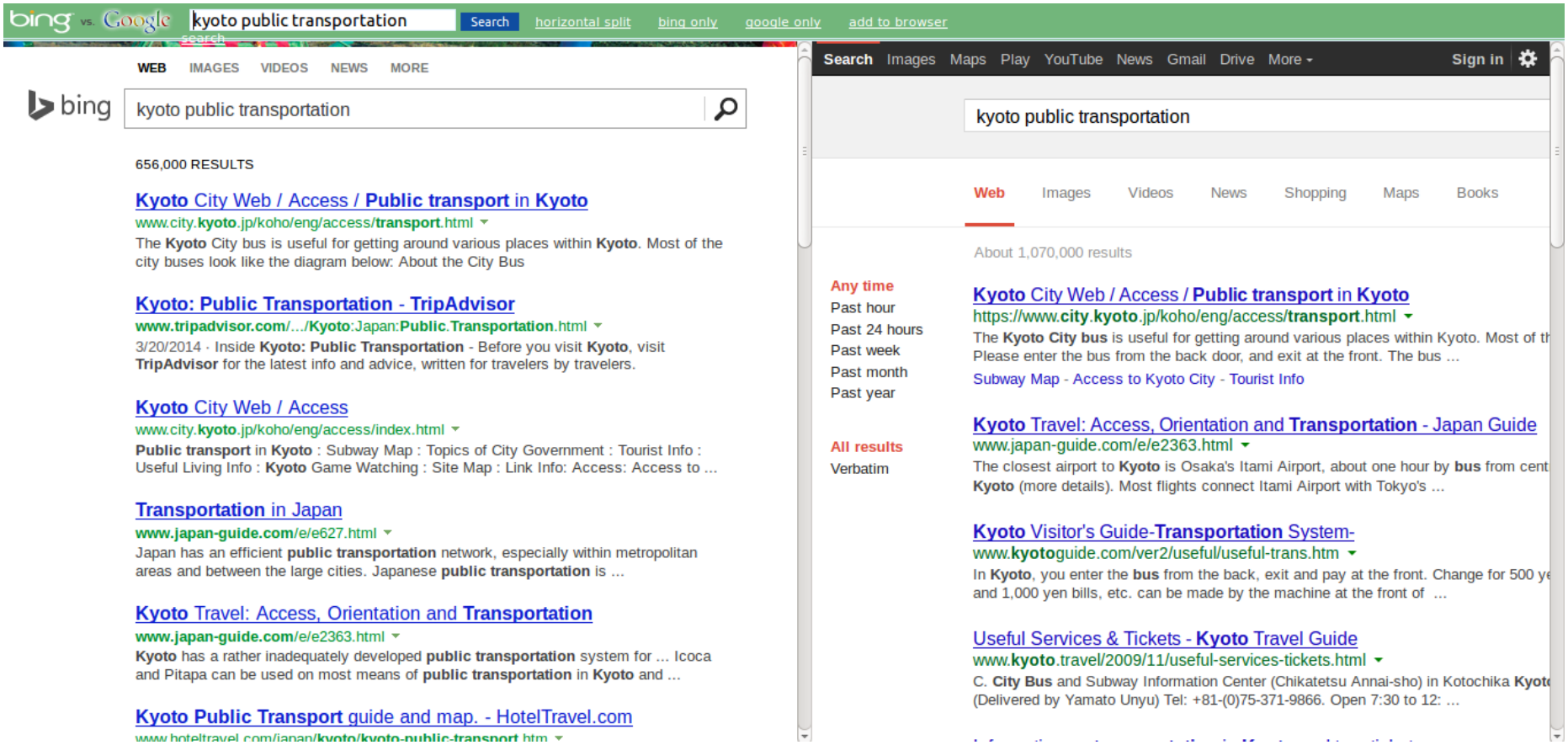
Outline

Learning outcomes 2

1. Explain what an IR test collection consist of and what it's used for
2. Define and calculate the standard IR evaluation metrics

IR evaluation

Clash of the giants: Which one is better?



IR evaluation

1. **Retrieval effectiveness** (standard IR evaluation)
   - Relevance of search results
2. **System quality**
   - Indexing speed (documents per hour)
   - Search speed (search latency as a function of index size)
   - Coverage (document collection size and diversity)
   - Query language expressiveness
3. **User utility**
   - User happiness based on the relevance, speed, and user interface
   - User return rate, user productivity (difficult to measure)
     $\Rightarrow$ Measured with user studies (expensive)
   - A/B test: slight change on a deployed system visible to a fraction of users, difference evaluated using clickthrough log analysis

IR test collection

- IR test collection is comprised of:
  1. Document collection
  2. Set of information needs (descriptions + queries)
     - At least 50 information needs
  3. Set of relevance judgments for each query–document pair
     - Binary relevance (document is relevant or not relevant) or graded relevance judgments (less common)
- Used for:
  - Evaluating retrieval effectiveness w.r.t. different settings (stemming, ranking model, etc.)
  - Comparing against other systems, typically in evaluation campaigns
  - Fine-tuning of system parameters, done on a development test collection (to prevent overfitting)

IR test collections – topic examples

TREC topic 351

Title: Falkland petroleum exploration
Description: What information is available on petroleum exploration in the South Atlantic near the Falkland Islands?
Narrative: Any document discussing petroleum exploration in the South Atlantic near the Falkland Islands is considered relevant. Documents discussing petroleum exploration in continental South America are not relevant.

TREC topic 409

Title: Legal, Pan Am, 103
Description: What legal sanctions have resulted from the destruction of Pan Am Flight 103 over Lockerbie, Scotland, on December 21, 1988?
Narrative: Documents describing any charges, claims, or fines presented to or imposed by any court or tribunal are relevant, but documents that discuss charges made in diplomatic jousting are not relevant.

Relevance judgments ("qrel file")

```
401 0 FBIS3-18916 0
401 0 FBIS3-18926 0
401 0 FBIS3-18943 1
401 0 FBIS3-18946 0
401 0 FBIS3-18972 0
401 0 FBIS3-18997 0
401 0 FBIS3-19003 0
401 0 FBIS3-19032 1
401 0 FBIS3-19037 0
401 0 FBIS3-19038 1
401 0 FBIS3-19042 0
401 0 FBIS3-19080 0
401 0 FBIS3-19103 0
401 0 FBIS3-19107 1
401 0 FBIS3-19110 0
401 0 FBIS3-19126 0
401 0 FBIS3-19133 0
401 0 FBIS3-19212 0
401 0 FBIS3-19213 0
401 0 FBIS3-19251 0
401 0 FBIS3-19290 0
401 0 FBIS3-19302 0
401 0 FBIS3-19303 0
401 0 FBIS3-19304 0
```

Standard test collections

- **Cranfield** – first IR test collection (1957)
  - 1,398 abstracts of aerodynamics journal articles, 225 queries, complete relevance judgments
- **TREC collections** – NIST Text Retrieval Conferences (1992 –)
  - Ad Hoc retrieval task: 1.89M docs, 450 information needs (topics), incomplete relevance judgments
  - Many other tasks: blog track, cross-language track, enterprise track, QA track, ...
- **GOV$_2$** – NIST 25M pages web page collections
- **CLEF** – Conference and Labs of the Evaluation Forum (Cross Language Evaluation Forum), focus on European languages
  - Mono-lingual and X-language tasks, QA tasks, ...

Evaluation metrics

- Compare retrieved documents against relevant documents
- Each document is either retrieved or not, and either relevant or not. This gives a 2×2 confusion matrix:

$$\begin{array}{lcc} & \text{relevant} & \text{not relevant} \\ \text{retrieved} & tp & fp \\ \text{not retrieved} & fn & tn \end{array}$$

We could compute accuracy as the fraction of correct decisions:

$$Acc = \frac{tp + tn}{tp + tn + fp + fn}$$

- **Q:** Why using accuracy is not a good idea?
  A: Given a query, most documents (say 99%) are irrelevant. A search engine that retrieves nothing will already be 99% accurate

Precision and Recall

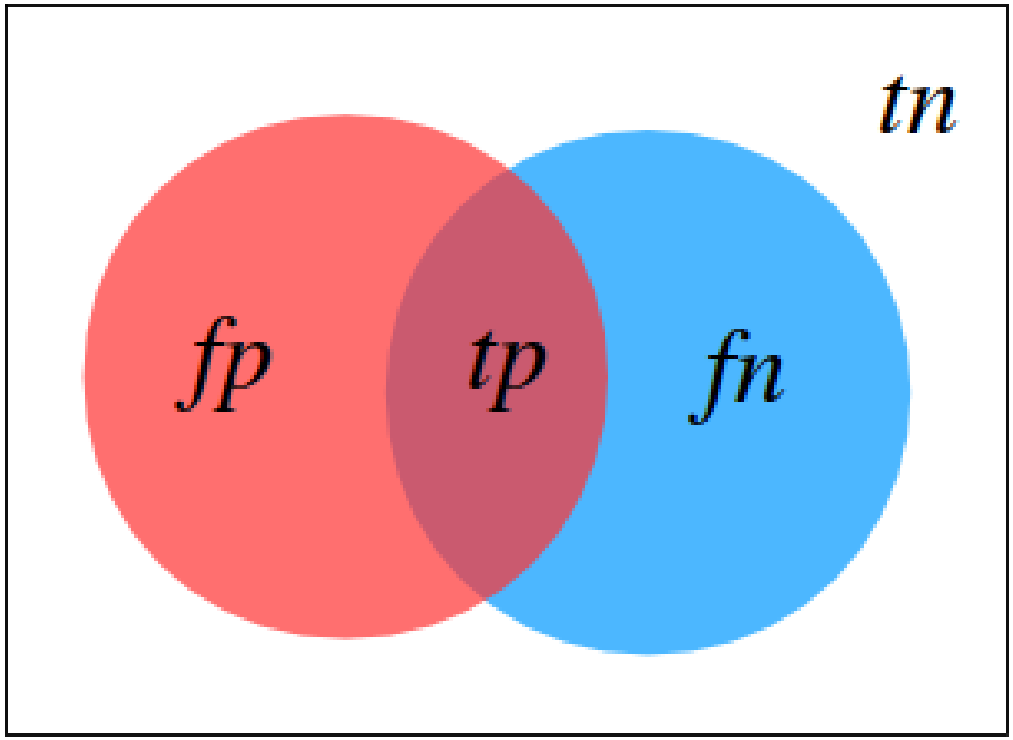
- **Precision (P)** is a fraction of retrieved documents that are relevant

$$P = \frac{\#(\text{relevant documents retrieved})}{\#(\text{retrieved documents})} = \frac{tp}{tp + fp}$$

- **Recall (R)** is a fraction of relevant documents that are retrieved

$$R = \frac{\#(\text{relevant documents retrieved})}{\#(\text{relevant documents})} = \frac{tp}{tp + fn}$$

Precision and Recall



$$P = \frac{tp}{tp + fp} \qquad R = \frac{tp}{tp + fn}$$

F-measure

- Combining P and R into a single number (the harmonic mean)

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \qquad \text{where } \beta^2 = \frac{1-\alpha}{\alpha}$$

- β controls the precision–recall trade-off:
  - $\beta = 1$ gives equal weight to precision and recall (**F1-score**):

$$F_{\beta=1} = \frac{2PR}{P+R}$$

  - $\beta = 0.5$ emphasizes precision twice as much as recall
  - $\beta = 2$ emphasized recall twice as much as precision
- **Q:** Why use the harmonic mean (instead, say, the arithmetic mean)?
  A: Because it is more strict: $P = 1$, $R = 0.01$, $F_1 = 0.02$, $\text{avg} = 0.5$

Precision and Recall – example

For a query q, there are four documents in the collection that are relevant (**R**), while all other are not relevant (**N**).

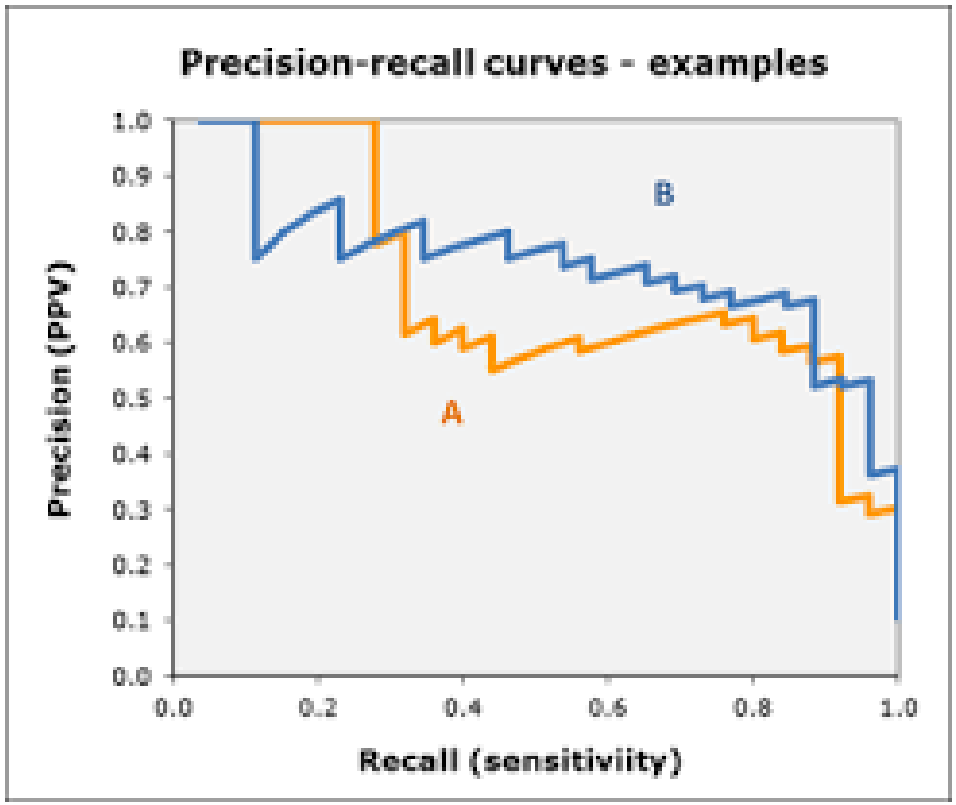
Given q, the system returns six documents: **N, R, N, R, N, N**.

Compute the P, R, and F_1-score for query q.

Evaluation of ranked results

- Modern search engines produce ranked results, but P, R, and F-score do not account for ranks
- Ideally, a search engine should rank all the relevant documents before the non-relevant ones
- Rank-based metrics:
  - Precision-recall curve
  - 11-point precision
  - MAP
  - P@k
  - R-precision
  - MRR

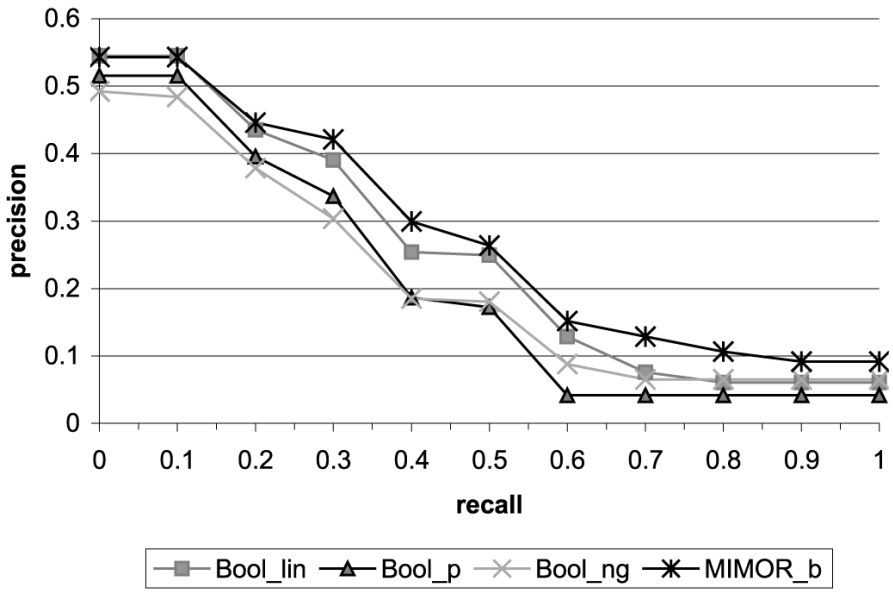
Precision-recall curve



Interpolated precision: $p_{interp}(r) = \max_{r' \geq r} p(r')$

11-point precision

- PR-curve is informative, but difficult to compare between systems
- **11-point precision** describes the performance more succinctly
  - For each information need, calculate interpolated precision at 11 recall levels: $0.0, 0.1, 0.2, \ldots, 1.0$
  - For each level, average across all information needs



Mean Average Precision (MAP)

- We'd like to have a single-figure measure of retrieval effectiveness across recall levels
- **Average precision** for a query q resulting in *relevant* documents $\{d_1, \ldots, d_m\}$:

$$\mathrm{AP}(q) = \frac{1}{m} \sum_{k=1}^{m} P(R_k)$$

  where R_k are the top documents (both relevant and non-relevant) down to the k-th relevant document
- **Mean average precision** is AP averaged over the set of queries Q:

$$\mathrm{MAP}(q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} P(R_{jk})$$

- Often used, but shown to have high variance across information needs

P@k and R-precision

- MAP takes into account all recall levels, even at very low ranks
  - This is inappropriate for web search: less than 6% users look at the second page of results
- **Precision at k (P@k)** computes precision for the top k-ranked documents (e.g., $P@5$, $P@10$, $P@20$)
- **R-precision** computes precision at top-k documents, where k equals the number of relevant documents for the query

Ranked-based evaluation – example

For a query q, there are four documents in the collection that are relevant (**R**), while all other are not relevant (**N**).

Given q, the system returns a ranked list of eight documents:

1	2	3	4	5	6	7	8
N	**R**	**N**	**R**	**N**	**N**	**N**	**R**

Compute AP, P@5, and R-precision.

Relevance judgments: problem of incompleteness

- Collecting relevance is tedious and expensive
- Test collections rarely have complete relevance judgments (judgments for all query–document pairs)
- **Pooling method:**
  - incomplete relevance judgments in evaluation campaigns
  - relevance judgments for the union of top k results for each participating system ("the pool")
  - assumes that if a document is relevant, it will be ranked among top k by at least one of the systems
  - good enough for comparing the relative performance of systems

Learning outcomes 2 – CHECK!

1. Explain what an IR test collection consist of and what it's used for
2. Define and calculate the standard IR evaluation metrics

Outline

Learning outcomes 3

1. State the PageRank hypothesis
2. Define and explain the PageRank iterative update formula in matrix form

PageRank hypothesis

- Web is a massive directed graph in which edges denote hyperlinks between web pages
- What does a web graph have to do with the search results?
  - It does if we interpret hyperlinks as recommendations
  - A page with more recommendations is more important
  - Status of the recommender matters
    - Recommendations from more important recommenders are worth more
    - The overall number of recommendation issued by the recommender also matters

Web page importance

A web page is important if it is pointed to by other important pages that do not point to too many other pages

PageRank formula

Heinrich Hertz (on Maxwell's equation)

One cannot escape the feeling that these mathematical formulae have an independent existence and an intelligence on their own, that they are wiser than we are, wiser even than their discoverers, that we get more out of them than was originally put into them.

- One of the most important formulas in Computer Science

$$\boldsymbol{\pi}^T = \boldsymbol{\pi}^T(\alpha\mathbf{S} + (1-\alpha)\mathbf{E})$$

- But lets take it step by step...

Original PageRank summation

- The PageRank of a page P_i is the sum of importances of all pages that have hyperlinks to P_i, each normalized with the total number of hyperlinks on that page

$$r(P_i) = \sum_{P_j \in B_{P_i}} \frac{r(P_j)}{|P_j|} \tag{1}$$

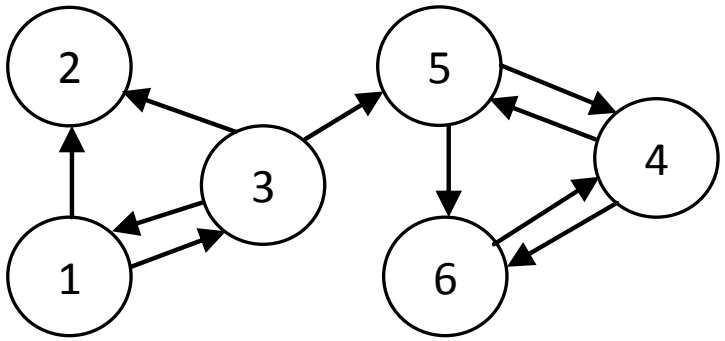
  - B_{P_i} is the set of all pages that link to P_i
  - $|P_j|$ is the number of outgoing links from page P_j
  - PageRank scores of pages P_j linking to P_i are unknown
  - How to determine the PageRank scores for pages?

Iterativeness over original PageRank summation

- The idea
  1. Assign the same initial score to all pages: $P_i = \frac{1}{n}$, where n is the total number of pages
  2. Run the equation (1) iteratively, until the PageRank scores for all pages converge (remain the same in two consecutive iterations)

$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|} \tag{2}$$

Iterative computation of PageRank scores – example



Iteration 0	Iteration 1	Iteration 2	Rank
$r_0(P_1) = 1/6$	$r_1(P_1) = 1/18$	$r_2(P_1) = 1/36$	5
$r_0(P_2) = 1/6$	$r_1(P_2) = 5/36$	$r_2(P_2) = 1/18$	4
$r_0(P_3) = 1/6$	$r_1(P_3) = 1/12$	$r_2(P_3) = 1/36$	5
$r_0(P_4) = 1/6$	$r_1(P_4) = 1/4$	$r_2(P_4) = 17/72$	1
$r_0(P_5) = 1/6$	$r_1(P_5) = 5/36$	$r_2(P_5) = 11/72$	3
$r_0(P_6) = 1/6$	$r_1(P_6) = 1/6$	$r_2(P_6) = 14/72$	2

PageRank summation in the matrix form

- In equation (2), PageRank scores are computed for one page at a time
- Using a matrix form, all PageRank scores can be updated at once
- Notation:
  - $\boldsymbol{\pi}$ – the vector of PageRank scores: $\boldsymbol{\pi}_i = r(P_i)$
  - $\mathbf{H}$ – the row normalized Web graph adjacency matrix

$$\mathbf{H}_{ij} = \begin{cases} 1/|P_i|, & \text{if there is a link from page } P_i \text{ to page } P_j \\ 0, & \text{otherwise.} \end{cases}$$

$$\mathbf{H} = \begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

PageRank in matrix form

- The iterative update formula can now be rewritten in matrix form

$$\boldsymbol{\pi}_{k+1}^T = \boldsymbol{\pi}_k^T \mathbf{H} \tag{3}$$

- Each iteration requires one vector-matrix multiplication: $\mathcal{O}(n^2)$
- However, $\mathbf{H}$ is a sparse matrix
  - Most pages have links to only few other pages: $\mathcal{O}(nnz(\mathbf{H}))$
    - Effectively, the complexity is $\mathcal{O}(kn) = \mathcal{O}(n)$
- Will this iterative process converge?
  - Does the convergence depend on the initial PageRank vector $\boldsymbol{\pi}_0^T$?
  - Under what properties of $\mathbf{H}$ is the iterative process guaranteed to converge?

PageRank in matrix form

- Eq. (3) is power method applied to matrix $\mathbf{H}$
- $\mathbf{H}$ is a substochastic transition probability matrix of a Markov chain
  - Rows that are not empty are probability distributions
  - But there are empty rows – nodes that have no outgoing edges (rank sinks)
- Power method applied to some matrix $\mathbf{P}$ converges iff $\mathbf{P}$ is:
  - Stochastic – each row is a probability distribution
  - Irreducible – there is a probability assigned to transition from each node to every other node
  - Aperiodic – no such cycle of length k for which $P = P^k$, $P^1 = P^{k+1}$, $P^2 = P^{k+2}$, $\ldots$
- Being the row-normalized adjacency matrix of the Web graph, $\mathbf{H}$ is, in general, neither of those

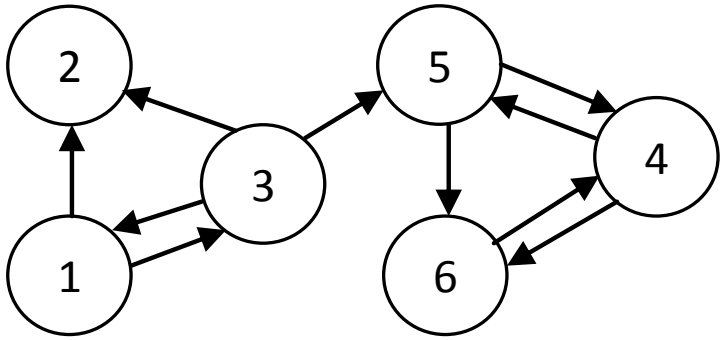
Adjustments to the basic PageRank model

- To ensure convergence of the power method, some adjustments to matrix $\mathbf{H}$ need to be made
- Brin & Page introduce a **random surfer**
  - A random surfer surfs the web by following the hyperlink structure of the Web (on each page she randomly selects the hyperlink to follow)
  - **Stochasticity adjustment**: when on a page containing no hyperlinks (i.e., a rank sink), surfer may hyperlink to any other page

$$\mathbf{S} = \mathbf{H} + \mathbf{a}(\frac{1}{n} \cdot \mathbf{e}^T) \tag{4}$$

where $a_i = 1$ if page i is a rank sink (has no hyperlinks) and $\mathbf{e}$ is a vector of ones

Stochasticity adjustment example



$$\mathsf{S} = \begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

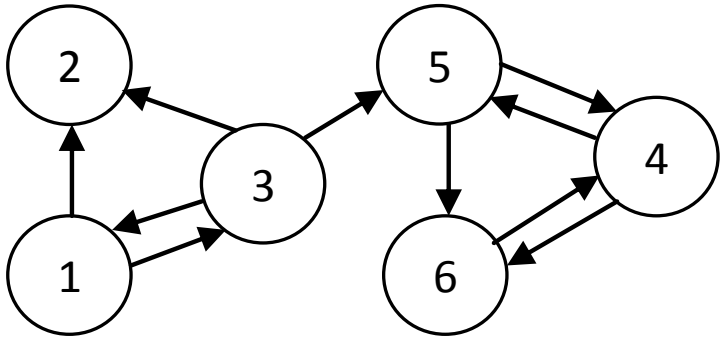
Adjustments to the basic PageRank model

- Matrix $\mathbf{S}$ in (4) is *stochastic* but, in general, it is not *primitive* (*primitive* = *irreducible* and *aperiodic*)
- **Primitivity adjustment**: at times, a random surfer gets bored and abandons the hyperlink method of surfing, entering a new destination into the browser's URL box

$$\mathbf{G} = \alpha\mathbf{S} + (1 - \alpha)\mathbf{E} \tag{5}$$

  where α is a scalar between 0 and 1 determining the probability of "teleports" and $\mathbf{E} = \frac{1}{n} \cdot \mathbf{e}\mathbf{e}^T$ is a normalized matrix of ones
- $\mathbf{G}$ is the "Google matrix" – *stochastic*, *irreducible*, and *aperiodic*
- $\boldsymbol{\pi}_{k+1}^T = \boldsymbol{\pi}_k^T \mathbf{G}$ converges to a positive vector $\boldsymbol{\pi}$

Primitivity adjustment example



- $\alpha = \frac{9}{10}, n = 6, G_{ij} = \alpha S_{ij} + (1-\alpha) \cdot \frac{1}{n} = \frac{9}{10} \cdot S_{ij} + \frac{1}{10} \cdot \frac{1}{6}$

$$\mathbf{G} = \begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{pmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{pmatrix} \end{array}$$

Computational complexity concerns

- Unlike the initial Web graph adjacency matrix $\mathbf{H}$, Google matrix $\mathbf{G}$ is a dense matrix
  - Complexity of the multiplication $\boldsymbol{\pi}^T\mathbf{G}$ is $\mathcal{O}(n^2)$
- Fortunately, Google matrix $\mathbf{G}$ can be written as the rank-one update to the very sparse matrix $\mathbf{H}$

$$\begin{aligned}\mathbf{G} &= \alpha\mathbf{S} + (1-\alpha)\mathbf{E} \\ &= \alpha(\mathbf{H} + \frac{1}{n}\mathbf{a}\mathbf{e}^T) + (1-\alpha)\frac{1}{n}\mathbf{e}\mathbf{e}^T \\ &= \alpha\mathbf{H} + \big(\alpha\mathbf{a} + (1-\alpha)\mathbf{e}\big)\cdot\frac{1}{n}\mathbf{e}^T\end{aligned}$$

- The power method applied to $\mathbf{G}$ without ever computing $\mathbf{S}$ or $\mathbf{G}$ – with $\boldsymbol{\pi}^T\mathbf{e} = 1$, we get:

$$\begin{aligned}\boldsymbol{\pi}_{k+1}^T &= \boldsymbol{\pi}_k^T\mathbf{G} \\ &= \alpha\boldsymbol{\pi}_k^T\mathbf{H} + (\alpha\boldsymbol{\pi}_k^T\mathbf{a} + \boldsymbol{\pi}_k^T(1-\alpha)\mathbf{e})\frac{\mathbf{e}^T}{n}\end{aligned}$$

Learning outcomes 3 – CHECK

1. State the PageRank hypothesis
2. Define and explain the PageRank iterative update formula in matrix form