DIS17 – Search Engine Technologies

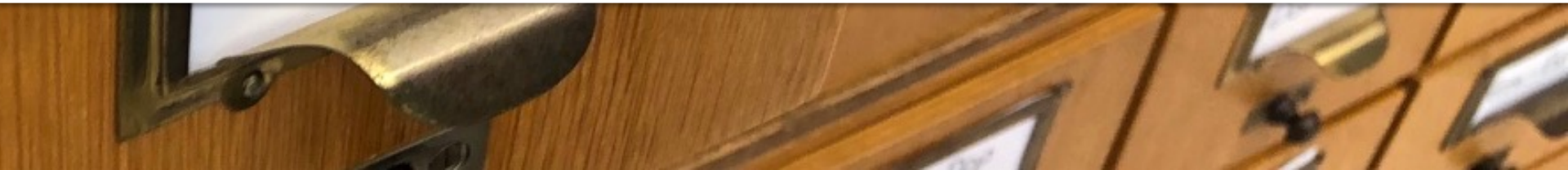
02 – Information Retrieval in a Nutshell and TREC Evaluations

Philipp Schaer, Technische Hochschule Köln, Cologne, Germany

Version: WS 2021

Technology
Arts Sciences
TH Köln

A long time ago…



What was retrieval again…?

This course is about **applied information retrieval** and we use the following temporary working definition:

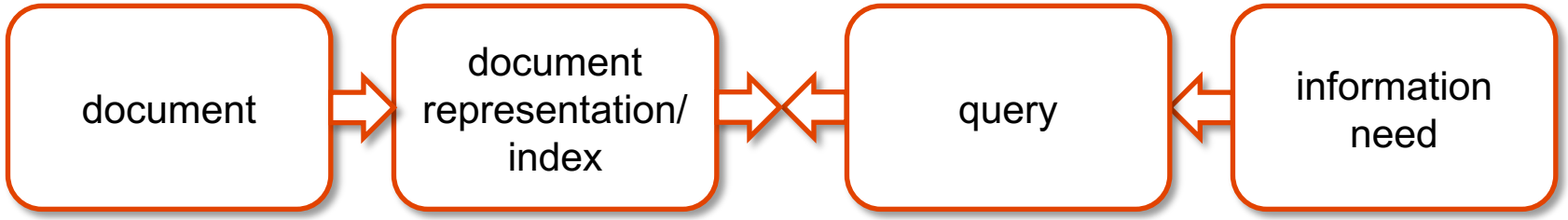
"Given a query and a corpus of documents,
find relevant documents."

- **Query**: a query is a description of an information need sent to the IR system. Can be natural language or formal (query language).
- **Corpus/Collection**: A collection of searchable documents / resources. In our case, mostly text documents.
- **Relevance**: Satisfaction of a user's information need.

Classic ad-hoc retrieval model

The classical ad-hoc retrieval is based on the matching of **document terms** (document representation) and **query terms** (query).

In the classical information retrieval model, the **information need** as well as the query are rigid and do not change.



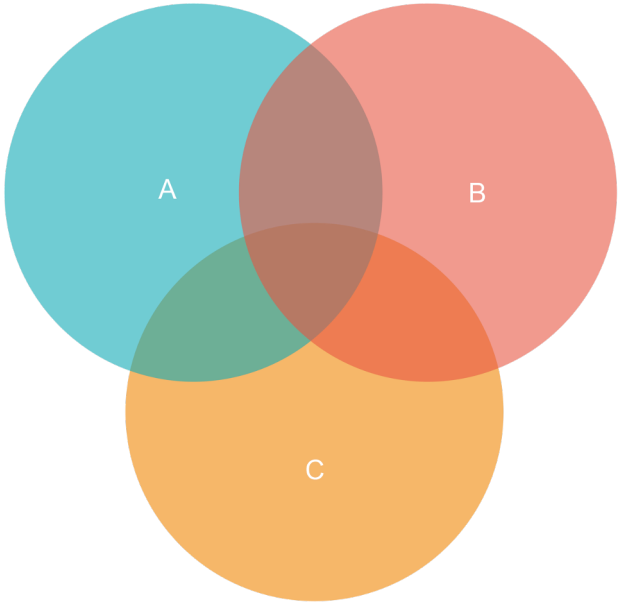
Binary

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calphurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Each document is represented by a **binary vector** (consists only of 0/1), which was precalculated!

1 if play contains term,
0 otherwise

Boolean Retrieval



Simple Boolean/binary decisions

- using AND, OR, NOT to join terms
- Result fulfils requirements or not

Advantages:

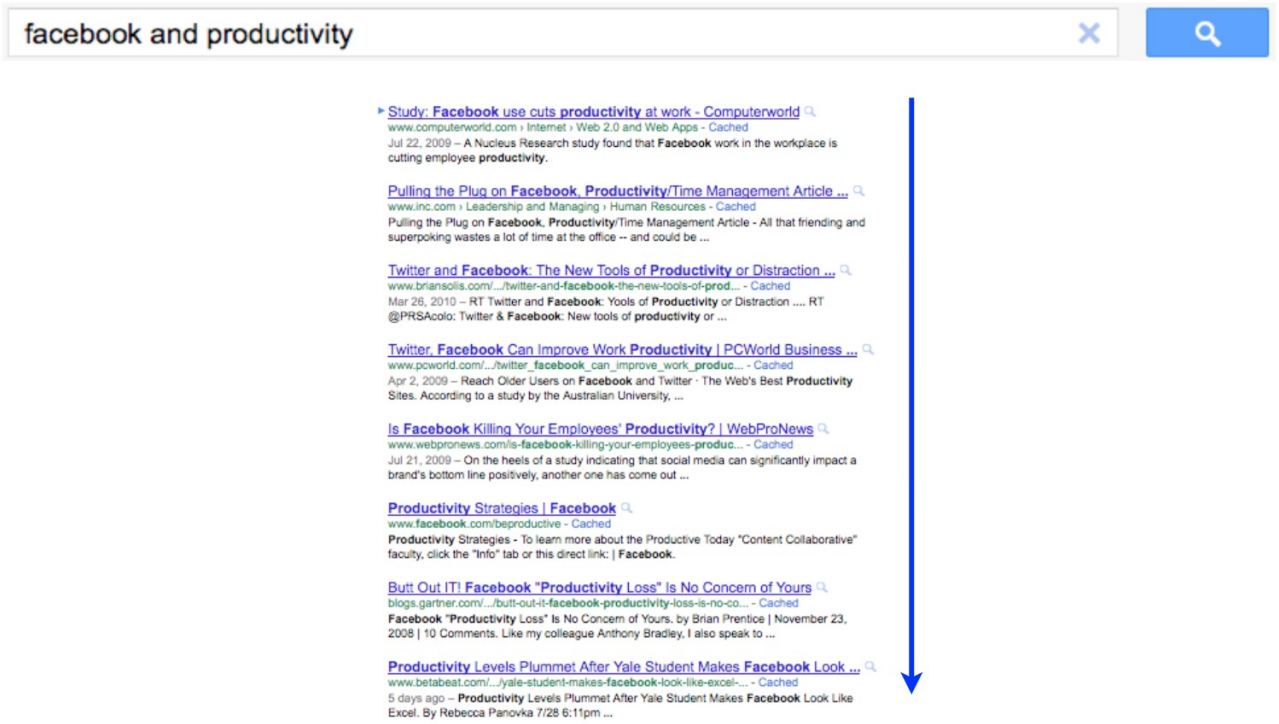
- simple queries; easy to understand
- relatively easy to implement (Term-Document-Matrix)

Disadvantages:

- difficult to specify exact requests
- too much / too little results (Feast or Famine)
- sorting, but not ranking

Most used IR model until the breakthrough of the web. Still used for E-Mail searching, various library catalogue, …

Retrieval as a search task



- **Output:** A **ranking** of documents, in descending order of your **estimated relevance** (makes it easier!).
- **Assumption**: The user looks at **the first few documents** and is satisfied if he found something suitable.

Binary → Frequencies

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	1
Brutus	4	157	0	2	0	0
Caesar	232	227	0	2	1	0
Calphurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	8	5	8
worser	2	0	1	1	1	5

Each document is represented by a vector of **term frequencies**.

Term frequency tf

- The **term frequency $tf_{t,d}$** of a term t in a **document d** is the number of time t appears in d.
- We want to **rank** the documents based on their **score**, which describes the similarity between query and document. For this we want to use the term frequency.
- But how?

- Pure term frequencies are unsuitable because:
  - A document with tf = 10 is certainly more relevant than a document with tf = 1....
  - But not necessarily 10 times more relevant....
  - The relevance **does not increase proportionally** with the term frequency.

Log term frequency weighting

To dampen the effect of the term frequency, the **logarithmized** term frequency is used:

$$w_{t,d} = \begin{cases} 1 + \log_{10}(\mathrm{tf}_{t,d}), & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

"The weight of the term t for the document d".

The score for a query-document pair is the **sum over all weights of all terms t contained in both q and d**:

$$\mathrm{Score}_{q,d} = \sum_{t \in q \cap d} w_{t,d} = \sum_{t \in q \cap d} \left(1 + \log_{10}(\mathrm{tf}_{t,d})\right)$$

Document frequency

Frequent terms are **less informative** than rare terms.

- Imagine a query term that occurs often, e.g., high, certain, expensive....
- A document that contains such a term is **probably more relevant** than one that does not (the basic principle of tf).
- But: it is **not a sure indicator of relevance**.

We want positive weights for words like high, sure, expensive, but **these should be lower than those for rare terms**.

- For this we use the **document frequency (df)**.

idf weighting

df_t is the **document frequency** for t: The number of documents that contain t.

- df_t is a measure for the **inverse information value** of t.
- $df_t \leq N$ (N is the number of all documents)

We define **idf** (inverse document frequency) of t:

$$idf_t = \log_{10}(\frac{N}{\mathrm{df}_t})$$

We use $\log_{10}(N/df_t)$ instead of N/df_t to dampen the effect of idf.

tf-idf weighting

The **tf-idf weighting** of terms is defined as the **product of tf and idf** values for a specific term t:

$$\text{tf}-\text{idf}_{t,d} = \text{tf}_{t,d} * \log_{10}\left(\frac{N}{\text{df}_t}\right)$$

tf-idf is **THE best-known weighting scheme** in IR.

- Be aware: The „-“ is a hyphen, not a minus.
- Other naming patterns: tf.idf, tf x idf, tf*idf, TF*IDF, etc...
- Different combinations are possible... (like different logarithms)

The tf-idf **increases** for

- the number of term frequencies in documents and
- the rarity of a term in the collection.

Binary → Frequency → Weights

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	4,73	2,20	0	0	0	0,03
Brutus	0,12	4,73	0	0,06	0	0
Caesar	4,09	4,00	0	0,04	0,02	0
Calphurnia	0	0,78	0	0	0	0
Cleopatra	4,44	0	0	0	0	0
mercy	0,02	0	0,02	0,06	0,04	0,06
worser	0,02	0	0,01	0,01	0,01	0,04

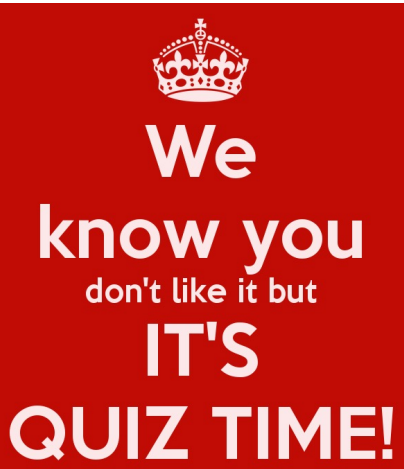
Every document is now represented through a vector with **tf-idf weights** $\in \mathbb{R}^{|V|}$

Computation of the score per document

$$Score(q,d) = \sum_{t \in q \cap d} \text{tf-idf}_{t,d}$$

There are very, very many variations of this scoring, like how to

- calculate tf (with or without logarithms)
- weight the terms in the query, and, and, and, …



At this point we – given a query – are able to rank documents. But how?

Vector Space Model

Documents can now be represented with an TF-IDF vector

- documents are in a |V|-dimensional vector space, |V| is the number of words in our space
- Words in V are the axes and documents are the pointers
- Queries are transformed into vectors in the same vector space
- now **cosine similarity between query vector and document vectors** can be calculated

Key idea:

- Get away from the 0/1 Boolean Model; ranking instead of filtering
- Documents are ranked by their calculated cosine similarity score.
- the user can now get the top-k (e.g. 10) documents, that are the closest to the query vector

Probabilistic Models (like BM25)

Calculates probability of relevancy given a document and query

- mostly based on Bayes' theorem on conditional probability
- generative models of documents and queries as bags-of-words
- with probability estimation similar to vector space similarities

Probability of relevance is mostly estimated,

- assumes that probability of relevance depends on the query and document representations
- assumes that relevant and irrelevant documents differ in their term distribution

Theoretically sound and better relevance than vector space

Language Models

Language models

- assign a **probability to a sequence of words** with a specific probability distribution
- are are trained on specific corpora, e.g., the Wikipedia corpus

They can be used for interpreting entire text, not only words

- considers sentence structure
- knows entities, like names, brands, etc.
- takes surrounding words into account

Better results in retrieval compared to vector space or conventional probabilistic approaches

Ranking – Which one is better?

- Relevant hits are marked in red.

A long time ago… Retrieval measures

	1	2	3	4	5	6
System A	R	R	R	R	R	R
System B	N	N	N	N	N	N
System C	R	R	N	N	R	R
System D	N	N	R	R	R	N

What are the values for the following measures?

- Recall
- Precision
- P@4
- (M)AP (hint: there are a total of 6 relevant documents!)

5 minutes break – **try to calculate**!

Evaluation styles…

- Academia (TREC, CLEF, …)
- Industry

Left image from: https://trec.nist.gov/presentations/TREC2002/overview11.pdf | Right image by leaflet, licenced under CC BY-SA 3.0, https://commons.wikimedia.org/wiki/File:Ralls_Texas_Grain_Silos_2010.jpg

How to evalute ranked lists? Offline.

IR **offline evaluations** require four components

- One common **set of documents**
- A lot of requests, so-called **topics** (min. 25, better 50)
- Relevance ratings, so-called **qrels**
- Our retrieval results, so-called **runs**

Different **measurements** can be determined from this

- Set-based: **Precision** (accuracy), **Recall** (completeness), F1
- List-based: P@k, MAP, R-Precision, MRR, …

Relevance judgements depending on **human assessments**

- problems with manual relevance assessments: experts contradict each other. Kappa values to determine the agreement of assessors

Relevane Feedback. Online.

IR **online evaluations** are based on interactions of real users

- Click-Through Rate (CTR)
  - can be extended by bounce-rate to calculate intended click
- Digestions numbers
  - Number of views, downloads, buyers, query reformulation, …

Main Advantages

- a lot of signals
- user are not aware of the eval. setting and behave normally

Main Disadvantages

- with a lot of signals comes a lot of noise → results are only somewhat meaningful
- bots are a nightmare

NIST – National Institute of Standards and Technology

TREC – Simplified workflow

TREC provides the **task**, **corpora**, **topics**, and in most cases **relevance judgements** every year

- can be downloaded on their website

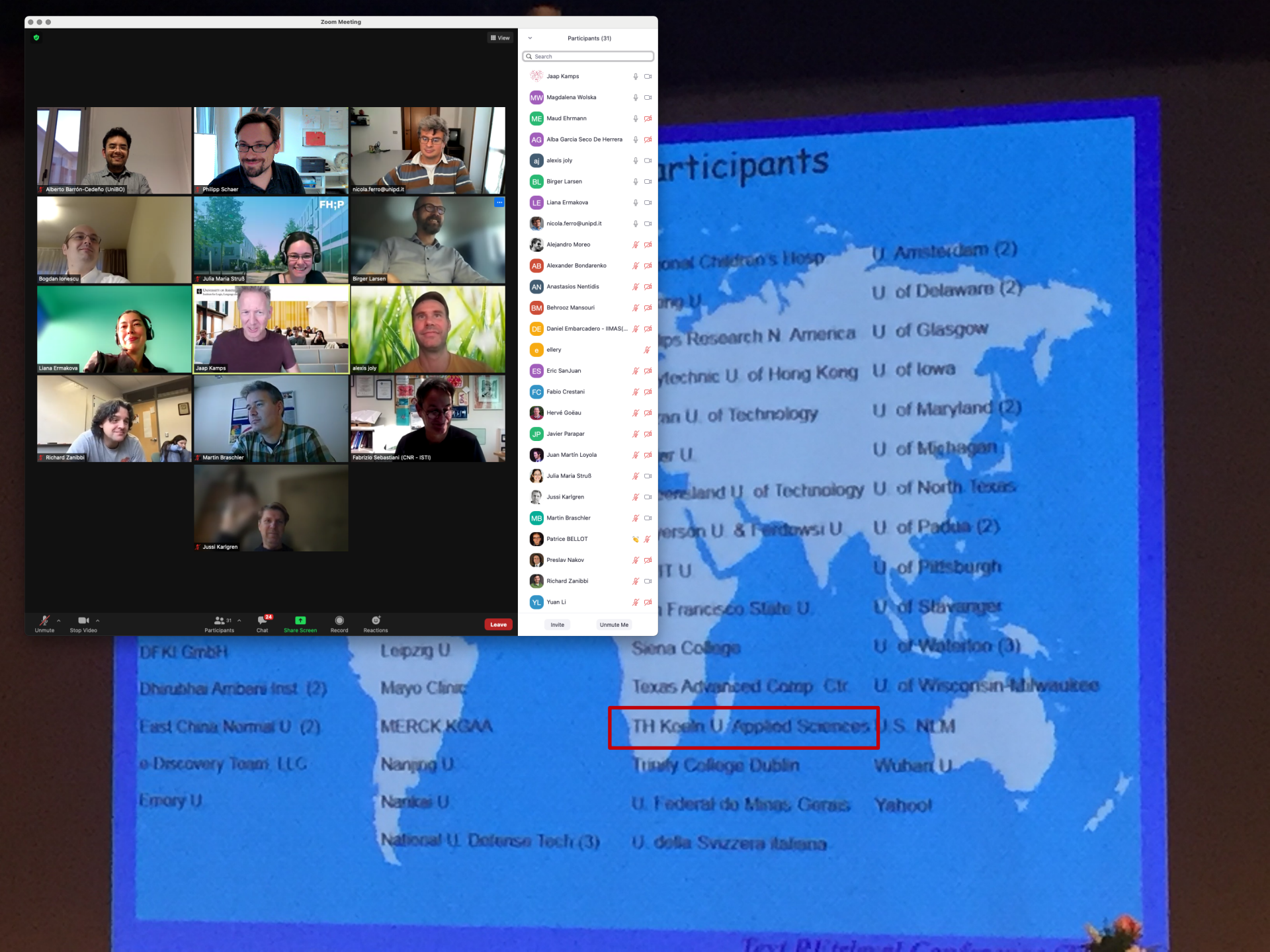
Everyone can work on the task in the best possible way

- all participants of a track compute a **ranking solution** for the task
- all participants **send in their retrieval results**

Results are collected and evaluated at TREC

- TREC illustrates a leader board and the best participants are invited to the conference
- invited participants present their approach and explain how they achieved the results

And guess what! We are going to do exactly this!


Zoom Meeting
View
Participants (31)
Search
Jaap Kamps
Magdalena Wolska
Maud Ehrmann
Alba Garcia Seco De Herrera
alexis joly
Birger Larsen
Liana Ermakova
nicola.ferro@unipd.it
Alejandro Moreo
Alexander Bondarenko
Anastasios Nentidis
Behrooz Mansouri
Daniel Embarcadero - IIMAS(...
ellery
Eric SanJuan
Fabio Crestani
Hervé Goëau
Javier Parapar
Juan Martín Loyola
Julia Maria Struß
Jussi Karlgren
Martin Braschler
Patrice BELLOT
Preslav Nakov
Richard Zanibbi
Yuan Li
Invite
Unmute Me
Alberto Barrón-Cedeño (UniBO)
Philipp Schaer
nicola.ferro@unipd.it
Bogdan Ionescu
Julia Maria Struß
Birger Larsen
Liana Ermakova
Jaap Kamps
alexis joly
Richard Zanibbi
Martin Braschler
Fabrizio Sebastiani (CNR - ISTI)
Jussi Karlgren
Unmute
Stop Video
Participants
Chat
Share Screen
Record
Reactions
Leave
U. Amsterdam (2)
U. of Delaware (2)
U. of Glasgow
U. of Iowa
U. of Maryland (2)
U. of Michigan
U. of North Texas
U. of Padua (2)
U. of Pittsburgh
U. of Stavanger
U. of Waterloo (3)
U. of Wisconsin-Milwaukee
U.S. NLM
Wuhan U.
Yahoo!
DFKI GmbH
Dhirubhai Ambani Inst. (2)
East China Normal U. (2)
e-Discovery Team, LLC
Emory U.
Leipzig U.
Mayo Clinic
MERCK KGAA
Nanjing U.
Nankai U.
National U. Defense Tech. (3)
Siena College
Texas Advanced Comp. Ctr.
TH Koeln U. Applied Sciences
Trinity College Dublin
U. Federal de Minas Gerais
U. della Svizzera italiana

TREC – Topics

- A typical topic (taken from the GIRT4 collection)

```
<top>
  <num>100</num>
      <DE-title>Kneipenkultur</DE-title>
      <DE-desc>Finde Dokumente, die die Gewohnheiten von
      Kneipengängern thematisieren.</DE-desc>
      <DE-narr>Relevante Dokumente analysieren die
      Gepflogenheiten von Kneipengängern und die Kultur
      der Räumlichkeiten und des Ambientes von
      Szenekneipen.</DE-narr>
</top>
```

TREC – Topics

A topic **describes a need for information**

- But it is **not a query**!!!
- The information need must be translated into a query!

The relevance is always evaluated (manually) based on the description of the information need - not the query!

- Example: “I'm looking for information on whether drinking red wine is more effective at reducing your risk of heart attacks than white wine”
- Query: “wine red white heart attack effective”

The manual evaluation of the retrieval results are not simply about whether the query terms are comprised in the documents

TREC – Document collections

```
<?xml version="1.0" encoding="UTF-8"?>
<doc>
   <field name="id">GIRT-DE19901300229</field>
   <field name="title_txt_de">Vorwärts um jeden Preis - wie der DFB seine Fans
ideologisch ausstattet</field>
   <field name="pubyear_i">1990</field>
   <field name="lang_s">DE</field>
   <field name="abstract_txt_de">Der Beitrag zeigt am Beispiel der
Fußballweltmeisterschaft in Italien, wie der DFB seine "Fans" ideologisch
"ausgestattet" hat und welche negativen Folgen eine auf Kommerz und Nationalgefühl
orientierte Begeisterung haben kann. Völlig vernachlässigt wurden die Möglichkeiten
Einfluß zu nehmen auf den Lebens- und Arbeitsalltag der Fans, auf sinnvolle Förderung
gemeinschaftlicher Aktivität, auf Zusammenarbeit mit anderen Institutionen, auf bewußte
Thematisierung von (männlichen) Widersprüchlichkeiten. (BA2)</field>
   <field name="author_ss">Hartmann, Gerold</field>
   <field name="author_ss">Hering, Wolfgang</field>
   <field name="controlledterm_txt">Fußball</field>
   <field name="controlledterm_txt">Fan</field>
   <field name="controlledterm_txt">Nationalismus</field>
   <field name="controlledterm_txt">Rechtsradikalismus</field>
   <field name="controlledterm_txt">Jugendlicher</field>
   <field name="controlledterm_txt">Massenmedien</field>
   <field name="method_txt">beschreibend</field>
</doc>
```

TREC_EVAL

TREC_EVAL is **THE standard evaluation toolkit** for TREC runs

- Remember, in TREC, retrieval results are called "**runs**"
- allows a common foundation for evaluating runs
- is open source and therefore completely reusable and open for verification
- allows the extension by own evaluation methods
- is actually used :-) (both by science and industry)
- is used via the Unix shell or the Windows command line

TREC_EVAL command line syntax

- `./trec_eval [-h] [-q] {-m measure}* trec_rel_file trec_top_file`

- wobei
  - **trec_eval**: name of the program
  - -h: allows the output of a help text
  - -q: is a parameter that allows to get results for each topic
  - -m: allows the selection of individual result types (e.g. map, bpref)
  - **trec_rel_file**: is the qrel-file containing the relevance ratings
  - **trec_run_file**: is the file containing the runs, i.e., the ranked retrieval results

Only the parameters in bold are mandatory, the others are optional

```
[1648][schaer@lapad33:~/aiw-suma/it3-set/solr-trec]$ trec_eval qrel.txt run.txt
runid                   all     0
num_q                   all     1
num_ret                 all     10
num_rel                 all     5
num_rel_ret             all     5
map                     all     0.5444
gm_map                  all     0.5444
Rprec                   all     0.4000
bpref                   all     0.4800
recip_rank              all     0.5000
iprec_at_recall_0.00    all     0.6667
iprec_at_recall_0.10    all     0.6667
iprec_at_recall_0.20    all     0.6667
iprec_at_recall_0.30    all     0.6667
iprec_at_recall_0.40    all     0.6667
iprec_at_recall_0.50    all     0.5556
iprec_at_recall_0.60    all     0.5556
iprec_at_recall_0.70    all     0.5556
iprec_at_recall_0.80    all     0.5556
iprec_at_recall_0.90    all     0.5556
iprec_at_recall_1.00    all     0.5556
P_5                     all     0.4000
P_10                    all     0.5000
P_15                    all     0.3333
P_20                    all     0.2500
P_30                    all     0.1667
P_100                   all     0.0500
P_200                   all     0.0250
P_500                   all     0.0100
P_1000                  all     0.0050
```

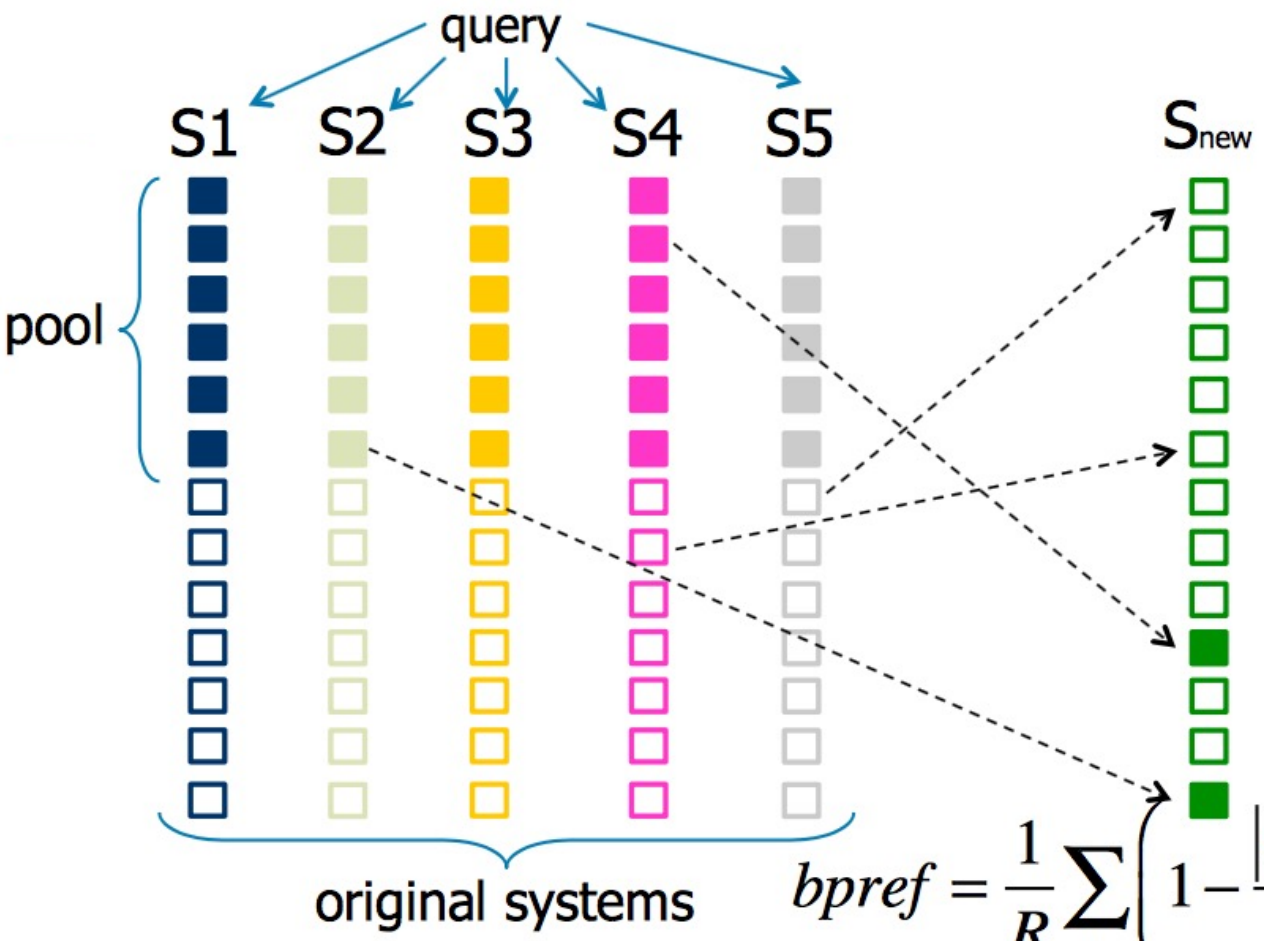
Quiz time! What is the Precision?

TREC_EVAL output

num_q	number of topics worked on in the runs
num_rel	total number of relevant documents
num_rel_ret	number of returned relevant documents
map	Mean Average Precision
R-prec	R-Precision
bpref	Binary Preference
recip_rank	Reciprocal Rank
P5	Precision at 5
P10	Precision at 10
...	...

new, and interesting!

bpref – unjudged documents



Normally unjudged documents are counted as unrelevant.

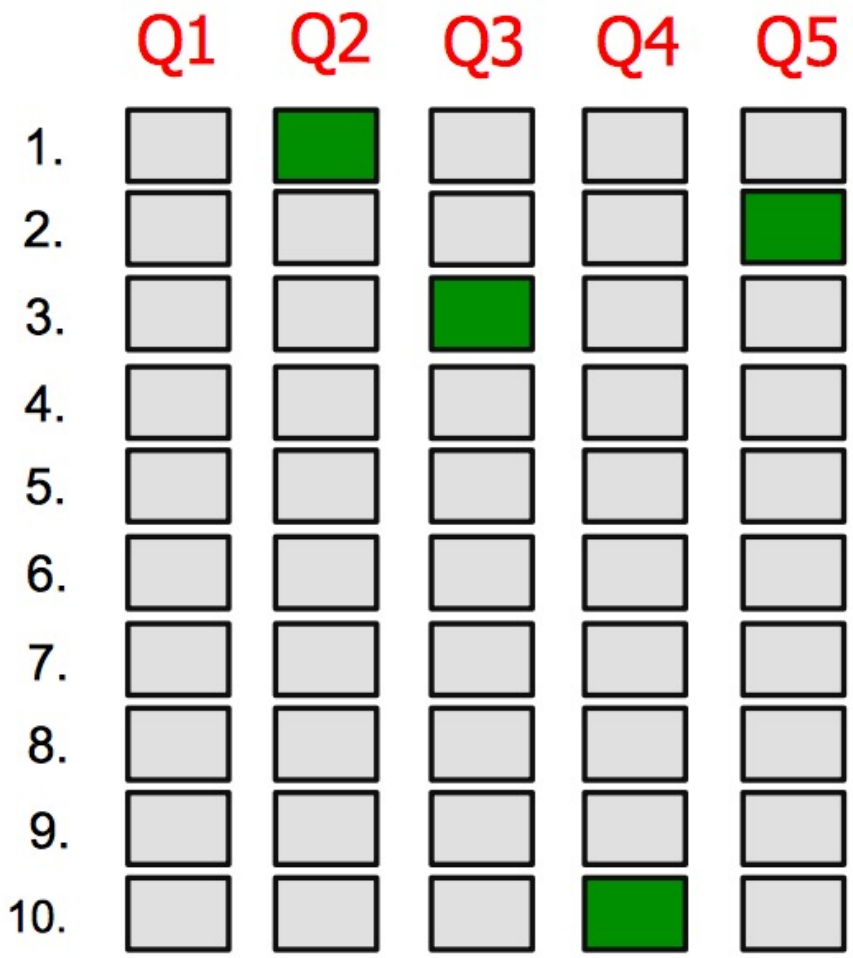
bpref ignored unjudged documents and doesn‘t downweight the measure.

Only **known non-relevant** documents are counted.

$$bpref = \frac{1}{R}\sum_{r}\left(1 - \frac{|n \text{ ranked higher than } r|}{R}\right)$$

Hauff (2012)

Mean Reciprocal Rank



RR	0.0	1.0	0.33	0.01	0.5

MRR=0.369

The key idea of this measure is simple: When does **the first relevant document** appear?

$$RR = \frac{1}{rank\ of\ relevant\ document}$$

Hauff (2012)

TREC_EVAL format: qrels

- **qrels** encode the relevance judgements of the assessors
- TREC_EVAL is a simple text format
  - `topic-id iter document-id relevance`

where

- **topic-id:** is the topic identifier of a topic in the topics fiel
- **iter:** is a constant about the iteration (not used, but needed)
- **document-id:** is the document-id from our collection
- **relevance:** the relevance value of the document-id for the query-id
  - “-1”: not assessed; “0”: not relevant; “1-255”: relevant

qrels – example

topic-id	iter	document-id	relevance
301	0	FR940202-2-00150	1
301	0	CR93E-10505	0
301	0	CR93E-1282	1
302	0	CR93E-10071	0
302	0	CR93E-10276	0
302	0	CR93E-10279	2

needed!,
but ignored

These relevance values are the key and ground truth for our experiments.

TREC_EVAL format: run results

- Runs are the individual retrieval results that are now “running for evaluation”. Again a simple text format is used :
  - `topic-id Q0 document-id rank score Exp`

where

- **topic-id:** is the topic identifier from the topic files
- **iter** and **Exp**: are constant (e.g. 0) and are rarely used, but ... ☺
  document-id: is the document-id from our collection
- **rank**: position in the result list (starting by 0, is usually ignored)
- **score**: is the estimated relevance value given by the search engine (this is the ranking)

run results – example

topic-id	iter	document-id	rank	sim	run_id
301	Q0	FBIS4-50478	1	3.340779	baseline
301	Q0	FR940202-2-00150	104	2.129133	baseline
301	Q0	FBIS4-45552	105	2.127882	baseline
301	Q0	FBIS4-49075	119	2.112576	algoX
301	Q0	FBIS3-27288	499	1.655729	algoX
302	Q0	FR940126-2-00106	1	3.903381	baseline
302	Q0	FBIS3-60449	200	1.374640	algoX
302	Q0	FBIS3-60572	499	1.099626	algoX

needed!,
but ignored

Demo time: TREC_EVAL in the wild