CS918: LECTURE 15

Relevance Feedback and Query Expansion

Arkaitz Zubiaga, 21st November, 2018

RECAP: INDEX ELIMINATION FOR EFFICIENCY

- Basic algorithm; cosine computation algorithm **only** considers docs **containing at least one query term**.

- Take this further:
  - **Only** consider **high-idf query terms**.
  - **Only** consider docs containing **many query terms.**
  - **Champion lists.**

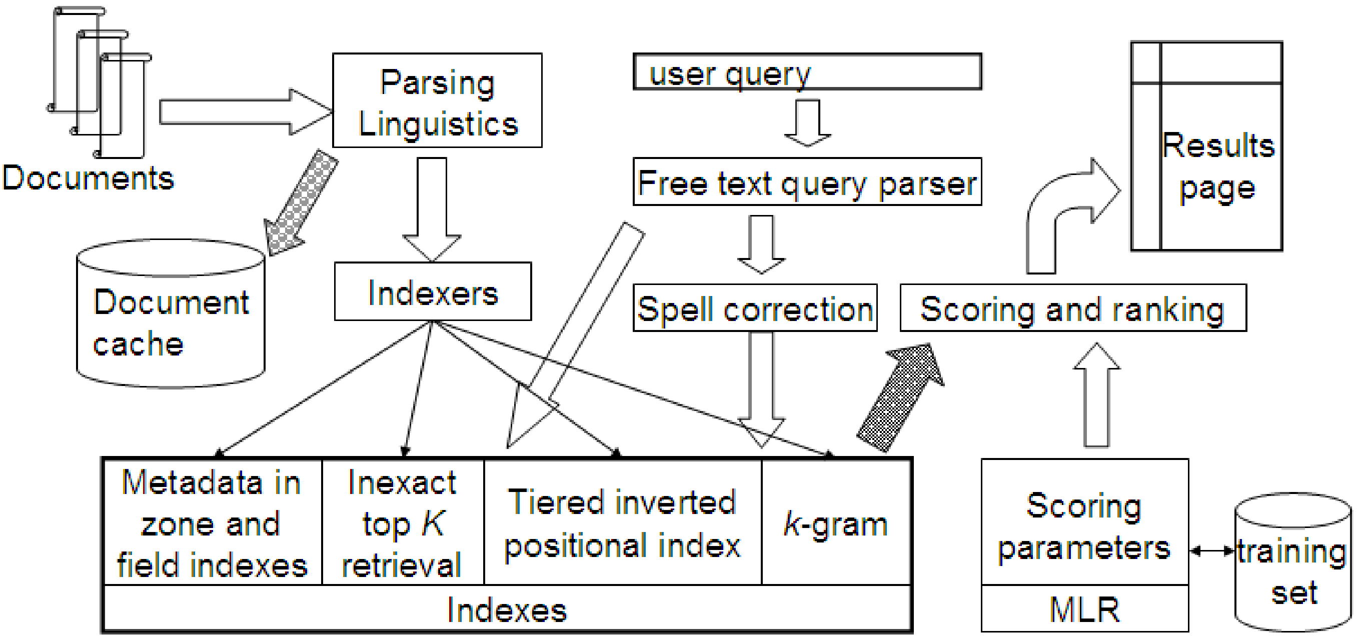
- And of course, **for frequent queries, cache results.**

RECAP: NET SCORE

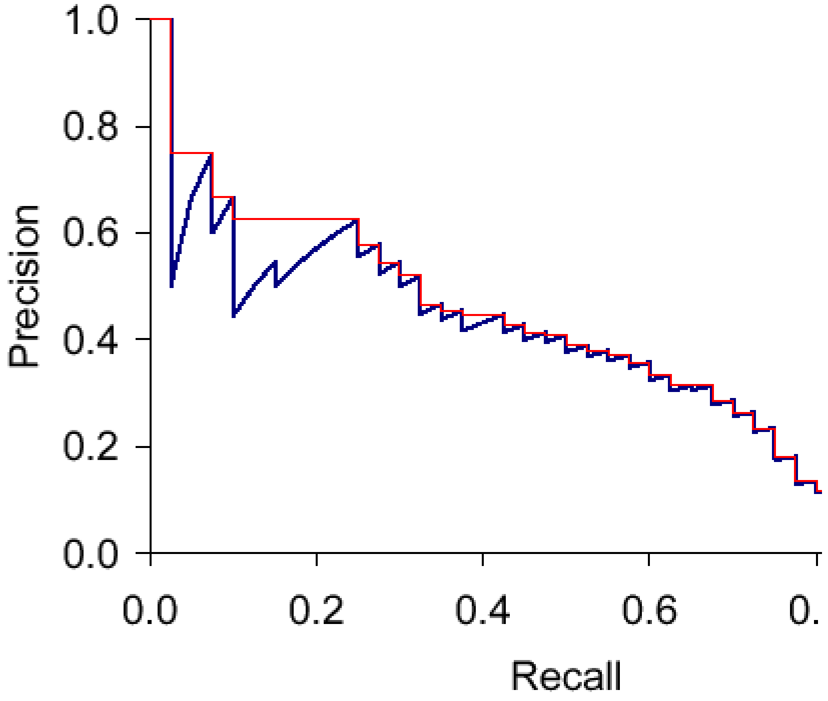
- The **net score** is a simple, total score combining **relevance and authority.**
  - **net-score(q,d) = g(d) + cosine(q,d)**

- Now we seek the top K docs by net score

RECAP: BUILDING AN ENTIRE SEARCH ENGINE



RECAP: A PRECISION-RECALL CURVE



RECAP: PRECISION AT K

- A **simple approach** is precision@k, i.e. ratio of relevant items within the top k results.
  - **Good if ranking is not important**, however these have the same precision@5 of 0.4:
    (regardless of relevant items being the top 2 or the bottom 2 items)
    - **1: R, 2: R, 3: NR, 4: NR, 5: NR**
    - **1: NR, 2: NR, 3: NR, 4: R, 5: R**

- **Better metrics** when **ranking** is important: **MAP, NDCG**.

RECAP: EVALUATION WITH MAP vs NDCG

WARWICK

- Both consider **top results are more important**.
- **MAP is simple**, easy to understand, **widely used**.
- NDCG can consider different levels of relevance.
  - Instead of **just relevant (1) vs non-relevant (0)**.
  - NDCG can take **relevance scores from e.g. 0 to 5**.
    - Especially used in these cases.
  - **MAP can only handle 0's and 1's**.

LECTURE 15: CONTENTS

- Why improve recall.

- Relevance feedback.
  - Interactive relevance feedback.
  - Rocchio' feedback.

- Query expansion.

MOTIVATION

- Search **queries** can be **ambiguous.**

- If I search for:
  - **jaguar**: am I looking for the animal or a car?
  - **windows**: the operating system, window frames or glasses?

MOTIVATION

- Search **queries** may need **expanding/altering**.

- If I search for:
  - **aircraft**: should we also return results for ‘plane’?
  - **study**: document containing ‘learn’ is also relevant?
  - **covetry**: did you mean ‘Coventry’?

- So far, we’re not dealing with this.

MOTIVATION

- Main challenge: we want to **improve recall**.
  - Generally more important than precision.

- **Losing a bit of precision is not that bad**: If I search for 'windows', I can accept having 5 results for the OS and 5 for frames.

- **Losing recall is**: If I search for 'automobile', not getting a relevant result that contains 'car' is a big mistake.

OPTIONS FOR IMPROVING RECALL

- **Local** methods:
  Adjust query based on returned documents.

  Relevance feedback (RF), Pseudo-RF, Indirect RF

- **Global** methods:
  Modify the query, without retrieving documents.

  Query expansion, Linguistic processing of query (spelling correction, stemming,...)

LOCAL METHODS: RELEVANCE FEEDBACK

INTUITION OF RELEVANCE FEEDBACK

- Rather than just a query-response paradigm,

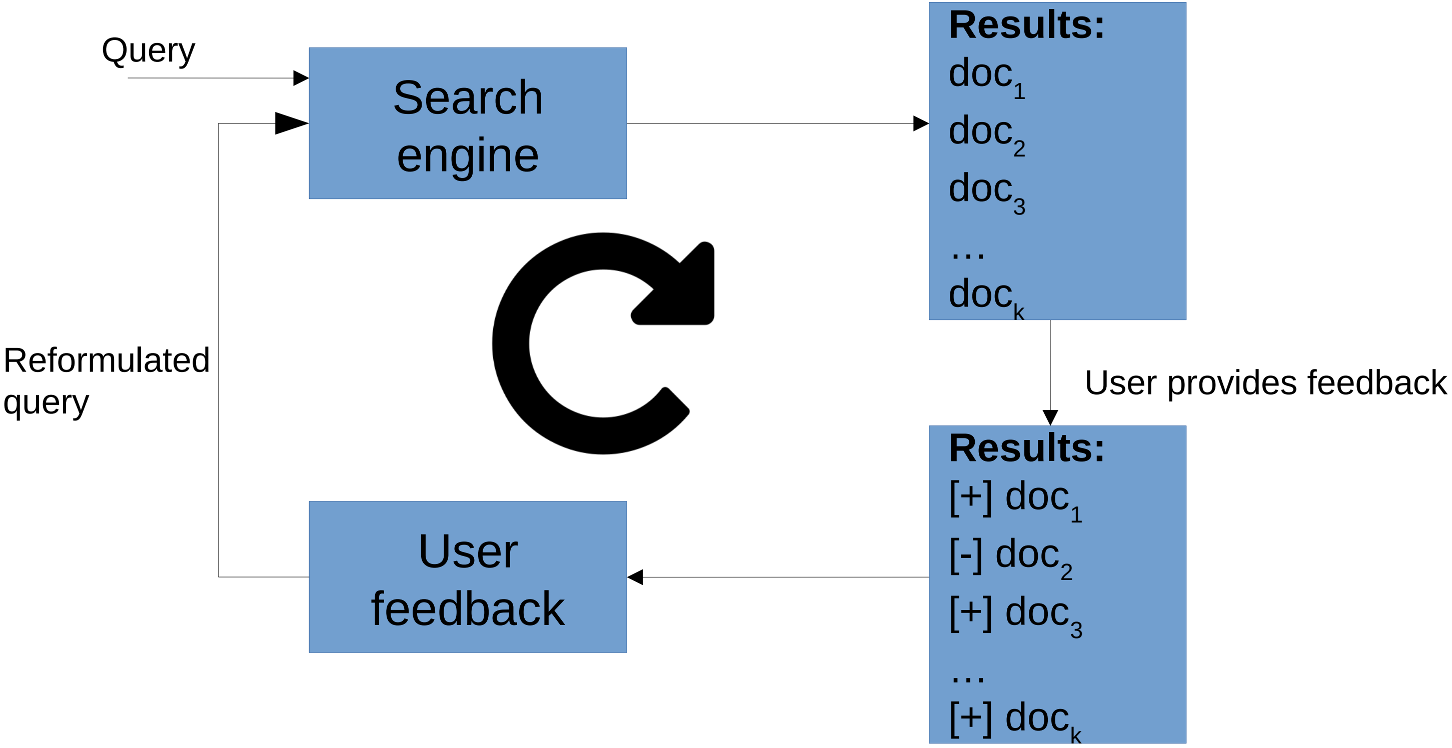
  we understand **search as a conversation** between searcher and engine.

- The user will be requested to **give feedback** on the returned results.

INTUITION OF RELEVANCE FEEDBACK

- User issues a **query**.
- Search engine returns **results**.
- User **marks** docs as **relevant** or **non-relevant**.
- Search engine **reformulates** the **query** based on feedback.
- Search engine returns **results** for **reformulated query**.
    - We expect recall to improve in 2^{nd} iteration.

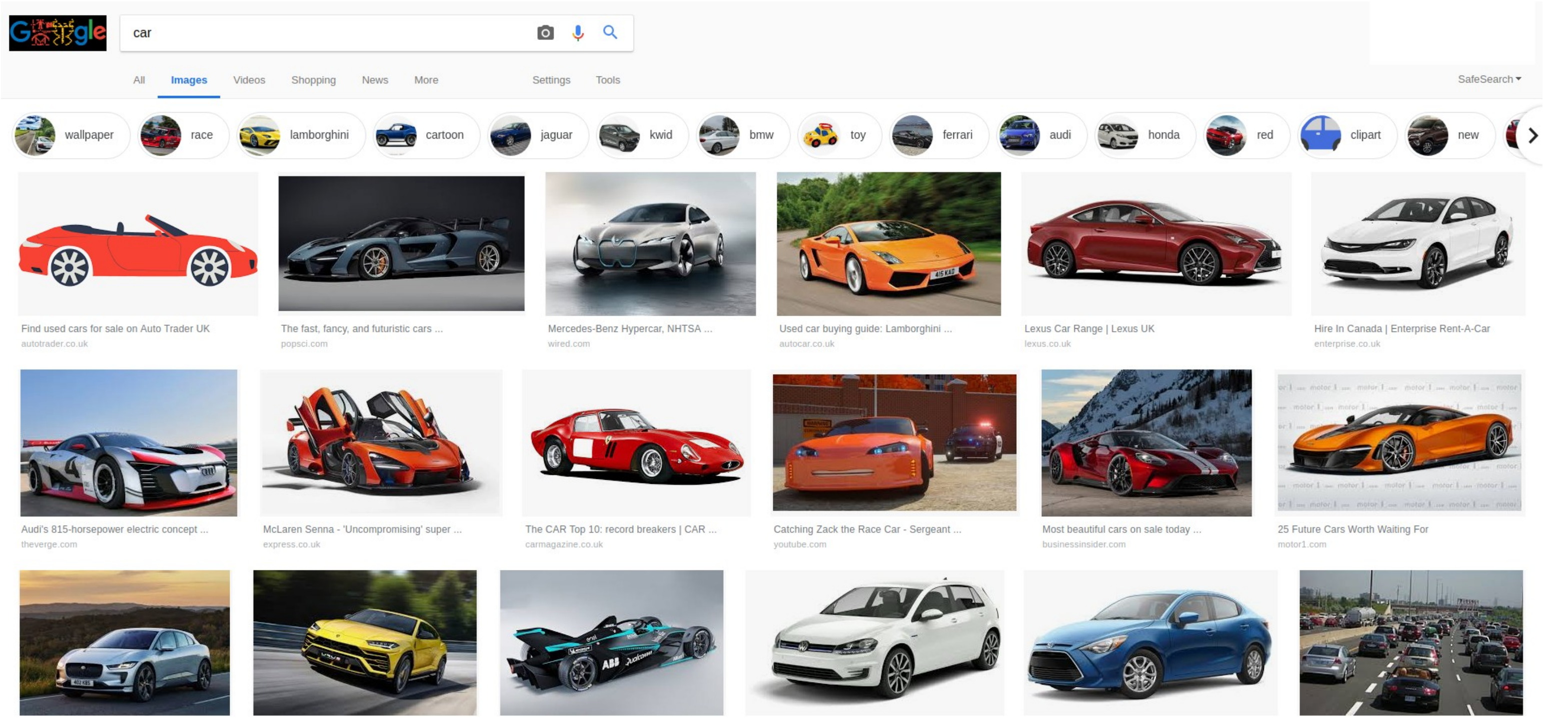
INTUITION OF RELEVANCE FEEDBACK



RELEVANCE FEEDBACK: EXAMPLE

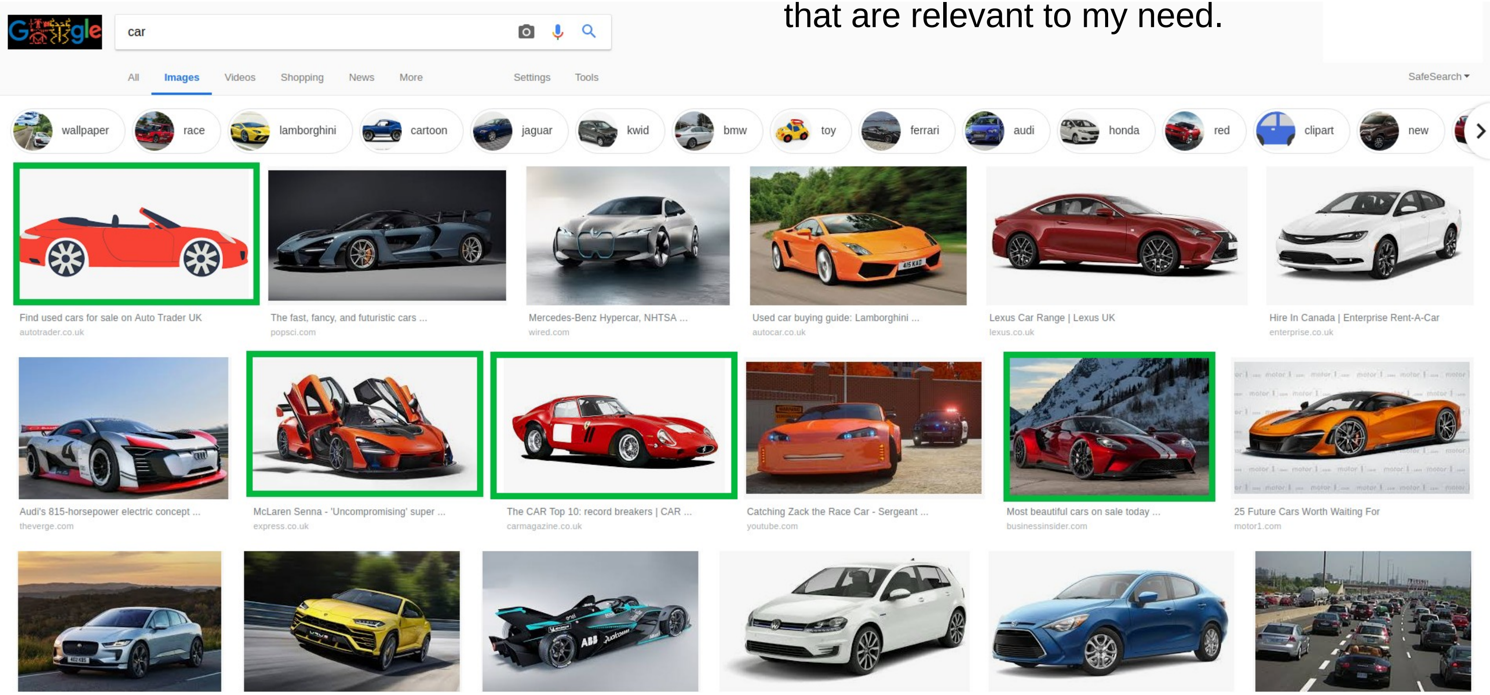
- I issue a search query ‘car’ in Google Images.

RELEVANCE FEEDBACK: EXAMPLE



RELEVANCE FEEDBACK: EXAMPLE

I provide feedback picking the ones that are relevant to my need.

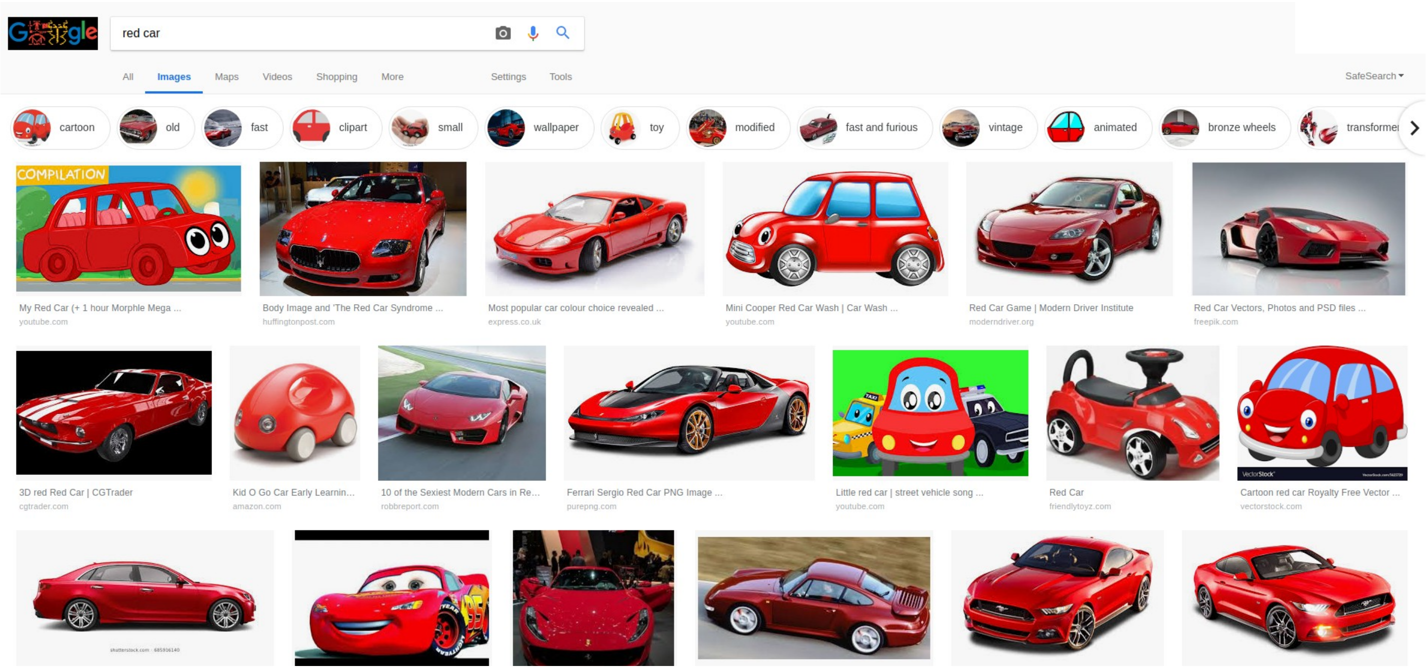


RELEVANCE FEEDBACK: EXAMPLE

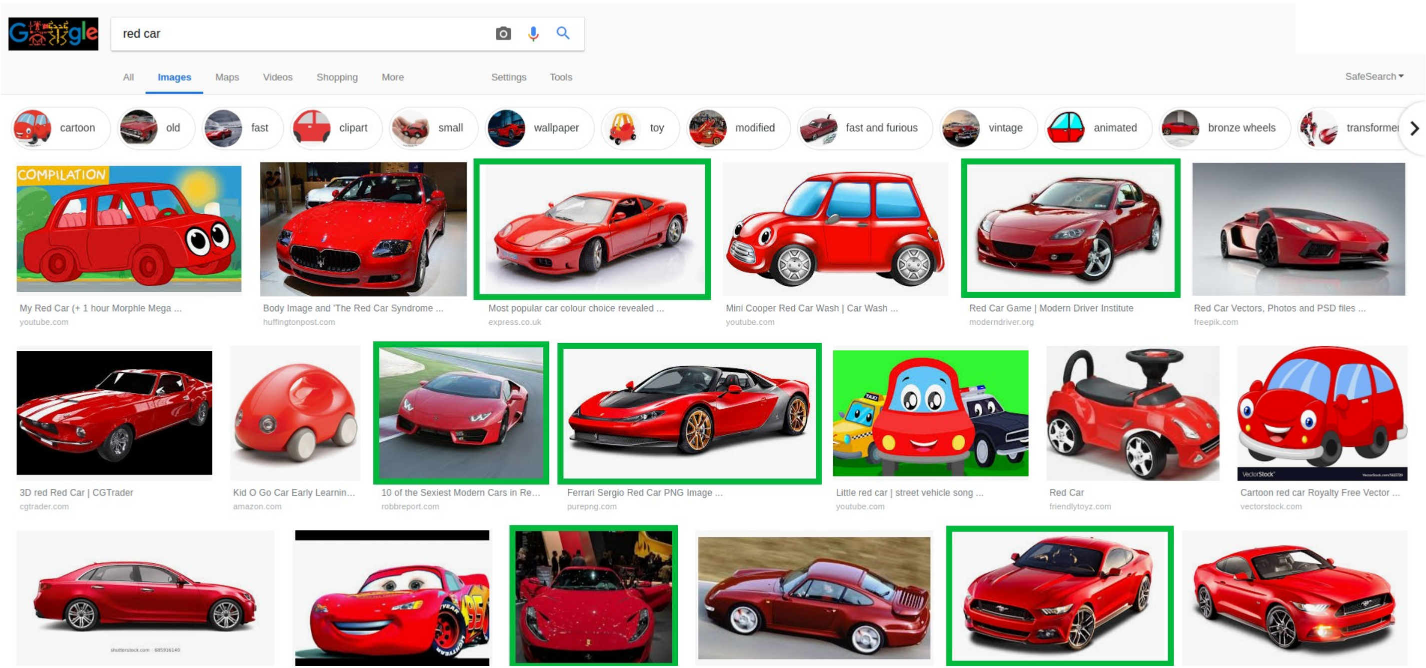
- Search engine: oh, you seem to be looking for red cars!

- Query: “car” → “red car”

RELEVANCE FEEDBACK: EXAMPLE



RELEVANCE FEEDBACK: EXAMPLE

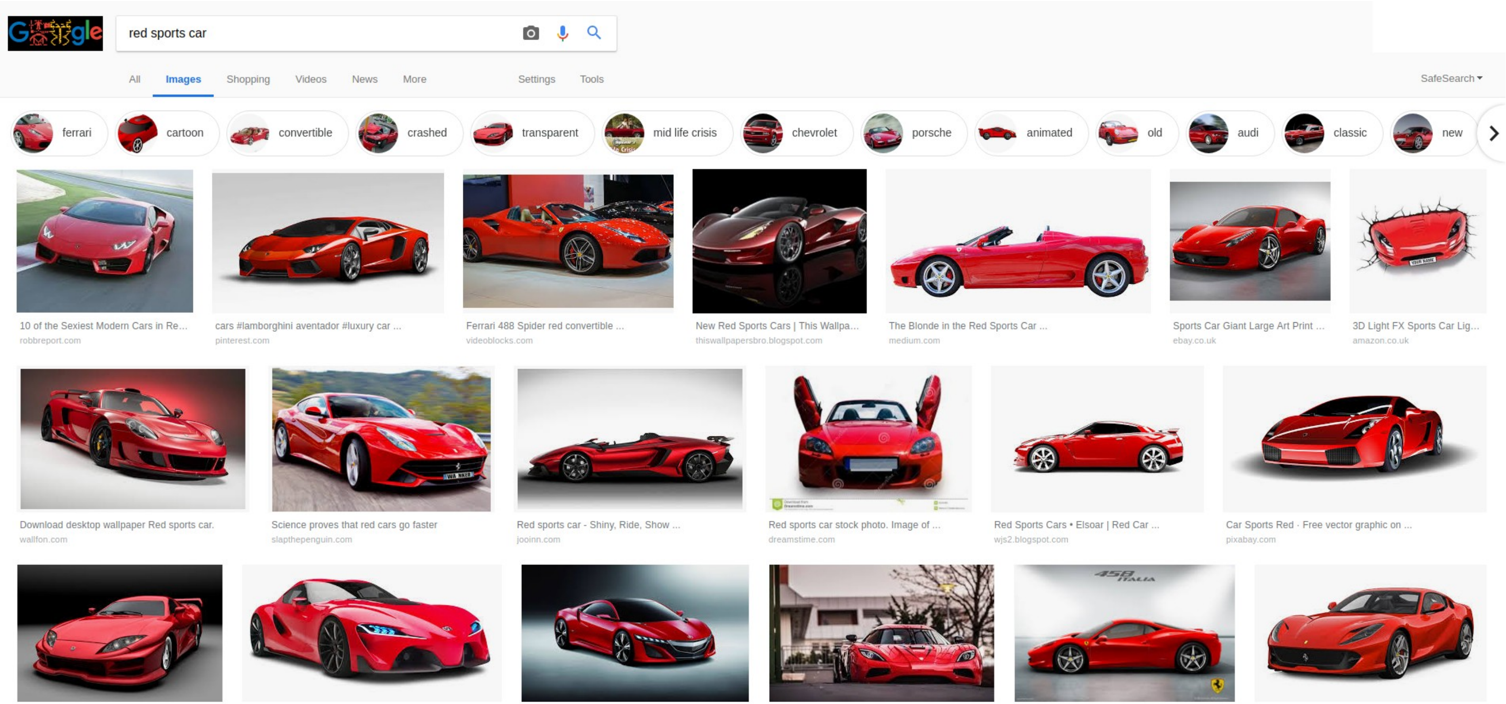


RELEVANCE FEEDBACK: EXAMPLE

- Search engine: oh, right, that’s sports car that you’re after.

- Query: “red car” → “red sports car”

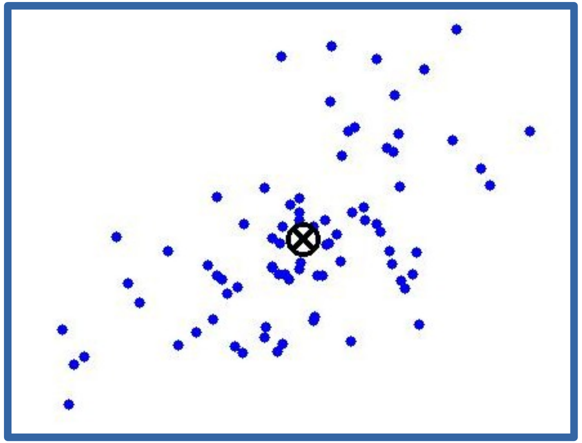
RELEVANCE FEEDBACK: EXAMPLE



HOW TO USE RELEVANCE FEEDBACK

- Key idea is to rely on centroids.
- A centroid is the average data point for a set of vectors/documents.

$$\vec{\mu}(D) = \frac{1}{|D|} \sum_{d \in D} \vec{v}(d)$$



ROCCHIO' ALGORITHM

- Given user's original query: **q**
- And user's relevance feedback: $\mathbf{D}_{\mathbf{R}}$ and $\mathbf{D}_{\mathbf{NR}}$.

- We aim to find the optimised query that satisfies the feedback: $\mathbf{q}_{\mathbf{OPT}}$.

ROCCHIO' ALGORITHM

- **Intuition**: new query q_{OPT} must be:
  - As **similar** as possible to docs deemed **relevant**.
  - As **dissimilar** as possible to docs deemed **non-relevant**.

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, \mu(D_r)) - \text{sim}(\vec{q}, \mu(D_{nr}))]$$

i.e. find **q** with max **similarity** wrt **centroid of relevant docs**, and max **dissimilarity** wrt **centroid of non-relevant docs**.

ROCCHIO' ALGORITHM

- This is however **hard to optimise.**
  - We'd **need to try many different q's – we don't have time**.

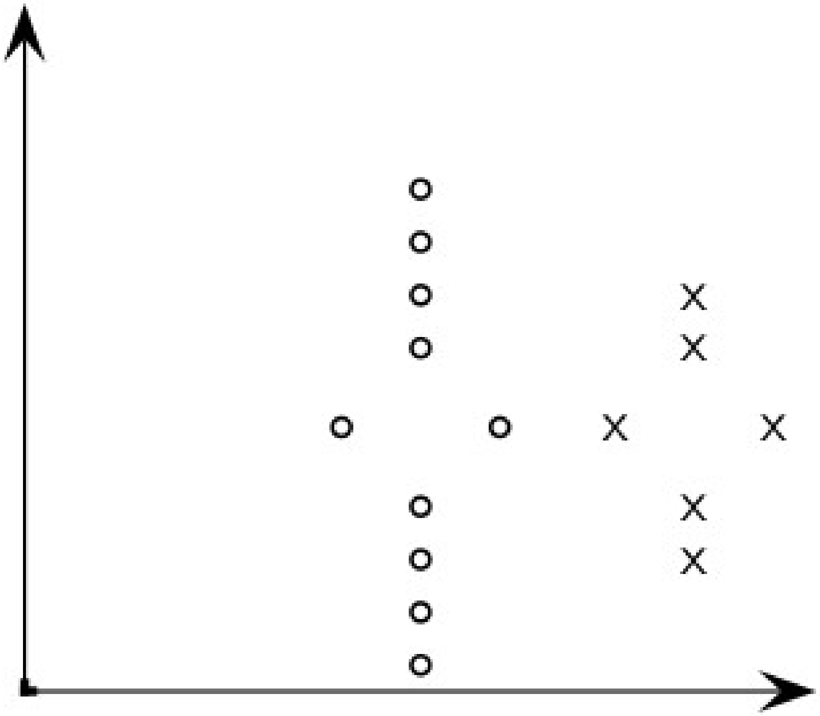
$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, \mu(D_r)) - \text{sim}(\vec{q}, \mu(D_{nr}))]$$

- The simpler alternative that Rocchio' relies on:

$$\vec{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_{nr})]$$

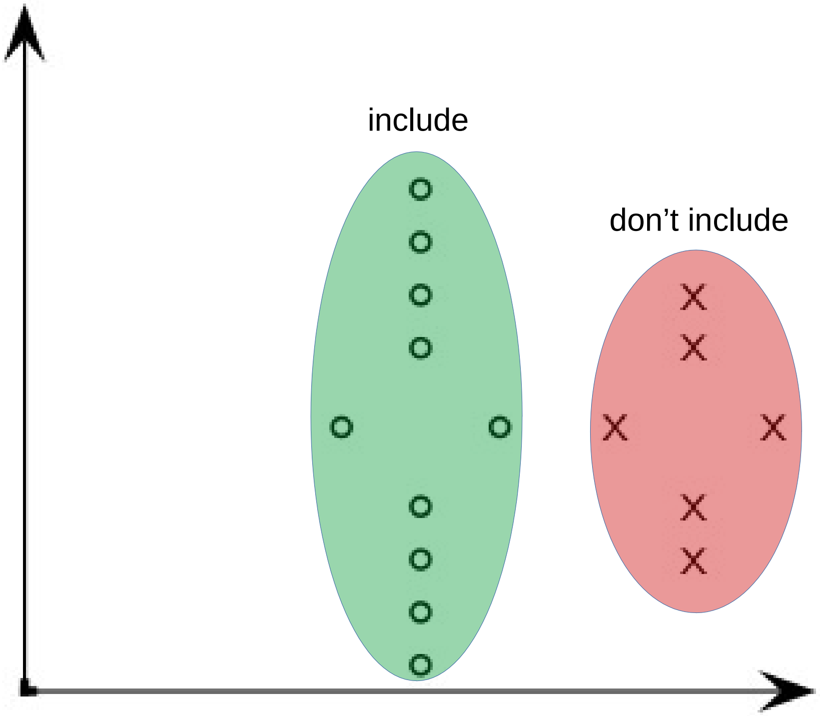
ROCCHIO' EXAMPLE

- Circles: relevant
- Crosses: non-relevant



ROCCHIO' EXAMPLE

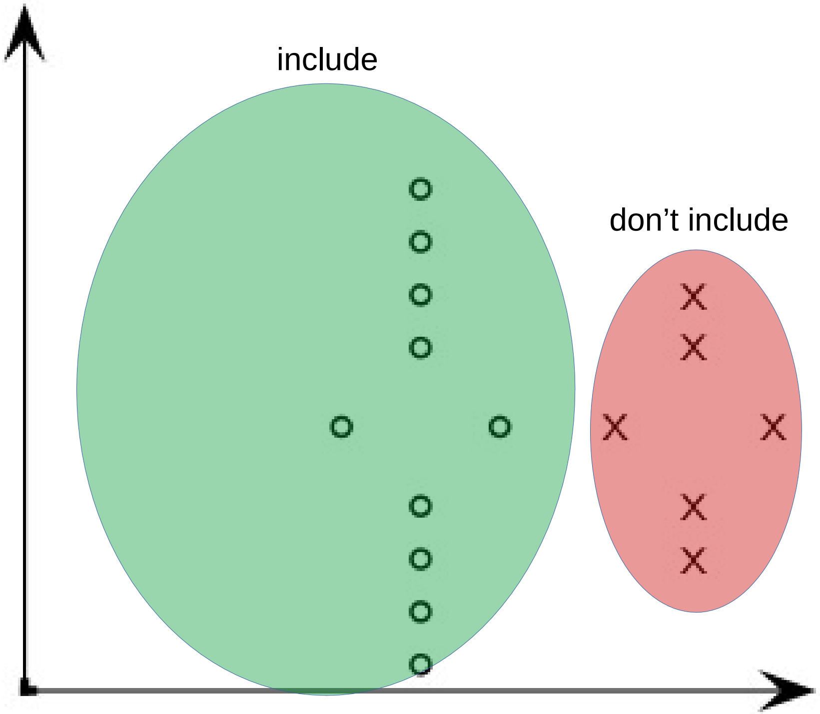
- Relevance feedback collected



ROCCHIO' EXAMPLE

- After applying Rocchio' we want to achieve something like this →

  i.e. the relevant bit
  + an additional area that's far from the non-relevant docs



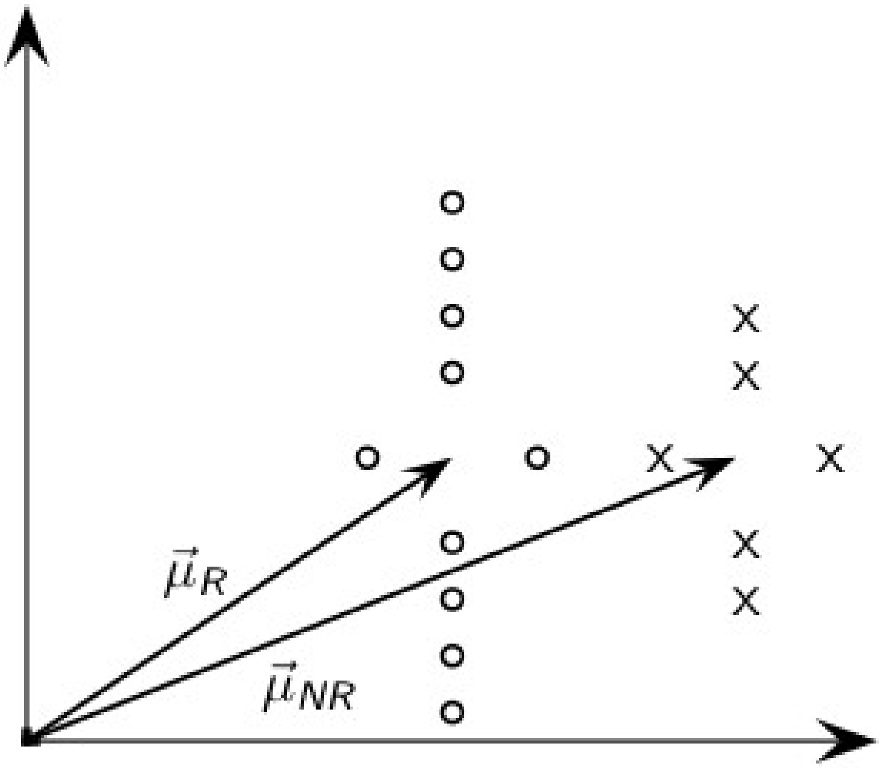
ROCCHIO' EXAMPLE

- With relevance feedback:

  we calculate the centroids for

  relevant docs: μ_R

  non-relevant docs: μ_{NR}



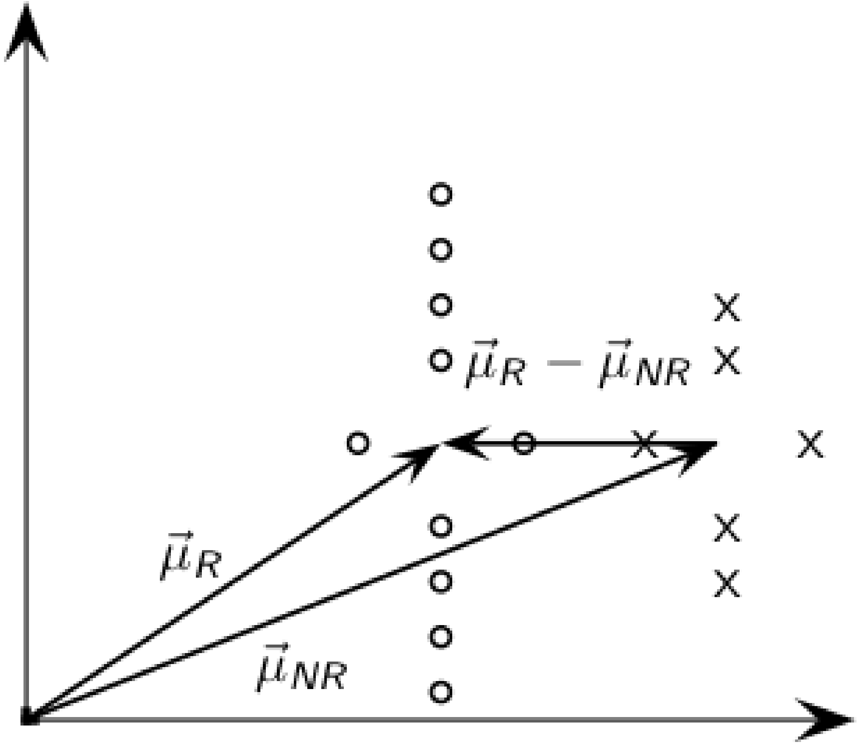
ROCCHIO' EXAMPLE

WARWICK

- We calculate:

  $\mu_{NR} - \mu_R$

  difference between the two centroids



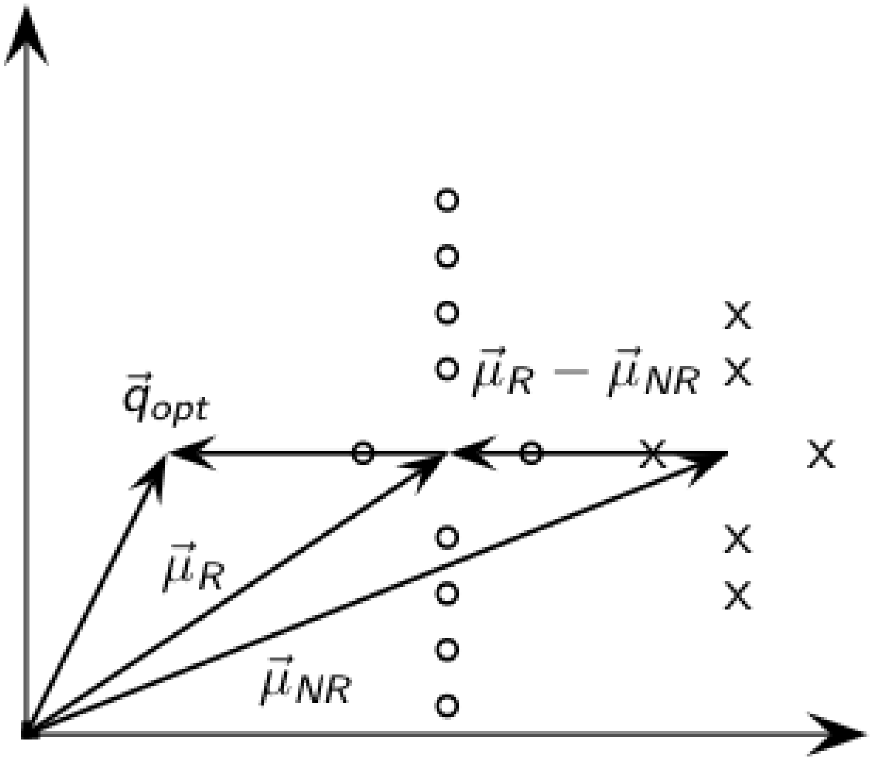
ROCCHIO' EXAMPLE

- Add the difference to μ_R.

- Which will give us q_{OPT}.

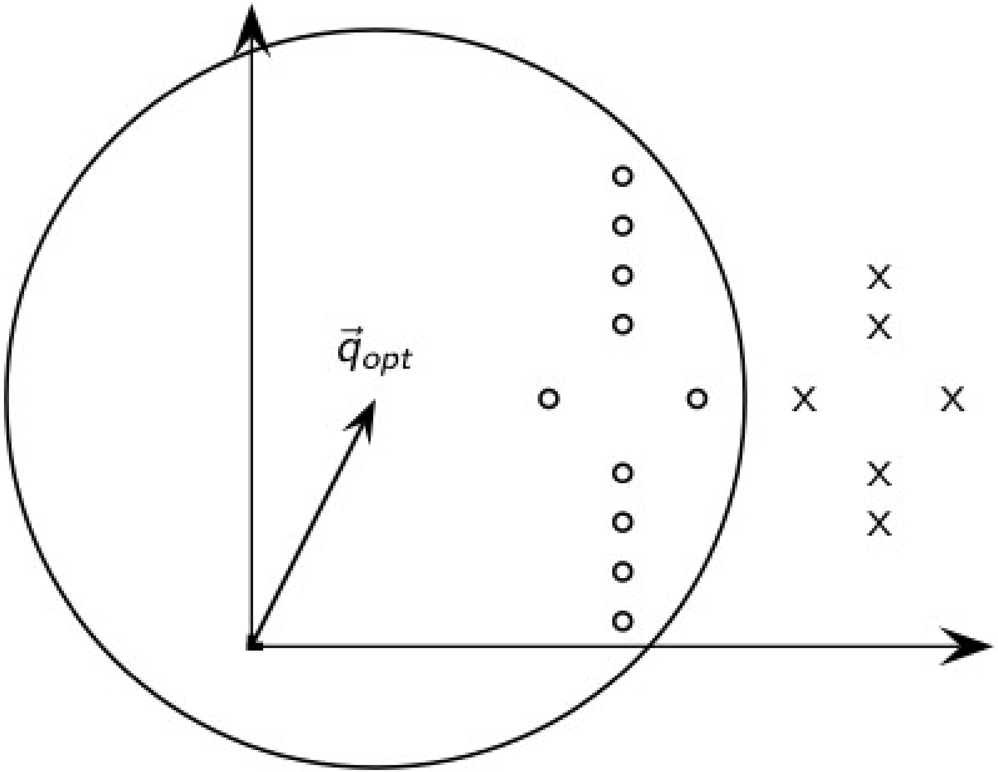
- Remember:

$$\vec{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_{nr})]$$



ROCCHIO' EXAMPLE

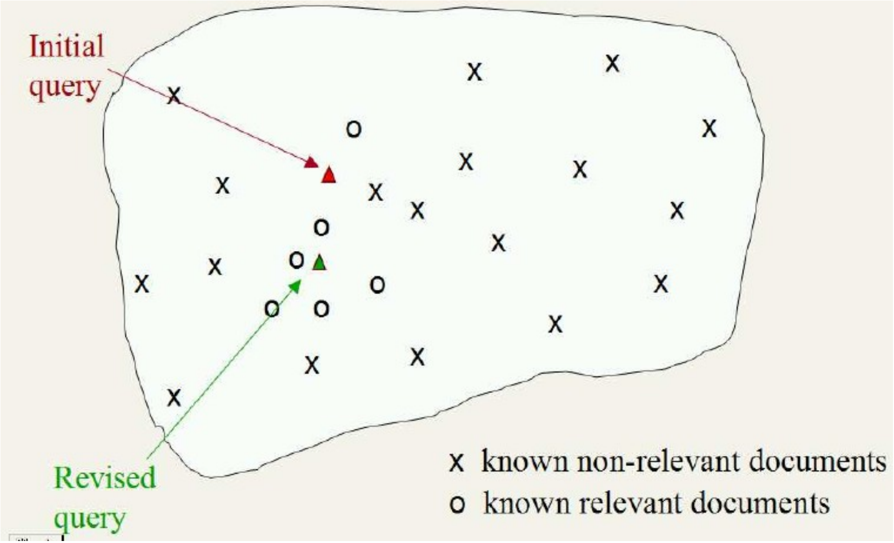
- q_{OPT} covers the desired area.



ROCCHIO' EXAMPLE

- A real example may not have so clearly separated rel vs non-rel docs.

  but we can still expect it to help.



ROCCHIO: ANOTHER ALTERNATIVE

- Another alternative of Rocchio’ (called Rocchio, no apostrophe):

$$\vec{q}_m = \alpha\vec{q}_0 + \beta\mu(D_r) - \gamma\mu(D_{nr})$$

  q_m: modified query
  q_0: initial query
  $\mu(D_R)$, $\mu(D_{NR})$: centroid of relevant and non-relevant docs
  α, β, γ: weights

- Weights can be adjusted based on relevance feedback, i.e. are there more rel (higher α, β) or non-rel docs (higher γ).

USE OF RELEVANCE FEEDBACK

- Relevance feedback **can be useful** in some search engines, but often **not used by search engines** (at least not visible to the end user)

- Why?
  - Do web users really **want to provide feedback**, or would they **rather revise their query**?
  - The **user doesn’t see the alterations** made to the query. Why have certain documents now been retrieved?

- But, of course, you can have **people in-house providing relevance feedback**. This can be used to provide improved results to end users.

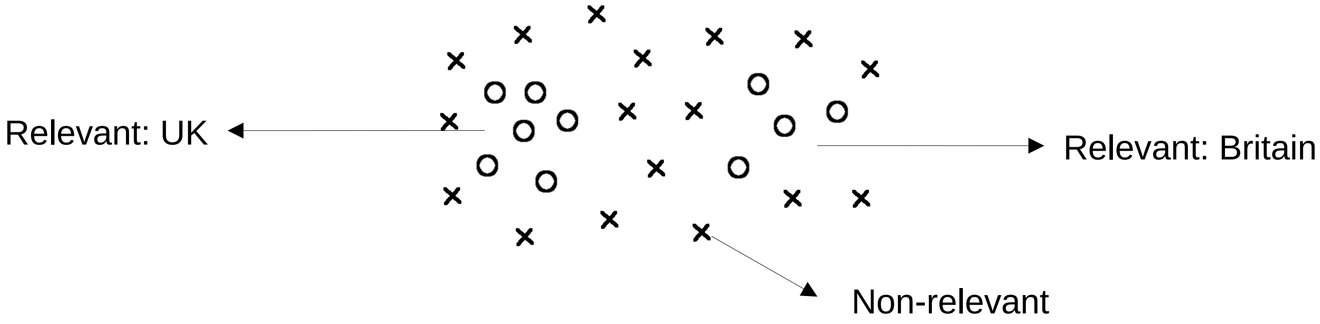
LIMITATIONS OF RELEVANCE FEEDBACK

- Relevance feedback will **fail to capture**:
  - **Misspellings**: to correct misspellings, we'll need to rewrite queries (e.g. query expansion), adjusting the query vector will not help.
  - **Cross-language information retrieval**: to search for docs in multiple language, we'll need machine translation, adjusting the query vector will not suffice.

ASSUMPTION FOR RF TO WORK

- We need **relevant docs to be similar to each other**, i.e. to form a single cluster.
- It will not work for cases with multiple clusters.

  e.g. a cluster for "UK", and another for "Britain".



PSEUDO RELEVANCE FEEDBACK

- Automating collection of relevance feedback.
- **Pseudo relevance feedback (PRF)**:
  - Retrieve top k docs for user's query.
  - **Assume top m are relevant** (for a small m).
  - Use Rocchio' algorithm using those m docs as the relevant set.

- It tends to **work well**, while **saving user's time**.
- **Several iterations** of PRF can lead to **query drift**.

INDIRECT RELEVANCE FEEDBACK

- Avoids asking the user for judgements.

- Instead, make assumptions, e.g.:
    - **user clicking on a search result indicates it is relevant** → this will help us improve results for future repetitions of the same query.
    - user **scrolling down on search results** means first few visible results were not enough.

INDIRECT RELEVANCE FEEDBACK

WARWICK

- **Pro:** easy to collect, no explicit user input needed.
- **Con:** no guarantee that scrolling down means top results weren't good enough.
- **Con:** user may find the web page wasn't relevant after clicking.

  Note: user clicks looking only at the snippet.

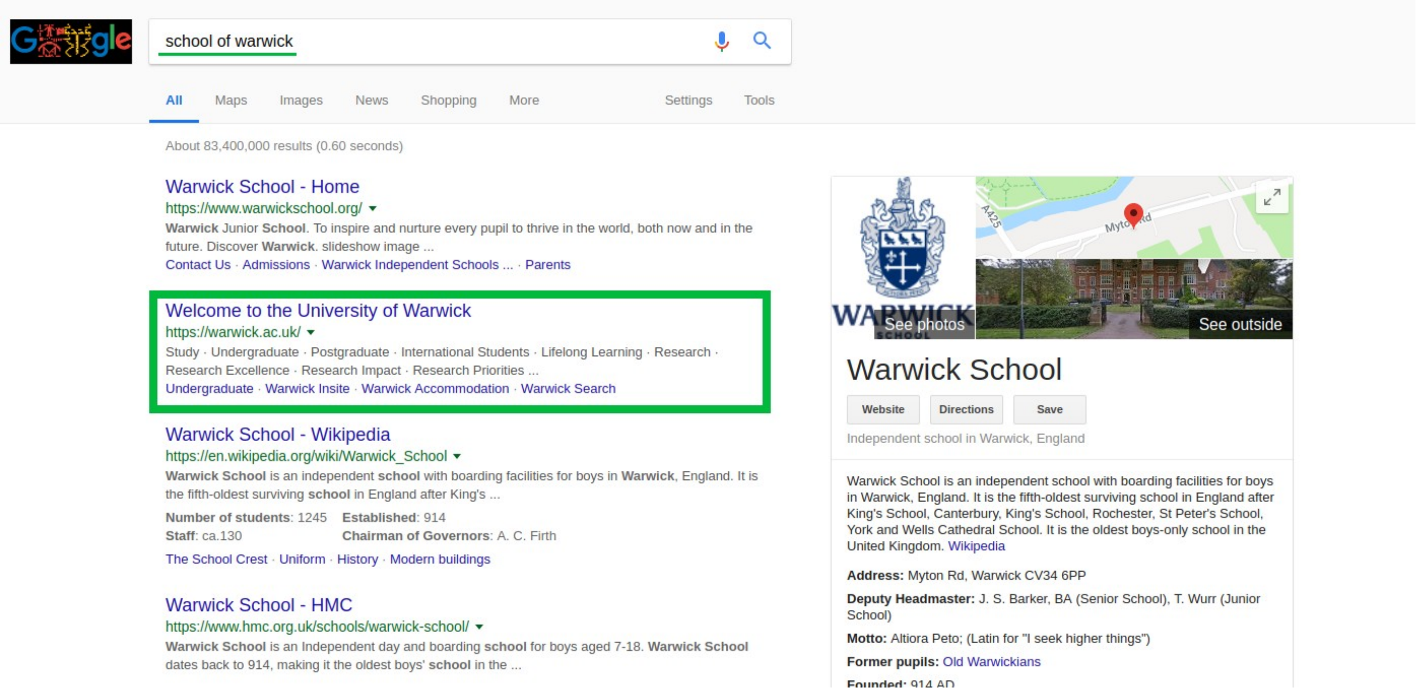
GLOBAL METHODS: QUERY EXPANSION

QUERY EXPANSION

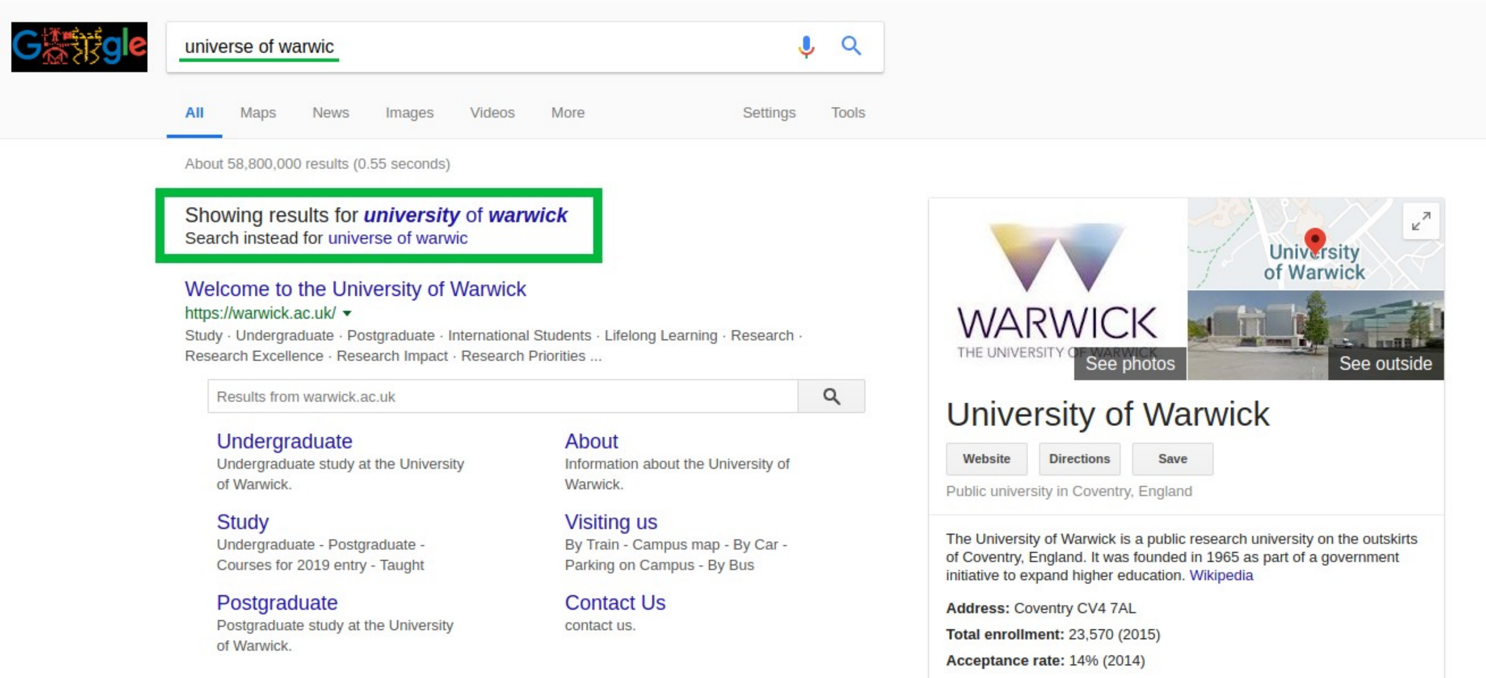
- With **relevance feedback**, we were looking at the **results** to **reformulate the query**.
  - e.g. move query away from non-relevant docs.

- For **query expansion**, we only **look at the query**, e.g.:
  - User enters: t_1 t_2
  - We consider query needs adding t_3 and t_4.
  - We will actually search for: t_1 t_2 t_3 t_4.

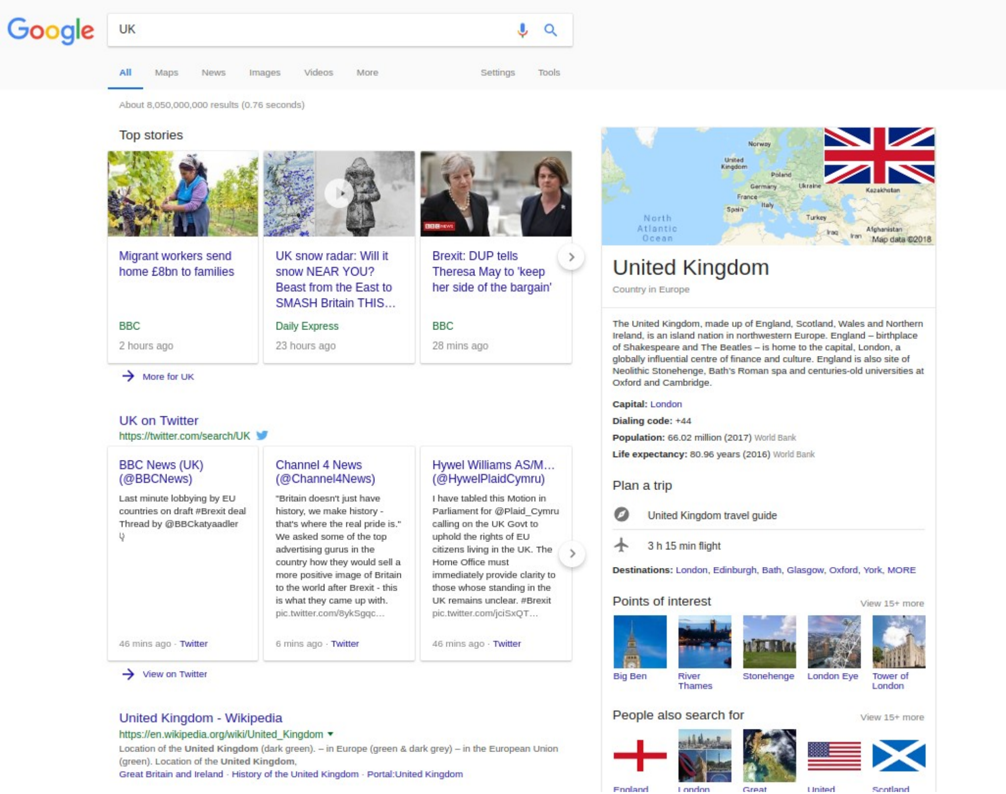
QUERY EXPANSION: EXAMPLE



QUERY EXPANSION: EXAMPLE



QUERY EXPANSION: EXAMPLE



TYPES OF QUERY EXPANSION

- Many possible options for query expansion, e.g.:
  - Acronyms: “UK” → “United Kingdom”
  - Misspellings: “warwik” → “warwick”
  - Synonyms: “connexion” → “connection”
  - Spelling variations: “realise” → “realize”
  - Translations: “Londra” → “London”

QUERY EXPANSION

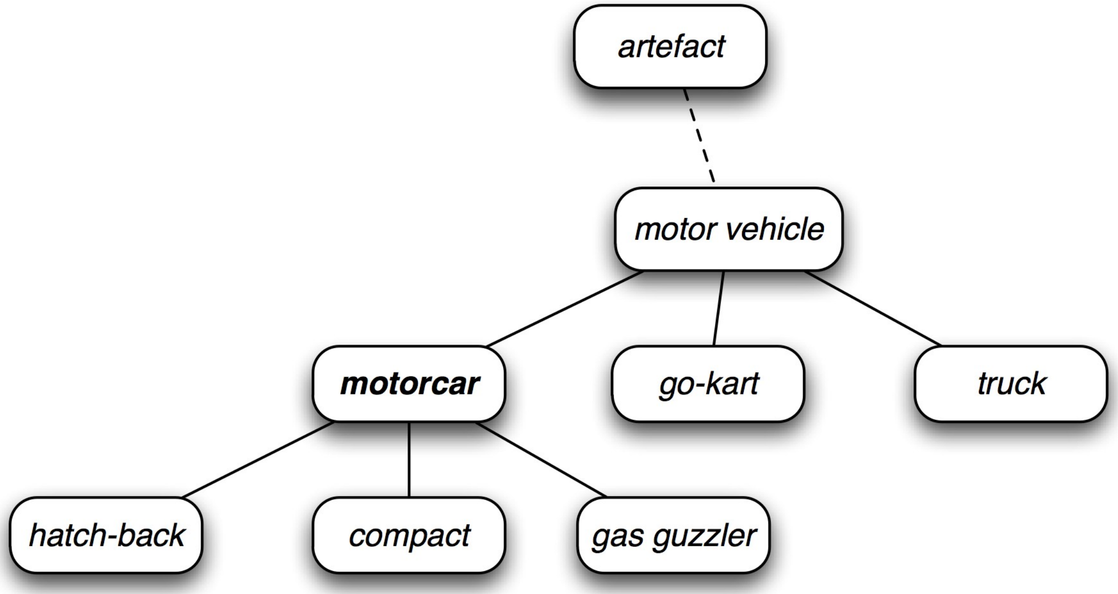
- Query expansion is usually performed using a **global resource**, e.g. **thesauri** or **list of common misspellings**.
  - This global resource is **not query-dependent**.

- How do we generate these global resources?

THESAURI AND OTHER RESOURCES

- Some types of resources:
  - Existing **thesauri** (e.g. WordNet).
  - Thesauri derived **from our collections**:
    - Use PPMI, word embeddings, etc.
  - Collect **user feedback**:
    - Store **user's query log**, find when they **retype queries**.
    - When we said "**did you mean Warwick**", did they click on it?

THESAURI



- Use hypernyms.
- User issues query “truck repair”.
- If we have few relevant results, give them results for “vehicle repair” instead?

WORDNET SYNSETS

- dog = domestic dog = canis familiaris

- We can use **WordNet synset to expand the query with synonyms**, in different cases:
  - Original query (dog) returns too many results, use more specialised word (canis familiaris).
  - Original query (canis familiaris) returns too few results, use more widely used words (dog).

WORDNET: ANTONYMS

WARWICK

- Use **antonyms from WordNet to exclude search results**.
- Particularly useful when original query returns too many results.
- e.g. user issues query “short stay car park”.
  - exclude everything having “long stay car park”, despite having significant overlap of ¾ keywords.

QUERY LOG MINING

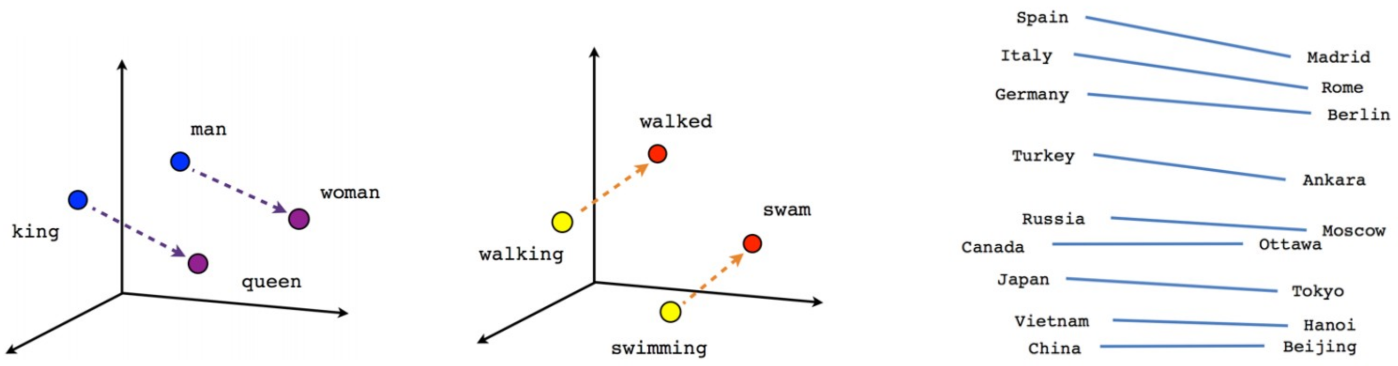
- Store queries + clicks of users.
- For instance, for users searching for “warwic”, we’ll give them a link: “did you mean Warwick?”.
- If many users click on it, we’re confident it needs to be corrected.
- For future users, directly show results for “Warwick” instead when they search for “warwic”.

USER-SPECIFIC QUERY EXPANSION

- You can do many more, user-specific expansions using query logs.
- Say I've searched for "Coventry hotels" and for "Birmingham hotels".
- Next, if I type "London", I'll likely be looking for "London hotels"?

WORD EMBEDDINGS

- Expand by using similar/related words.



Male-Female Verb tense Country-Capital

QUERY EXPANSION: CAVEAT

- We need to be careful with the expansions we make.

- For instance, for a query:

  apple computer

- I may end up expanding it to:

  apple **fruit** computer

QUERY EXPANSION: WORD2VEC

WARWICK

With Word2Vec, use as many words as possible to find similar words.

model.most_similar(positive=[‘apple’, ‘computer’])

rather than

model.most_similar(positive=[‘apple’])

or even better if we’re able to determine that the user is not interested in fruits:

model.most_similar(positive=[‘apple’, ‘computer’], negative=[‘fruit’])

WARWICK

SUMMARY

- **Local methods:** (pseudo-/indirect) relevance feedback.
  - Tends to be effective.
  - Can be cumbersome for the user.
  - Better to have people in-house providing feedback.

- **Global methods:** query expansion.
  - Widely used.
  - Provided that queries are short, lacking specificity, expanding helps.
  - We need to be careful not to incorporate wrong keywords (apple fruit vs computer).

ASSOCIATED READING

- Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval (Vol. 1, No. 1, p. 496). Cambridge: Cambridge university press. **Chapter 9**. https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf