Introducing Ranked Retrieval

Boolean Retrieval

- Thus far, our queries have all been Boolean
  - Documents either match or don't
- Good for expert users with precise understanding of their needs and the collection
  - Also good for applications : applications can easily consume 1000s of results
- **Not good for the majority of users**
  - Most users incapable of writing Boolean queries (or they are, but they think it's too much work)
  - Most users don't want to wade through 1000s of results
    - This is particularly true of web search

Problem with Boolean Search : Feast or Famine

- Boolean queries often result in either **too few** (≈ 0) or **too many** (1000s) results
- example
  - Query 1 : "stanford user dlink 650" → 200,000 hits
  - Query 2 : "stanford user dlink 650 no card found" → 0 hit
- It takes skill to come up with a query that produces a manageable number of hits
  - AND gives too few; OR gives too many

Ranked Retrieval Models

: Boolean Retrieval의 단점 보완

- **Ranked Retrieval model** returns an **ordering** over the (top) documents in the collection with respect to a query
- **Free text queries** : Rather than a query language of operators and expressions (and, or, not...), the user's query is just **one or more words** in a human language

Feast or Famine : not exist in Ranked Retrieval

- When a system produces a ranked result set, large result sets are not a issue
  - Indeed, the size of the result set is not an issue
  - we just show the top k results
  - we don't overwhelm the user

Scoring as the basis of Ranked Retrieval

- We wish to return in order the documents most likely to be **useful** to the searcher

- How can we rank-order the documents in the collection with respect to a query?
  - Assign a **score** - say in [0, 1] - to each document
- This score measures how well document and query **"match"**
  - 즉, "query와 document의 유사도"

Query-Document Matching Scores

- we need a way of assigning a score to a query/document pair
- Let's start with a one-term query
  - If the query term docs not occur in the document : score should be 0
  - **The more frequent** the query term in the document, **the higher the score** should be

정리

- Boolean Retrieval : 사용하기도 어렵고, 시스템 결과로 나오는 document의 수가 너무 많거나 너무 적다.
- 이러한 불편성을 해소하는 Ranked Retrieval
  - Free text queries에 적합
  - 사용자가 원하는 만큼의 문서를 보면 되어, 시스템 결과로 나오는 document의 수에 따라 만족도가 달라지지 않음

출처 : stanford IR 강의(https://www.youtube.com/watch?v=ZrNmCtSrL48&list=PLaZQkZp6WhWwoDuD6pQCmgVyDbUWl_ZUi&index=7)