BM25 is so Yesterday

Modern Techniques for Better Search Relevance in Solr

Grant Ingersoll
CTO Lucidworks
Lucene/Solr/Mahout Committer
😊 iPAD CASE
iPad case

"ipad accessory"~3 OR "ipad case"~5
1. 
{
 "shortDescription": "Designed for Apple® iPad® 2; polyurethane construction; converts to a stand; Cleveland Browns design",
 "id": "2789054",
 "name": "Tribeca - Cleveland Browns Folio Case for Apple® iPad® 2",
}

15. 
{
 "shortDescription": "Compatible with Apple® iPad® 2; ABS material; slim, lightweight design; team design",
 "id": "2789946",
 "name": "Tribeca - Minnesota Vikings Hard Shell Case for Apple® iPad® 2"
}
So, what do you do?
if (doc.name.contains("Vikings")){
    doc.boost = 100
}

OR

q:(MAIN QUERY) OR (name:Vikings)^Y
TF*IDF

• Term Frequency: “How well a term describes a document?”
  • Measure: how often a term occurs per document

• Inverse Document Frequency: “How important is a term overall?”
  • Measure: how rare the term is across all documents
BM25 (aka Okapi)

Score(q, d) =
\[ \sum_{t \in q} \text{idf}(t) \cdot (\text{tf}(t \text{ in } d) \cdot (k + 1)) / (\text{tf}(t \text{ in } d) + k \cdot (1 - b + b \cdot |d| / \text{avgdl})) \]

Where:
- t = term; d = document; q = query; i = index
- \text{tf}(t \text{ in } d) = \text{numTermOccurrencesInDocument}^{\frac{1}{2}}
- \text{idf}(t) = 1 + \log(\text{numDocs} / (\text{docFreq} + 1))
- |d| = \sum_{t \in d} 1
- \text{avgdl} = (\sum |d|) / (\sum 1)

k = Free parameter. Usually ~1.2 to 2.0. Increases term frequency saturation point.
b = Free parameter. Usually ~0.75. Increases impact of document normalization.
Lather, Rinse, Repeat
WWGD?
Measure, Measure, Measure

• Capture and log pretty much everything
  • Searches, Time on page/1st click, What was not chosen, etc.
• Precision — Of those shown, what’s relevant?
• Recall — Of all that’s relevant, what was found?
• NDCG — Account for position
Magic

Guessing

Core Information Theory (aka Lucene/Solr)

Search Aids (Facets, Did You Mean, Highlighting)

Machine Learning (Clicks, Recs, Personalization, User feedback)

Rules, Domain Specific Knowledge

fuhgeddaboudit
Core Solr capabilities: text matching, faceting, spell checking, highlighting

Business Rules for content: landing pages, boost/block, promotions, etc.

Leverage collective intelligence to predict what users will do based on historical, aggregated data

Recommenders, Popularity, Search Paths

Who are you? Where are you? What have you done previously?

User/Market Segmentation, Roles, Security, Personalization
But What About the Real World? Indexing Edition

Extraction

NER, Topic Detection, Clustering Word2Vec, etc.

Domain Rules: Synonyms, Regexes, Lexical Resources

Solr

Content

Solr

Models

Build W2V, PageRank, Topic, Clustering Models

Load Into Spark

Offline
But What About the Real World? Query Edition

- iPad case
- Parse
- Query Intent: Strategic, Tactical, Semantic
- Head/Tail/Clickstream enhancement
- User Factors: Segmentation, Location, History, Profile, Security
- Transform Results
- Cascading Rerankers: Learn To Rank (multi-model), Bias corrections
- Domain Specific Rules
But What About the Real World? Signals Edition

Signals → Solr Raw → Query Edition → Solr Models → Recommenders/Persomalization → Load Into Spark → Clickstream Models → Query Analysis Jobs

iPad case
The Perfect(?!?) Query*  YMMV!

(Exact/Original Match)$^X$
(Sloppy Phrase)$^M$
(AND Q)$^Z$
(OR Q)$^{XX}$
(Expansions/Click/Head/Tail Boosts)$^{YY}$
(Personalization Biases)$^{ZZ}$
({!ltr ...})

Filters+Options: security, rules, hard preferences, categories

* Note: there are a lot of variations on this. edismax handles most
Experimentation, Not Editorialization

• Don’t take my word for it, experiment!
• Good primer:
  • http://www.slideshare.net/InfoQ/online-controlled-experiments-introduction-insights-scaling-and-humbling-statistics
• Rules are fine, as long as the are contained, have a lifespan and are measured for effectiveness
Show Us Already, Will You!
Fusion Architecture

- **Core Services**
  - ETL and Query Pipelines
  - Recommenders/Signals/Rules
  - NLP
  - Machine Learning
  - Scheduling
  - Alerting and Messaging
  - Security
  - Connectors

- **Rest API**
  - Twigkit
  - Admin UI

- **Apache Solr**
  - Shards
  - Leader Election
  - Load Balancing
  - Shared Config Management

- **Apache Spark**
  - Worker
  - Cluster Manager

- **Apache Zookeeper**
  - ZK 1
  - ZK N

- **Connectors**

- **Log Management**
  - Logs
  - File
  - Web
  - Database
  - Cloud
Key Features

• Solr:
  • Extensive Text Ranking Features
    • Similarity Models
    • Function Queries
    • Boost/Block
  • Pluggable Reranker
  • Learn to Rank contrib
  • Multi-tenant

• Spark
  • SparkML (Random Forests, Regression, etc.)
  • Large scale, distributed compute
Demo Details

- Best Buy Kaggle Competition Data Set
  - Product Catalog: ~1.3M
  - Signals: 1 month of query, document logs
- Fusion 3.1 Preview + Recommenders (sampled dataset) + Rules (open source add-on module) + Solr LTR contrib
- Twigkit UI ([http://twigkit.com](http://twigkit.com))
Resources

- http://lucidworks.com
- http://lucene.apache.org/solr
- http://spark.apache.org/
- https://github.com/lucidworks/spark-solr
- https://cwiki.apache.org/confluence/display/solr/Learning+To+Rank
- Bloomberg talk on LTR https://www.youtube.com/watch?v=M7BKwJoh96s