Query Categorization at Scale
About Magnetic

First company to focus 100% on applying search intent to display

Proprietary media platform and targeting algorithm

One of the largest aggregators of intent data

Display, mobile, video and social retargeting capabilities

Strong solution for customer acquisition and retention
Search Retargeting

Search retargeting combines the purchase intent from search with the scale from display.

1) Magnetic collects search data
2) Magnetic builds audience segments
3) Magnetic serves retargeted ads
4) Magnetic optimizes campaign
Natural Searches and Navigational Searches

Navigational Search: “iPad Accessories”

Natural Search: “iPhone”
Page keywords from article metadata: “Recipes, Cooking, Holiday Recipes”
Article Titles: “Microsoft is said to be in talks to acquire Minecraft”
Search Data – Why Categorize?

- Targeting categories instead of keywords = Scale
- Use category name to optimize advertising as an additional feature in predictive models
- Reporting by category is easier to grasp as compared to reporting by keyword
Query Categorization Problem

- Input: Query
- Output: Classification into a predefined taxonomy
Query Categorization – Academic Approach

• Usual approach (academic publications):
  – Get documents from a web search
  – Classify based on the retrieved documents
Query Categorization

- Usual approach:
  - Get results for query
  - Categorize returned documents
  - Best algorithms work with the entire web (search API)

<table>
<thead>
<tr>
<th>Query</th>
<th>Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>apple</td>
<td>Computers \ Hardware \ Living \ Food &amp; Cooking</td>
</tr>
<tr>
<td>FIFA 2006</td>
<td>Sports \ Soccer \ Sports \ Schedules &amp; Tickets \ Entertainment \ Games &amp; Toys</td>
</tr>
<tr>
<td>cheesecake recipes</td>
<td>Living \ Food &amp; Cooking \ Information \ Arts &amp; Humanities</td>
</tr>
<tr>
<td>friendships poem</td>
<td>Information \ Arts &amp; Humanities \ Living \ Dating &amp; Relationships</td>
</tr>
</tbody>
</table>
• Relying on Bing Search API:
  – Get search results using the query we want to categorize
  – See if some category-specific “characteristic” keywords appear in the results
  – Combine scores
  – Not too bad....
Long Time Ago …

• … But ….

• .... We have ~8Bn queries per month to categorize ....

• $2,000 * 8,000 = Oh My!

A transaction is one request that returns one page of results. Retrieving multiple pages will result in multiple transactions executed.
Our Query Categorization Approach

- Use web replacement – Wikipedia
  - 4.5 million articles
  - 9 million of unique titles
  - 40 GB text
- DBPedia
  - Good categories for articles
  - Additional structured data
- Freebase
  - 170 GB triples
  - 40 million topics

All available for download!
Our Query Categorization Approach

- Assign a category to each Wikipedia document (with a score)
- Load all documents and scores into an index
- Search within the index
- Compute the final score for the query
Results Quality - Precision/Recall

![Chart showing precision and recall for different categories such as Automotive, Electronics & Computing, Health, Style & Fashion, World Localities, Food & Drink, Sports/Precision Sports, Finance, Home & Garden. Each category has bars indicating Recall%, Precision%, and F1%.](image)
Categorizing Other Languages

- In development
- Combining indexes for multiple languages into one common index
- Focus:
  - Spanish
  - French
  - German
  - Portuguese
  - Dutch
Preprocessing Workflow

• Automatized Hadoop and local jobs
• Luigi library and scheduler
• Steps:
  – Download
  – Uncompressing
  – Parse Wikipedia/Freebase/DBpedia
  – Generate N-grams
  – JOIN together (Wikipage) – JSON per page
  – Preprocess Wikipage categories
  – Produce JSON for Solr or local index
  – Load to SOLR + Check quality
Query Categorization: Scale

- Scale is achieved by combination of multiple categorization boxes, load balancing, and Varnish (open source) cache layer in front of Solr
- We have 6 servers in production today
- Load Balancer - HAProxy
- Capacity – 1,000 QPS/server
- More servers can be added if needed
Architected for Scale

- Bidders, AdServers developed in Python and use PyPy VM with JIT
- Response time critical - typically under 100ms as measured by exchange
- High volume of auctions – 200,000 QPS at peak
- Hadoop – 25 nodes cluster
- 3 DC – US East, West and London
- Data centers have multiple load balancers – HAProxy
- Overview of servers in production:
  - US East: 6LB, 45 Bidders, 6 AdServers, 4 trackers, 25 Hadoop, 9 Hbase, 8 Kyoto DB
  - US West: 3LB, 17 Bidders, 6 AdServers, 4 trackers, 4 Kyoto DB
  - London: 8 Bidders, 2 AdServers, 2 trackers, 4 Kyoto DB
ERD Challenge

- ERD'14: Entity Recognition and Disambiguation Challenge
- Organized as a workshop at SIGIR 2014 Gold Coast
- Goal: Submit working systems that identify the entities mentioned in text
- We participated in the “Short Text” track
  - Entities (people, locations, organizations …) in queries
- 19 team participated in the challenge
- We took 4th place
Ďakujeme za pozornosť