Pre-computing Search Features for Fast and Accurate Query Classification

Arnd Christian König
Venkatesh Ganti
Xiao Li

Microsoft Research
Query Classification important for

- Matching advertisements for a query
- Retrieval of additional non-web search results from verticals
- Load optimization for search verticals
Problem Statement

Query Classification:
- **Problem**: ~3 words in queries => little ‘signal’ for classification.
- Large vocabulary size => large, sparse feature space.
- Difficult to generalize across queries.

Post-Retrieval Features:
- Use search to obtain more context to derive features.
General approach:

- Issue the search query against a document corpus.
- Identify relevant sub-components of top results (e.g., titles, captions, key terms, etc.)
- Derive additional features from these components.

\[ F = \sum_{d \in \text{Result}} (...) \]
**Problem Statement**

**Query Classification:**
- **Problem:** ~3 words in queries => little ‘signal’ for classification.
- Large vocabulary size => large, sparse feature space.
- Difficult to generalize across queries.

**Post-Retrieval Features:**
- Use search to obtain more context to derive features.
- => significant improvements in classification accuracy.
- **Problem:** Search Latency
  - Even slight (100 ms) increases in latency decrease user satisfaction, increase in fraction of abandoned searches.

⇒ **Task:** Realize benefits of post-retrieval features at low overhead.
Features based on the incidence of tags in the documents returned in response to a query.

⇒ Small feature space, features generalize across queries.
⇒ Less information to store, helping pre-computation.

**Other examples:**

**Corpus:** Sponsored Search Bids
- **Tags:** Advertiser-IDs
- Advertisers can be thought of as ‘topics’

**Corpus:** Wikipedia
- **Tags:** Wikipedia-Category Tags
Documents $\mathcal{E}$  
Tag Corpus $\mathcal{I}$

**Pre-computation of Tag-ratios**

- Collection of $(query, tag-ratio)$ pairs

**Offline**

Retrieval Semantics: word-containment
- Search engine not involved in retrieval
  $\Rightarrow$ Fast pre-computation of query sets
- Tradeoff: result relevance vs. result size

- Tag ratios are pre-computed and indexed in memory

**Online**

**Query Classifier**

**Feature Generation**

The rest of this talk:
- How do we generate features from the ratios?
- Size-constraints: for which queries do we pre-compute ratios?
- How do we deal with query that is not pre-computed?
Creating Features from Tag Ratios

Features = Ratios?

\[ F = [\text{ratio}(q, t_1), \cdots, \text{ratio}(q, t_k)] \]

- **Problem 1: Small result sizes**

---

**Overall Accuracy**

- **Corpus used for Training / Testing**

<table>
<thead>
<tr>
<th>Overall Accuracy</th>
<th>Corpus used for Training / Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>76%</td>
<td>All Items</td>
</tr>
<tr>
<td>82%</td>
<td>Result-size &gt; 0</td>
</tr>
<tr>
<td>84%</td>
<td>Result-size &gt; 10</td>
</tr>
<tr>
<td>86%</td>
<td>Result-size &gt; 100</td>
</tr>
<tr>
<td>88%</td>
<td>Result-size &gt; 200</td>
</tr>
<tr>
<td>90%</td>
<td>Result-size &gt; 500</td>
</tr>
</tbody>
</table>
Creating Features from Tag Ratios

Features = Ratios?

\[ F = [\text{ratio}(q, t_1), \ldots, \text{ratio}(q, t_k)] \]

Overall Accuracy

<table>
<thead>
<tr>
<th>Ratios used for Training / Testing</th>
<th>Overall Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>result(q) only</td>
<td>81.77%</td>
</tr>
<tr>
<td>result(q'),</td>
<td>q'</td>
</tr>
<tr>
<td>result(q'),</td>
<td>q'</td>
</tr>
<tr>
<td>result(q'),</td>
<td>q'</td>
</tr>
<tr>
<td>result(q'),</td>
<td>q'</td>
</tr>
</tbody>
</table>
Creating Features from Tag Ratios

Query Q

Pre-computed Tag Ratios

Q_1 \rightarrow G_1 \rightarrow \text{Features}(G_1)

Q_2 \rightarrow G_2 \rightarrow \text{Features}(G_2)

Q_3 \rightarrow G_3 \rightarrow \text{Features}(G_3)

Q_i \subseteq Q

With Q_i \subseteq Q

Group by similarity to Q

Features based on Aggregates over ratios in a group, such as SUM, AVG, STDIV, MAX, MIN, etc…
Creating Features from Tag Ratios

Pre-computed Tag Ratios

Query Q

{Camera} {Canon} {SD 2}

Q₁

{Canon, Camera}

Q₂

{Canon, Camera, SD2}

Q₃

Features(G₁)

Features(G₂)

Features(G₃)

G₁

G₂

G₃

Group by similarity to Q

With Qᵢ ⊆ Q

Features based on Aggregates over ratios in a group, such as

SUM, AVG, STDIV, MAX, MIN, etc…
Selecting queries to precompute

- $|V| > 10^7 \Rightarrow$ intractable # of keyword combinations to pre-compute

**Pruning Logic**

- **Short queries:** limit query-length to $w_{\text{max}}$ words.
- **Significant correlation:**
  
  $$ratio(q,t) \geq \Theta_{\text{high}} \frac{|\text{#tags } t \in D|}{D} \quad ratio(q,t) \leq \Theta_{\text{low}} \frac{|\text{#tags } t \in D|}{D}$$

- **Ratio-support:**
  
  $$|\text{result}_C(q)| \geq \alpha$$

Very few keyword (combinations) satisfy *Correlation Condition*

No keyword (combination) can satisfy *Support Condition*
Experimental Evaluation

Task I: Indentifying ‘Consumer-Electronics’ queries
  - C = Wikipedia, T = Entity-Categories (contained in pages)
  - **Accuracy:** 93.0% (n-grams only)
    - 93.2% (n-grams + Brand/Models/Product Type/Product Attribute lexicons)
    - 95.6% (Tag ratios only)
    - 96.5% (Tag ratios + n-grams)

Task II: Indentifying ‘Retail’ queries
  - C = Wikipedia, T = Top Wikipedia Categories (contained in pages)
  - + C = Sponsored Search Bids, T = Advertiser IDs (top advertisers)
  - Large training corpus (~330K labeled examples)
  - **Accuracy:** 92.5% (n-grams only) => 93.3% (Tag Ratios + ngrams)

Task III: Indentifying ‘Health’-related queries
  - Same corpora/tags as before
  - Very large training corpus (~800K labeled examples)
  - **Accuracy:** 98.2% (n-grams only) => 98.8% (Tag Ratios + ngrams)
Experimental Evaluation: Generalization

Overall Accuracy

- Tag Ratio Features
- N-Gram Features

100% Training Data: 95.60%
33% Training Data: 94.30%
10% Training Data: 93.00%
3.33% Training Data: 91.90%
1% Training Data: 87.20%

- 70% Training
- 75%
- 80%
- 85%
- 90%
- 95%
Experimental Evaluation: Query Selection

Using earlier classification tasks, we evaluate features based on:

- Single-Word queries only
- Single-word queries + selected query/ratio combinations
- All queries in training/test data + all subsets

Results:

- Pruning results in very large reduction in space of ratios to store ($\Theta_{\text{low}} = 0.8$, $\Theta_{\text{high}} = 1.2$ => 0.8% of ratios (for frequent keywords) remain).
- Differences in classification accuracy slight: (0.17% or less)
Many thanks!

Any Questions?