Online Expansion of Rare Queries for Sponsored Search

Peter Ciccolo, Evgeniy Gabrilovich, Vanja Josifovski, Don Metzler, Lance Riedel, Jeff Yuan

Yahoo! Research
Sponsored Search in the Tail

• Many tail queries display no sponsored search results

• Advertising in the tail is challenging
  – Longer / rare queries are more difficult to interpret
  – Exact match and phrase match are much less likely
  – Click-based relevance predictors are poor, due to data sparseness

• Considerable monetization potential in tail, if done correctly
Sponsored Search in the Tail

**Smoked-Sausage, Cabbage, and Potato Soup Recipe | Food & Wine**
... for Smoked-Sausage, Cabbage, and Potato Soup. Dishes created by Food ... Smoked-Sausage, Cabbage, and Potato Soup. Recipe by Quick From Scratch Soups & Salads ...
foodandwine.com/recipes/smoked-sausage-cabbage-and-potato-soup?...  

**Cabbage-and-White-Bean Soup with Prosciutto Recipe | Food & Wine**
A recipe for Cabbage-and-White-Bean Soup with Prosciutto. Dishes ... Cabbage-and-White-Bean Soup with Prosciutto. Recipe by Quick From Scratch Soups & Salads ...
foodandwine.com/recipes/cabbage-and-white-bean-soup-with-prosciutto

**Savory Soup Recipes for Homecooks**
Cabbage, Chicken, Dumpling, Onion, Potato, Tomato and Vegetable Soup ... Yes, you can cook soup from scratch with the right recipes for homemade soup. ...
www.soulsoufflesoutherncooking.com/soup-recipes.html - Cached

**Soup Recipes like Chicken Soup, Potato Soup or Cabbage Soup Recipes**
Recipes for soup with photos and reviews. Recipes like Bean Soup, Tomato Soup, ... a can and adding a few extra ingredients or you can create soup from scratch. ...
www.cdkitchen.com/recipes/cat/20

**Cabbage & Dumpling Soup Recipe - YumYum.com**
View the free recipe for Cabbage & Dumpling Soup ... Cabbage & Dumpling Soup Instructions: Dumplings: Blend tofu with water till smooth. ...
www.yumyum.com/recipe.htm?ID=227 - Cached
Advanced Match Scoring

- Advanced match (or broad match) refers to an inexact match between the query and an ad’s bid phrase
  - Advertisers opt in
  - Tend to have lower click-through rates

- Matching queries to bid phrases is challenging
  - Sparse representation (just a few terms)
  - Vocabulary mismatch
  - Misspellings

- Query and/or bid phrase expansion is often used to overcome these issues
Offline Query Expansion

• Query expansion is generally expensive to perform in real time

• It is possible to compute expansions for popular queries offline and cache the results

• Query expansion sources
  – Query reformulations (query logs, co-clicks on search results and ads, etc.)
  – Top features from web search results (next slide)

• Circumstantial evidence can convince better than one witness!

• Expanded query is a combination of the original query and additional information from expansion sources (more details later)
Query Expansion using Search Results

- Query expansion using web search results has been shown to be effective for sponsored search in the past [Broder et al., SIGIR ’07, Radlinski et al., SIGIR ‘08]

- General overview:
  - Run query against web search engine
  - Retrieve the top $N$ pages (we use 40)
  - Extract features and assign weights ($TF.IDF$)
  - Select (about 100) features by sorting features by weight
  - End result is a query -&gt; feature vector

- Helps provide uniformity of the search engine result page!
What about the Tail?

- Expansion is effective, but only if the incoming query is in the expansion cache.

- Can we leverage the offline expansion cache to improve ad matching for queries that are not in the cache (rare queries)?
  
  ... and do it in real time?

- Our proposed methodology
  
  - ‘Map’ incoming rare query to related popular queries
  
  - Expand using offline expansion cache
  
  - Use expanded version of rare query for ad matching
System Architecture

Diagram showing the system architecture with components such as Query Feature Hash Table, Ad Selection, and Query Feature Indexer connected by arrows indicating flow and relationships.
Online Query Expansion

• Map rare queries to head/torso queries that have been processed offline via similarity search

• Build inverted index of queries that have been pre-expanded
  – Document retrieval analogy: query ≈ docid, expansion vector ≈ document text
  – Queries run against index will match against the expanded versions of the pre-expanded queries

• If an incoming query was not pre-processed, it is run against the inverted index of pre-expanded query vectors
  • Retrieve $k$ most related head/torso queries
  • Construct expanded query vector
  • This query can be executed very efficiently, adding little overhead
### Online Query Expansion Example

**cabbage soup from scratch**

<table>
<thead>
<tr>
<th>feature</th>
<th>inverted list</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabbage</td>
<td>cabbage (10), cabbage soup (8), cabbage soup diet (5), leafy vegetable (3), vegetable (2)</td>
</tr>
<tr>
<td>soup</td>
<td>soup (10), homemade soup (8), recipes (7), tomato soup (5), cabbage soup (2), food (2)</td>
</tr>
<tr>
<td>“cabbage soup”</td>
<td>cabbage soup (10), cabbage soup diet (8), cabbage (7), diet (5), weight loss (3)</td>
</tr>
<tr>
<td>scratch</td>
<td>scratch (10), scratchy eyes (3), cat scratch (3)</td>
</tr>
<tr>
<td>“from scratch”</td>
<td>recipes (10), cooking (7), food (5)</td>
</tr>
</tbody>
</table>
## Online Query Expansion Example

**cabbage soup from scratch**

<table>
<thead>
<tr>
<th>feature</th>
<th>inverted list</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabbage</td>
<td>cabbage (10), cabbage soup (8), cabbage soup diet (5), leafy vegetable (3), vegetable (2)</td>
</tr>
<tr>
<td>soup</td>
<td>soup (10), homemade soup (8), recipes (7), tomato soup (5), cabbage soup (2), food (2)</td>
</tr>
<tr>
<td>“cabbage soup”</td>
<td>cabbage soup (10), cabbage soup diet (8), cabbage (7), diet (5), weight loss (3)</td>
</tr>
<tr>
<td>scratch</td>
<td>scratch (10), scratchy eyes (3), cat scratch (3)</td>
</tr>
<tr>
<td>“from scratch”</td>
<td>recipes (10), cooking (7), food (5)</td>
</tr>
</tbody>
</table>

1. cabbage soup
2. cabbage soup diet
3. recipes
4. food
Online Query Expansion Example

cabbage soup from scratch

<table>
<thead>
<tr>
<th>feature</th>
<th>inverted list</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabbage</td>
<td>cabbage (10), cabbage soup (8), cabbage soup diet (5), leafy vegetable (3), vegetable (2)</td>
</tr>
<tr>
<td>soup</td>
<td>soup (10), homemade soup (8), recipes (7), tomato soup (5), cabbage soup (2), food (2)</td>
</tr>
<tr>
<td>“cabbage soup”</td>
<td>cabbage soup (10), cabbage soup diet (8), cabbage (7), diet (5), weight loss (3)</td>
</tr>
<tr>
<td>scratch</td>
<td>scratch (10), scratchy eyes (3), cat scratch (3)</td>
</tr>
<tr>
<td>“from scratch”</td>
<td>recipes (10), cooking (7), food (5)</td>
</tr>
</tbody>
</table>

1. cabbage soup
2. cabbage soup diet
3. recipes
4. food

Used to expand original query
Query Feature Weighting

Original query weights:

\[ w(f, Q) = (1 + \log \#(f, Q)) \cdot \text{idf}(f) \]

Offline expansion query feature weights:

\[ w(f, E(Q)) = \left(1 + \log \sum_{D \in \text{Results}(Q)} \#(f, D)\right) \cdot \text{idf}(f) \]

Similarity between the query vectors:

\[ \text{sim}(Q, E(Q')) = \sum_{F \in \{F_u, F_p, F_c\}} \lambda_F \cdot \text{sim}_F(Q, E(Q')) \]

Final query weights:

\[ w(f, Q^*) = (1 - \lambda) \cdot w(f, Q) + \lambda \sum_{Q' \in \text{Related}(Q)} \frac{w(f, Q')}{|\text{Related}(Q)|} \]
Ad Feature Weighting and Scoring

• Ad features are weighted using BM25F

\[ w(f, A) = \frac{(k + 1) \cdot \#(f, A)}{k \cdot \left( (1 - b) + b \cdot \frac{|A|}{|A|_{avg}} \right) + \#(f, A)} \cdot \text{idf}(f) \]

  – \#(f, A) is a field weighted term frequency

• Final ad scoring function:

\[ S(Q, Q^*, A) = \sum_{F \in \{F_u, F_p, F_c\}} \lambda_F \cdot \text{sim}_F(Q^*, A) \cdot (1 + \text{prox}_F(Q, A)) \]

  – \text{sim}(Q^*, A) is the cosine similarity between the expanded query vector and the ad vector

  – \text{prox}(Q,A) is a simple proximity measure between original query and the ad
Experimental Evaluation

- Expansion cache consists of a large set of pre-expanded queries
  - Most frequent Yahoo! web search queries
  - Bid phrases from Yahoo!’s textual ad corpus
- Randomly sampled 400 rare queries from a web search query log
  - 121 in lookup table
  - 179 not in lookup table
- Human editors judged top 3 results for several system variants
  - Total of 3,556 judgments
  - Judgments are on an integral scale 1-5 (1=highly attractive, 5=poor)
- Evaluate using interpolated precision-recall curves and DCG
Queries Not Found in Lookup Table

![Graph showing precision vs recall for baseline and online expansion methods.](image-url)
Summary: Our proposed online expansion approach significantly improves DCG over the baseline approach

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Online Expansion</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCG@1</td>
<td>0.99</td>
<td>1.07 (+8.1%)†</td>
</tr>
<tr>
<td>DCG@2</td>
<td>1.57</td>
<td>1.66 (+5.7%)†</td>
</tr>
<tr>
<td>DCG@3</td>
<td>1.97</td>
<td>2.10 (+6.6%)‡</td>
</tr>
</tbody>
</table>
## Summary

Online expansion less useful for queries found in lookup table. Offline expansion tends to be more effective for these queries.

<table>
<thead>
<tr>
<th>Expansion Type</th>
<th>DCG@1</th>
<th>DCG@2</th>
<th>DCG@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>2.89</td>
<td>4.56</td>
<td>5.75</td>
</tr>
<tr>
<td>Online Expansion</td>
<td>2.83</td>
<td>4.43</td>
<td>5.54</td>
</tr>
<tr>
<td>Offline Expansion</td>
<td>3.07‡</td>
<td>4.75</td>
<td>5.87</td>
</tr>
<tr>
<td>Online+Offline Expansion</td>
<td>2.91</td>
<td>4.44</td>
<td>5.59</td>
</tr>
</tbody>
</table>
Hybrid Online / Offline Approach

• Results so far…
  – Offline expansion works best for queries in lookup table
  – Online expansion works best for queries not in lookup table

• Rather than blindly applying one approach to all rare queries, we can use an adaptive approach that gets the best of both worlds
  – Use offline expansion for queries in lookup table
  – Use online expansion for queries not in lookup table

• We refer to this as the “hybrid” approach
Summary: Online expansion is necessary to achieve good results for rare queries. The hybrid approach is superior to any other single method.
Conclusions

• Producing relevant advanced match ads for tail queries is challenging

• We proposed a method that leverages pre-processed head/torso query expansions to improve rare/tail queries

• Tail queries are mapped to head/torso queries, via similarity search against an inverted index of offline expansions
  – Can (theoretically) expand 100% of the query volume
  – Nominal latency overhead

• Experimental results show significant improvements in DCG (+8%) and precision at all recall levels
Thank you!