Learning to **Attend, Copy, and Generate** for Session-Based Query Suggestion

Mostafa Dehghani, Sascha Rothe, Enrique Alfonseca, and Pascal Fleury
Query suggestion

● Task:
  ○ Assisting users to formulate their queries to better represent their intent during Web search by providing suggestions for the next query.

● Purpose:
  ○ Disambiguation
  ○ Tackling the lack of user knowledge on search topic
  ○ In general accelerates search satisfaction.
Query suggestion

● Task:
  o Assisting users to formulate their queries to better represent their intent during Web search by providing suggestions for the next query.

● Purpose:
  o Disambiguation
  o Tacking the lack of user knowledge on search topic

● In general accelerates search satisfaction.
How to do query suggestion?

- Wisdom of crowds:
  - Analyzing the search logs to use either **query co-occurrences** in the search logs, or document **clicks information**.
  - Suffer from data **sparsity**.

- Context queries:
  - As a **sequence** of attempts for finding relevant information:
    - Input: a **sequence of words** in a **sequence of queries**.
    - Output: a **sequence of words**.
Reformulation patterns

- Users reformulation patterns:
  - Term addition
  - Term removal
  - Term retention
Term retention

● Makes up a large proportion of query reformulation in search sessions!
  ○ An average of 62% of the terms in a query are retained from their preceding queries
  ○ More than 39% of the users repeat at least one term from their previous query

● Retained terms are clearly core terms indicating the user’s information need, hence, they are usually discriminative terms and entities.
  ○ More than 67% of the retained terms in the sessions are from the bottom 10% of terms ordered by their frequency.
Attend, Copy, Generate!

- Our idea is to consider:
  - Structure of the data (the notion of hierarchy)
  - Natural reformulation pattern

- IDEA:
  - Augmenting the standard seq2seq with
    - Hierarchical attention
    - Copy mechanism
Example

Consider an example session is composed of three queries:

- *bob dylan* → *forever young dylan* → *dylan photo*
Evaluation

○ Evaluating the model as a **discriminative** model
  - Select some candidates and use the model as scoring function (given input-output) to rank the candidates
  - Use mean reciprocal rank (MRR) to measure the quality of the ranking

○ Evaluating the model as a **generative** model → **HOW?**
Evaluating the model as a generative model

- Word Overlap Based query Similarity
  - Word Error Rate (PER)
  - Failure: `<city hall phone number , municipality contact information>`

- Embedding Based query Similarity
  - We calculate the query-level embedding using vector extrema, then we compute the cosine similarity ($sim_{ebm}$).
Evaluating the model as a generative model

- Retrieval Based query Similarity (Three setups, three metrics!)
  
  - Given an external collection of documents, we submit the target query leading to a **reference list**. Then we submit the generated query leading to a new **rank** list, the we calculate the agreement of these two ranked lists ($sim_{ret}$).
  
  - Using pseudo relevance feedback for creating the reference list ($sim_{ret}^+$). → idea: comparing to a (in average) better version of the target query!
  
  - Merging the reference list of the remaining query as the final reference list ($sim_{ret}^{++}$) → idea: Considering the cases where we don’t have an immediate agreement.
Experiments: Main Results

- Evaluation based on discrimination ability

Table 1: Performance of the different methods as discriminative models. \((x)\) indicates that the improvements with respect to the method in row \(x\) is statistically significant, at the 0.05 level using the paired two-tailed t-test with Bonferroni correction.

<table>
<thead>
<tr>
<th>#</th>
<th>Model</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MPS</td>
<td>0.5216</td>
</tr>
<tr>
<td>2</td>
<td>BaseRanker</td>
<td>0.5530(^1)</td>
</tr>
<tr>
<td>3</td>
<td>BaseRanker + Seq2Seq</td>
<td>0.5679(^{1,2})</td>
</tr>
<tr>
<td>4</td>
<td>BaseRanker + HRED [42]</td>
<td>0.5727(^{1,2})</td>
</tr>
<tr>
<td>5</td>
<td>BaseRanker + (Seq2Seq + QaA)</td>
<td>0.5744(^{1,2})</td>
</tr>
<tr>
<td>6</td>
<td>BaseRanker + (Seq2Seq + CM)</td>
<td>0.5851(^{1,2,3,4,5})</td>
</tr>
<tr>
<td>7</td>
<td>BaseRanker + ACG</td>
<td>0.5941(^{1,2,3,4,5,6})</td>
</tr>
</tbody>
</table>
## Experiments: Main Results

- Evaluation based on generation ability

<table>
<thead>
<tr>
<th>#</th>
<th>Method</th>
<th>Overlap Based</th>
<th>Embedding Based</th>
<th>Retrieval Based</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PER (%)</td>
<td>sim&lt;sub&gt;emb&lt;/sub&gt;</td>
<td>sim&lt;sub&gt;ret&lt;/sub&gt;</td>
</tr>
<tr>
<td>1</td>
<td>seq2seq</td>
<td>84.11 (± 6.3)</td>
<td>0.5170 (± 0.003)</td>
<td>0.1630 (± 0.008)</td>
</tr>
<tr>
<td>2</td>
<td>BaseRanker + seq2seq (top-1)</td>
<td>72.23 (± 8.1)</td>
<td>0.5019 (± 0.006)</td>
<td>0.4375 (± 0.009)</td>
</tr>
<tr>
<td>3</td>
<td>seqsSeq + QaA</td>
<td>80.90 (± 5.0)</td>
<td>0.5517 (± 0.004)</td>
<td>0.2012 (± 0.009)</td>
</tr>
<tr>
<td>4</td>
<td>seq2seq + CM</td>
<td>71.16 (± 3.5)</td>
<td>0.6119 (± 0.003)</td>
<td>0.3563 (± 0.009)</td>
</tr>
<tr>
<td>5</td>
<td>HRED [42]</td>
<td>81.50 (± 4.9)</td>
<td>0.5455 (± 0.004)</td>
<td>0.2667 (± 0.008)</td>
</tr>
<tr>
<td>6</td>
<td>BaseRanker + HRED [42] (top-1)</td>
<td>72.36 (± 7.3)</td>
<td>0.5200 (± 0.004)</td>
<td>0.4504 (± 0.009)</td>
</tr>
<tr>
<td>7</td>
<td>ACG</td>
<td>68.03 (± 5.6)</td>
<td>0.6473 (± 0.004)</td>
<td>0.3612 (± 0.008)</td>
</tr>
<tr>
<td>8</td>
<td>BaseRanker + ACG (top-1)</td>
<td>70.66 (± 7.1)</td>
<td>0.5196 (± 0.004)</td>
<td>0.4594 (± 0.008)</td>
</tr>
</tbody>
</table>
Analysis: Multiple Query Suggestion.

- During the decoding, after generating the first suggestion, we ignore the fact that the first suggestion was generated through the beam search, i.e. ⟨/q⟩ was generated. Instead, we continue decoding until the next suggestion is generated.

![Figure 3: Performance of the generated queries at different ranks.](image)
Analysis: Effect of Session Length

- It has been shown that when the length of a session increases, the percentage of repeating previously-used terms also increases.
  - The benefit of copying kicks in!

(a) Evaluation based on Discrimination  
(b) Evaluation based on Generative

Figure 4: Performance of ACG compared to HRED on sessions with different lengths.
Analysis: Attending the Promising Parts of the Context

- Noise term insertion
- Noise query insertion
- Noise session insertion

Table 3: Performance (and performance loss) of the different methods as generative models on noisy data, in terms of $sim_{emb}$.

<table>
<thead>
<tr>
<th>Method</th>
<th>Noise term insertion</th>
<th>Noise query insertion</th>
<th>Noise session insertion</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq2seq</td>
<td>0.4973 (-3.8%)</td>
<td>0.4419 (-14.5%)</td>
<td>0.3969 (-23.2%)</td>
</tr>
<tr>
<td>HRED [42]</td>
<td>0.5380 (-1.4%)</td>
<td>0.5140 (-5.8%)</td>
<td>0.4505 (-17.4%)</td>
</tr>
<tr>
<td>ACG</td>
<td>0.6366 (-1.6%)</td>
<td>0.6019 (-7.0%)</td>
<td>0.5878 (-9.1%)</td>
</tr>
</tbody>
</table>
Thank you!

mostafadehghani.com
@m__dehghani