Personalized Language Model for Query Auto-Completion

Aaron Jaech and Mari Ostendorf
University of Washington
Query Auto-Completion

- Search engine suggests queries as the user types

- Idea from Park & Chiba (2017): Use an LSTM to generate completions
  - Memory savings over most popular completion
  - Handles previously unseen prefixes

- Can we do better by adapting the LM to provide personalized suggestions?
RNN Language Model Adaptation

- Learn an embedding, $c$, for each user and use it to adapt the predictions
- **Method #1:** Concatenate the user embedding with the input at each step*
  - Same as applying a constant linear shift to the bias vector (in recurrent & output layers)
  - Leaves most of the recurrent model parameters unchanged
- **Method #2:** Low-rank adaptation of recurrent weight matrix (FactorCell model)

\[
\begin{align*}
\hat{W} &= [W \ V] \\
\hat{h}_t &= \sigma(\hat{W} [h_{t-1}, e_t, c] + b) \\
&= \sigma(W [h_{t-1}, e_t] + Vc + b)
\end{align*}
\]

* Referred to here as ConcatCell (Mikolov & Zweig, 2012)
FactorCell Model

- The adaptation matrix is formed from a product of the context embedding with left and right bases.
- The two bases tensors ($L$ and $R$) hold $k$ different rank $r$ matrices, each the same size as $W$. Context vectors give a weighted combination.
Learning

- User embeddings, recurrent layer weights and \{L, R\} tensor learned jointly
- Need online learning to adapt to users that were not previously seen
- In joint training, learn a cold-start embedding for set of infrequent users
- During evaluation
  - Initialize each users’ embedding with learned cold-start vector
  - Make query suggestions
  - After user selects a query, back-propagate and only update the user embedding
Data & Experiments

- Using AOL 2006 Query Log data, 173K users and 12 million queries for training
- User embedding size = 32, LSTM size = 600
- Evaluate on 500K queries with disjoint user population
- Mean reciprocal rank (MRR) as a metric
Experimental Results

Mean Reciprocal Rank

Performance for users with > 50 queries

Relative Improvement in MRR

Benefit improves over time!
Qualitative Comparison

What queries are boosted the most after searching for “high school softball” and “math homework help”?

<table>
<thead>
<tr>
<th>FactorCell</th>
<th>ConcatCell</th>
</tr>
</thead>
<tbody>
<tr>
<td>high school musical</td>
<td>horoscope</td>
</tr>
<tr>
<td>chris brown</td>
<td>high school musical</td>
</tr>
<tr>
<td>funnyjunk.com</td>
<td>homes for sale</td>
</tr>
<tr>
<td>funbrain.com</td>
<td>modular homes</td>
</tr>
<tr>
<td>chat room</td>
<td>hair styles</td>
</tr>
</tbody>
</table>

Queries that most decrease in likelihood with the FactorCell include travel agencies and plane tickets.
Recent Related Work: Florini & Lu, NAACL 2018

- Also personalized LSTM for query prediction
- ConcatCell adaptation framework
- User embedding learned separately
- No online learning
- Assessed on two datasets, but different split of AOL data
- Confirms benefit of adapted LM
Conclusions

- Personalization helps and the benefit increases as more queries are seen
- Stronger adaptation of the recurrent layer (FactorCell) gives better results than concatenating a user vector
  - No extra latency/computation due to caching of adapted weight matrix
- Try out the FactorCell on your data
  - [http://github.com/ajaech/query_completion](http://github.com/ajaech/query_completion)
THANKS!
Qualitative Comparison

What queries are boosted the most after searching for “prada handbags” and “versace eyewear”?

<table>
<thead>
<tr>
<th>FactorCell</th>
<th>ConcatCell</th>
</tr>
</thead>
<tbody>
<tr>
<td>neiman marcus</td>
<td>craigslist nyc</td>
</tr>
<tr>
<td>pottery barn</td>
<td>myspace layours</td>
</tr>
<tr>
<td>jc penny</td>
<td>verizon wireless</td>
</tr>
<tr>
<td>verizon wireless</td>
<td>jensen ackles</td>
</tr>
<tr>
<td>bed bath and beyond</td>
<td>webster dictionary</td>
</tr>
</tbody>
</table>
Backup Slides
The adapted weight matrix is a drop-in replacement for $W$

$$W_A = W_0 + LR$$

$$h_{t+1} = \sigma((W_0 + W_c)[h_t, e_t] + b)$$

Much larger change in recurrent layer than what ConcatCell does
Prefix and query length

- Longer queries are more difficult
- Suggestion quality improves as prefix length increases