Dissertation Talk

Temporal Context Modeling for Text Streams

Jinfeng Rao
jinfeng@cs.umd.edu
Department of Computer Science
University of Maryland

Committee:
Jimmy Lin (advisor), Alan Sussman (Dean’s rep),
Marine Carpuat, Jordan Boyd-Graber, John Dickerson

© Jinfeng Rao 2018
Information Retrieval

Query
University of Maryland

Document
University of Maryland
UMD, College Park
Maryland

Search Engine

Document Ranking

d1: -2.78
University of Maryland | A Preeminent Public Research University
https://www.umd.edu/
The University of Maryland, College Park is one of the nation's preeminent public research universities. A global leader...
UMD Undergraduate Admissions · Maryland Graduate School · Contact Us · Visit

d2: -4.26
University of Maryland, College Park - Wikipedia
The University of Maryland-College Park is a public research university located in the city of College Park in Prince George's County, Maryland. Founded in 1856...
Campus: Suburban, 1,339 acres (5.42 km²) Motto in English: Strong deeds, gentle words Students: 39,083 (Fall 2016) Motto: Fatti maschi, parole femine unofficial

d3: -12.5
University of Maryland Medical Center
https://www.umms.org/
University of Maryland Medical Center (UMMC) is a leading teaching hospital located in downtown Baltimore. As the flagship site, UMMC has a history...

© Jinfeng Rao 2018
Information Retrieval (state-of-the-art)

**Representation**

- **Bag of Words**

- Document is represented by individual words.

- Feature engineering:
  - text match
  - query/doc statistics
  - click…

**Ranking Model**

**Unsupervised**

- Language Model
- VSM
- TF-IDF
- BM25
- DPH
- COOR

**Learning to Rank**

- Pointwise: Logistic Regression, SVM, …
- Pairwise: RankSVM, LambdaMART…
- Listwise: ListNet…

**Neural Network**: DSSM, DRMM, DUET, …

© Jinfeng Rao 2018
Outline

1. Temporal Context Modeling for Tweet Search
   1.1 Pseudo trend modeling
   1.2 Query trend modeling

2. Multi-Perspective Lexical Modeling for Tweet Search

3. Temporal Context Modeling for Voice Search
   3.1 Voice session modeling
   3.2 Multi-task learning
Publications

Motivation

• Distribution of relevant documents in time
  • horizontal axis denotes time prior to query time
  • green for relevant document, red for highly relevant, gray for irrelevant

• Insights
  • relevant docs cluster in time
  • clustering patterns vary across queries
Background: Kernel Density Estimation

- A two-stage process for pseudo trend estimation:
  - Retrieve docs to estimate the ground truth distribution (pseudo trend)
  - Rerank docs with the estimated pseudo trend.

- **Kernel density estimation**, Efron et al. SIGIR’14 [10]

\[
\hat{f}_\omega(x) = \frac{1}{nh} \sum_{i=0}^{n} \omega_i K \left( \frac{x - x_i}{h} \right)
\]

Four weighting schemas \( \omega \):
1. Uniform
2. Score-based
3. Rank-based
4. Oracle (upper bound)

© Jinfeng Rao 2018
Continuous Hidden Markov Model (cHMM)

Traditional HMM models have a finite set of discrete observations, while our observations are the number of documents in a state.

Our assumption [1]:

\[ B_i(O_t) = P(O_t | q_t = i) \sim N(u_i, \sigma_i) \]

emission prob.

observed document count at time \( t \).

state \( i \)

Optimization with EM algorithm:

- **E-step:** \( Q(\lambda, \lambda') \propto \sum_q \log P(O, q | \lambda) P(O, q | \lambda') \)
- **M-step:** maximizes the \( Q \) function and updates parameter estimation.
- Viterbi algorithm to decode the most likely state sequence.
Temporal Query Expansion with cHMM

Pipeline:
1. Given a query, retrieve the top k documents for cHMM estimation

2. With estimated states, select terms in bursty states for query expansion:

\[
P(w|R) = \sum_{D \in C} P(D)P(w|D) \prod_{i=1}^{n} P(t_i|D)
\]
where \(C\) is the set of documents from bursty states.

3. With augmented query, retrieve again and evaluate.
Experimental Setup

Datasets: TREC Microblog Track 2011-2014

<table>
<thead>
<tr>
<th>Test Set</th>
<th>2011</th>
<th>2012</th>
<th>2013</th>
<th>2014</th>
</tr>
</thead>
<tbody>
<tr>
<td># of query topics</td>
<td>49</td>
<td>60</td>
<td>60</td>
<td>55</td>
</tr>
<tr>
<td># of query-doc pairs</td>
<td>39,780</td>
<td>49,879</td>
<td>46,192</td>
<td>41,579</td>
</tr>
<tr>
<td># of relevant docs</td>
<td>1,940</td>
<td>4,298</td>
<td>3,405</td>
<td>6,812</td>
</tr>
<tr>
<td># of unique words</td>
<td>21649</td>
<td>27470</td>
<td>24,546</td>
<td>22099</td>
</tr>
<tr>
<td># of unique OOV words</td>
<td>13067</td>
<td>17190</td>
<td>15724</td>
<td>14331</td>
</tr>
<tr>
<td># of URLs</td>
<td>20351</td>
<td>25405</td>
<td>23100</td>
<td>20885</td>
</tr>
<tr>
<td># of hashtags</td>
<td>6784</td>
<td>8019</td>
<td>7869</td>
<td>7346</td>
</tr>
</tbody>
</table>

Observations
- Out-of-vocabulary (OOV) word rates are above 50%
- Lots of URLs and Hashtags

Underlying Collections:
- Tweets2011 (16M tweets) for TREC 2011-2012 topic set
- Tweets2013 (243M tweets) for TREC 2013-2014 topic set

Evaluated as a ranking task on mean average precision (MAP) and precision at K (P@K)
cHMM Evaluation

Results on TREC 2011-2012 topic set:

<table>
<thead>
<tr>
<th>Method</th>
<th>P5</th>
<th>P15</th>
<th>P30</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>QL</td>
<td>0.465</td>
<td>0.411</td>
<td>0.354</td>
<td>0.268</td>
</tr>
<tr>
<td>RM3</td>
<td>0.500</td>
<td>0.433</td>
<td>0.378</td>
<td>0.302</td>
</tr>
<tr>
<td>RM3 + KDE (score)</td>
<td>0.494</td>
<td>0.436</td>
<td>0.379</td>
<td>0.300</td>
</tr>
<tr>
<td>RM3 + KDE (rank)</td>
<td>0.490</td>
<td>0.425</td>
<td>0.376</td>
<td>0.292</td>
</tr>
<tr>
<td>RM3 + KDE (oracle)</td>
<td>0.548*</td>
<td>0.492*</td>
<td>0.422*</td>
<td>0.319*</td>
</tr>
<tr>
<td>cHMM</td>
<td>0.528*</td>
<td>0.444*</td>
<td>0.391°</td>
<td>0.310°</td>
</tr>
</tbody>
</table>

RM3 denotes the classical query expansion method without temporal information.

○ and ● denote statistically significant to the RM3 baseline at p<0.10 and p<0.05, respectively.

Estimated states for topic 14 from day 1 to day 6, with an improvement of 0.22(MAP) and 0.57(P30) against RM3:

Blue for bursty state, grey for intermediate, white for quiet.
Neural Temporal Framework

• Lexical modeling
  • A Siamese network that generates an embedding for each <query, doc> pair.
  • Can be extended to any Siamese model.
  • We choose: SM[13], DSSM[14], MP-CNN[16]

• Temporal modeling [2]
  • Order docs by their timestamps
  • A sequence learning framework – neighboring docs contribute relevance signals to each other.
  • Implemented by a bi-directional LSTM
1.1 Pseudo Trend Modeling

Evaluation

- Train on TREC 2011 topic set, test on 2012 topic set.

<table>
<thead>
<tr>
<th>ID</th>
<th>Method</th>
<th>P15</th>
<th>P30</th>
<th>P100</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Query Likelihood (QL) [23]</td>
<td>0.381</td>
<td>0.329</td>
<td>0.234</td>
<td>0.200</td>
</tr>
</tbody>
</table>

**Temporal Baselines**

<table>
<thead>
<tr>
<th></th>
<th>Method</th>
<th>P15</th>
<th>P30</th>
<th>P100</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>uniform</td>
<td>0.366</td>
<td>0.326</td>
<td>0.243</td>
<td>0.203</td>
</tr>
<tr>
<td>3</td>
<td>score-based</td>
<td>0.383</td>
<td>0.334</td>
<td>0.244</td>
<td>0.203</td>
</tr>
<tr>
<td>4</td>
<td>rank-based</td>
<td>0.387</td>
<td>0.337</td>
<td>0.244</td>
<td>0.204</td>
</tr>
<tr>
<td>5</td>
<td>oracle</td>
<td>0.411</td>
<td>0.389</td>
<td>0.260</td>
<td>0.229</td>
</tr>
</tbody>
</table>

**Neural Ranking Approaches**

<table>
<thead>
<tr>
<th></th>
<th>Method</th>
<th>P15</th>
<th>P30</th>
<th>P100</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>DSSM [15]</td>
<td>0.187</td>
<td>0.168</td>
<td>0.153</td>
<td>0.102</td>
</tr>
<tr>
<td>7</td>
<td>SM [33]</td>
<td>0.203</td>
<td>0.188</td>
<td>0.170</td>
<td>0.116</td>
</tr>
<tr>
<td>8</td>
<td>Multi-Perspective CNN [11]</td>
<td>0.401</td>
<td>0.356</td>
<td>0.252</td>
<td>0.197</td>
</tr>
</tbody>
</table>

**Neural Ranking + Temporal Modeling**

<table>
<thead>
<tr>
<th></th>
<th>Method</th>
<th>P15</th>
<th>P30</th>
<th>P100</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>SM [33] + Temporal</td>
<td>0.222</td>
<td>0.196</td>
<td>0.169</td>
<td>0.116</td>
</tr>
<tr>
<td>10</td>
<td>Multi-Perspective CNN [11] + Temporal</td>
<td>0.418</td>
<td>0.366</td>
<td>0.257</td>
<td>0.203</td>
</tr>
</tbody>
</table>

- Observations:
  - Existing neural ranking approaches don’t work well on tweet search
  - Temporal modeling significantly outperforms models without temporal information.
Outline

1. Temporal Context Modeling for Tweet Search
   1.1 Pseudo trend modeling
   1.2 Query trend modeling

2. Multi-Perspective Lexical Modeling for Tweet Search

3. Temporal Context Modeling for Voice Search
   3.1 Voice session modeling
   3.2 Multi-task learning
Motivation

- **Research question:** can we make use of the temporal statistics of query terms (query trends) to predict the ground truth?
  - What is query trend? E.g., collection frequencies of query terms for each 5 minutes.
  - An example of ground truth and term trends for query MB127 “hagel nomination filibustered” from TREC 2013 topic set.
Regression of Query Trends

Goal: Approximate the ground truth ($Y$) by taking a weighted sum of all query trends ($ft$).

$$Y \approx \sum_{t} w_t f_t$$
Term Importance Modeling

• Bursty terms can be more informative.

• We adopt entropy definition to measure the importance of terms.

• Given the counts of a particular term t (unigram/bigram) \{c_1, c_2, \ldots, c_n\},

\[
\text{Entropy}(t) = - \sum_i \frac{c_i}{C} \log \frac{c_i}{C}
\]

lower entropy = bursty term trend = more important
Two questions in this non-linear regression modeling:
- Q1: How to model the weights of different query terms?
- Q2: How to differentiate the contribution from unigrams with bigrams?

Q1 solution: exponential mapping from entropy to term weight

$$w_t = \exp(\theta \cdot e_t) - 1$$

Q2 solution: assume unigram weight $u_i$, then bigram weight $(1 - u_i)$

$$u_i = \text{logistic}(R_i, \gamma) = \frac{1}{1 + \exp(-\gamma R_i)}$$

where $R_i$ is the difference between the maximum unigram entropy and maximum bigram entropy.

**Intuition:** $R_i > 0 \Rightarrow \max(\text{unigram_entropy}) > \max(\text{bigram_entropy}) \Rightarrow u_i > 0.5$
Regression of Query Trends

• Problem reformulation:

\[ Y = \sum_t w_t f_t \quad \Rightarrow \quad Y_i \approx u_i U_i w_i^u + (1 - u_i) B_i w_i^b \]

unigram  bigram

• Objective Loss:

\[ L = \sum_{i=1}^{N_q} \| Y_i - (u_i U_i (e^{\alpha E_i^u} - 1)^T + (1 - u_i) B_i (e^{\beta E_i^b} - 1)^T) \|^2 \]

\[ + \lambda (\alpha^2 + \beta^2 + y^2) \]

which can be solved with gradient descent algorithm (more details in paper [4]).
Combine Query Trend with Pseudo Trend

- Two ways to estimate the ground truth distribution:
  - **Document-level**: pseudo trend through an initial retrieval
  - **Term-level**: regression over query trends

- Combine query trend and pseudo trend in a linear ranking model:

\[
S_d = \sum_i \alpha_i \cdot F_i(d, q) \quad \text{s.t.} \quad \sum_i \alpha_i = 1.
\]
Experimental Setup

• Dataset: TREC Microblog Track 2013-2014, total 115 topics.

• Experimental conditions:
  • Odd-even: odd numbered topics (57 topics) for training, even (58 topics) for testing
  • Even-odd: switch train/test split
  • Cross: 4-fold cross validation

• Baselines:
  • Query Likelihood (QL)
  • Recency Prior, Li et al. CIKM’03 [8]
  • Moving Window, Dakka et al. TKDE’12 [9]
  • Kernel Density Estimation (KDE), SIGIR’14 [10]
    • Uniform (IRDu)
    • Score-based (IRDs)
    • Rank-based (IRDr)
    • Oracle (upper bound)
\section*{Main Results}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
ID & Method & \multicolumn{2}{c|}{Odd-Even} & \multicolumn{2}{c|}{Even-Odd} & \multicolumn{2}{c|}{Cross} \\
 & & AP & P30 & AP & P30 & AP & P30 \\
\hline
1 & Query Likelihood (QL) \cite{15} & 0.271 & 0.475 & 0.357 & 0.564 & 0.315 & 0.520 \\
2 & Recency prior \cite{12} & 0.277 & 0.499$^1$ & 0.359 & 0.574 & 0.313 & 0.534$^{1,4}$ \\
3 & Moving Window (WIN) \cite{5} & 0.283$^1$ & 0.487$^1$ & 0.358 & 0.567 & 0.319 & 0.527 \\
\hline
4 & KDE \cite{8} & IRD$_u$ & 0.273 & 0.481 & 0.350 & 0.566 & 0.308 & 0.515 \\
5 & & IRD$_s$ & 0.274 & 0.487$^1$ & 0.353 & 0.577$^1$ & 0.314 & 0.530$^{1,4}$ \\
6 & & IRD$_r$ & 0.288$^{1,4,5}$ & 0.517$^{1,3,4,5}$ & 0.360 & 0.588$^{1,2,3,4}$ & 0.327$^{1,2,4,5}$ & 0.552$^{1,2,3,4,5}$ \\
\hline
7 & This work & QT & 0.278 & 0.492$^{1,4}$ & 0.367$^{1,4,5}$ & 0.587$^{1,2,3,4}$ & 0.320 & 0.530$^{1,4}$ \\
8 & & Reg & 0.276 & 0.488$^1$ & 0.366$^{1,4,5}$ & 0.576$^1$ & 0.329$^{1,2,4,5}$ & 0.535$^{1,4}$ \\
9 & & QT-IRD$_r$ & 0.290$^{1,2,4,5}$ & 0.522$^{1,2,3,4,5}$ & 0.370$^{1,2,3,4,5}$ & \textbf{0.598}$^{1,2,3,4,5}$ & 0.328$^{1,2,4,5}$ & 0.565$^{1,2,3,4,5}$ \\
10 & & Reg-IRD$_r$ & \textbf{0.302}$^{1,2,3,4,5,6}$ & \textbf{0.535}$^{1,2,3,4,5}$ & \textbf{0.370}$^{1,2,3,4,5}$ & \textbf{0.598}$^{1,2,3,4,5}$ & \textbf{0.332}$^{1,2,3,4,5}$ & \textbf{0.566}$^{1,2,3,4,5}$ \\
11 & Oracle & 0.314$^{1,2,3,4,5,6}$ & 0.536$^{1,2,3,4,5,6}$ & 0.382$^{1,2,3,4,5,6}$ & 0.636$^{1,2,3,4,5,6}$ & 0.349$^{1,2,3,4,5,6}$ & 0.586$^{1,2,3,4,5,6}$ \\
\hline
\end{tabular}
\caption{Comparison of various methods on query trend modeling.}
\end{table}

- **Conclusions:**
  - KDE with rank-based weights (IRD$_r$) is the strongest baseline.
  - Our approach (Reg-IRD$_r$) significantly outperforms all baselines, and is even close to the upper bound in some splits.
Per-Topic Analysis

Per-topic P30 improvement against the Query Likelihood (QL) and the best KDE baseline (IRDr).
Analysis of the Best-Performing Topic 144

• How query trend signals help?
  • red color for ground truth distribution
  • green for pseudo trend estimated by the best KDE method (IRD\textsubscript{r})
  • blue for query trends.

• Conclusion: A combination of pseudo trend (KDE) and query trend (Our approaches) provides a more accurate estimation to the ground truth distribution.
Outline

1. Temporal Context Modeling for Tweet Search
   1.1 Pseudo trend modeling
   1.2 Query trend modeling

2. Multi-Perspective Lexical Modeling for Tweet Search

3. Temporal Context Modeling for Voice Search
   3.1 Voice session modeling
   3.2 Multi-task learning
2. Multi-Perspective Relevance Matching

Challenges

• Tweet Search vs. Web Search (ours paper [5])
  • Shorter document length
  • Informal texts
  • Heterogeneous relevance signals (URLs, hashtags, etc.)

• No existing work on neural networks for tweet search
Multi-Perspective Relevance Matching

- Multi-perspective similarity modeling [5]
  - Input-level: query-tweet, query-url
  - Semantic-level: char-level, word-level, phrase-level
  - Weighting-level: multi-level weightings

- Main Components
  - Separate char-level and word-level modeling
  - Hierarchical convolutional layers for long-phrase modeling
  - Pooling-based matching with external weightings

- Query: BBC world service cuts
- Tweet: BBC news – BBC world service cuts to be outlined to staff
- Hashtag: #bbcworldservice

* bigger model figure in the next slide
2. Multi-Perspective Relevance Matching
Experimental Setup

• Datasets: TREC Microblog Track 2011-2014 topic sets

• Experimental condition:
  • three topic sets for training, the left one for testing
  • Randomly selected 10% query set from training set for validation

• Baselines:
  • Non-neural baselines:
    • Query Likelihood (QL)
    • Query Expansion (RM3) [11]
    • Learning to rank (LambdaMART) [12]
      • text-based features (5)
      • URL-based features (2)
      • hashtag-based features (2)
  • Neural baselines:
    • DSSM (2013) [14]
    • C-DSSM (2014) [15]
    • DUET (2017) [19]
    • MatchPyramid (2016) [23]
    • DRMM (2016) [17]
    • K-NRM (2017) [18]
  • Interpolation baselines:
    • DUET+QL
    • DRMM+QL
    • K-NRM+QL
Main Results

2. Multi-Perspective Relevance Matching

Observations
- Existing neural approaches suffered, while a simple interpolation works!
- Our approaches (MP-HCNN) significantly outperformed all competitive baselines.
Ablation Study

Figure: MAP scores with different convolution depth on 2011-2014 dataset.

Table: ablation study. Star symbol denotes the score is significantly lower than MP-HCNN at p<0.05.
MP-HCNN Error Analysis

### Table

<table>
<thead>
<tr>
<th>ID</th>
<th>Query</th>
<th>Sample Tweet</th>
<th>Label</th>
<th>Score/Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>QL</td>
<td>MP-HCNN</td>
</tr>
<tr>
<td>1</td>
<td>2: 2022 fifa soccer</td>
<td>#ps3 best sellers: fifa soccer 11 ps3 #cheaptweet</td>
<td>I</td>
<td>7.33(#54)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="https://www.amazon.com/fifa-soccer-11-playstation-3">https://www.amazon.com/fifa-soccer-11-playstation-3</a></td>
<td></td>
<td>0.85(#1)</td>
</tr>
<tr>
<td>2</td>
<td>qatar’s 2022 fifa world cup stadiums:</td>
<td>R</td>
<td>10.58(#2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td><a href="https://wordlesstech.com/qatars-2022-fifa-world-cup-stadiums/">https://wordlesstech.com/qatars-2022-fifa-world-cup-stadiums/</a></td>
<td></td>
<td>0.41(#105)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2022 world cup could be held at end of year: fifa : lausanne switzerland the 2022 world cup in qatar:</td>
<td>R</td>
<td>11.25(#1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>world cup in qatar: <a href="http://www.reuters.com/article/us-soccer-world-blatter">http://www.reuters.com/article/us-soccer-world-blatter</a></td>
<td></td>
<td>0.31(#127)</td>
<td></td>
</tr>
</tbody>
</table>

Figure: error analysis for the worst-performing topic 2. Label I denotes irrelevant, and R denotes relevant. The pink ground color denotes the score of phrase-matching by MP-HCNN. The brighter the color, the higher the matching score.

- **Observations**
  - Our model put more emphasizes on phrase match (such as “fifa soccer”)
  - The OOV word “2022” underweights its contribution.
  - Relevance signals in URLs are important.
Outline

1. Temporal Context Modeling for Tweet Search
   1.1 Pseudo trend modeling
   1.2 Query trend modeling

2. Multi-Perspective Lexical Modeling for Tweet Search

3. Temporal Context Modeling for Voice Search
   3.1 Voice session modeling
   3.2 Multi-task learning
Comcast’s Voice Interface for TV

22 million customers in 40 states of United States since 2015
http://www.xfinity.com/voice-remote

NLP IN THE CLOUD

1. Speech-to-Text
2. Intent Understanding
3. Action Resolution
4. Message Management

© Jinfeng Rao 2018
Challenges

• Short length of voice queries
  • Analysis over a week of 32M voice queries shows the average length is 2.04 words, much shorter than published stat. on smartphone & computers.

• Ambiguity
  • Query-level: a query “Chicago fires” can refer to either a television series or a soccer team.
  • Program-level: many TV programs share similar names: *The Princess Diaries, The Princess Diaries II: Royal Engagement*, *The Princess Knight*, etc…

• Many speech recognition errors
  • Little query/program lexical overlap because of SR errors, such as query “you” to program “Calliou”
Voice Session as Contexts

Find me free HBO shows.

I: Silicon Valley
2. West World
3. Blacklist
......

Find me Game of Throw

Do you mean the Fish Throw game or Game of Thrones?

Time
Session Analysis (~32M voice queries)

- Observations
  - More than 30% sessions have multiple queries, accounting for 57% of all queries.
  - The longer of a session, the higher of unsatisfactory rate
Problem Formulation

• Assumption
  • User will keep issuing queries until he find the intended program, i.e., [“hbo series”, “game of throw”, “game of throne”]

• Given a voice query session \( S_i = [q_1, q_2, \ldots, q_n] \), we aim to predict the intended program at each query time, exploiting previous queries as contexts:

\[
\text{Data: } D = \{(s_i, p_i) | s_i = [q_{i_1}, \ldots, q_{i_{|s_i|}}], p_i \in \Phi\}_{i=1}^{|D|} \\
\text{Model: } \hat{\theta} = \arg \max_\theta \prod_{i=1}^{|D|} \left( \prod_{t=1}^{|s_i|} P(p_i | q_{i_1}, \ldots, q_{i_t}; \theta) \right)
\]

where \( S_i \) denotes a voice session, \( P_i \) is the ground truth program,
for each session
for each query
Bayesian Decomposition

Model: $\hat{\theta} = \arg\max_\theta \prod_{i=1}^{D} \prod_{t=1}^{s_i} P(p_i|q_{i_1}, ..., q_{i_t}; \theta)$

$$P(p_i|q_{i_1}, ..., q_{i_t}) = P(p_i|c_{i_t}) \cdot P(c_{i_t}|v_{i_1}, ..., v_{i_t}) \cdot \prod_{j=1}^{t} P(v_{i_j}|q_{i_j})$$

$v_{i_t} \sim F(q_{i_t}; \theta_F)$, $c_{i_t} \sim G(v_{i_1}, ..., v_{i_t}; \theta_G)$, $p_i \sim H(c_{i_t}; \theta_H)$, $1 \leq t \leq |s_i|$

- prediction
- contextual embedding
- query embedding
3.1 Voice Session Modeling

Model: Hierarchical Recurrent Neural Networks (HRNN)

Program/Type Prediction

Query-level LSTM

Query Embedding Vector

Char-level LSTM

Char-level 1-Hot Vector

User's Query Session

'Game of '

'Game of Throw '

'Game of Thrones'

P('Game of Thrones') = 0.4

P('Game of Thrones') = 0.8

P('Game of Thrones') = 1.0

© Jinfeng Rao 2018
Model Variants

• Three ways of query representations
  • **character-level**: better for resolving speech recognition errors (i.e., “you” to “Caillou”)
  • **word-level**: better for semantic modeling
  • **combined**: combine the above two embedding ways

• Three model options
  • **basic**: remove the contextual modeling part
  • **full context**: full architecture, end-to-end training
  • **constrained context**: full architecture, pretrain the underlying layers, then fine-tune the top layers

• Total 3*3 model configurations
Experimental Setup

• Dataset

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#sessions</th>
<th>#queries</th>
<th>Avg. session len</th>
<th>Avg. query len</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>126016</td>
<td>181058</td>
<td>1.44</td>
<td>2.34</td>
</tr>
<tr>
<td>Validation</td>
<td>28427</td>
<td>82828</td>
<td>2.91</td>
<td>2.30</td>
</tr>
<tr>
<td>Test</td>
<td>28173</td>
<td>82272</td>
<td>2.92</td>
<td>2.30</td>
</tr>
</tbody>
</table>

• Metrics
  • Precision at 1 (P@1), Precision at 5 (P@5), Mean Reciprocal Rank (MRR)

• Baselines
  • TF-IDF: match query with programs by character-level 3grams.
  • BM25
  • SVM-Rank with three sets of features: 1) BM25 score, 2) popularity prior of the program, and 3) the word embedding based similarity features.
  • Deep Structure Semantic Models (DSSM): 3gram based neural matching method
  • Basic model with three query representations.
## Main Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Query</th>
<th>P@1</th>
<th>P@5</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF</td>
<td>3-gram</td>
<td>0.518</td>
<td>0.593</td>
<td>0.543</td>
</tr>
<tr>
<td>BM25</td>
<td>3-gram</td>
<td>0.533</td>
<td>0.596</td>
<td>0.565</td>
</tr>
<tr>
<td>SVM-Rank</td>
<td>-</td>
<td>0.547</td>
<td>0.621</td>
<td>0.582</td>
</tr>
<tr>
<td>DSSM</td>
<td>-</td>
<td>0.568</td>
<td>0.617</td>
<td>0.584</td>
</tr>
</tbody>
</table>

**Our Approach**

<table>
<thead>
<tr>
<th>Model</th>
<th>Query</th>
<th>P@1</th>
<th>P@5</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>char</td>
<td>0.605</td>
<td>0.647</td>
<td>0.690</td>
</tr>
<tr>
<td>Basic</td>
<td>word</td>
<td>0.609</td>
<td>0.644</td>
<td>0.677</td>
</tr>
<tr>
<td>Basic</td>
<td>comb</td>
<td>0.614</td>
<td>0.651</td>
<td>0.687</td>
</tr>
<tr>
<td>Full Context</td>
<td>char</td>
<td>0.482</td>
<td>0.532</td>
<td>0.580</td>
</tr>
<tr>
<td>Full Context</td>
<td>word</td>
<td>0.599</td>
<td>0.638</td>
<td>0.687</td>
</tr>
<tr>
<td>Full Context</td>
<td>comb</td>
<td>0.598</td>
<td>0.643</td>
<td>0.688</td>
</tr>
<tr>
<td>Cons-Context</td>
<td>char</td>
<td>0.639</td>
<td>0.684</td>
<td>0.731</td>
</tr>
<tr>
<td>Cons-Context</td>
<td>word</td>
<td>0.639</td>
<td>0.683</td>
<td>0.729</td>
</tr>
<tr>
<td><strong>Cons-Context</strong></td>
<td><strong>comb</strong></td>
<td><strong>0.643</strong></td>
<td><strong>0.687</strong></td>
<td><strong>0.734</strong></td>
</tr>
</tbody>
</table>

Two key factors: Session Context Modeling Query Combination
HRNN Context Analysis

- How does context signal help?
  - For each session length, we plot the average precision at different query position.
  - We take sessions with length of 6 as example

Observations
- Non-context methods (SVM-Rank, DSSM, Basic) stay flat at different query positions.
- Context methods consistently goes up with more queries as context.
# Case Study

### Session

**Cacio** -> **You** -> **You** -> **Calliou**

### Program

**Calliou**

<table>
<thead>
<tr>
<th>Model</th>
<th>Query</th>
<th>Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM25</td>
<td>3gram</td>
<td>Pacific Rim -&gt; Now You See Me -&gt; Now You See Me -&gt; 😊</td>
</tr>
<tr>
<td>SVM-Rank</td>
<td></td>
<td>Pacific Rim -&gt; Now You See Me -&gt; Now You See Me -&gt; 😊</td>
</tr>
<tr>
<td>DSSM</td>
<td>3gram</td>
<td>Pacific Rim -&gt; Young -&gt; Young -&gt; 😊</td>
</tr>
</tbody>
</table>

### Our Approach

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>char</td>
<td>😊 (0.81) -&gt; 😊 (0.80) -&gt; 😊 (0.80) -&gt; 😊 (1.0)</td>
</tr>
<tr>
<td>Basic</td>
<td>word</td>
<td>Child Genius (0.03) -&gt; 😊 (0.57) -&gt; 😊 (0.57) -&gt; 😊 (1.0)</td>
</tr>
<tr>
<td>Basic</td>
<td>comb</td>
<td>Paw Patrol (0.17) -&gt; 😊 (0.83) -&gt; 😊 (0.83) -&gt; 😊 (1.0)</td>
</tr>
<tr>
<td>Cons-Context</td>
<td>char</td>
<td>😊 (0.96) -&gt; 😊 (0.99) -&gt; 😊 (0.99) -&gt; 😊 (1.0)</td>
</tr>
<tr>
<td>Cons-Context</td>
<td>word</td>
<td>Wallykazam (0.07) -&gt; 😊 (0.59) -&gt; 😊 (0.86) -&gt; 😊 (1.0)</td>
</tr>
<tr>
<td>Cons-Context</td>
<td>comb</td>
<td>Paw Patrol (0.17) -&gt; 😊 (0.93) -&gt; 😊 (1.0) -&gt; 😊 (1.0)</td>
</tr>
</tbody>
</table>

© Jinfeng Rao 2018
Product Deployment

• Run after a number of simpler NLP modules and processed 8% of total queries which the current system provided no response.

• Among 70M queries in a week, 5.7M queries were sent to our model, which provided response to 4.2M queries, increasing end-to-end coverage from 92% to 98%.

• Human annotation shows 2/3 response are relevant, while the left 1/3 are considered irrelevant.
Outline

1. Temporal Context Modeling for Tweet Search
   1.1 Pseudo trend modeling
   1.2 Query trend modeling

2. Multi-Perspective Lexical Modeling for Tweet Search

3. Temporal Context Modeling for Voice Search
   3.1 Voice session modeling
   3.2 Multi-task learning
Building a Comprehensive Voice Navigation Platform

• Drawbacks of the previous HRNN approach
  • Classification-based nature, unable to retrieve unseen TV programs
  • Can’t scale to hundreds of thousands of programs

• Towards a comprehensive voice navigation platform
  • In reality, users can say anything to their TVs (i.e., looking for movies, set reminders, checking the weather, etc.)
  • A comprehensive system that can response to any general query.
1. Channel (30%): “HBO”
2. Movie/Series (27%): “Find me game of throne”
3. Event (9%): “Oscar show”
4. Browse (6%): “free action movies on HBO”
5. Record (1%): “record game of throne”
6. Command (1%): “turn on/off TV”
7. Hundreds of intents more…
Tag Distribution

Word-level Freq(%)

Examples

“Find me game of throne”

Context  Title

“Free action movies on HBO”

Cost  Genre  Category  Channel

© Jinfeng Rao 2018
Multi-Task Learning

• Given a voice session, perform three tasks:
  • Program prediction: predict the intended TV program if it exists
  • Intent classification: predict the query intent (movie/channel/browse/etc.)
  • Tagging: generate the word tag sequence for each query

• These three information together are necessary to understand an arbitrary query:
  • For simple queries, the three predictions should reinforce each other.
  • For ambiguous queries, combine the three evidences to interpret the query correctly.

“Find me game of throne”
  intent = series
  program = “game of throne”
  tags – not important here

“Record game of throne”
  intent = record
  program = “game of throne”
  tags - not important here

“Free action movies on HBO”
  intent = browse
  program = NA
  tags = [cost, genre, category, context, channel]
Multi-Task Problem Formulation

• Given a voice query session $S_i = [q_1, q_2, ..., q_n]$, we aim to predict the query intent, intended program and tag sequence at each query time, exploiting previous queries as contexts:

Data: $D = \{(s_i, p_i, A_i, T_i) | s_i = [q_{i_1}, ..., q_{i_n}], p_i \in \Phi, A_i = [a_{i_1}, ..., a_{i_n}], T_i = [\tau_{i_1}, ..., \tau_{i_n}]\}$

Model: $\hat{\theta} = \arg \max_{\theta} \prod_{i=1}^{\left|D\right|} \prod_{t=1}^{n} P(p_i, A_i, T_i | q_{i_1}, ..., q_{i_t}; \theta)$

where $S_i$ denotes a voice session, $P_i$ is the intended program (one per session), $A_i$ is the query intent sequence (one per query), $T_i$ is the tag sequences (one tag per word).
Bayesian Decomposition

Model: \( \hat{\theta} = \arg \max_{\theta} \prod_{i=1}^{D} \prod_{t=1}^{n} P(p_i, A_i, T_i | q_{i_1}, \ldots, q_{i_t}; \theta) \)

\( = \arg \max_{\theta} \prod_{i=1}^{D} \prod_{t=1}^{n} P(p_i | c_{i_t}) \cdot P(a_{i_t} | c_{i_t}) \cdot P(\tau_{i_t} | q_{i_t}) \)

\( \cdot P(c_{i_t} | v_{i_1}, \ldots, v_{i_t}) \cdot \prod_{j=1}^{t} P(v_{i_j} | q_{i_j}) \)

- Program prediction
- Intent classification
- Tagging
- Contextual embedding
- Query embedding
Multi-Task Learning Framework

- Query embedding component
  - Lookup layer + BiLSTM

- Contextual embedding component
  - another LSTM

- Task-specific components
  - Program prediction
  - Intent classification
  - Tagging

© Jinfeng Rao 2018
Program Prediction

• A ranking formulation

\[ P(p_i | c_{i_t}) \rightarrow P(\text{rel} | c_{i_t}, p^+) > P(\text{rel} | c_{i_t}, p^-), \forall p^- \in \Phi \]

• Model query and program jointly
  • Search-based program representation
  • Title-based
  • Combination-based

• Contrastive list-wise loss to maximize the pos relevance given all negatives

\[ P(p^+ | q_i_1, ..., q_i_t) = \frac{\exp(P(\text{rel} | p^+, q_i_1, ..., q_i_t))}{\sum_{p^- \in C} \exp(P(\text{rel} | p^-, q_i_1, ..., q_i_t))} \]

\[ P(\text{rel} | c_{i_t}, p) = \text{cosine}(c_{i_t}, p) \]
Intent Classification & Tagging

• Classification modeling since the intent vocab is small and stable
  • Modeled using a fully connected layer + cross-entropy loss

• Tagging is a sequential labeling task
  • Modeled using a conditional random field (CRF)
  • MLE for training and Viterbi algorithm for decoding
Multi-Task Optimization

• Stage 1: weighted sum of the losses of three tasks and optimize simultaneously

\[ L = w_p \cdot L_p + w_i \cdot L_i + w_t \cdot L_t \]

• Stage 2: fix the underlying shared components (the query embedding + contextual layer), fine-tune the task-specific layer with their own loss.
Experimental Setup

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#sessions</th>
<th>#queries</th>
<th>Avg. session len</th>
<th>Avg. query len</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>0.87M</td>
<td>1.2M</td>
<td>1.36</td>
<td>2.34</td>
</tr>
<tr>
<td>Validation</td>
<td>0.62M</td>
<td>0.82M</td>
<td>1.32</td>
<td>2.30</td>
</tr>
<tr>
<td>Test</td>
<td>0.62M</td>
<td>0.82M</td>
<td>1.33</td>
<td>2.30</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#programs</th>
<th>#channels</th>
<th>#intents</th>
<th>#tags</th>
<th>single-query vs. multi-query session</th>
<th>Unseen sessions (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>26247</td>
<td>244</td>
<td>109</td>
<td>11</td>
<td>80:20</td>
<td>10%</td>
</tr>
</tbody>
</table>

- We reused the baselines and evaluation metrics in the previous HRNN model.
## Main Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Program P@1</th>
<th>Program P@5</th>
<th>Intent Acc. (%)</th>
<th>Tagging Acc. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF</td>
<td>0.663</td>
<td>0.745</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BM25</td>
<td>0.674</td>
<td>0.750</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SVM-Rank</td>
<td>0.682</td>
<td>0.758</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DSSM</td>
<td>0.703</td>
<td>0.765</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Stanford CRF Tagger</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.821</td>
</tr>
</tbody>
</table>

Our Previous Work in CIKM’17

<table>
<thead>
<tr>
<th>Method</th>
<th>Program P@1</th>
<th>Program P@5</th>
<th>Intent Acc. (%)</th>
<th>Tagging Acc. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRNN w/ LSTM</td>
<td>0.724</td>
<td>0.783</td>
<td>0.915</td>
<td>0.884</td>
</tr>
<tr>
<td>HRNN w/ BiLSTM</td>
<td>0.725</td>
<td>0.786</td>
<td>0.916</td>
<td>0.939</td>
</tr>
</tbody>
</table>

Our Approach

<table>
<thead>
<tr>
<th>Method</th>
<th>Program P@1</th>
<th>Program P@5</th>
<th>Intent Acc. (%)</th>
<th>Tagging Acc. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-Task Learning</td>
<td>0.738</td>
<td>0.801</td>
<td>0.917</td>
<td>0.944</td>
</tr>
<tr>
<td>Multi-Task Learning</td>
<td><strong>0.757</strong></td>
<td><strong>0.824</strong></td>
<td><strong>0.925</strong></td>
<td><strong>0.945</strong></td>
</tr>
</tbody>
</table>
## Error Analysis: Program Prediction

<table>
<thead>
<tr>
<th>Method</th>
<th>Program P@1</th>
<th>Program P@5</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM-Rank</td>
<td>0.682</td>
<td>0.758</td>
</tr>
<tr>
<td>DSSM</td>
<td>0.703</td>
<td>0.765</td>
</tr>
<tr>
<td>HRNN</td>
<td>0.725</td>
<td>0.786</td>
</tr>
</tbody>
</table>

### Single-Task Learning

<table>
<thead>
<tr>
<th>Method</th>
<th>Program P@1</th>
<th>Program P@5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search-based</td>
<td>0.715</td>
<td>0.770</td>
</tr>
<tr>
<td>Title-based</td>
<td>0.720</td>
<td>0.796</td>
</tr>
<tr>
<td>Combination-based</td>
<td>0.738</td>
<td>0.801</td>
</tr>
</tbody>
</table>

### Multi-Task Learning

<table>
<thead>
<tr>
<th>Method</th>
<th>Program P@1</th>
<th>Program P@5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search-based</td>
<td>0.731</td>
<td>0.790</td>
</tr>
<tr>
<td>Title-based</td>
<td>0.728</td>
<td>0.808</td>
</tr>
<tr>
<td>Combination-based</td>
<td>0.757</td>
<td>0.824</td>
</tr>
</tbody>
</table>

### Observations

- Ranking-based approaches suffered
  - Lexical mismatch of query/program
  - Speech recognition errors

- Combination-based is effective

- Multi-task learning improved the performance by a lot
Error Analysis: Intent Classification and Tagging

Intent Classification

<table>
<thead>
<tr>
<th>Intent</th>
<th>Channel</th>
<th>Movie</th>
<th>Series</th>
<th>Event</th>
<th>Browse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel</td>
<td>97.9%</td>
<td>0.4%</td>
<td>0.5%</td>
<td>0</td>
<td>0.2%</td>
</tr>
<tr>
<td>Movie</td>
<td>0.4%</td>
<td>89.7%</td>
<td>2.4%</td>
<td>0.1%</td>
<td>3.3%</td>
</tr>
<tr>
<td>Series</td>
<td>0.2%</td>
<td>0.8%</td>
<td>96.2%</td>
<td>0</td>
<td>1.3%</td>
</tr>
<tr>
<td>Event</td>
<td>0.4%</td>
<td>3.7%</td>
<td>1.6%</td>
<td>87.3%</td>
<td>0</td>
</tr>
<tr>
<td>Browse</td>
<td>0.1%</td>
<td>1.9%</td>
<td>1.8%</td>
<td>0</td>
<td>94.1%</td>
</tr>
</tbody>
</table>

Table 4: Confusion matrix for the top 5 intent types.

Findings:
- Event intent is the most ambiguous one because of its blurred definition.
- For tagging, neural models win the Stanford CRF tagger in popular tags, while lose in the infrequent tags.

Accuracy distributions for the top 5 tags.
This work also won the 69th Emmy Award (2017) for the technical contribution in advancing television technologies.
Summary

• I present families of modeling techniques to model different granularities of time information and demonstrated their effectiveness on various applications:
  • Document-level pseudo trend for tweet search with cHMM and neural temporal model
  • Term-level query trend for tweet search with non-linear regression model
  • Session-level contexts for voice search with HRNN and multi-task learning framework

• I present a novel multi-perspective relevance matching model for tweet search to copy with the domain-specific challenges.
Thanks!

Q & A
<table>
<thead>
<tr>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. J. Rao, H. He, J. Lin, Integrating Lexical and Temporal Signals in Neural Ranking Models for Social Media Search, SIGIR NeuIR 2017</td>
</tr>
<tr>
<td>5. J. Rao, V. Yang, Y. Zhang, F. Ture, J. Lin, Multi-Perspective Relevance Matching with Hierarchical ConvNets for Social Media Search, arXiv 1805.08519</td>
</tr>
<tr>
<td>7. Multi-Task Learning with Neural Networks for Voice Query Understanding on an Entertainment Platform, KDD 2018</td>
</tr>
<tr>
<td>8. Xiaoyan Li and W. Bruce Cro, Time-Based Language Models, CIKM 2003, 469–475.</td>
</tr>
</tbody>
</table>
Reference

16. Hua He, Kevin Gimpel, and Jimmy Lin, 2014, Multi-Perspective Sentence Similarity Modeling with Convolutional Neural Networks, in EMNLP
17. Jiafeng Guo, Yixing Fan, Qingyao Ai, W Bruce Croft, 2016, A Relevance Matching Model for Ad-hoc Retrieval, in CIKM
Reference

Compression of Query Trends

- Query trends = number of terms * number of time intervals (i.e., every five mins)
- Large but sparse, amenable for compression

Compression methods:

1. Variable-byte Encoding (VB) [20]
2. Simple16 [21]
3. PForDelta (P4D) [22]
4. Discrete Wavelet Transform
5. Huffman Encoding (ours [3]):
   5.1 Partition a list of integers to continuous blocks, with each block having 8 consecutive integers.
   5.2 Construct a Huffman tree based on frequencies of blocks.
   5.3 Concatenate the binary Huffman codes of blocks, then convert back to compact integer list.
Evaluation of Compression

Compression on Tweets2013 corpus (243M tweets)
Evaluated on compression size and decoding time.

<table>
<thead>
<tr>
<th>Tweets2013</th>
<th>Unigrams</th>
<th>Bigrams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>size (GB)</td>
<td>percentage</td>
</tr>
<tr>
<td>Raw</td>
<td>52.5</td>
<td>-</td>
</tr>
<tr>
<td>VB</td>
<td>13.1</td>
<td>+446%</td>
</tr>
<tr>
<td>Simple16</td>
<td>2.2</td>
<td>-8.33%</td>
</tr>
<tr>
<td>P4D</td>
<td>2.4</td>
<td>-</td>
</tr>
<tr>
<td>Wavelet+VB</td>
<td>14.8</td>
<td>+517%</td>
</tr>
<tr>
<td>Wavelet+P4D</td>
<td>3.8</td>
<td>+58.3%</td>
</tr>
<tr>
<td>Huffman</td>
<td>0.71</td>
<td>-70.4%</td>
</tr>
<tr>
<td>Huffman+VB</td>
<td>0.48</td>
<td>-80.0%</td>
</tr>
<tr>
<td>Optimal</td>
<td>0.33</td>
<td>-86.2%</td>
</tr>
</tbody>
</table>

Observation:
1. Huffman based methods achieve state-of-the-art compression sizes, while still maintaining acceptable decoding speed.
2. The performance of Huffman-based method are close to the optimal.
HRNN Data Generation

- Collected ~32M raw voice queries during Feb. 22-28th 2016 from 2.5M unique viewers in the Comcast’s Xfinity X1 platform.

- Selected 45 seconds to sessionize queries, yielding ~20M sessions.

- The ground truth labels of sessions are collected by viewers’ watching behaviors after their queries, yielding 13M session-program pairs.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#sessions</th>
<th>#queries</th>
<th>Avg. session len</th>
<th>Avg. query len</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>126016</td>
<td>181058</td>
<td>1.44</td>
<td>2.34</td>
</tr>
<tr>
<td>Validation</td>
<td>28427</td>
<td>82828</td>
<td>2.91</td>
<td>2.30</td>
</tr>
<tr>
<td>Test</td>
<td>28173</td>
<td>82272</td>
<td>2.92</td>
<td>2.30</td>
</tr>
</tbody>
</table>

© Jinfeng Rao 2018
HRNN Training Details

• GloVe word embeddings
  • 1.8K/20.4K words were not found in the vocabulary
  • Unseen word embeddings were randomly initialized from [-0.05, 0.05]

• RMS-PROP algorithm for parameter updating
  • learning rate was set to 0.001 initially, then decreased a factor of 3 if validation set loss
doesn’t go down for three epoches.

• Categorical loss function was adopted

• At test time, we selected the model that obtained the highest P@1 accuracy at
validation set for evaluation.
Conclusion

• As the first time, we present a comprehensive voice search system that allows users talk to their TVs.

• We decompose the system into jointly learning three tasks: program prediction, intent classification and tagging.

• Experiments on large real datasets show multi-task learning can improve the performance than learning individually.

• Our work was also awarded by the 69th Emmy Award (2017) for the technical contribution in advancing television technologies.