YJTI at the NTCIR-13 STC Japanese Subtask

Dec. 7, 2017
Toru Shimizu
Overview
Retrieval or Generation

• Retrieval-based system
  – Effective if you have a good matching model and enough candidate responses
  – Pros
    • Human-written, fluent sentences for responses
    • The conversation can sometimes actually be interesting.
      – Hence more practical
  – Cons
    • Lack of flexibility
      – This can be mitigated with large amount of candidates and the variety in them.
      – 1.2M unique sentences in the training data
Architecture

- **DSSM (Deep Structured Semantic Model)**
  - Huang et al., 2013
  - A method for IR, query-document matching

\[
R_{\Theta}(Q, D) = \frac{z_Q^T z_D}{\|z_Q\| \|z_D\|}
\]

- **LSTM-DSSM**
  - Palangi et al., 2014
  - LSTM-RNN for generating query and document representations
The Overall Process: Three Stages

Model Training

- Train two models:
  - a comment encoder
  - a reply encoder

Reply Text Preparation and Indexing

- Preprocess the training data to obtain candidate replies
- Generate vector representations of the replies
- Build the reply index

Runtime

- Produce actual reply lists using the runtie system
Submissions

• Two runs:
  – YJTI-J-R1
    • Trained by Twitter conversation data
  – YJTI-J-R2
    • Trained mainly by Yahoo! Chiebukuro QA data

• The runtime system is the same.
• Only the models are different.
Runtime System
Runtime System Overview

Model training stage
- comment encoder model
- reply encoder model

Reply text preparation and indexing stage
- candidate replies
- reply encoder

Runtime stage
- query (comment)
- comment encoder
- comment vector
- retriever
- reply vectors
- top-200 replies
- ranker
- top-10 ranked replies

- data
- component
Runtime System Overview: Software Components

Model training stage
- comment encoder model
- reply encoder model

Reply text preparation and indexing stage
- candidate replies
- reply encoder

Runtime stage
- query (comment)
- comment encoder
- comment vector
- retriever
- reply vectors
- ranker
- top-200 replies
- top-10 ranked replies

- data
- component
Runtime System Overview: Data

Model training stage
- comment encoder model
- reply encoder model

Reply text preparation and indexing stage
- candidate replies
- reply encoder

Runtime stage
- query (comment)
- comment encoder
- comment vector
- retriever
- reply vectors
- top-200 replies
- ranker
- top-10 ranked replies

- data
- component
Runtime System Overview: The 1st Stage

**Model training stage**
- comment encoder model
- reply encoder model

**Reply text preparation and indexing stage**
- candidate replies
- reply encoder

**Runtime stage**
- query
  - (comment)
- comment encoder
- comment vector
- retriever
  - reply vectors
- ranker
- top-200 replies
- top-10 ranked replies
Runtime System Overview: The 2\textsuperscript{nd} Stage

\begin{itemize}
  \item **Model training stage**
    \begin{itemize}
      \item comment encoder model
      \item reply encoder model
    \end{itemize}
  
  \item **Runtime stage**
    \begin{itemize}
      \item query (comment)
      \item comment encoder
      \item comment vector
      \item retriever
      \item reply vectors
      \item top-200 replies
      \item ranker
      \item top-10 ranked replies
    \end{itemize}

  \item **Reply text preparation and indexing stage**
    \begin{itemize}
      \item candidate replies
      \item reply encoder
    \end{itemize}
\end{itemize}
Runtime System Overview: The 3rd Stage

Model training stage

- Comment encoder model
- Reply encoder model

Reply text preparation and indexing stage

- Candidate replies
- Reply encoder

Runtime stage

- Query (comment)
- Comment encoder
- Comment vector
- Retriever
- Reply vectors
- Top-200 replies
- Ranker
- Top-10 ranked replies
Indexer and Retriever

- Generate 1024-element representations of reply candidates by the reply encoder model
- NGT
  - Open source software for graph-based approximate similarity search over dense vectors
    - Developed by M. Iwasaki
    - [https://research-lab.yahoo.co.jp/software/ngt/](https://research-lab.yahoo.co.jp/software/ngt/)
- Retrieve the nearest 200 reply vectors from a given comment vectors
  - L2-distance, cosine similarity
- Return the list of their texts and metadata
• Three tiers for dealing with metadata matching: THEME, GENRE, and OTHER

**THEME**
The Theme is matched btw. the comment and a reply.
(At most 3)

**GENRE**
The Genre is matched btw. the comment and a reply
(At most 3)

**OTHER**
No metadata match.
(No limitation of number)

The final top-10 replies

• reply 1
• reply 2
• reply 3
• reply 4
• reply 5
• reply 6
• reply 7
• reply 8
• reply 9
• reply 10
Model, Data, and Training
Comment/Reply Encoder Model

- 3-layer LSTM RNN
  - Formulation: Graves, 2013
  - LSTM's hidden layer size: 1024 (for all the
  - Embedding layer size: 256
  - Representation size: 1024
Comment/Reply Encoder Model

- Training

Consider this as a classification problem and maximize the probability for the right choice over a given dataset

\[
P_\Theta(D_i^k | Q_i) = \frac{\exp(\beta R_\Theta(Q_i, D_i^k))}{\sum_{j=1}^{5} \exp(\beta R_\Theta(Q_i, D_i^j))}
\]
Comment/Reply Encoder Model

- Training cont'd

<table>
<thead>
<tr>
<th>run</th>
<th>model type</th>
<th>data name</th>
<th>records consumed</th>
</tr>
</thead>
<tbody>
<tr>
<td>YJTI-J-R1</td>
<td>DSSM</td>
<td>Twitter conversation</td>
<td>135.0M</td>
</tr>
<tr>
<td>YJTI-J-R2</td>
<td>LM</td>
<td>Y! Chiebukuro LM</td>
<td>171.5M</td>
</tr>
<tr>
<td>DSSM</td>
<td>Twitter conversation</td>
<td>85.8M</td>
<td></td>
</tr>
<tr>
<td>DSSM</td>
<td>Y! Chiebukuro QA</td>
<td>42.9M</td>
<td></td>
</tr>
</tbody>
</table>
## Data for Model Training

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>no. of records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twitter LM</td>
<td>posts</td>
<td>100.0M</td>
</tr>
<tr>
<td>Twitter conversation</td>
<td>pairs</td>
<td>65.1M</td>
</tr>
<tr>
<td>Y! Chiebukuro LM</td>
<td>posts</td>
<td>202.0M</td>
</tr>
<tr>
<td>Y! Chiebukuro QA</td>
<td>pairs</td>
<td>66.3M</td>
</tr>
</tbody>
</table>
Results
Analysis and Results

- Performances measured by the validation data

<table>
<thead>
<tr>
<th>matching task</th>
<th>YJTI-J-R1</th>
<th>YJTI-J-R2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twitter conversation</td>
<td>0.835</td>
<td>0.759</td>
</tr>
<tr>
<td>Chiebukuro QA</td>
<td>0.864</td>
<td>0.967</td>
</tr>
</tbody>
</table>
### Analysis and Results

- The official results under Rule-2

<table>
<thead>
<tr>
<th>metric</th>
<th>YJTI-J-R1</th>
<th>YJTI-J-R2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean $nG@1$</td>
<td>0.4171</td>
<td>0.4726</td>
</tr>
<tr>
<td>Mean $nERR@2$</td>
<td>0.4544</td>
<td>0.5288</td>
</tr>
<tr>
<td>Mean $Acc_{L2}@1$</td>
<td>0.1860</td>
<td>0.2040</td>
</tr>
<tr>
<td>Mean $Acc_{L2}@2$</td>
<td>0.1490</td>
<td>0.2030</td>
</tr>
<tr>
<td>Mean $Acc_{L1,L2}@1$</td>
<td>0.6100</td>
<td>0.7200</td>
</tr>
<tr>
<td>Mean $Acc_{L1,L2}@2$</td>
<td>0.5750</td>
<td>0.6900</td>
</tr>
</tbody>
</table>
Conclusions

- Effectiveness of the overall approach:
  - Retrieval-based system
  - DSSM-like matching powered by LSTM-RNNs trained over a large amount of linguistic resources
- Social QA data was surprisingly useful for modeling topic-oriented conversations seen in this Yahoo! News comments data