Knowledge-Grounded Dialogue Generation with Pre-trained Language Models

Xue Liang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, Rui Yan
EMNLP 2020
Task and Method

- **Knowledge-aware Dialogue Generation:**
  - **Input:**
    - Conversation Context: The previous sentences in a conversation
    - A Document: Background knowledge that the next sentence in the conversation should be generated with respect to it
  - **Output:**
    - A text to complete the conversation

- **Method:**
  - The conversation and the document are fed into GPT-2 to generate the response

- **Challenge:**
  - The document is long and it will exceed the GPT-2 limit as the input document
  - Part of the document should be used to represent the knowledge
  - No label is available for knowledge selection from the document
Model
Task

- **Dataset:** \( \{(U_i, D_i, r_i)\}_{i=1}^{N} \)
- **Goal:** \( P(r|U, D; \theta) \)
- With GPT the goal is defined in autoregressive manner:
  \[
P(r|U, D; \theta) = P(r|g(U, D); \theta)
  = \prod_{t=1}^{l_r} P(r_t|g(U, D), r_{1:t-1}; \theta),
\]
- Optimize \( g(U,D) \) and GPT-2 parameters
- No label for \( g(U,D) \)
Knowledge Encoder

- Given $U = (u_1, \ldots, u_n)$ and $D = (d_1, \ldots, d_m)$ construct the sequence $S = (S_1, \ldots, S_m)$ where (all words of U concatenated with each sentence in D):

  $$S_i = [CLS]w_1^u \ldots w_{l_u}^u[SEP]w_1^d \ldots w_{l_d}^d[SEP]$$

- Using BERT we obtain the representation of CLS for each sentence in the document to have the following sequence of vectors:

  $$E = (e_1, \ldots, e_m)$$
Sequential Knowledge Selector

- A sequential process in which a randomly initialized vector is the initial state, using it and the knowledge encoded in $E$, a sentence is selected and the state is updated until the termination criterion is matched (Max length or Special Token)

$$P(d_t|U, d_{j_1:t-1}) = \exp(\alpha_{t,i})/\sum_i \exp(\alpha_{t,i})$$

$$\alpha_{t,i} = v^\top \tanh(W_e e_i + W_s s_t + b),$$

- At each step the $j_t$ sentence is selected: $\arg\max_{i \in \{1, \ldots, m\}} P(d_t|U, d_{j_1:t-1})$
- Update state:

$$s_{t+1} = \text{LSTM}(e_{j_t}, s_t)$$

$$U \cup D'$$
Pre-train \( g(U,D) \) on Pseudo-Label

- Sort the sentences in \( D \) with respect to their similarity to the gold response:
  \[
  \text{Sim}(d_t, r)
  \]

- Truncate the sorted document in a way that maximizes the similarity between the truncated document and the response:
  \[
  \bar{D} = \{d_{j_1}, \ldots, d_{j_{\bar{m}}}\},
  \bar{m} = \arg\max_t (\text{Sim}(d_{j_{1:t}}, r)),
  \]

- Use the new constructed pseudo-labels to train \( g(U,D) \) and GPT-2 using MLE
  \[
  \mathcal{D}_K = \{(U_i, D_i, \bar{D}_i)\}_{i=1}^N
  \]
  \[
  \mathcal{D}_G = \{(U_i, \bar{D}_i, r_i)\}_{i=1}^N
  \]
Training $g(U,D)$

- Use the similarity between response generated by GPT-2 using the document constructed by $g(U,D)$ as the supervision signal
- Increase probability of constructed documents that result in better responses

\[
\mathcal{L}_K = -\frac{1}{N} \sum_{i=1}^{N} \left( \tilde{R}_i \sum_{t=1}^{|	ilde{D}_i|} \log P(d_{i,j_t} | U_i, d_{i,j_1:t-1}) \right)
\]

\[
\tilde{R}_i = R(\tilde{D}_i) - b,
\]

\[
R(\tilde{D}_i) = \text{Sim}(r'_i, r_i)
\]

\[
b = \frac{\sum_{i=1}^{N} R(\tilde{D}_i)}{N}
\]
Training GPT-2

- Mix the Pseudo-label and the constructed documents as the input to the GPT-2 and fine-tuned it in an autoregressive manner

\[
\mathcal{L}_G = -\frac{1}{N} \sum_{i=1}^{N} \left( z_i \sum_{t=1}^{l_r} \log P(r_{i,t}|U_i, \bar{D}_i, r_{i,1:t-1}) 
+
(1 - z_i) \sum_{t=1}^{l_r} \log P(r_{i,t}|U_i, D'_i, r_{i,1:t-1}) \right),
\]

- Lower the coefficient of loss for the pseudo-label document at each training step
Algorithm 1 Optimization Algorithm

1: **Input**: Training data \( D \), pre-trained GPT-2, initial curriculum rate \( p_0 \), exponential decay constant \( \lambda \), maximum step \( M \).
2: Construct \( \mathcal{D}_K \) and \( \mathcal{D}_G \).
3: Optimize \( g(U, D) \) and GPT-2 using MLE on \( \mathcal{D}_K \) and \( \mathcal{D}_G \) respectively.
4: for \( m \leftarrow 1 \) to \( M \) do
5: Sample a mini-batch \( \{(U_i, D_i, r_i)\} \) from \( D \).
6: Update the parameters of \( g(U, D) \) based on Eq.6. \( \triangleright \) the Reinforcement Step.
7: Sample \( \{z_i\} \) from a Bernoulli distribution parameterized by \( p \), where \( p = p_0 e^{-\lambda m} \).
8: Update the parameters of the GPT-2 model based on Eq.7. \( \triangleright \) the Curriculum Step.
9: end for
10: return \( g(U, D) \) and GPT-2.
## Results

<table>
<thead>
<tr>
<th>Models</th>
<th>Test Seen</th>
<th></th>
<th></th>
<th></th>
<th>Test Unseen</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PPL</td>
<td>F1</td>
<td>Average</td>
<td>Extrema</td>
<td>Greedy</td>
<td>PPL</td>
<td>F1</td>
<td>Average</td>
</tr>
<tr>
<td>TMN (Dinan et al., 2019)</td>
<td>66.5</td>
<td>15.9</td>
<td>0.844</td>
<td>0.427</td>
<td>0.658</td>
<td>103.6</td>
<td>14.3</td>
<td>0.839</td>
</tr>
<tr>
<td>ITDD (Li et al., 2019)</td>
<td>17.8</td>
<td>16.2</td>
<td>0.841</td>
<td>0.425</td>
<td>0.654</td>
<td>44.8</td>
<td>11.4</td>
<td>0.826</td>
</tr>
<tr>
<td>SKT* (Kim et al., 2020)</td>
<td>52.0</td>
<td>19.3</td>
<td>0.846</td>
<td>0.440</td>
<td>0.665</td>
<td>81.4</td>
<td>16.1</td>
<td>0.839</td>
</tr>
<tr>
<td>DRD (Zhao et al., 2020)</td>
<td>19.4</td>
<td>19.3</td>
<td>0.852</td>
<td>0.452</td>
<td>0.674</td>
<td>23.0</td>
<td>17.9</td>
<td>0.849</td>
</tr>
<tr>
<td>SKT+GPT-2*</td>
<td>17.6</td>
<td>20.3</td>
<td>0.866</td>
<td>0.460</td>
<td>0.679</td>
<td>23.7</td>
<td>17.8</td>
<td>0.860</td>
</tr>
<tr>
<td>GPT-2_{trunc}</td>
<td>14.6(2.2)</td>
<td>18.7(0.7)</td>
<td>0.864(0.002)</td>
<td>0.451(0.006)</td>
<td>0.674(0.004)</td>
<td>16.9(3.1)</td>
<td>18.3(0.6)</td>
<td>0.862(0.002)</td>
</tr>
<tr>
<td>KnowledGPT</td>
<td>19.2</td>
<td>22.0</td>
<td>0.872</td>
<td>0.463</td>
<td>0.682</td>
<td>22.3</td>
<td>20.5</td>
<td>0.870</td>
</tr>
</tbody>
</table>
Thanks