9. End-to-end systems (2)

Ondřej Dušek & Vojtěch Hudeček

http://ufal.cz,npfl099

12. 12. 2019
Sequicity + explicit state

- the same context encoder as Sequicity

- state decoder:
  - individual slots decoded separately
    - prevents decoding invalid states
  - the same decoder run for each slot
  - informal:
    - decode values, seq2seq way
  - requestable:
    - classify 0/1 if user requested

- response generation:
  - 1\textsuperscript{st} step – classify which slots to include
  - then seq2seq delexicalized generation

$\text{(Shu et al., 2019) https://www.aclweb.org/anthology/W19-5922/}$
Structured Fusion Nets: End-to-end on top of individual modules

- **1**\(^{st}\) step: optimize separate NLU/DM/NLG modules
- **2**\(^{nd}\) step: optimize end-to-end network over the outputs of modules

(Mehri et al., 2019)
https://www.aclweb.org/anthology/W19-5921/
Structured Fusion Nets

- high-level module on top of NLU/DM/NLG modules works better than just joining, even with joint optimization
- modules can be fine-tuned (end-to-end differentiable)
  - this helps in either case (with modules only or high-level network)
  - multi-task learning doesn’t help more (alternating fine-tuning with module-specific tasks)
- RL: only high-level
  - this way the base generator maintains fluency
  - BLEU OK & success much higher

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
<th>Inform</th>
<th>Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised Learning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seq2Seq (Budzianowski et al., 2018)</td>
<td>18.80</td>
<td>71.29%</td>
<td>60.29%</td>
</tr>
<tr>
<td>Seq2Seq w/ Attention (Budzianowski et al., 2018)</td>
<td>18.90</td>
<td>71.33%</td>
<td>60.96%</td>
</tr>
<tr>
<td>Seq2Seq (Ours)</td>
<td>20.78</td>
<td>61.40%</td>
<td>54.50%</td>
</tr>
<tr>
<td>Seq2Seq w/ Attention (ours)</td>
<td>20.36</td>
<td>66.50%</td>
<td>59.50%</td>
</tr>
<tr>
<td>Naive Fusion (Zero-Shot)</td>
<td>7.55</td>
<td>70.30%</td>
<td>36.10%</td>
</tr>
<tr>
<td>Naive Fusion (Fine-tuned Modules)</td>
<td>16.39</td>
<td>66.50%</td>
<td>59.50%</td>
</tr>
<tr>
<td>Multitasking</td>
<td>17.51</td>
<td>71.50%</td>
<td>57.30%</td>
</tr>
<tr>
<td>Reinforcement Learning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Structured Fusion (Frozen Modules)</td>
<td>17.53</td>
<td>65.80%</td>
<td>51.30%</td>
</tr>
<tr>
<td>Structured Fusion (Fine-tuned Modules)</td>
<td>18.51</td>
<td>77.30%</td>
<td>64.30%</td>
</tr>
<tr>
<td>Structured Fusion (Multitasked Modules)</td>
<td>16.70</td>
<td>80.40%</td>
<td>63.60%</td>
</tr>
<tr>
<td>Structured Fusion (Frozen Modules) + RL</td>
<td>16.34</td>
<td>82.70%</td>
<td>72.10%</td>
</tr>
</tbody>
</table>

NPFL099 L9 2019

MultiWOZ (multi-domain data)
DA-based self-attention

- DAs represented as a graph
  - 3-level: domains – intents – slots

- ignores DB & tracker
  - uses ground truth from data

- NLU:
  - BERT over all history tokens
  - feed-forward/attention + sigmoid
  - predict domains-intents-slots graph

- Decoder: modified self-attention
  - optimized separately
  - gated sum instead of concatenation
    - gating follows predicted DA graph
  - delexicalized – DB & tracker provide lexicalization

- Supervised learning only

See https://www.aclweb.org/anthology/P19-1360 for more details.
Latent Action RL

- Making system actions latent, learning them implicitly
- Like a VAE, but **discrete latent space** here ($M^k$-way variables)
  - using Gumbel-Softmax trick for backpropagation
  - using Full ELBO (KL vs. prior network) or “Lite ELBO” (KL vs. uniform $1/k$)
- RL over latent actions, not words
  - avoids producing disfluent language
  - “fake RL” based on supervised data
    - generate outputs, but use original contexts from a dialogue from training data
    - success & RL updates based on generated responses
  - on par with Structured Fusion Nets (slightly higher success, lower BLEU)
- again, ignores DB & belief tracking

(Zhao et al., 2019)
https://www.aclweb.org/anthology/N19-1123
“Hello, it’s GPT-2 – How can I help?”

- Simple adaptation of the GPT pretrained LM
  - system/user embeddings
    - added to Transformer positional embs. & word embs.
  - training to generate as well as classify utterances (good vs. random)
    - all supervised

- Again, no DB & belief tracking
  - using gold-standard belief & DB, no way of updating belief

(Budzianowski & Vulić, 2019)
https://www.aclweb.org/anthology/D19-5602
Soft DB Lookups

• incorporating NLU/tracker uncertainty into DB results
• making the system fully differentiable
  • but less interpretable
• DB output = distribution over all items
  • plain MLE estimation: \[ p(\text{row } i) = \prod_{\text{slots } j} \frac{p(v=j)}{\# \text{ of } v's \text{ in table}} \] if \( j \) specified & in table
  \[ \frac{1}{\# \text{ rows (uniform)}} \] otherwise
• NLU(trackers – per-slot GRUs + softmaxes
  • input: counts of n-grams
• policy = GRU + softmax
• trained by RL
  • shown to outperform hard DB on a movie domain

(Dinghra et al., 2017)
https://www.aclweb.org/anthology/P17-1045
Key-value Retrieval Nets

- using attention to model DB access
- LSTM encoder, no specific tracker/NLU
- DB in a “key-value” format
  - subject-relation-object (subject-property-value)
  - dinner-time-8pm
  - key = subject + relation
  - value = subject_relation
    - i.e. delexicalized values
- generator: seq2seq with 2 attentions
  - over inputs (as usual)
  - over keys in the DB – increases generator output probs. of DB values
    - doesn’t change probs. of regular vocabulary
- supervised training, better than seq2seq/copy
DB Table Attention

- **Input/State tracking:**
  - LSTM encoder over whole history
  - slot states = per-slot attention over encoder

- **DB representation:**
  - **cell embedding** = column/slot emb. & value emb. + linear + tanh
  - **row similarity** with dialogue state: \( \sum_{\text{slots}} \text{cell emb} \cdot \text{slot state} \)
  - **info matrix:** softmax-weighted sum of row similarities
  - **memory:** weights \( \cdot \) (slot states & info matrix)

- **Response decoder:** seq2seq + “copy”
  - with attentions over input & memory
  - copying: choosing to generate a slot & filling in value based on info matrix

(Wen et al., 2018)
http://arxiv.org/abs/1806.04441
Memory networks

- not a full dialogue model, just ranker of candidate replies
- no explicit modules
- based on attention over history
  - sum of bag-of-words embeddings
  - added features (user/system, turn no.)
  - weighted match against last user input (dot + softmax)
  - linear transformation to produce next-level input
- last input matched (dot + softmax) against a pool of possible responses

\[ o = R \sum p_i m_i \]

within a few iterations

\[ o = \text{softmax} \quad \sum_{i} p_i m_i \]

for multiple steps
**Mem2Seq: memory nets & pointer-generator**

- “standard” MemNN encoder:
  - special memory:
    - token-level dialogue history (whole history concatenated, no hierarchy)
    - with added turn numbers & user/system flags
    - DB tuples (sums of subject-relation-object)
    - “sentinel” (special token)

- decoder: MemNN over GRU
  - GRU state is MemNN initial query
  - last level attention is copy pointer
  - if copy pointer points at sentinel, generate from vocabulary
    - copies whenever it can
  - vocabulary distribution comes from 1st level of memory + GRU state
    - linear transform + softmax

---

(Madotto et al., 2018)
https://www.aclweb.org/anthology/P18-1136
Mem2Seq attention visualization

discussion...
Few-shot dialogue generation

• Domain transfer:
  • source domain training dialogues
  • target domain “seed responses” with annotation

• encoding all into latent space
  • keeping response & annotation encoding close
  • keeping context & response encoding close
  • decoder loss + matching loss

• encoder: HRE (hierarchical RNN)

• decoder: copy RNN (with sentinel)
  • “copy unless attention points to sentinel” (see Mem2Seq)

• DB queries & results treated as responses/inputs
  • DB & user part of environment

(Zhao & Eskenazi, 2018) http://aclweb.org/anthology/W18-5001
Few-shot & Latent Actions

- Latent discrete encoder-decoder
  - discrete VAE for dialogue turns
  - discrete Variational Skip Thought
    - predicting next turn
  - trained jointly

- Full model:
  - LAED to predict next action
  - DI-VAE for user input representation
  - HRED with ELMo
  - KVRET-like DB representation
    - DB is treated as part of context
  - decoder: same as previous
    - copy with sentinel
  - uses NER/entity linking instead of handcrafted annotations

(Zhao et al., 2018) [http://aclweb.org/anthology/P18-1101](http://aclweb.org/anthology/P18-1101)
Summary

• RL for end-to-end systems helps if it’s not on token level
  • RL over latent system actions (embeddings / discrete)

• Pretrained LMs can work as end-to-end DS

• Soft DB lookups – making the whole system differentiable
  • “transparent” (directly based on tracker)
  • attention/memory nets (multi-hop attention)

• Few-shot: lot of autoencoding
Thanks

Contact us:
  odusek@ufal.mff.cuni.cz
  hudecek@ufal.mff.cuni.cz
(or on Slack)

Get these slides here:
  http://ufal.cz/npfl099

References/Inspiration/Further:
• Gao et al. (2019): Neural Approaches to Conversational AI: https://arxiv.org/abs/1809.08267