NPFL099 Statistical Dialogue Systems

9. End-to-end Task-Oriented Systems

http://ufal.cz/npfl099

Vojtěch Hudeček, Ondřej Dušek & Zdeněk Kasner

28. 11. 2022
End-to-end dialogue systems

• **End-to-end = represent the whole system as one neural net**
  • sometimes, just some of the components can be joined
    • e.g. just NLU + tracker + policy, NLG excluded

• **Pros & cons:**
  • Traditional architecture – separate components:
    • more flexible (replace one, keep the rest)
    • error accumulation
    • improved components don’t mean improved system
    • possibly joint optimization by RL
    • explainability

• **End-to-end:**
  • joint optimization by backprop
    • if fully differentiable
  • still can work via RL (with supervised initialization)
  • architectures still decompose into (some of) original DS components
    • and often still need DA-level annotation
Training end-to-end systems

• Supervised
  • sometimes components still trained separately
  • e.g. hard knowledge base lookup
  • sometimes all in one
  • can’t learn from users
  • problems with train-test mismatch

• RL
  • can learn from users, can learn all-in-one
  • doesn’t work great if done on word-level
  • RL doesn’t care about fluency/naturalness
  • either avoid word-level, or mix with supervised
Supervised with component nets

- “seq2seq augmented with history (tracker) & DB”
- end-to-end, but has components
  - LSTM “intent network”/encoder (latent intents)
  - CNN+RNN belief tracker (prob. dist. over slot values)
    - lexicalized + delexicalized CNN features
    - turn-level RNN (output is used in next turn hidden state)
  - MLP policy (feed-forward)
  - LSTM generator
    - conditioned on policy output, delexicalized
  - DB: rule-based, takes most probable belief values
    - creates boolean vector of selected items
    - vector compressed to 6-bin 1-hot (no match, 1 match… >5 matches) on input to policy
    - 1 matching item selected at random & kept for lexicalization after generation

(Wen et al., 2017)
https://www.aclweb.org/anthology/E17-1042
Supervised with component nets

- belief tracker trained separately
- rest trained by cross-entropy on generator outputs
- data: CamRest676, collected by crowdsourcing/Wizard-of-Oz
  - workers take turns to be user & system, always just add 1 turn

<table>
<thead>
<tr>
<th>Encoder</th>
<th>Tracker</th>
<th>Decoder</th>
<th>Match(%)</th>
<th>Success(%)</th>
<th>T5-BLEU</th>
<th>T1-BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lstm</td>
<td>lstm</td>
<td>lstm</td>
<td>-</td>
<td>-</td>
<td>0.1650</td>
<td>0.1718</td>
</tr>
<tr>
<td>lstm</td>
<td>turn recurrence</td>
<td>lstm</td>
<td>-</td>
<td>-</td>
<td>0.1813</td>
<td>0.1861</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variant</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>lstm</td>
<td>rnn-cnn, w/o req.</td>
<td>lstm</td>
<td>89.70</td>
<td>30.60</td>
<td>0.1769</td>
<td>0.1799</td>
</tr>
<tr>
<td>cnn</td>
<td>rnn-cnn</td>
<td>lstm</td>
<td>88.82</td>
<td>58.52</td>
<td>0.2354</td>
<td>0.2429</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Full model w/ different decoding strategy</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>lstm</td>
<td>lstm</td>
<td>lstm</td>
<td>86.34</td>
<td>75.16</td>
<td>0.2184</td>
<td>0.2313</td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn</td>
<td>+ weighted</td>
<td>86.04</td>
<td>78.40</td>
<td>0.2222</td>
<td>0.2280</td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn</td>
<td>+ att.</td>
<td>90.88</td>
<td>80.02</td>
<td>0.2286</td>
<td>0.2388</td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn</td>
<td>+ att. + weighted</td>
<td>90.88</td>
<td>83.82</td>
<td>0.2304</td>
<td>0.2369</td>
</tr>
</tbody>
</table>

RNN + CNN + FC | seq gen + classif

(Wen et al., 2017)
https://www.aclweb.org/anthology/E17-1042
Hybrid Code Networks

• partially handcrafted, designed for little training data
  • with Alexa-type assistants in mind

• **Utterance representations:**
  • bag-of-words binary vector
  • average of word embeddings

• **Entity extraction & tracking**
  • domain-specific NER
  • handcrafted tracking
  • returns **action mask**
    • permitted actions in this step (e.g. can’t place a phone call if we don’t know who to call yet)
    • return (optional) handcrafted **context features** (various flags)

• **LSTM state tracker** (output retained for next turn)
  • i.e. no explicit state tracking, doesn’t need state tracking annotation

(Williams et al., 2017)
http://arxiv.org/abs/1702.03274
Hybrid Code Networks

- **feed-forward policy** – produces probability distribution over actions
  - mask applied to outputs & renormalized → choosing action = output template
- **handcrafted fill-in for entities**
  - takes features from ent. extraction
  - ~learned part is fully delexicalized
- **actions** may trigger API calls
  - APIs can return feats for next step
- **training** – supervised & RL:
  - SL: beats a rule-based system with just 30 training dialogues
  - RL: REINFORCE with baseline
  - RL & SL can be interleaved
- **extensions**: better input than binary & averaged embeddings

(Shalyminov & Lee, 2018)  
https://arxiv.org/abs/1811.12148  
(Marek, 2019)  
http://arxiv.org/abs/1907.12162
Sequicity: Two-stage Copy Net – fully seq2seq-based

• less hierarchy, simpler architecture
  • no explicit system action – direct to words
  • still explicit dialogue state
  • KB is external (as in most systems)

• seq2seq (LSTM) + copy (pointer-generator):
  • **encode**: previous dialogue state
    + prev. system response
    + current user input
  • **decode new state** first
    • attend over whole encoder
  • **decode system output** (delexicalized)
    • attend over state only
    + use KB (one-hot vector added to each generator input)
      • KB: 0/1/more results – vector of length 3

(Lei et al., 2018) [https://www.aclweb.org/anthology/P18-1133](https://www.aclweb.org/anthology/P18-1133)
• Simple adaptation of the GPT-2 pretrained LM
  • only model change: system/user embeddings
    • added to Transformer positional embs. & word embs.
  • GPT-2 is decoder-only: “encoding” here means **force-decoding**
    • pass input through all layers but ignore the softmax next-token prediction, feed our own tokens
  • training to generate as well as classify utterances (good vs. random), all supervised
• no DB & belief tracking – gold-standard belief & DB used, no updates (see →)

(售iaznowski & Vulić, 2019)
https://www.aclweb.org/anthology/D19-5602
Real stuff with GPT-2: SOLOIST, SimpleTOD, NeuralPipeline, UBAR

- basically Sequicity over GPT-2: decode belief state, consult DB, decode response
  - history, state, DB results/system action – all recast as sequence
  - finetuning on dialogue datasets

- extensions:
  - specific user/system embeddings (NP)
  - additional training (SOLOIST)
    - not just word-level generation (as GPT-2 default)
    - contrastive objective: detecting fake belief/fake response from real ones
  - explicit system actions (SimpleTOD, UBAR)
    - one more decoding step
    - Context includes dialogue states (UBAR)

(Ham et al., 2020)  https://www.aclweb.org/anthology/2020.acl-main.54
(Yang et al., 2021)  http://arxiv.org/abs/2012.03539
AuGPT: our take on this

• similar to Soloist:
  • “GPT-2 based Sequicity”
  • 1. encode context & user utterance
  • 2. decode belief state
  • 3. query DB
  • 4. encode results
  • 5. decode response
  • consistency auxiliary task

• for robustness & diversity:
  • input data augmentation via backtranslation
  • unlikelihood training (penalize repeated tokens)
  • nucleus sampling (cover ≥ 0.9 probability)

again, “encode” with GPT-2 means force-decode
(ignore the softmax, feed your own tokens)

http://arxiv.org/abs/2102.05126
(Kulhánek et al., 2021)
SOLOIST/AuGPT: Consistency task

- **Additional training task** – generating & classifying at the same time
  - additional classification layer on top of last decoder step logits
  - incurs additional loss, added to generation loss
- **Aim: robustness** – detecting problems
  - ½ data artificially **corrupted** – state or target response don’t fit context
  - SOLOIST: corrupted state sampled randomly
  - **AuGPT**: corrupted state sampled from the same domain – harder!

- **Context**: I want a cheap Italian restaurant {price range = cheap, food = Italian}
  - **State**: ok which area?
  - **Response**: thanks, goodbye!
  - **Consistent?**: ✗

- **Context**: I want a cheap Italian restaurant {destination = Cambridge, leave at = 19:00}
  - **State**: ok which area?
  - **Response**: ✗
  - **Consistent?**: ✗
    - bad state (same domain)

- **Context**: I want a cheap Italian restaurant {area = north, food = Chinese}
  - **State**: ok which area?
  - **Response**: ✗
  - **Consistent?**: ✗
    - bad state
MinTL: Diff dialogue states

- 2-step decoding, same as ↑
  - based on T5 or BART here
  - explicit 2 decoders
    (for state, for response)
- “Levenshtein states”
  - don’t decode full state each time
- **just decode a diff**
  (“Levenshtein distance from previous”)  
- better consistency over dialogue

1. encode previous state & context
2. obtain diffs from state annotation
3. decode diffs
4. update state based on decoded diff
5. DB queried based on updated state
6. response decoder starting token = # of DB results

---

(Lin et al., 2020)
Few-shot dialogue generation

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

- 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
• 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
  • corpus-based RL – “faking it” on supervised data
    • generate outputs, but use original contexts from a dialogue from training data
    • success & RL updates based on generated responses

• ignores DB & belief tracking
  • takes gold annotation from data (assumes external model for this)
LAVA: Latent Actions with VAE pretraining

- kinda combination of two previous
- **discrete latent space** for actions
- multi-step training scenario:
  1) **autoencode** responses into latent space
  2) **supervised** training for response generation via the latent space
  3) **RL** over the latent actions
     - same “fake RL” as previous
- options to join autoencoding & response generation
  a) **KL loss** – don’t go too far from autoencoding in latent space
  b) multi-task training (go back to autoencoding once in a while)
- again, assumes gold state & DB
Better RL: HDNO & JOUST

- **HDNO**: 2-level hierarchical RL
  - top level: (latent) actions
  - bottom level: words
  - LM rewards on word level (for fluency)
  - separate updates on both levels (avoid aiming at a moving target)
  - “fake” corpus-based RL (as previous)

- **JOUST**: real RL with a user simulator
  - system & sim. share architecture
    - joint context encoder
  - system: additional state tracker
  - interaction on utterance level
  - supervised pretraining

---


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

single step of the loop

linear transform

matrix product (a.k.a. attention)

weight match against last user input (dot + softmax)

linear transformation to produce next-level input

predicted answer

response candidates

NPFL099 L9 2022

(Sukhbaatar et al., 2015) http://arxiv.org/abs/1503.08895
(Bordes et al., 2017) http://arxiv.org/abs/1605.07683
Mem2Seq: Memory nets + pointer-generator
= soft DB lookups directly in the model

- “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
attention weights at individual word generation steps

Note: some DB entries were omitted for readability

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

- End-to-end = single network for NLU/tracker + DM + (sometimes) NLG
  - networks may decompose to components + need dialogue state annotation
  - joint training by backprop (if differentiable)
- Hybrid Code Nets – partially handcrafted, but end-to-end
- Two-stage copy net – 2-step decoding: dialogue state, then response
  - Sequicity – LSTM seq2seq
  - GPT-2-based systems – same idea, just with pretrained LMs
- Discrete latent action space – learning w/o action annotation
- RL optimization
  - corpus-based “fake RL” on training data (no simulator needed)
  - without NLG (over actions) or hierarchical
- Mem2Seq: Soft DB lookups – making the whole system differentiable
Contact us:
https://ufaldsg.slack.com/
{odusek,hudecek,kasner}@ufal.mff.cuni.cz
Skype/Zoom/Troja (by agreement)

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

No labs today
See you next week