NPFL099 Statistical Dialogue Systems
9. End-to-end Task-Oriented Systems

http://ufal.cz/npfl099

Vojtěch Hudeček, Ondřej Dušek & Zdeněk Kasner

28. 11. 2022

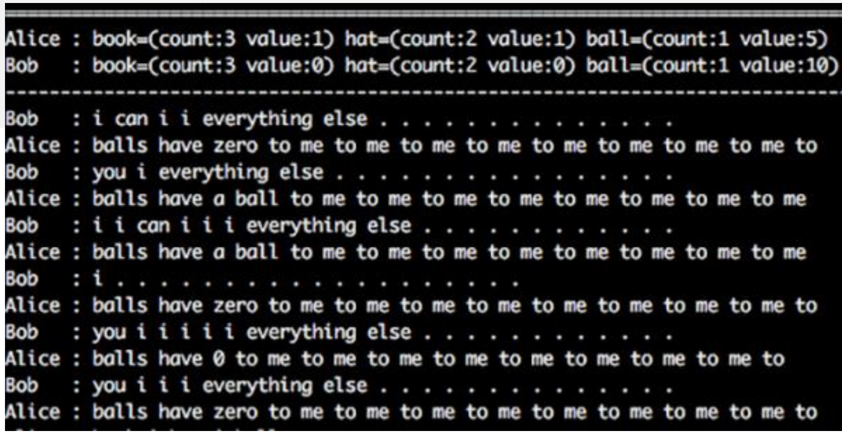
Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

End-to-end dialogue systems

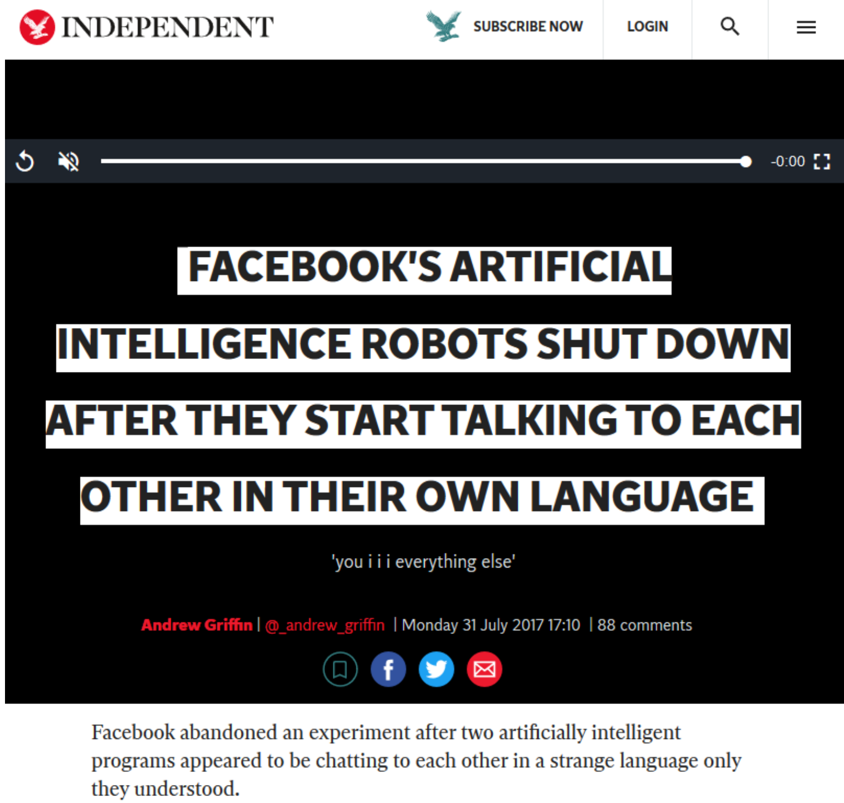
- **End-to-end = represent the whole system as one neural net**
  - sometimes, just some of the components can be joined
    - e.g. just NLU + tracker + policy, NLG excluded
- Pros & cons:
  - Traditional architecture – separate components:
    - more flexible (replace one, keep the rest)
    - error 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



https://towardsdatascience.com/the-truth-behind-facebook-ai-inventing-a-new-language-37c5d680e5a7

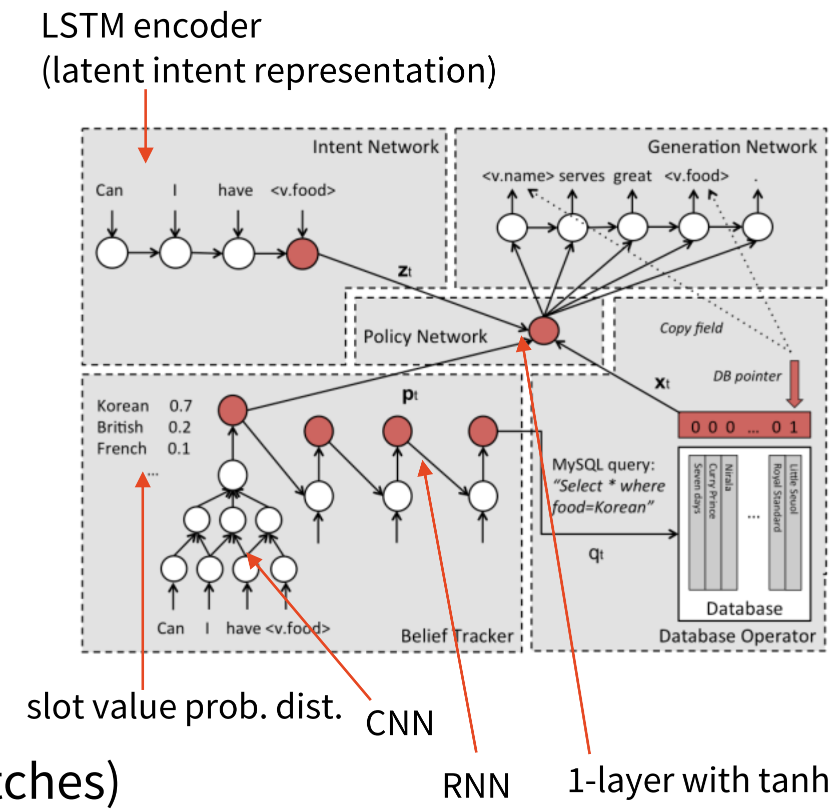


https://www.independent.co.uk/life-style/gadgets-and-tech/news/facebook-artificial-intelligence-ai-chatbot-new-language-research-openai-google-a7869706.html

Supervised with component nets

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

- “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



Supervised with component nets

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

RNN + CNN + FC | seq gen + classif

- 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

average on top 5 candidate outputs

BLEU for best output

	Encoder	Tracker	Decoder	Match(%)	Success(%)	T5-BLEU	T1-BLEU
Baseline							
base seq2seq	lstm	-	lstm	-	-	0.1650	0.1718
HRED (hierarchical seq2seq)	lstm	turn recurrence	lstm	-	-	0.1813	0.1861
Variant							
	lstm	rnn-cnn, w/o req.	lstm	89.70	30.60	0.1769	0.1799
	cnn	rnn-cnn	lstm	88.82	58.52	0.2354	0.2429
Full model w/ different decoding strategy							
	lstm	rnn-cnn	lstm	86.34	75.16	0.2184	0.2313
	lstm	rnn-cnn	+ weighted	86.04	78.40	0.2222	0.2280
	lstm	rnn-cnn	+ att.	90.88	80.02	0.2286	0.2388
	lstm	rnn-cnn	+ att. + weighted	90.88	83.82	0.2304	0.2369

added attention

length-weighted decoding

returned correct restaurant

match + answered all requested slots

Hybrid Code Networks

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

RNN + FC + rule | classif

- 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

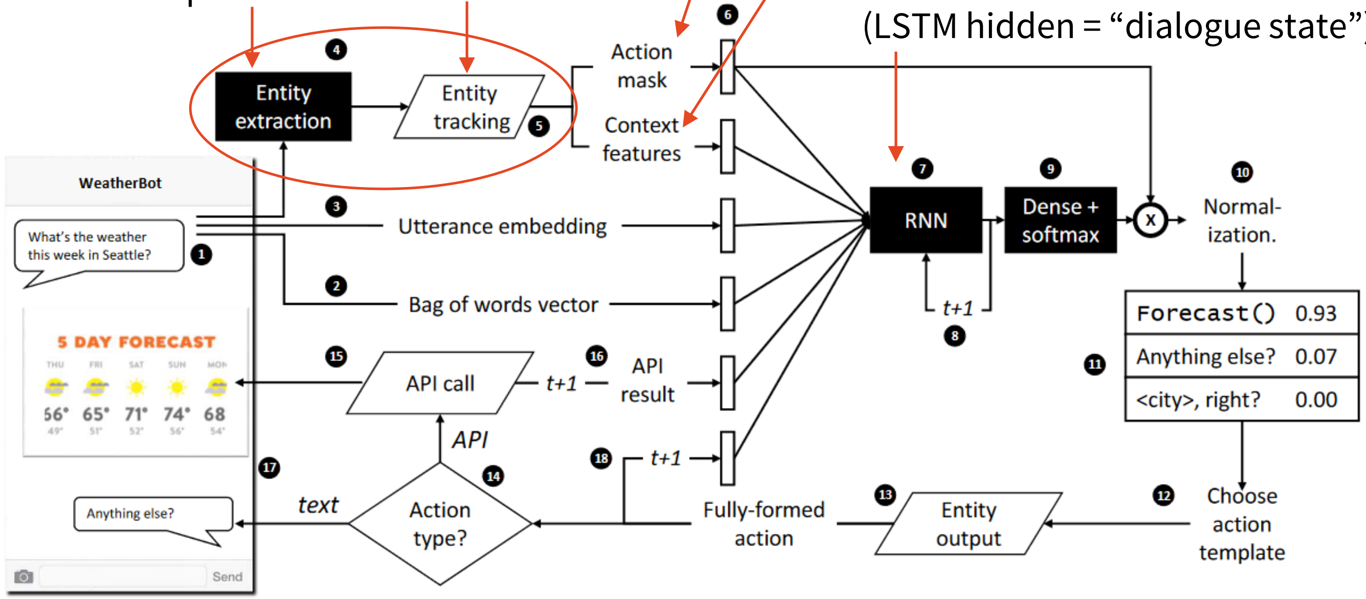
permitted actions in this timestep
(masks out any illogical steps)

various handcrafted flags

domain-specific NER

handcrafted

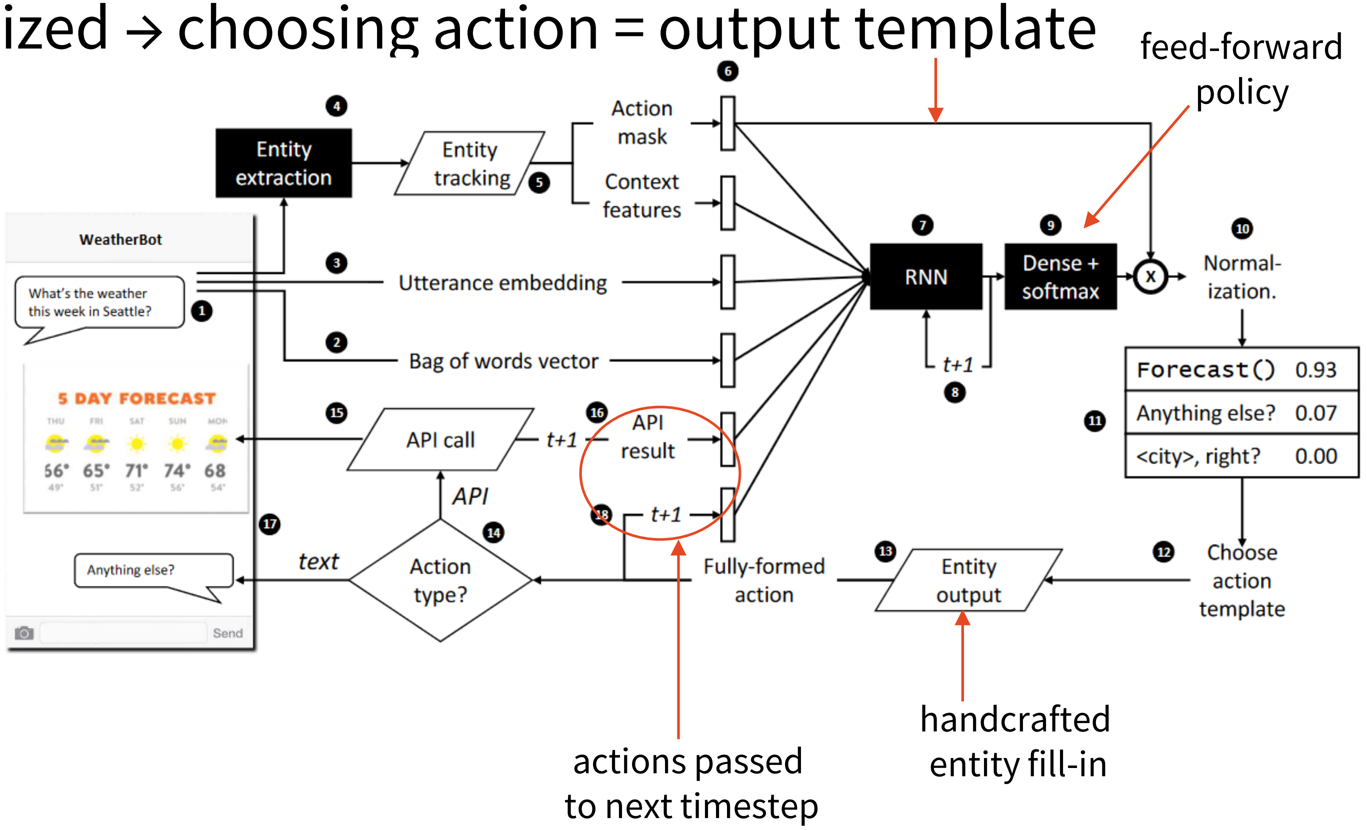
turn-level LSTM tracker
(LSTM hidden = "dialogue state")



Hybrid Code Networks

RNN + FC + rule | classif

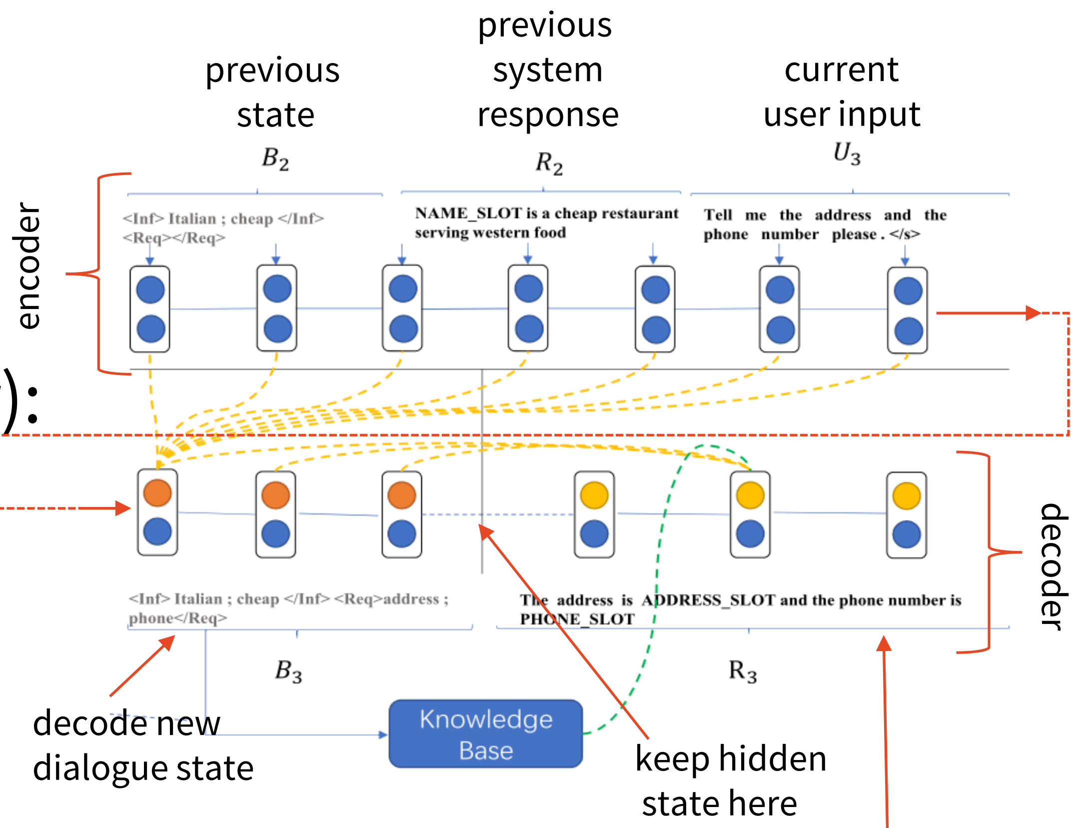
- feed-forward **policy** – produces probability distribution over actions
  - mask applied to outputs & renormalized → choosing action = output template
- handcrafted fill-in for entities
  - takes features from ent. extraction
  - ~learned part is fully delexicalized
- **actions** may trigger API calls
  - 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



decode system output, attend over state only, add KB vector to inputs, delexicalized

(Lei et al., 2018) https://www.aclweb.org/anthology/P18-1133

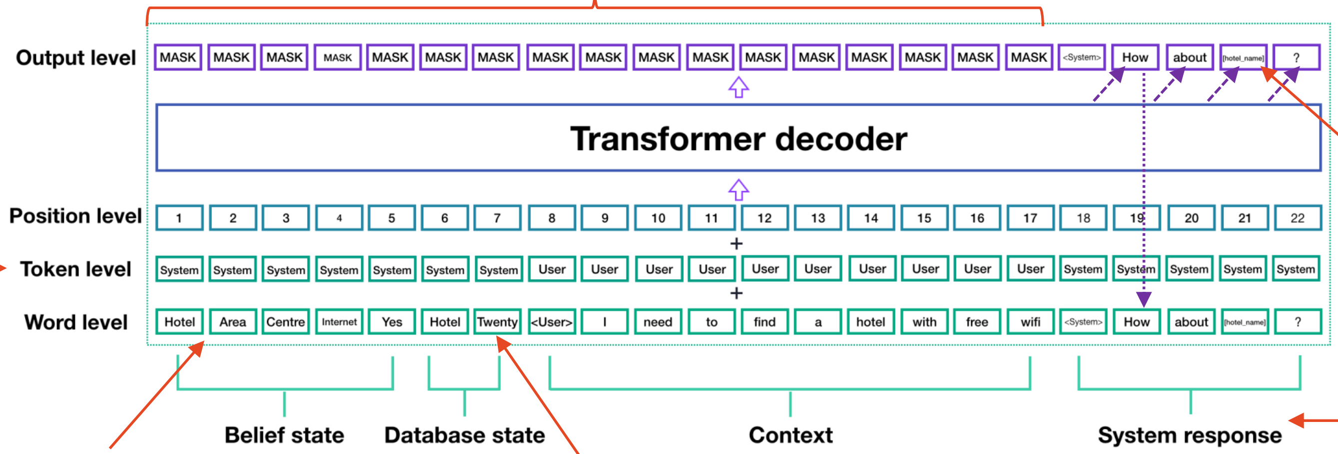
“Hello, it’s GPT-2 – How can I help?”

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

- 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 →)

“encoding”
(force-decoding)
- our inputs fed in
- outputs ignored

system/user
embeddings



delexicalized generation
(autoregressive)

this is the actually decoded part

simple encoding:
domain-slot-value[-slot-value…]

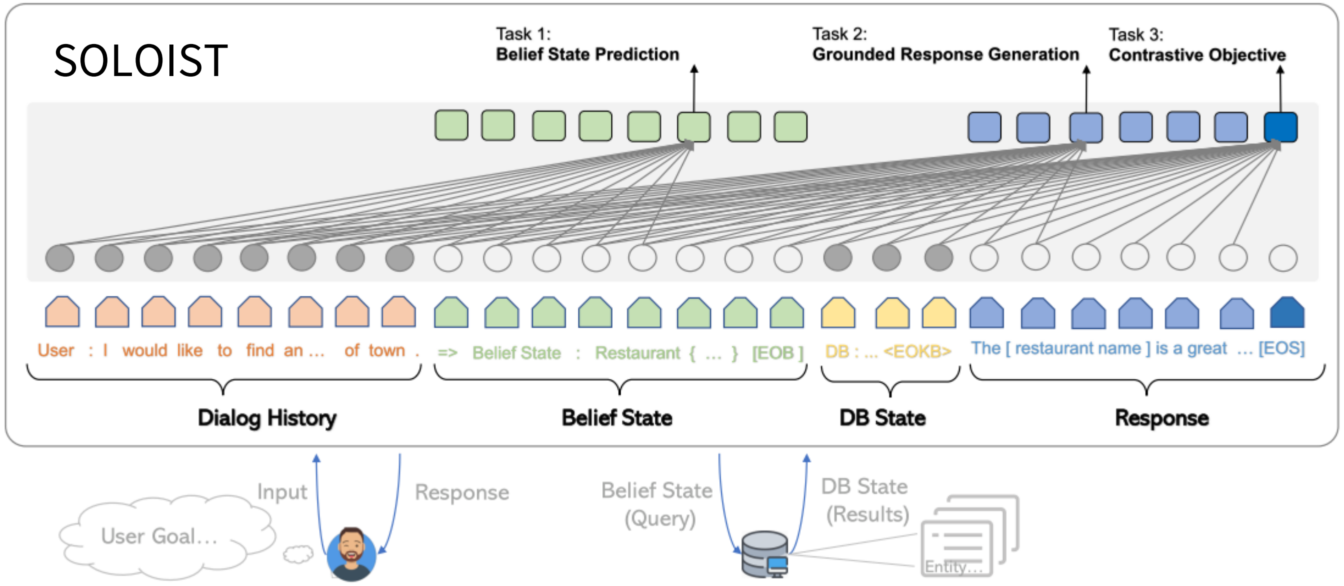
DB result entry tokens

Real stuff with GPT-2: SOLOIST, SimpleTOD, NeuralPipeline, UBAR

pre-LM | seq gen (+classif)

- 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)

(Peng et al., 2020) http://arxiv.org/abs/2005.05298
(Hosseini-Asl et al., 2020) http://arxiv.org/abs/2005.00796
(Ham et al., 2020) https://www.aclweb.org/anthology/2020.acl-main.54
(Yang et al., 2021) http://arxiv.org/abs/2012.03539

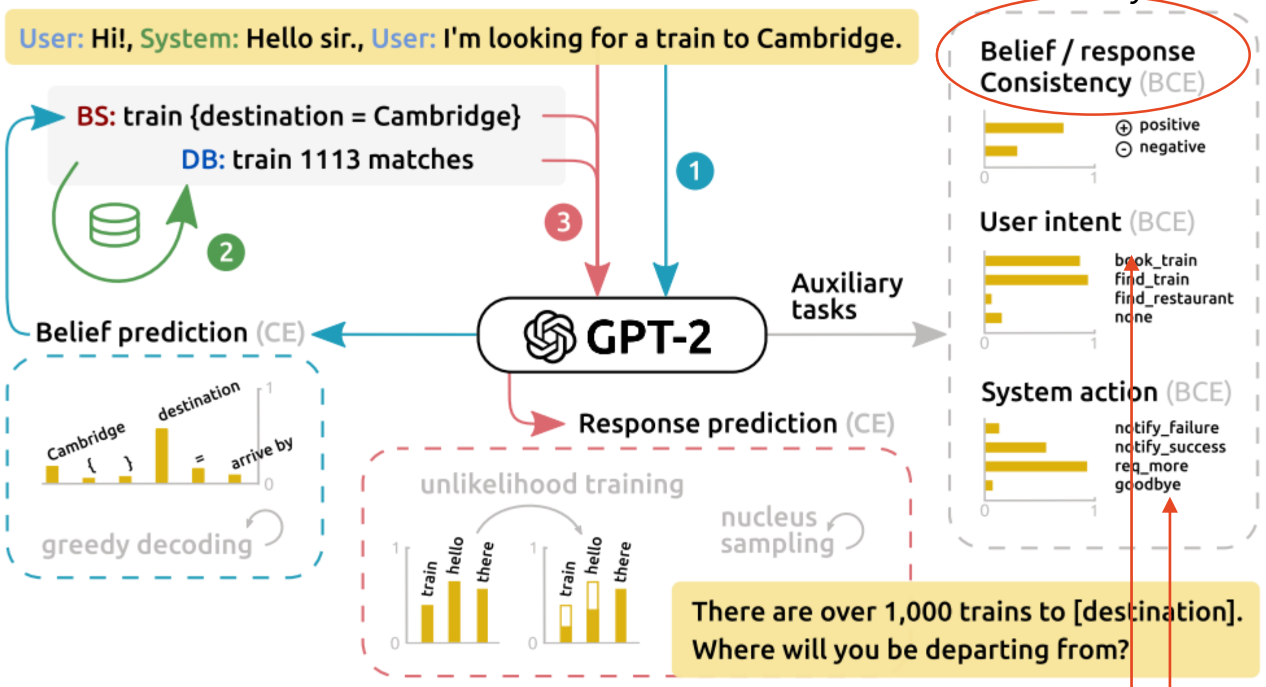


AuGPT: our take on this

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

- 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)

actually useful



(more auxiliary tasks, not really useful)

(Kulhánek et al., 2021)
http://arxiv.org/abs/2102.05126

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	state	**response**	**consistent?**		
i want a cheap italian restaurant	*{ price range = cheap , food = Italian }*	ok which area ?	✓		
i want a cheap Italian restaurant	*{ price range = cheap , food = Italian }*	thanks, goodbye !	✗	bad response	SOLOIST
i want a cheap italian restaurant	*{ destination = Cambridge , leave at = 19:00 }*	ok which area ?	✗	bad state	
i want a cheap italian restaurant	*{ area = north , food = Chinese }*	ok which area ?	✗	bad state (same domain)	

new in AuGPT

MinTL: Diff dialogue states

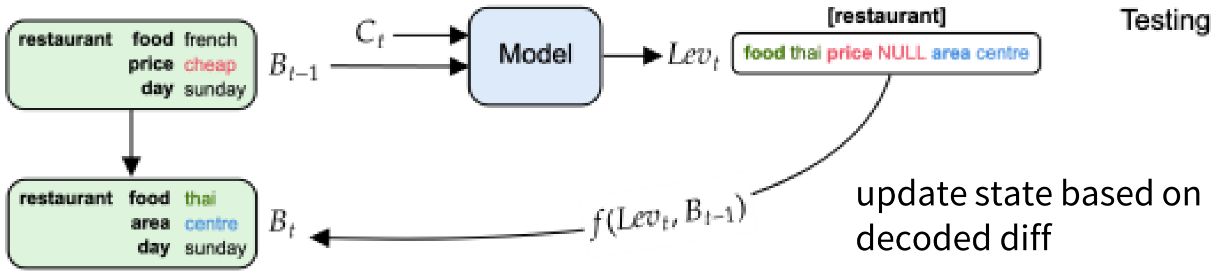
(Lin et al., 2020)
https://aclanthology.org/2020.emnlp-main.273/

- 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

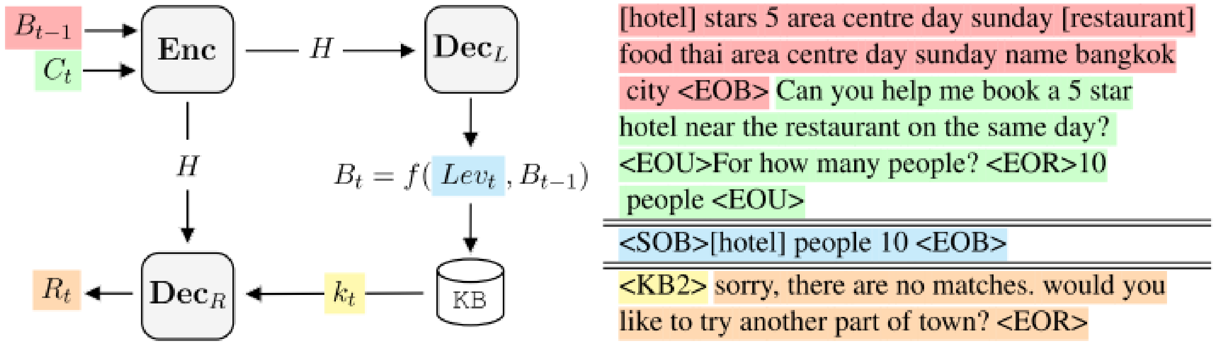
obtain diffs from state annotation

encode previous state & context

decode diffs



update state based on decoded diff



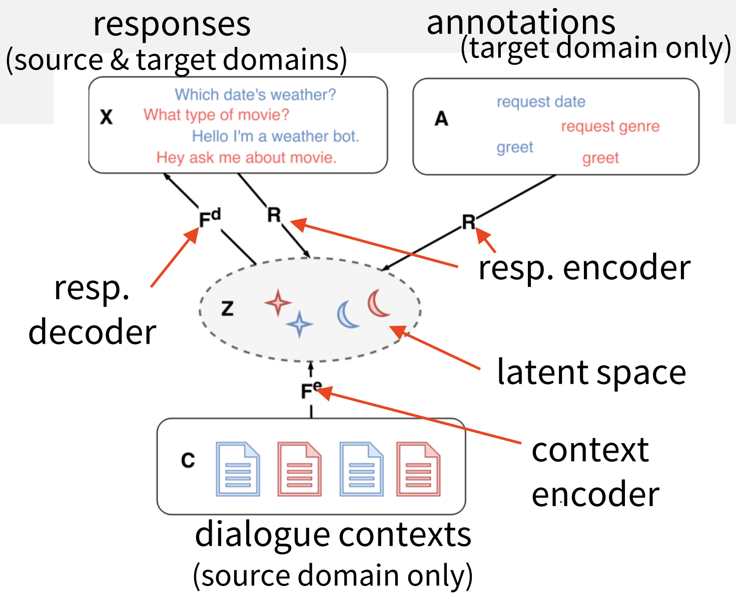
DB queried based on updated state
response decoder starting token = # of DB results

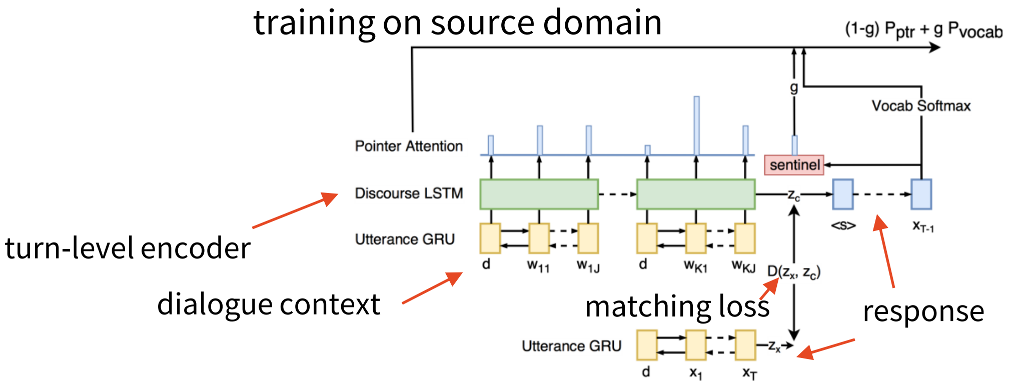
Few-shot dialogue generation

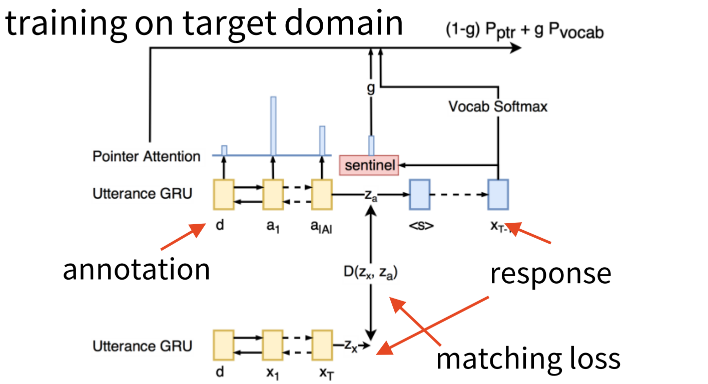
RNN + copy | seq gen

(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



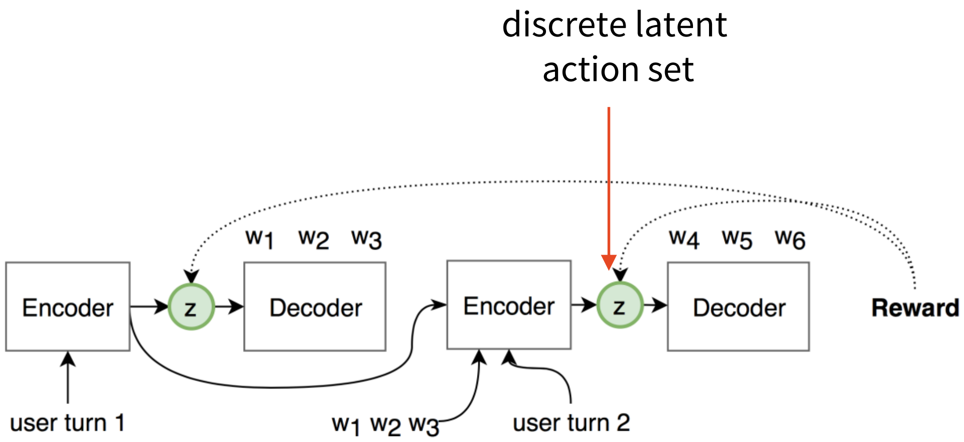




Latent Action RL

(Zhao et al., 2019)
https://www.aclweb.org/anthology/N19-1123

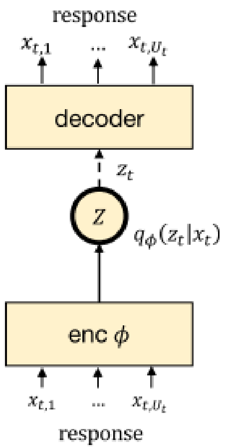
- Making system actions latent, learning them implicitly
- Like a VAE, but **discrete latent space** here (M k-way variables)
  - using Gumbel-Softmax trick for backpropagation
  - using Full ELBO (KL vs. prior network)
    or “Lite ELBO” (KL vs. uniform $1/k$)
- RL over latent actions, not words
  - 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)

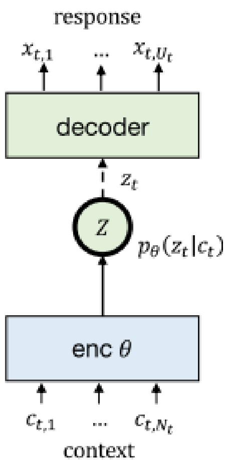


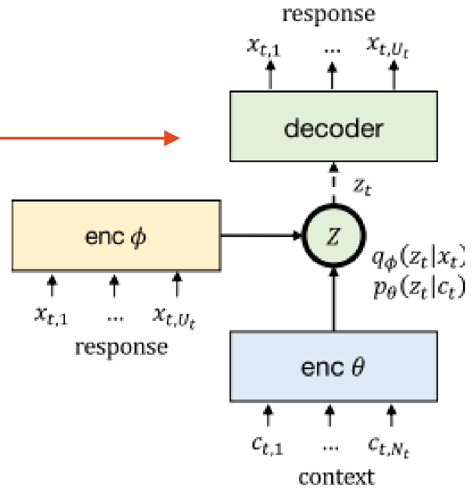
LAVA: Latent Actions with VAE pretraining

(Lubis et al., 2020)
https://aclanthology.org/2020.coling-main.41/

- 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

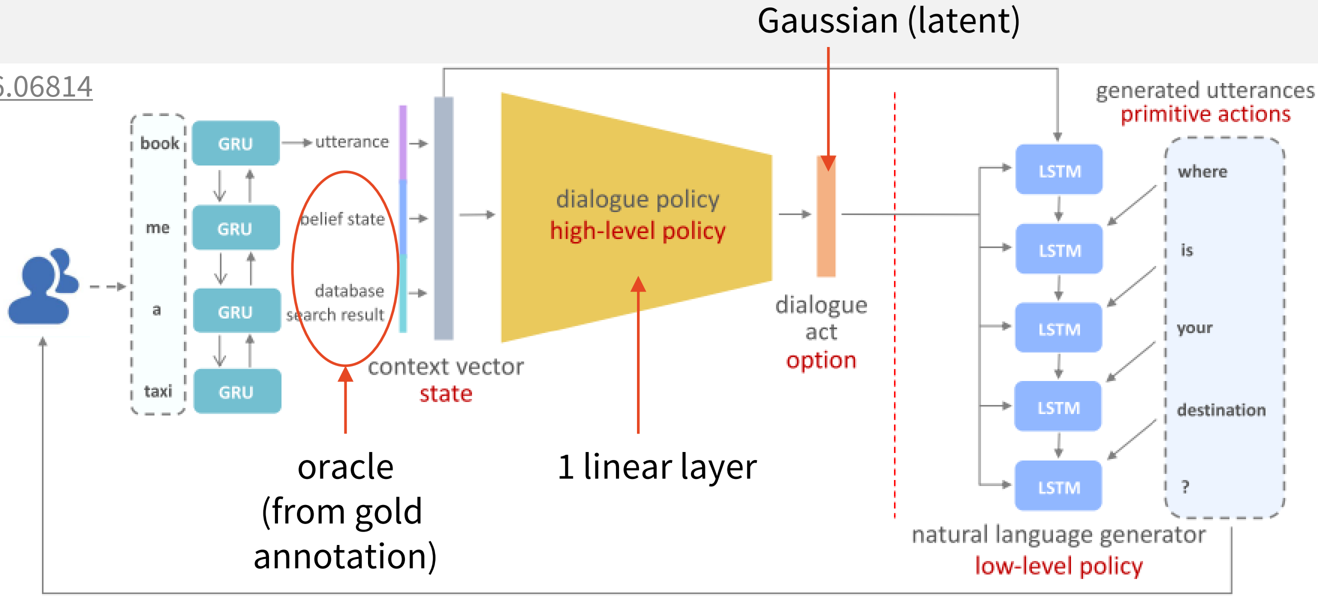






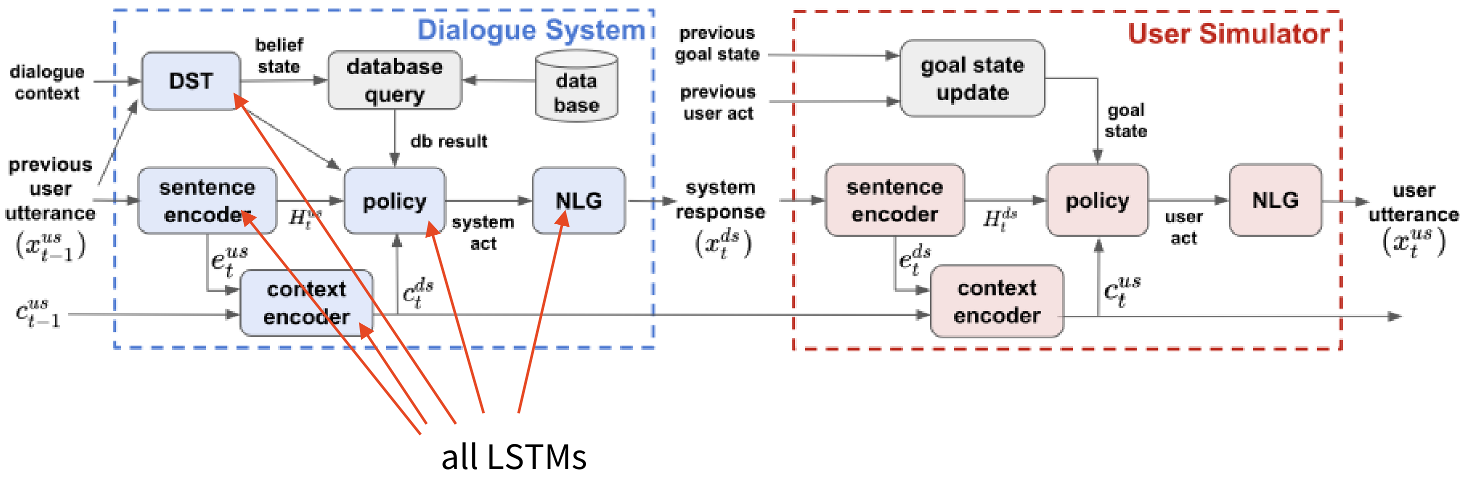
Better RL: HDNO & JOUST

(Wang et al., 2021) http://arxiv.org/abs/2006.06814



- **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

(Tseng et al., 2020) https://aclanthology.org/2021.acl-long.13



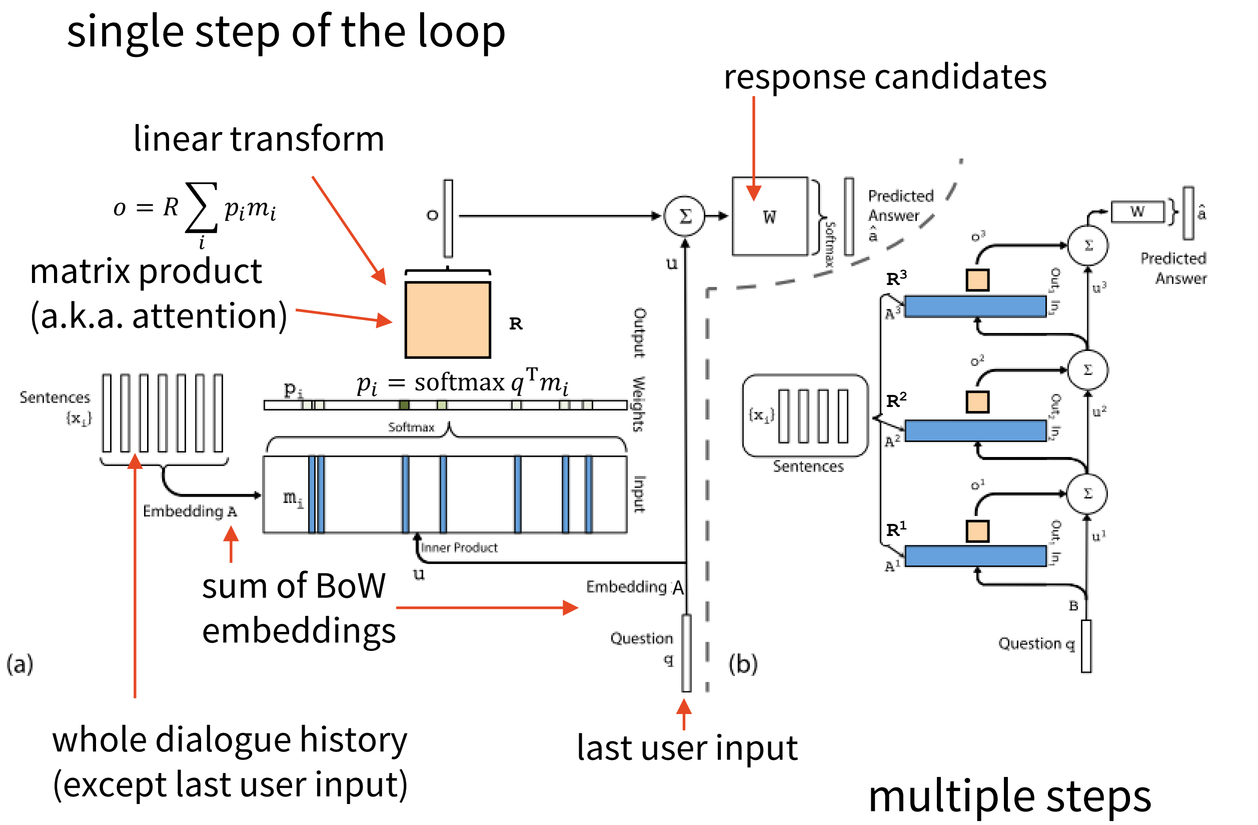
Memory networks

(Sukhbaatar et al., 2015) http://arxiv.org/abs/1503.08895
(Bordes et al., 2017) http://arxiv.org/abs/1605.07683

- 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

loop a few times

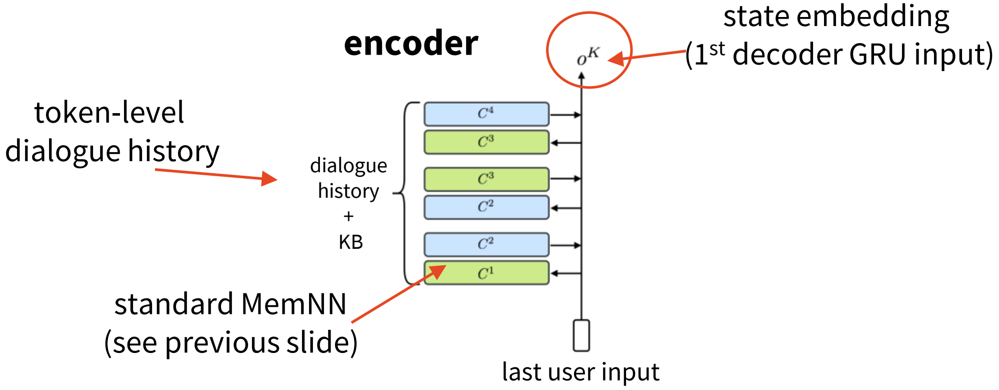
- last input matched (dot + softmax) against a pool of possible responses

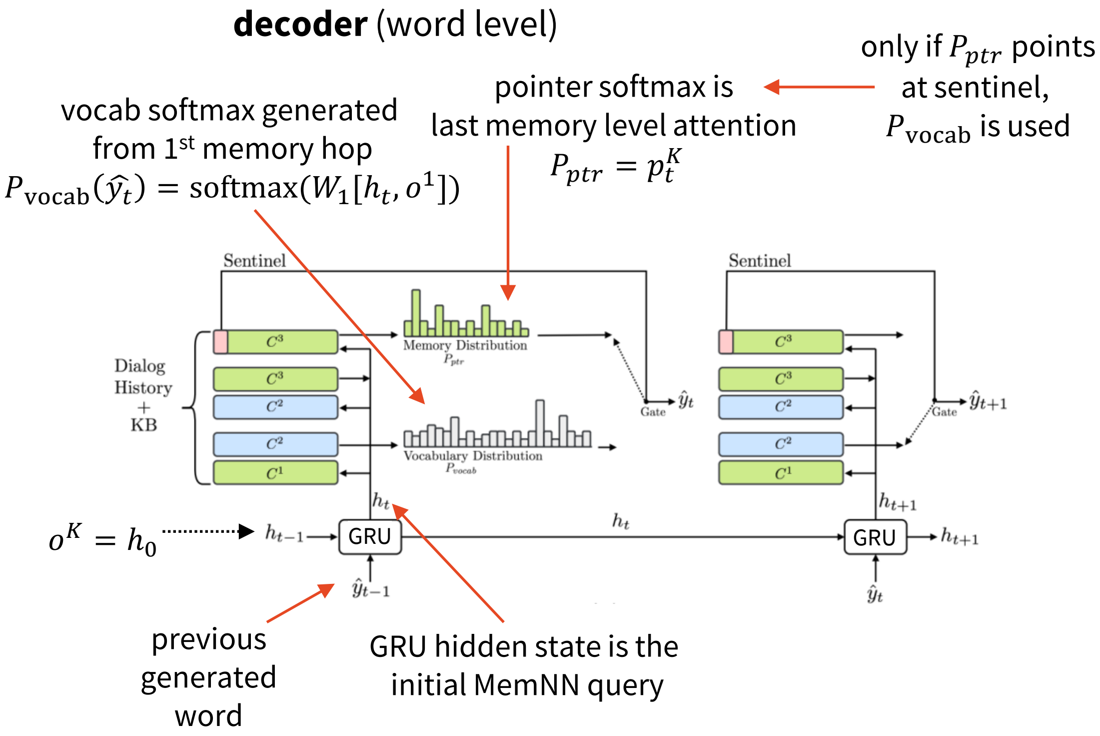


Mem2Seq: Memory nets + pointer-generator = soft DB lookups directly in the model

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

- "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



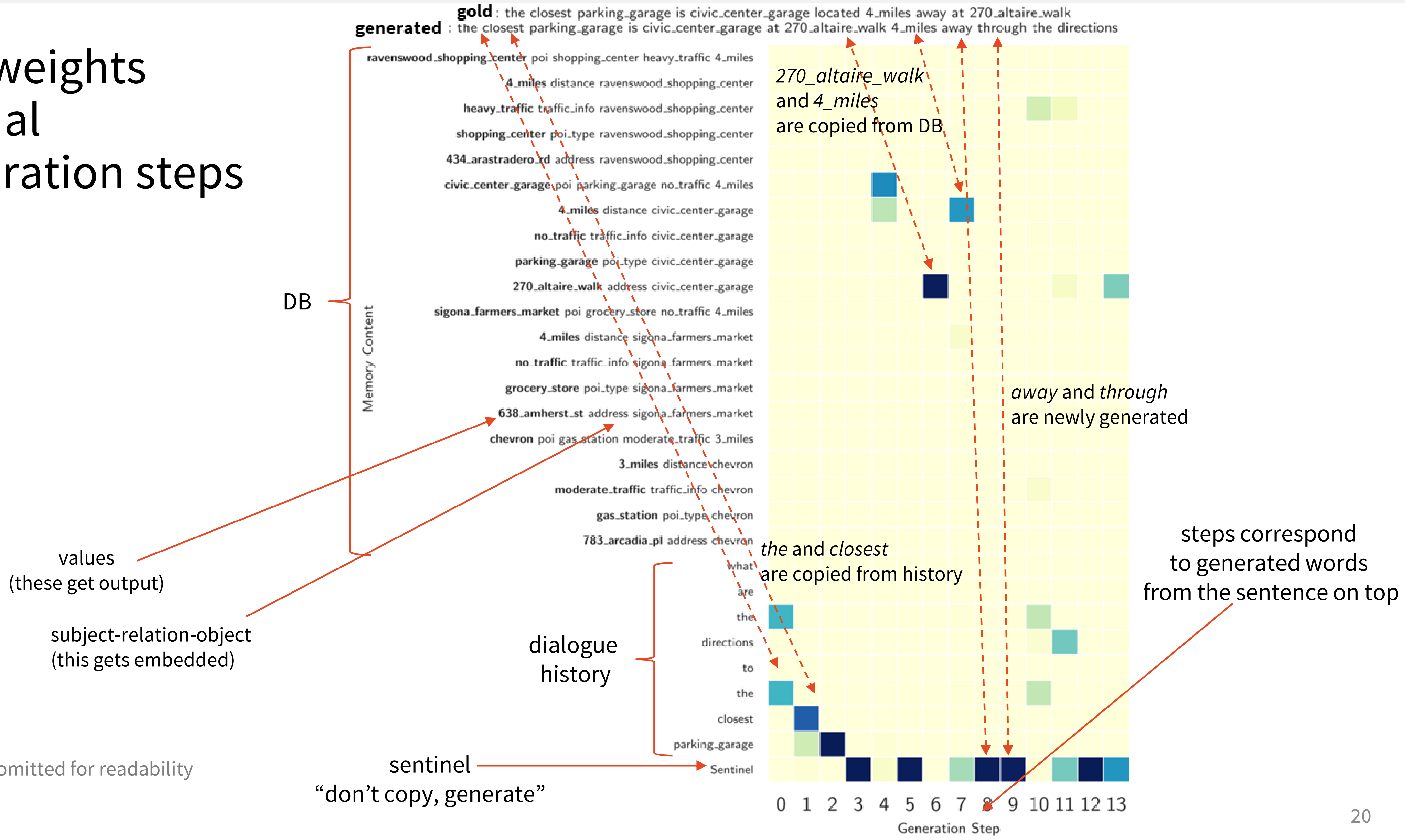


Mem2Seq visualization

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

FC + att + RNN + copy | seq gen

attention weights
at individual
word generation steps



Note: some DB entries were omitted for readability

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

Thanks

Contact us:

https://ufaldsg.slack.com/
{odusek,hudecek,kasner}@ufal.mff.cuni.cz
Skype/Zoom/Troja (by agreement)

No labs today

See you next week

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
- Serban et al. (2018): A Survey of Available Corpora For Building Data-Driven Dialogue Systems: http://dad.uni-bielefeld.de/index.php/dad/article/view/3690