NPFL099 Statistical Dialogue Systems

5. Dialogue State Tracking

http://ufal.cz/npfl099

Ondřej Dušek & Vojtěch Hudeček

27. 10. 2020

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

Dialogue State Tracking

- Dialogue management consists of:
  - **State update** ← here we need DST
  - Action selection (later)
- **Dialogue state** needed to remember what was said in the past
  - tracking the dialogue progress
  - summary of the whole dialogue history
  - basis for action selection decisions

U: I'm looking for a restaurant in the city centre.
S: OK, what kind of food do you like?
U: Chinese.
✗ *S: What part of town do you have in mind?*
✗ *S: Sure, the Golden Dragon is a good Chinese restaurant. It is located in the west part of town.*
✓ *S: Sure, the Golden Dragon is a good Chinese restaurant. It is located in the city centre.*

Dialogue State Contents

- "All that is used when the system decides what to say next" (Henderson, 2015)
- **User goal**/preferences ~ NLU output
  - slots & values provided (search constraints)
  - information requested
- Past **system actions**
  - information provided
    - slots and values
    - list of venues offered
  - slots confirmed
  - slots requested
- **Other** semantic context
  - user/system utterance: bye, thank you, repeat, restart etc.

U: *Give me the address of the first one you talked about.*
U: *Is there any other place in this area?*

S: *OK, Chinese food. […]*

S: *What time would you like to leave?*

Problems with Dialogue State

- NLU is unreliable
  - takes unreliable ASR output
  - makes mistakes by itself – some utterances are ambiguous
  - output might conflict with ontology
- Possible solutions:
  - detect contradictions, ask for confirmation
  - ignore low-confidence NLU input
    - what's "low"?
    - what if we ignore 10x the same thing?
- Better solution: make the state probabilistic – **belief state**

ASR: 0.5 *I'm looking for an expensive hotel*
0.5 *I'm looking for inexpensive hotels*

NLU: 0.3 inform(type=restaurant, stars=5)

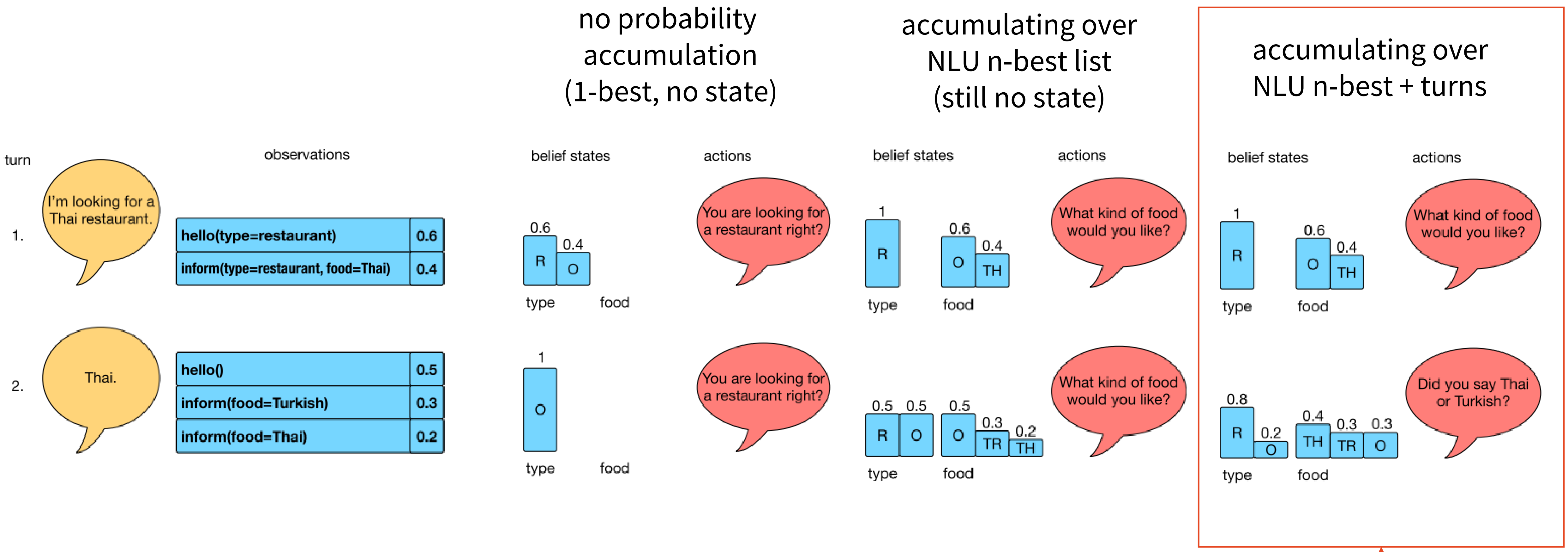
only hotels have stars!

Belief State

- Assume we don't know the true current dialogue state s_t
  - states (what the user wants) influence **observations** o_t (what the system hears)
  - based on observations o_t & system actions a_t, we can estimate a probability distribution $b(s)$ over all possible states – **belief state**
- More robust than using dialogue state directly
  - accumulates probability mass over multiple turns
    - low confidence – if the user repeats it, we get it the 2nd time
  - accumulates probability over NLU n-best lists
- Plays well with probabilistic dialogue policies (POMDPs)
  - but not only them – rule-based, too

Belief State

(from Milica Gašić's slides)



this is what we need (=belief state)

Basic Discriminative Belief Tracker

- **Partition the state** by assuming conditional independence
  - simplify – assume each slot is independent:
    - state $\mathbf{s} = [s^1, \ldots s^N]$, belief $b(\mathbf{s}_t) = \prod_i b(s_t^i)$
- **Always trust the NLU**
  - this makes the model parameter-free
  - …and basically rule-based
  - but very fast, with reasonable performance

NLU output

“user mentioned this value”

$$p(s_t^i | a_{t-1}^i, s_{t-1}^i, o_t^i) = \begin{cases} p(o_t^i) \text{ if } \overbrace{s_t^i = o_t^i \wedge o_t^i \neq \text{🤫}} \\ p(o_t^i) \text{ if } \underbrace{s_t^i = s_{t-1}^i \wedge o_t^i = \text{🤫}} \\ 0 \text{ otherwise} \end{cases}$$

“no change”

user silent about slot i

update rule

$$b(s_t^i) = \sum_{s_{t-1}^i, o_t^i} \underbrace{p(s_t^i | a_{t-1}^i, s_{t-1}^i, o_t^i)}_{\text{discriminative model}} b(s_{t-1}^i)$$

substitution

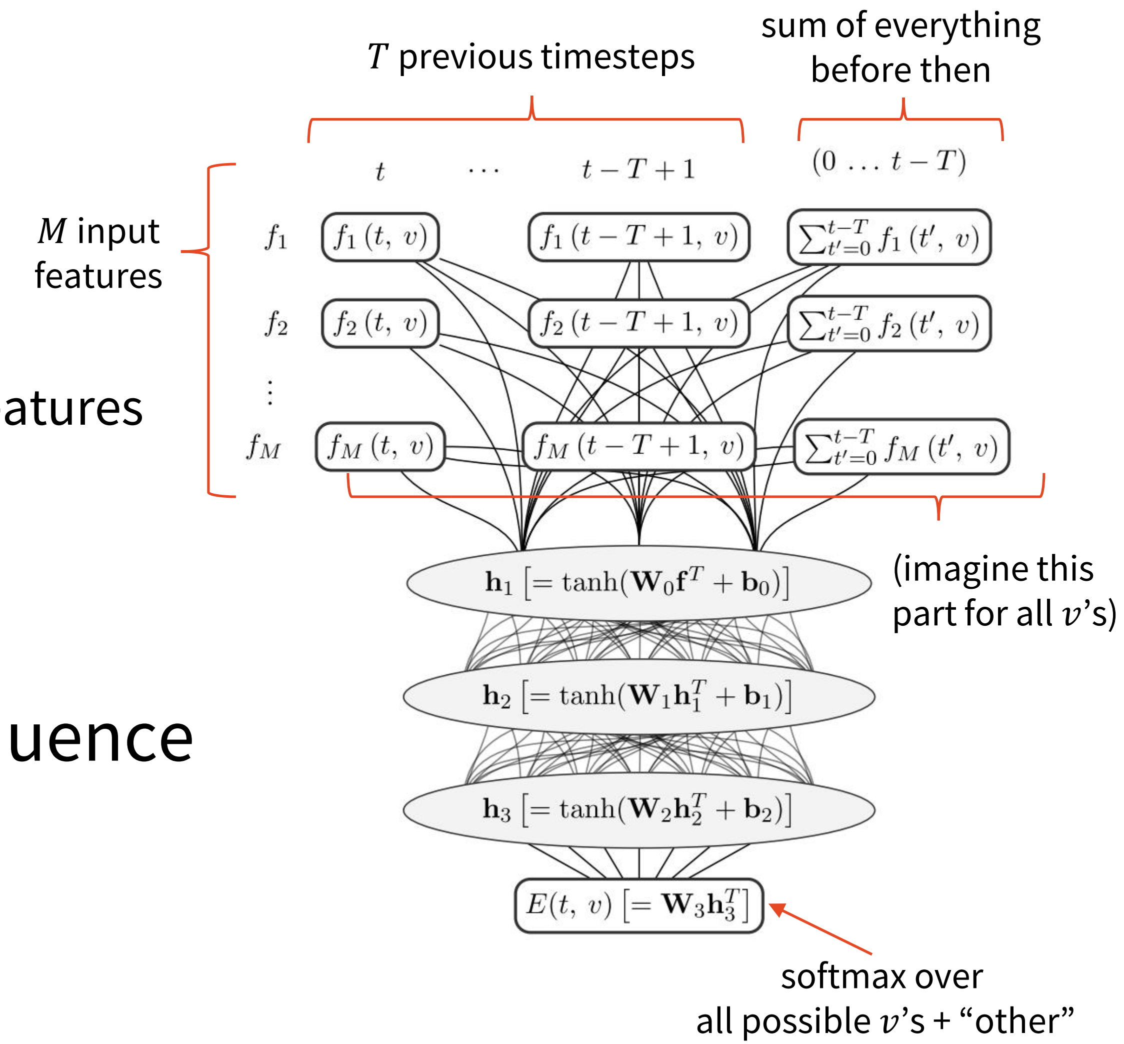
$$b(s_t^i) = \begin{cases} p(s_t^i = \text{🤫}) p(o_t^i = \text{🤫}) & \text{if } s_t^i = \text{🤫} \\ p(o_t^i = s_t^i) + p(o_t^i = \text{🤫}) p(s_t^i = s_{t-1}^i) & \text{otherwise} \end{cases}$$

the belief state update rule is deterministic

(Žilka et al., 2013)
http://www.aclweb.org/anthology/W13-4070

Basic Feed-forward Neural Tracker

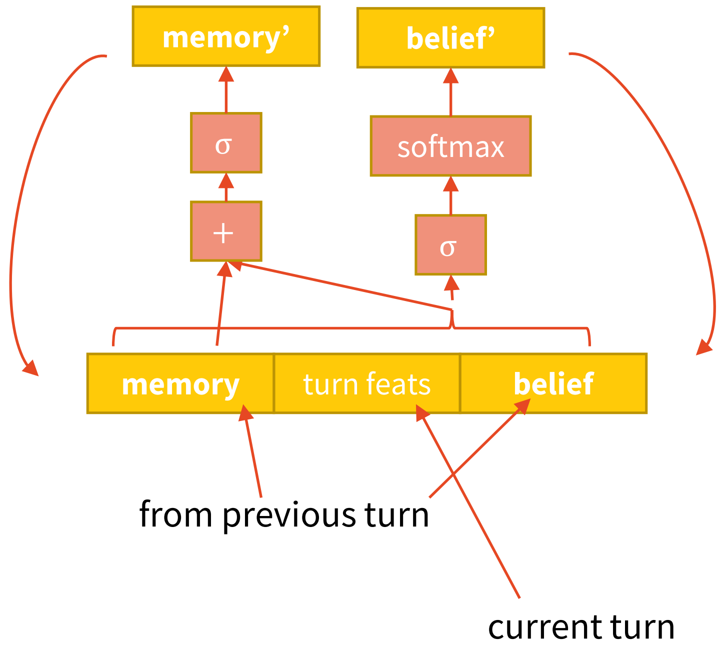
- a simple feed-forward network
  - input – features (w.r.t. slot-value v & time t)
    - NLU score of v
    - n-best rank of v
    - user & system intent (*inform*/*request*)
    - … – other domain-independent, low-level NLU features
  - 3 tanh layers
  - output – softmax
    (= probability distribution over values)
- **static** – does not model dialogue as a sequence
  - uses a **sliding window**:
    current time t + few steps back + $\sum$previous



(Henderson et al., 2013)
https://aclweb.org/anthology/W13-4073

Basic RNN Tracker

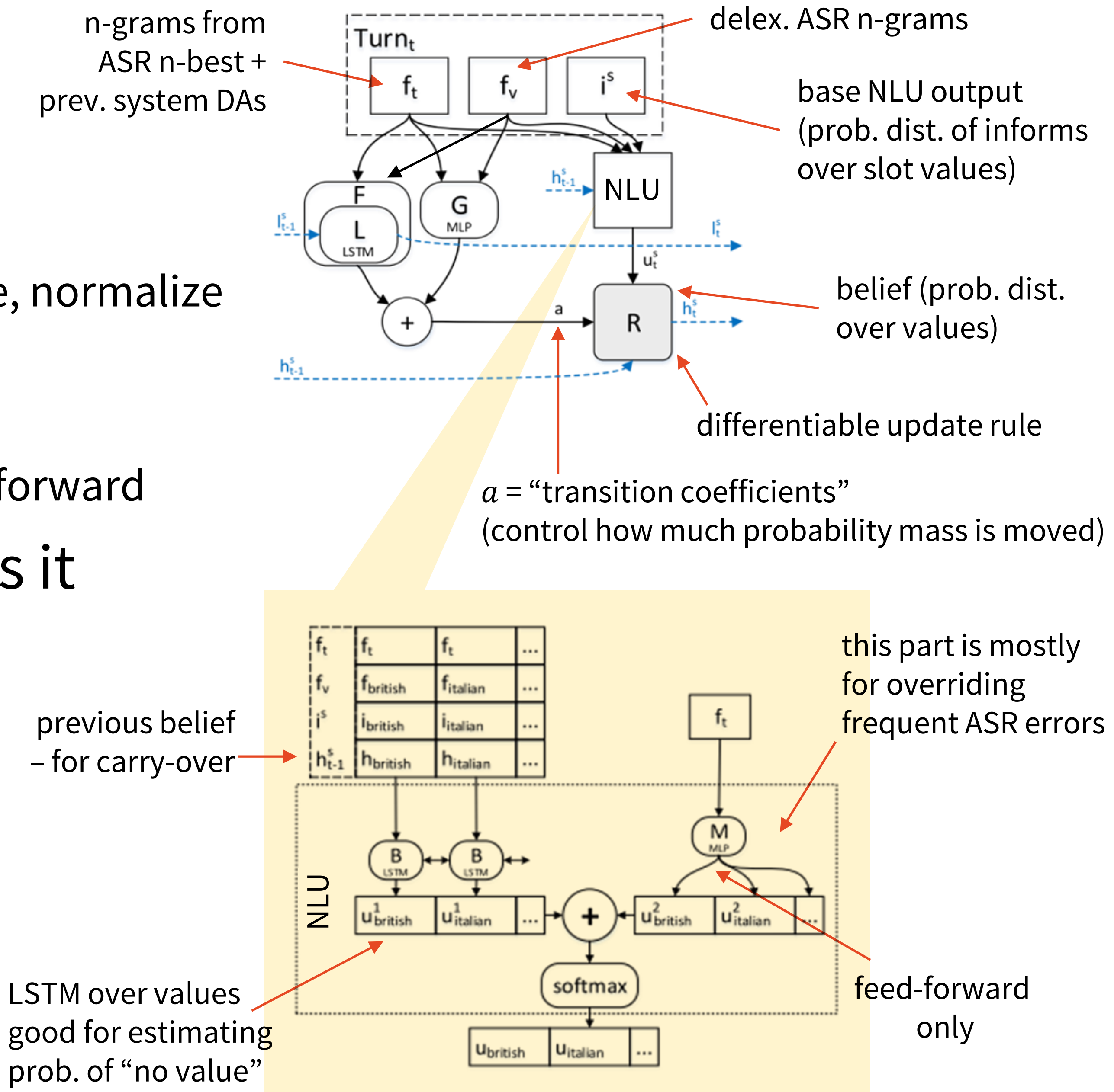
- plain sigmoid RNN with a memory vector
  - not quite LSTM/GRU, but close
  - memory updated separately, used in belief update
- does not need NLU
  - turn features = lexicalized + delexicalized *n*-grams from ASR n-best list, weighted by confidence
- delexicalization is very harsh: <slot> <value>
  - you don't even know which slot it is
  - this apparently somewhat helps the system generalize across domains
- **dynamic** – explicitly models dialogue as sequence
  - using the network recurrence



(Mrkšić et al., 2015)
http://arxiv.org/abs/1506.07190

Neural/Rule Hybrid

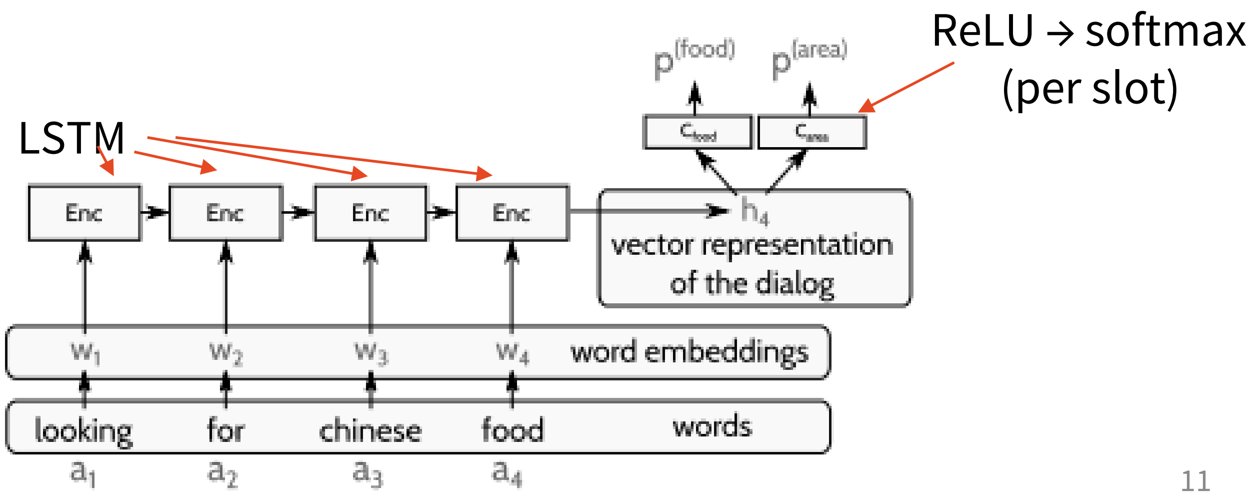
- Dynamic: explicit update of belief
  - per-slot model (separate for each slot)
  - simple update rule R
    - for a value: add a · current NLU confidence, normalize
    - differentiable, can be trained end-to-end
  - trained models F, G provide a
    - F is generic LSTM, G is value specific feed-forward
- Needs a base NLU, but postprocesses it
  - input & output of tracker NLU step
    = prob. dist. of informs
    over slot values in current turn
  - generic & specific part again



(Vodolán et al., 2017)
http://arxiv.org/abs/1702.06336

Incremental Recurrent Tracker

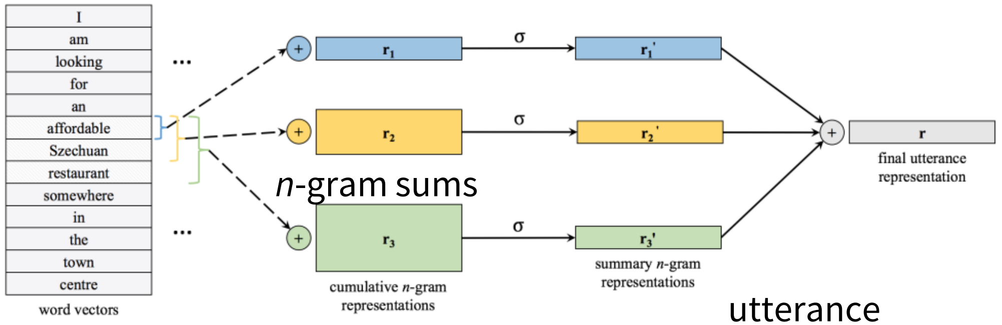
- Simple: LSTM over words + classification on hidden states
  - runs over the whole dialogue history (user utterances + system actions)
  - classification can occur after each word, right as it comes in from ASR
- Dynamic/sequential
- Doesn’t use any NLU
  - infrequent values are delexicalized (otherwise it can’t learn them)
- Slightly worse performance – possible causes:
  - only uses ASR 1-best
  - very long recurrences (no hierarchy)

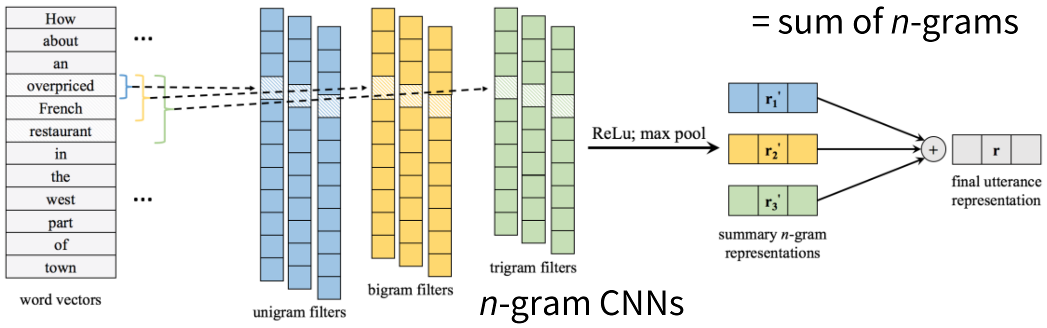


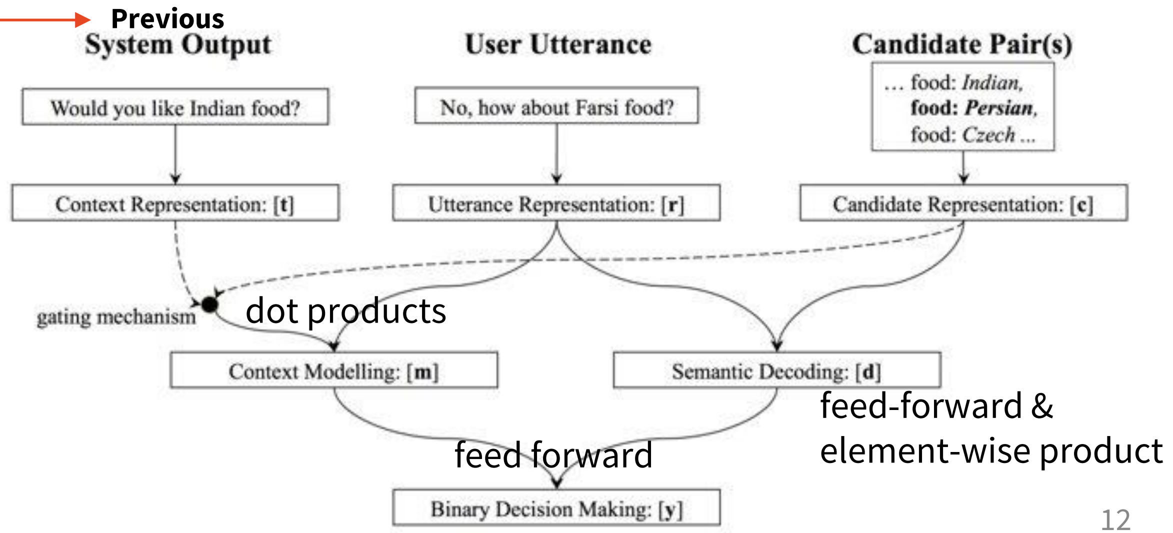
(Žilka & Jurčíček, 2015)
https://dl.acm.org/citation.cfm?id=2955040
http://arxiv.org/abs/1507.03471

NBT: Pretrained Word Embeddings

- No delexicalization needed
- Current turn + rule-based updates (=**static** tracker)
- Pretrained word vectors (kept fixed)
  - GloVe enhanced with paraphrases
- Text = *n*-gram sums/CNNs, summed
  - same parameters + handling for all inputs
    - contextual: requested/confirmed slot (+value)
    - current user utterance
    - candidate slot-value pair (run once for each)
- Simple combinations
  - dot product, feed-forward
  - binary decision: is the candidate correct?







Candidate Ranking

(Rastogi et al., 2017)
https://arxiv.org/abs/1712.10224

- Previous systems consider all values for each slot
  - this is a problem for open-ended slots (e.g. restaurant name)
  - enumerating over all takes ages, some are previously unseen
- Alternative: always consider just K candidates
  - use last K candidates from system actions and NLU output
    - NB: only way history is incorporated here (~static)
  - select from them using a per-slot softmax

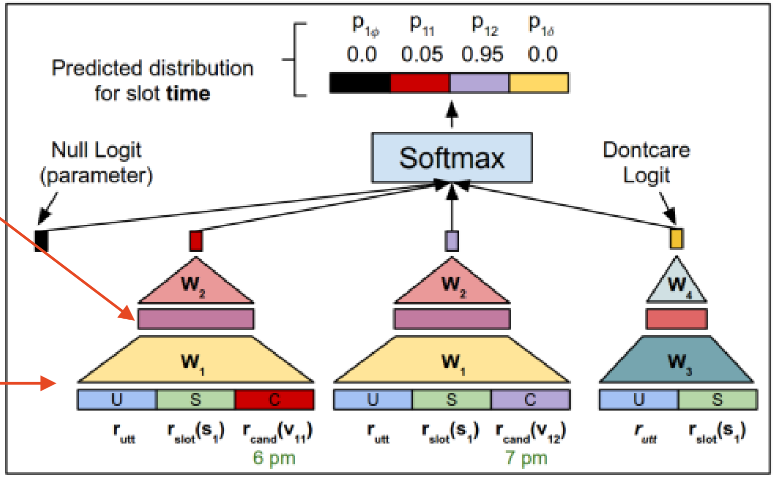
padding (not enough values mentioned)

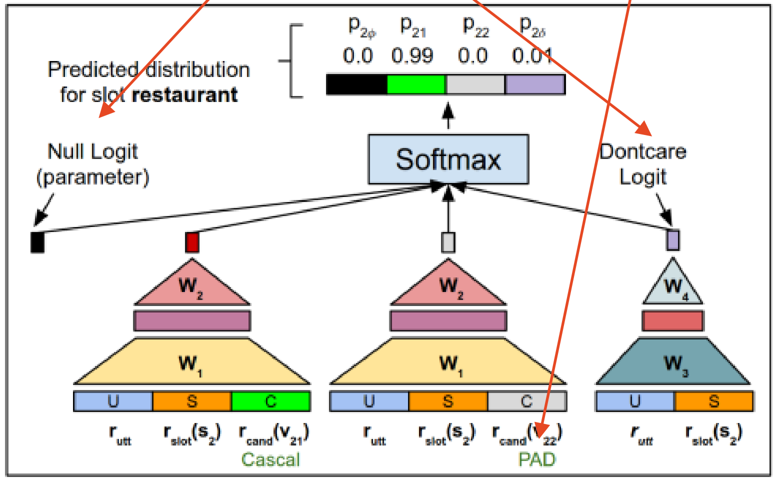
additional values to consider (even if not mentioned in NLU)

pictures assume $K = 2$

2 sigmoid layers

representation of i-th candidate: utterance/slot/candidate features (next slide)





Candidate Ranking – representation

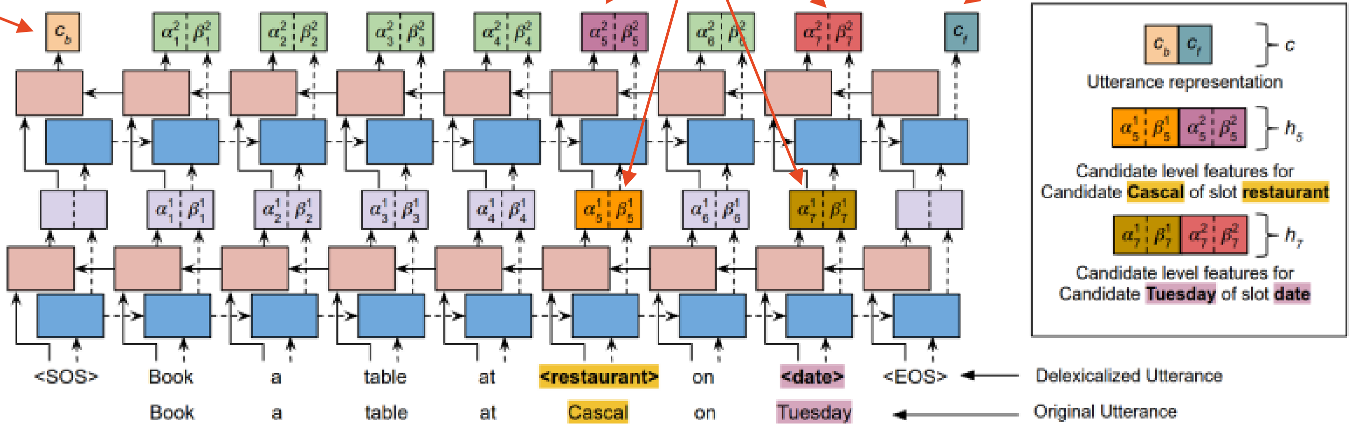
(Rastogi et al., 2017)
https://arxiv.org/abs/1712.10224

- Using BiGRU over lexicalized & delexicalized utterance
- Features:
  - **utterance** – last GRU state + NLU indicators for non-slot DAs (user & prev. system) *bye(), affirm()*
  - **slot** – NLU indicators for DAs with this slot (user & prev. system) *inform(slot=*), request(slot)*
    + last turn scores for *null* & *dontcare*
  - **candidate** – GRU states over matched value words
    + NLU indicators for DAs with this slot & value (user & prev. system) *inform(slot=value)*

utterance feature

candidate features

utterance feature



Multi-value Candidate Ranking

- What if multiple values are true?
  - previous approach picks one (softmax)
  - use set of binary classifiers (log loss) instead (similar to NBT)
- More flexible regarding candidates (still a fixed max. number)
  - can be past k from NLU, but also just current ASR n-grams
    - ElMo helps with ASR n-grams
- Dynamic –keeps context by itself
  - embedding previous states,
    system actions, text of the whole dialogue

multiple per-slot models
share info about previous beliefs

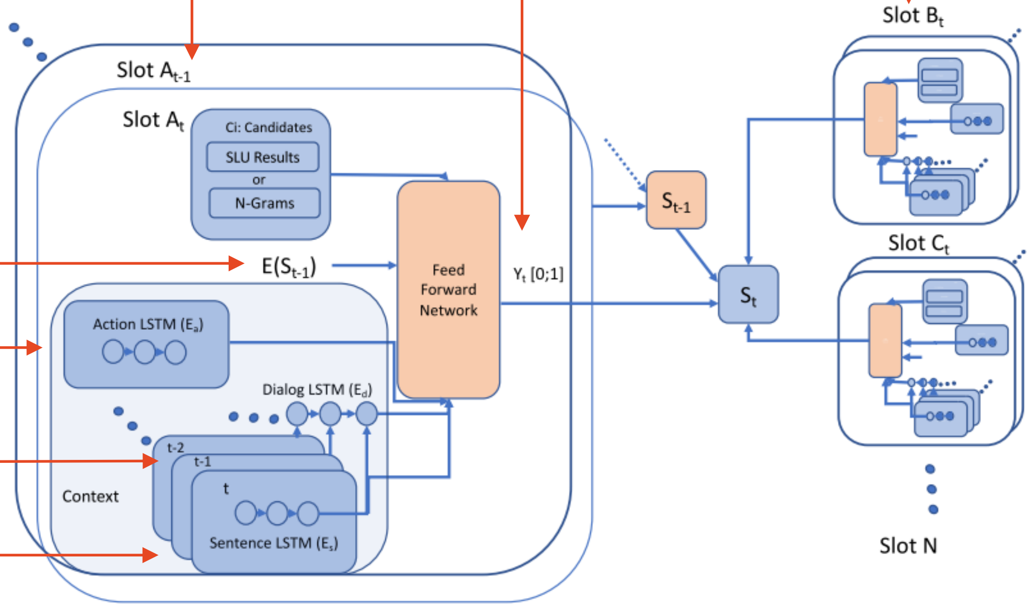
binary decision
for a candidate

previous belief state embedding

past system actions turn-level LSTM

past sentences over
hierarchical LSTM

current sentence



(Goel et al., 2018)
http://arxiv.org/abs/1811.12891

Hybrid Classify/Rank

metric: **joint goal accuracy**
– exact match on dialogue state
(most probable value only)

- Ranking is faster & more flexible
- Classification over all values is more accurate
  - at least for most slots, where # of values is limited
- Solution: combine classification & ranking
  - **choose best model for each slot** based on dev data performance
- Ranking approach – multi-value from previous slide
- Classification approach – straightforward:
  - hierarchical LSTM
  - per-slot feed-forward
  - softmax

Method	Accuracy
Majority Baseline	1.5%
MultiWOZ-2.0 Benchmark	25.83%
Ranking only	31.11% (29.73%)
Classification only	40.74% (38.42%)
Hybrid	44.24% (42.33%)

ensemble (majority vote of 3 models)
single model

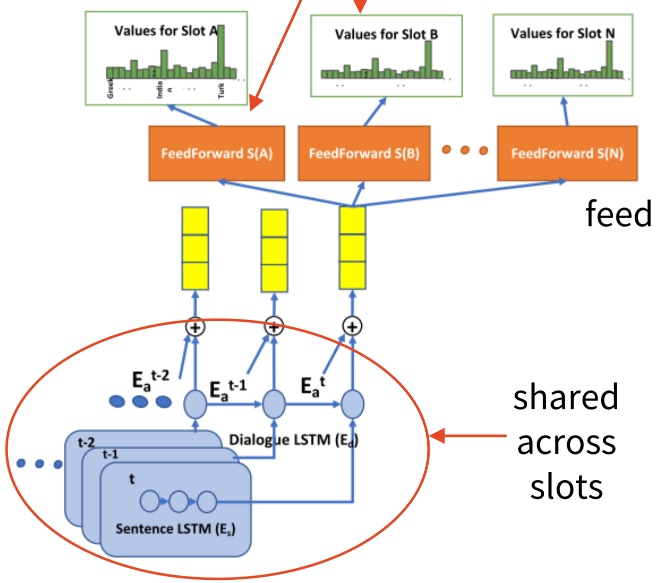
separate for each slot

softmax

feed forward

hierarchical LSTM

shared across slots

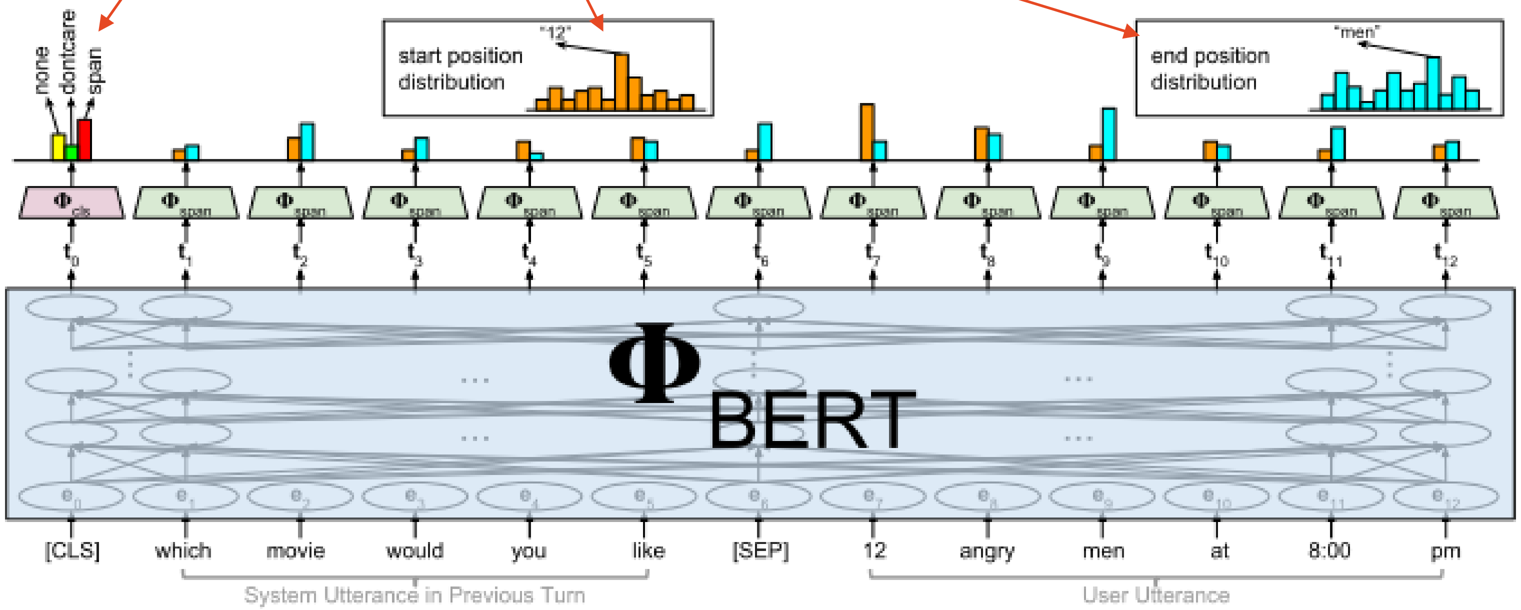


(Goel et al., 2019)
http://arxiv.org/abs/1907.00883

BERT & Span Tagging (~similar to reading comprehension)

(Chao & Lane, 2019)
http://arxiv.org/abs/1907.03040

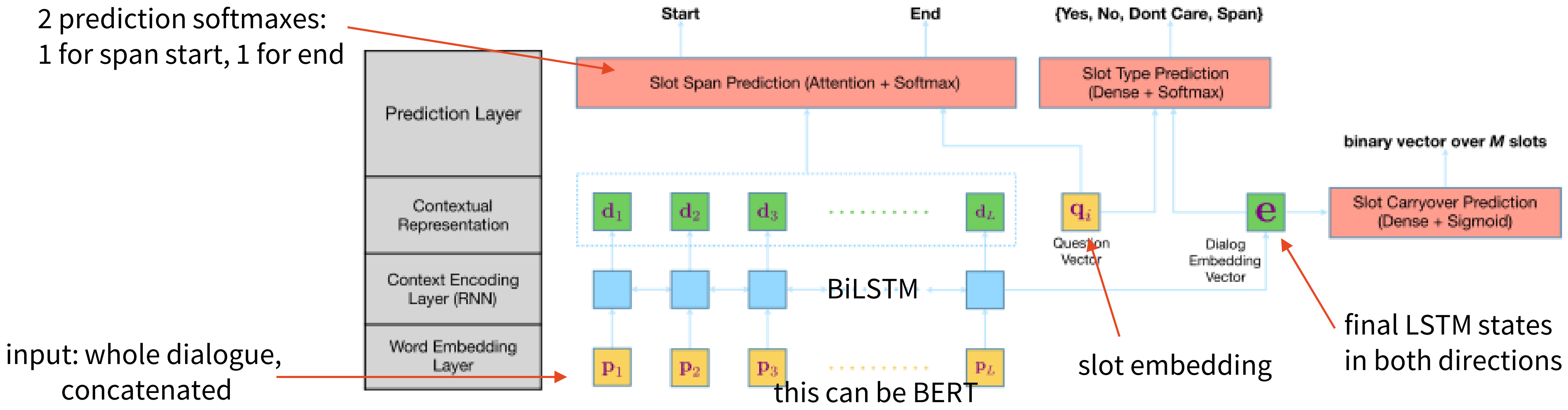
- BERT over previous system & current user utterance
- from 1st token's representation, get a **decision:** ***none*/*dontcare*/span**
  - per-slot (BERT is shared, but the final decision is slot-specific)
- span = need to find a concrete value as a span somewhere in the text
  - **predict start & end token** of the span using 2 softmaxes over tokens
- rule-based update (static):
  - if *none* is predicted, keep previous value



Span Tagging with Modelled Update

(Gao et al., 2019)
https://www.aclweb.org/anthology/W19-5932/

- Also uses BERT, but not necessarily
  - works slightly worse with random-initialized word embeddings
- sequence of 3 decisions
  - do we carry over last turn's prediction? (Yes/No) (~static tracking, but not so rigid)
  - if no: what kind of answer are we looking for? (*yes*/*no*/*dontcare*/span of text)
  - if span: predict span's start and end



Span Tagging & Better Copying

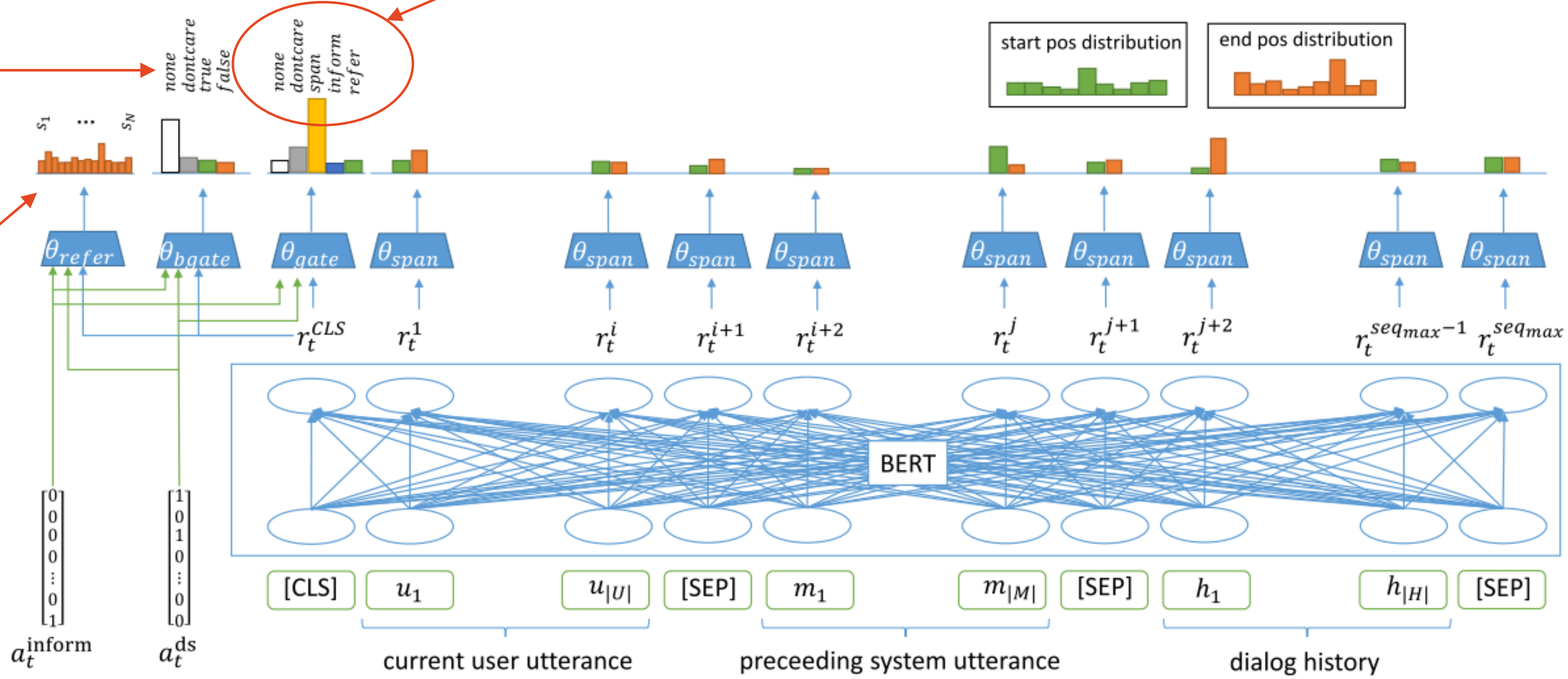
(Heck et al., 2020)
https://aclweb.org/anthology/2020.sigdial-1.4/

- “triple-copy” – gets the value from 3 sources:
  - user utterance (same as previous span tagging models)
  - system informs (last value the system mentioned)
  - another slot (coreference), e.g. a taxi ride to a hotel (hotel name = destination)
- rule-based update (static)

same decision as previously, just different options:
none/*dontcare*/span/inform/refer

boolean slots are handled separately (classification)

coreference – distribution over slots to copy from

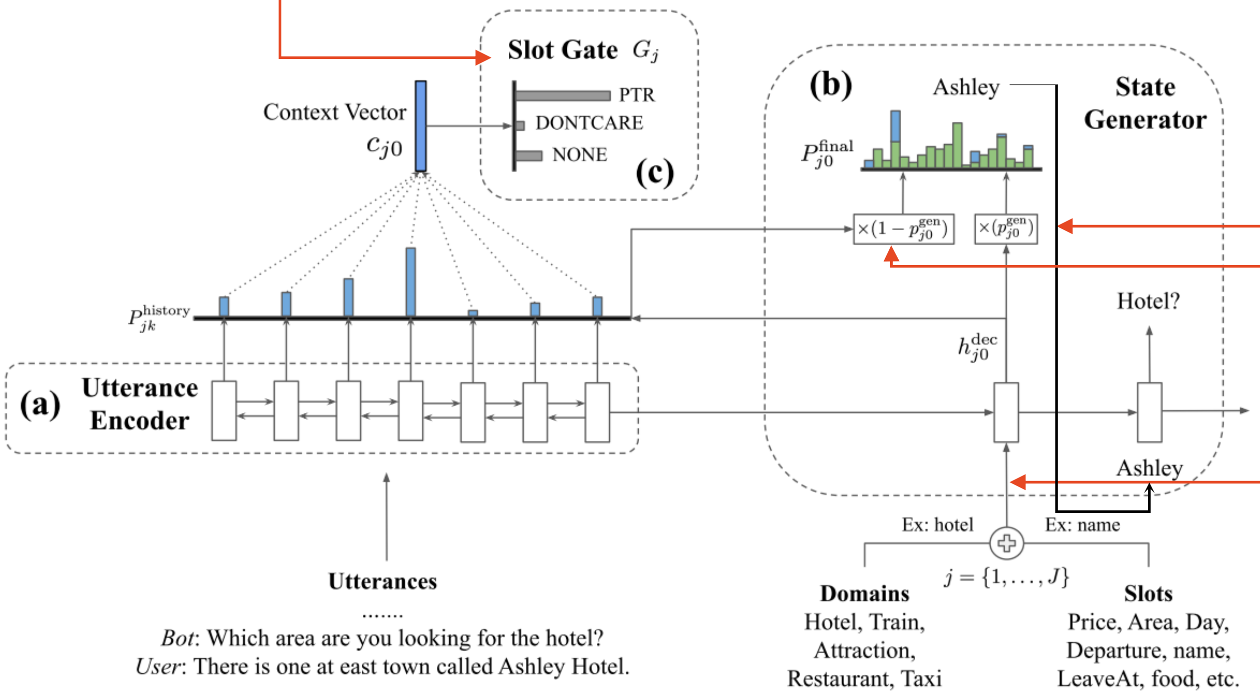


Generator-based Tracker

(Wu et al., 2019)
https://www.aclweb.org/anthology/P19-1078

- Similar to span tagging: encodes whole dialogue history (static)
- Pointer-generator seq2seq decoder produces values
  - specific start token for each slot -- copies from input & generates new tokens
- Slot gate: "use generated"/*dontcare*/*none*
  - same as the decisions done in span tagging, just applied *after* getting the value

takes concatenated dialogue history

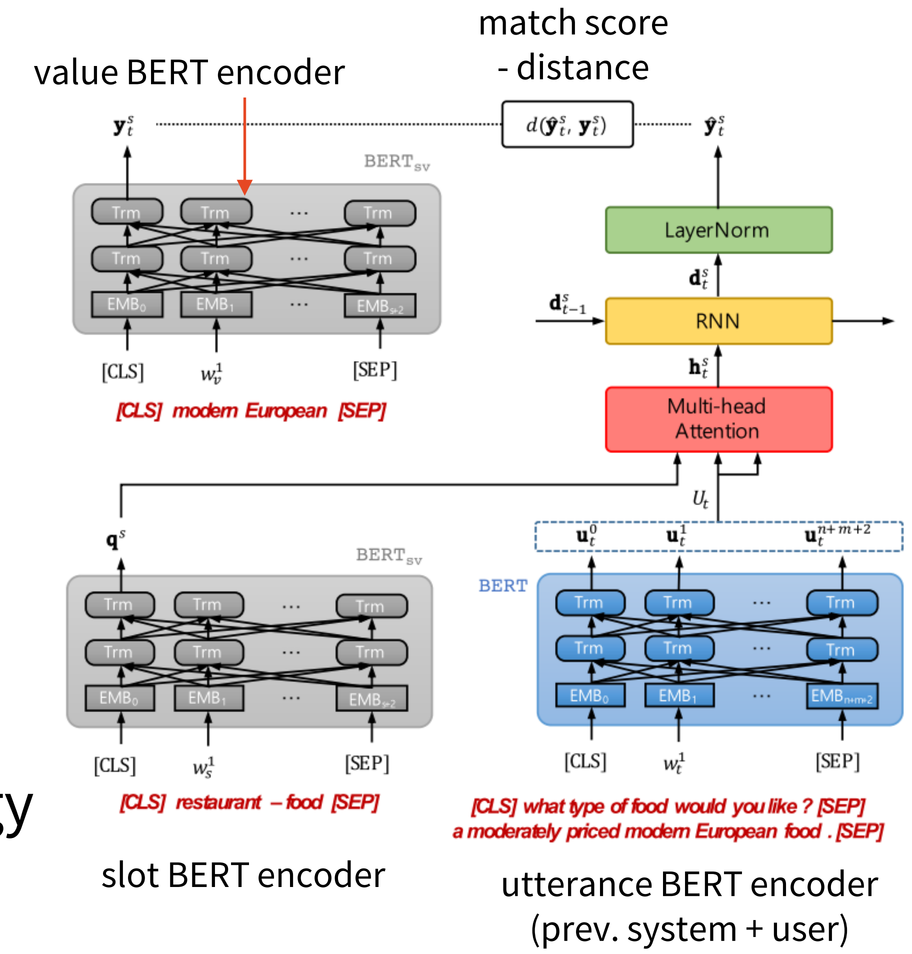


pointer-generator net (see NLU lecture): can **generate** tokens from vocabulary or **copy** tokens from attention

specific start token for each slot (& domain)

Slot-Utterance Matching

- different take on BERT reading comprehension
  - considers “domain – slot” a question & tries to find the best-matching value
  - ~ candidate ranking/binary classification approach
- tracker over BERT
  - attention + turn-based RNN (dynamic)
    - attention over current utterance
      - with BERT-encoded slot name as guidance
    - RNN (LSTM/GRU) keeps past values
    - RNN output layer-normalized to match BERT outputs
  - trained to match the correct values from the ontology
    - loss: distance of true value’s BERT encoding from the tracker output (Euclidean/Cosine)
    - BERT encodings of all possible values can be precomputed



(Lee et al., 2019)
https://aclweb.org/anthology/P19-1546/

Dialogue State as SQL

(Yu et al., 2019)
http://arxiv.org/abs/1909.05378
http://arxiv.org/abs/1906.02285

- User goal is a query → why not SQL query?
- Text-to-SQL models used for tracking
  - with contextual enhancements, input:
    - all user inputs so far
    - previous system response
    - database schema
- Seq2seq-based model example:
  - hierarchical LSTM for encoding user & system
  - database column embeddings
    = averaged embeddings over table + column name
  - decoder:
    - decide between SQL keyword vs. column
    - then select which keyword / column via softmax
- So far, experimental – performance is low

D_1	Database about student dormitories containing 5 tables	
Q_1	What are the names of all the dorms?	INFORM_SQL
S_1	SELECT dorm name FROM dorm	
A_1	(Result table with many entries)	
R_1	This is the list of the names of all the dorms.	CONFIRM_SQL
Q_2	Which of those dorms have a TV lounge?	INFORM_SQL
S_2	SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV Lounge'	
A_2	(Result table with many entries)	
R_2	This shows the names of dorms with TV lounges.	CONFIRM_SQL
Q_3	What dorms have no study rooms as amenities?	AMBIGUOUS
R_3	Do you mean among those with TV Lounges?	CLARIFY
Q_4	Yes.	AFFIRM
S_4	SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV Lounge' EXCEPT SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'Study Room'	
A_4	Fawlty Towers	
R_4	Fawlty Towers is the name of the dorm that has a TV lounge but not a study room as an amenity.	CONFIRM_SQL
	...	
Q_8	Thanks!	THANK_YOU
R_8	You are welcome.	WELCOME

Summary

- State tracking is needed to maintain user goal over multiple turns
- Best to make the state probabilistic – **belief state**
- Architectures – many options
  - good NLU + rules – works well!
  - **static** (sliding-window or with rule-based value update)
    vs. **dynamic** (modelling dialogue as sequence, modelling value update)
  - with vs. without NLU
  - **classification** vs. candidate **ranking** vs. span **tagging** vs. **generation**
    - classifiers are more accurate than rankers but slower, limited to seen values
    - tagging is a rather new approach, works nicely but probably slow
  - using BERT & co. as usual – good but slow
  - incremental – not used too much so far

Thanks

Contact us:

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

Labs in 10 minutes
Lab Projects Intro

Next Tue 9:50am: Dialogue Policy

Get these slides here:

http://ufal.cz/npfl099

References/Inspiration/Further:

- Filip Jurčíček's slides (Charles University): https://ufal.mff.cuni.cz/~jurcicek/NPFL099-SDS-2014LS/
- Milica Gašić's slides (Cambridge University): http://mi.eng.cam.ac.uk/~mg436/teaching.html
- Henderson (2015): Machine Learning for Dialog State Tracking: A Review https://ai.google/research/pubs/pub44018