

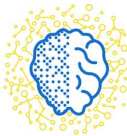


Dialog Systems (2)

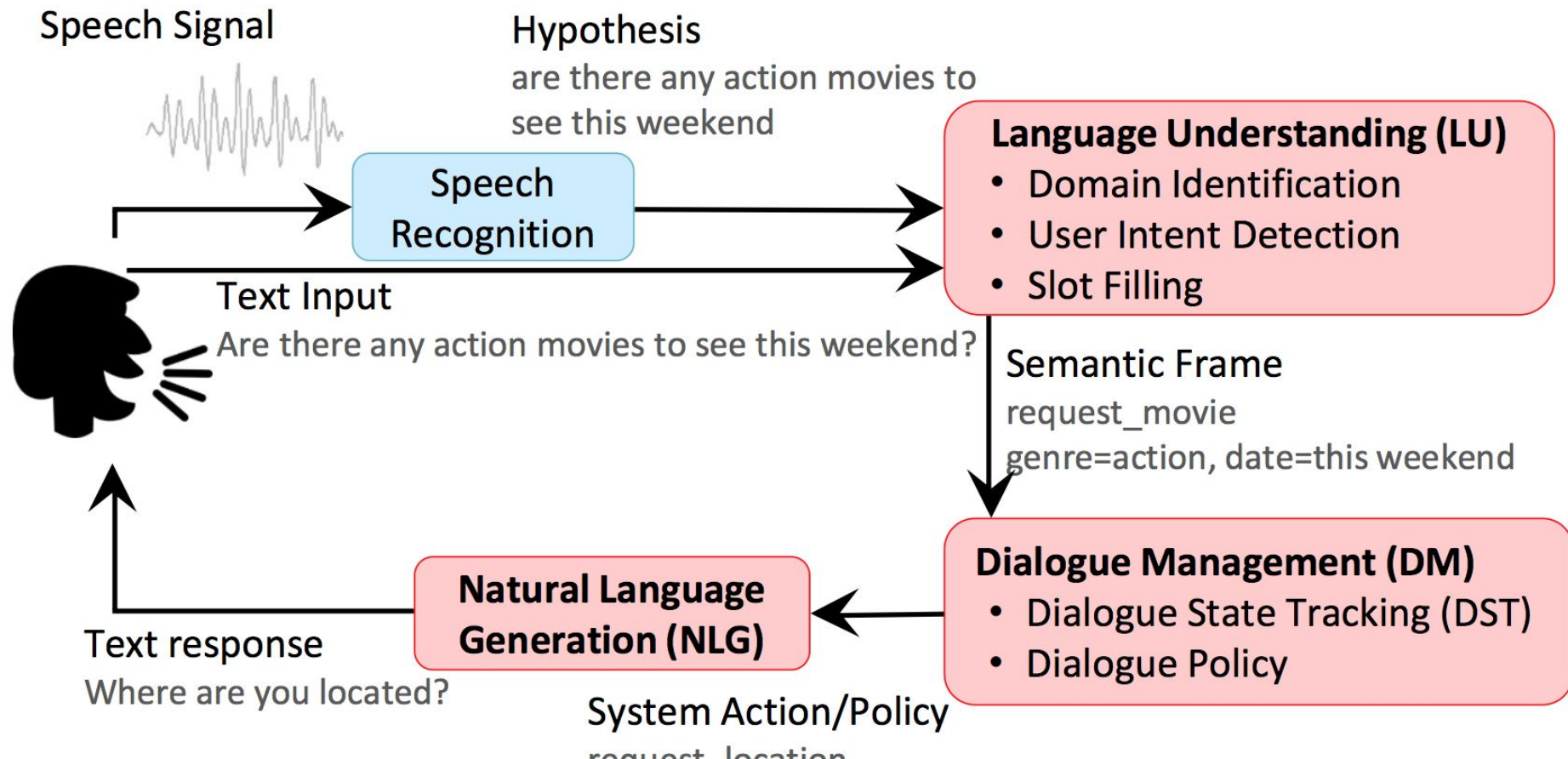
Modern Perspective
by

Valentin Malykh

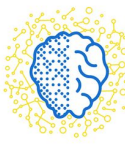
valentin.malykh@phystech.edu



Task-Oriented Dialogue System



Language Modeling



Goal: estimate the probability of a word sequence

$$P(w_1, \dots, w_m)$$

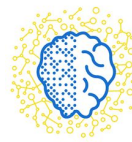
Example task: determinate whether a sequence is grammatical or makes more sense



recognize speech
or
wreck a nice beach

If $P(\text{recognize speech})$
> $P(\text{wreck a nice beach})$

Output =
“recognize speech”



N-Gram Language Modeling

Goal: estimate the probability of a word sequence

$$P(w_1, \dots, w_m)$$

N-gram language model

- Probability is conditioned on a window of (n-1) previous words

$$P(w_1, \dots, w_m) = \prod_{i=1}^m P(w_i \mid w_1, \dots, w_{i-1}) \approx \prod_{i=1}^m P(w_i \mid w_{i-(n-1)}, \dots, w_{i-1})$$

- Estimate the probability based on the training data

$$P(\text{beach} \mid \text{nice}) = \frac{C(\text{nice beach})}{C(\text{nice})}$$

Count of "nice beach" in the training data

Count of "nice" in the training data



N-Gram Language Modeling

Training data:

- The dog ran
- The cat jumped

$$P(\text{jumped} \mid \text{dog}) = \cancel{0} \text{ } 0.0001$$

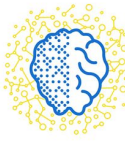
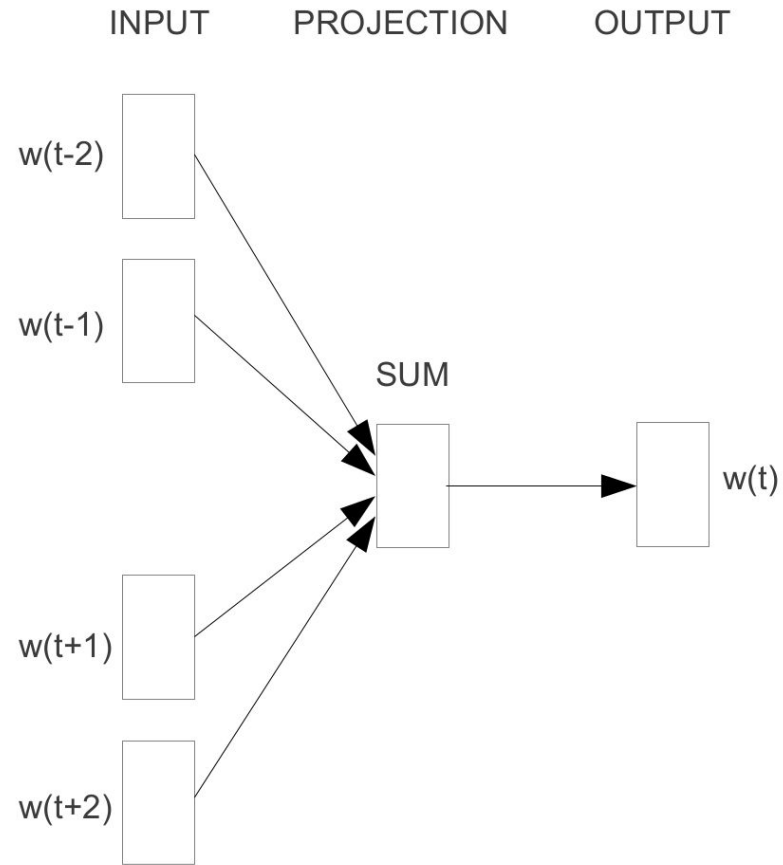
$$P(\text{ran} \mid \text{cat}) = \cancel{0} \text{ } 0.0001$$

give some small
probability
→ smoothing

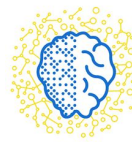
- The probability is not accurate.
- The phenomenon happens because we cannot collect all the possible text in the world as training data.

Modern Language Modeling

- Word2Vec
- fastText
- etc.



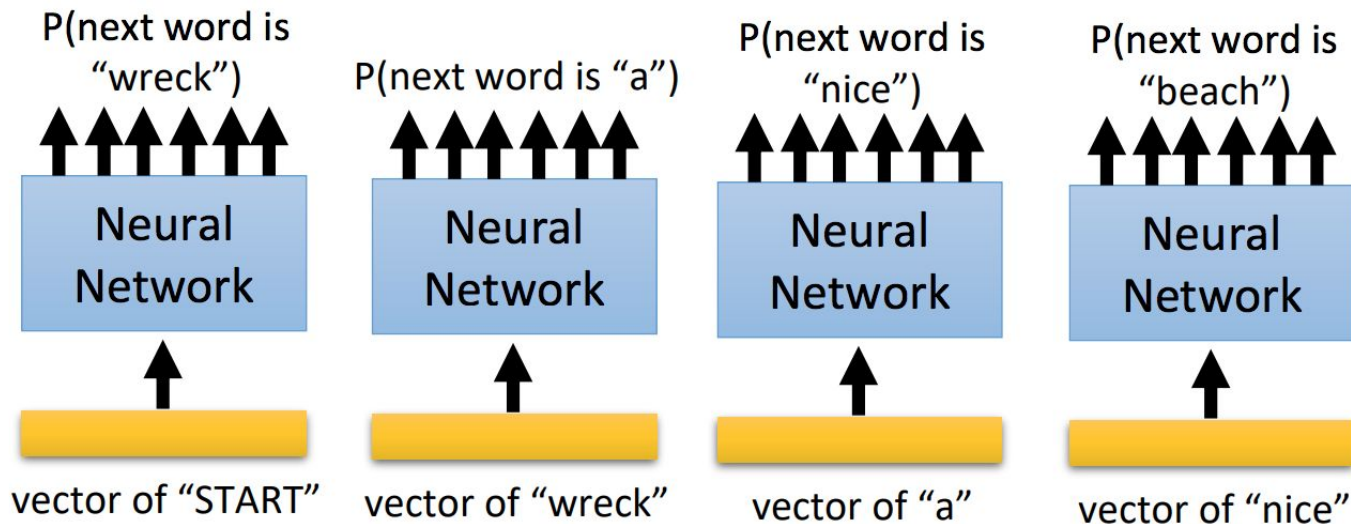
CBOW



Neural Language Modeling

Idea: estimate $P(w_i \mid w_{i-(n-1)}, \dots, w_{i-1})$ not from count, but from the NN prediction

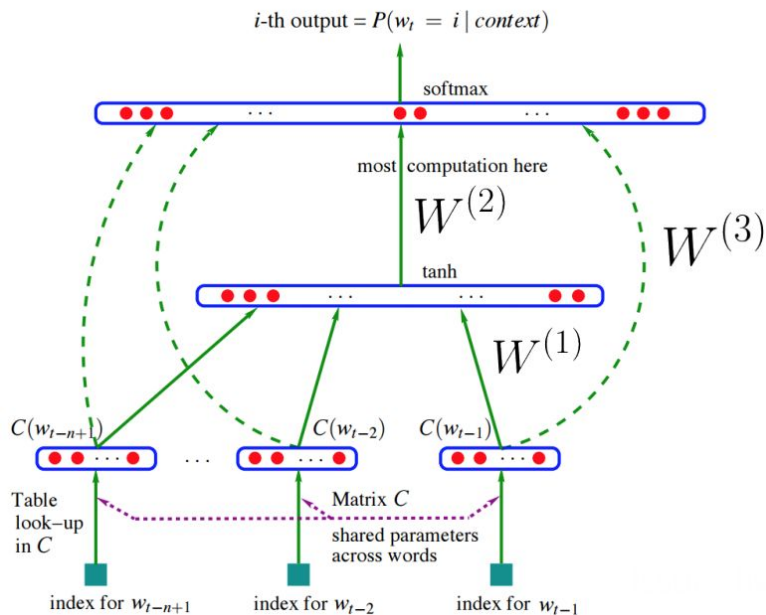
$$P(\text{"wreck a nice beach"}) = P(\text{wreck} \mid \text{START})P(a \mid \text{wreck})P(\text{nice} \mid a)P(\text{beach} \mid \text{nice})$$





Neural Language Modeling

$$\hat{y} = \text{softmax}(W^{(2)}\sigma(W^{(1)}x + b^{(1)}) + W^{(3)}x + b^{(3)})$$



Probability distribution
of the next word

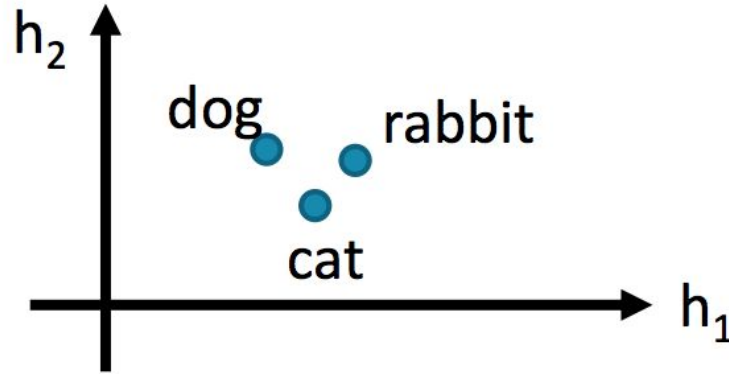


context vector

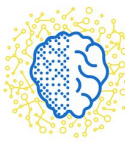
Neural Language Modeling



The input layer (or hidden layer) of the related words are close



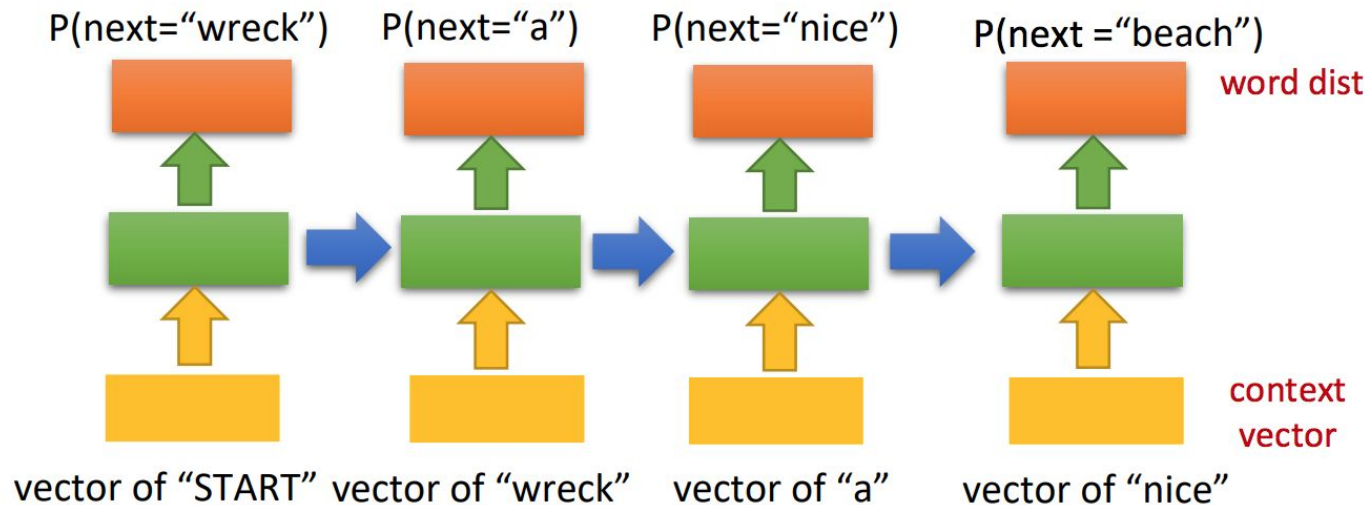
If $P(\text{jump}|\text{dog})$ is large, $P(\text{jump}|\text{cat})$ increase accordingly (even there is not "... cat jump ..." in the data)

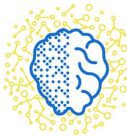


RNNLM

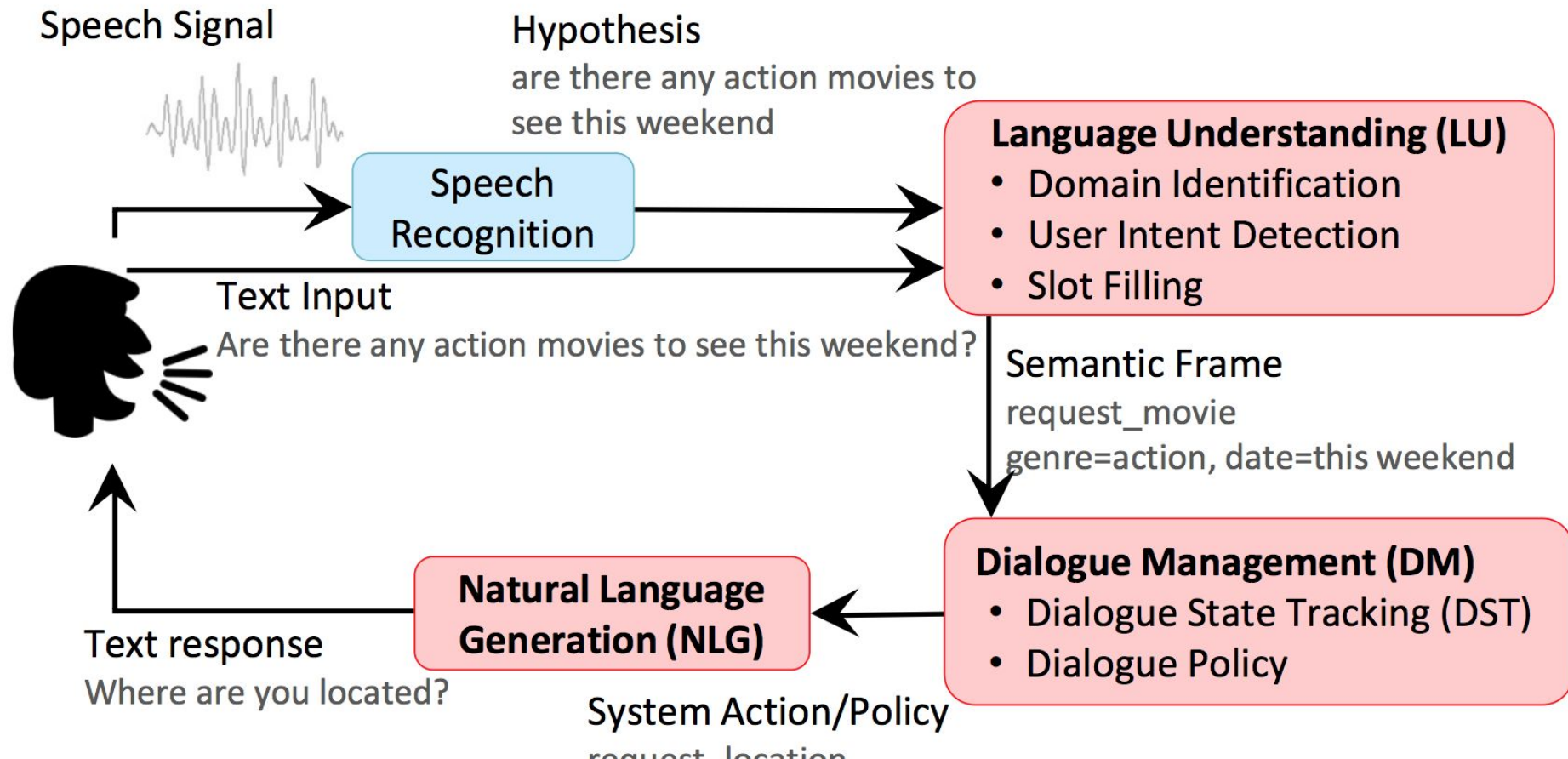
Idea: condition the neural network on all previous words and tie the weights at each time step

Assumption: temporal information matters

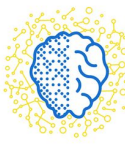




Task-Oriented Dialogue System



Natural Language Generation

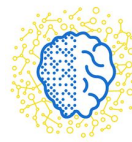


inform(name=Seven_Days, foodtype=Chinese)



Seven Days is a nice Chinese restaurant

Template-Based NLG

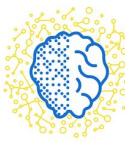


Define a set of rules to map frames to NL

Semantic Frame	Natural Language
confirm()	"Please tell me more about the product your are looking for."
confirm(area=\$V)	"Do you want somewhere in the \$V?"
confirm(food=\$V)	"Do you want a \$V restaurant?"
confirm(food=\$V,area=\$W)	"Do you want a \$V restaurant in the \$W."

Pros: simple, error-free, easy to control

Cons: time-consuming, rigid, poor scalability



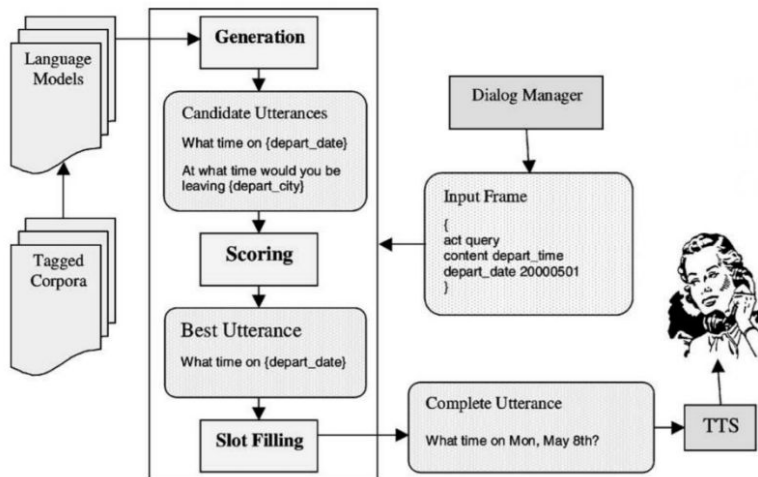
Class-Based LM NLG (Oh and Rudnicky, 2000)

Class-based language modeling

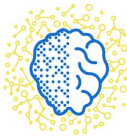
$$P(X | c) = \sum_t \log p(x_t | x_0, x_1, \dots, x_{t-1}, c)$$

NLG by decoding $X^* = \arg \max_X P(X | c)$

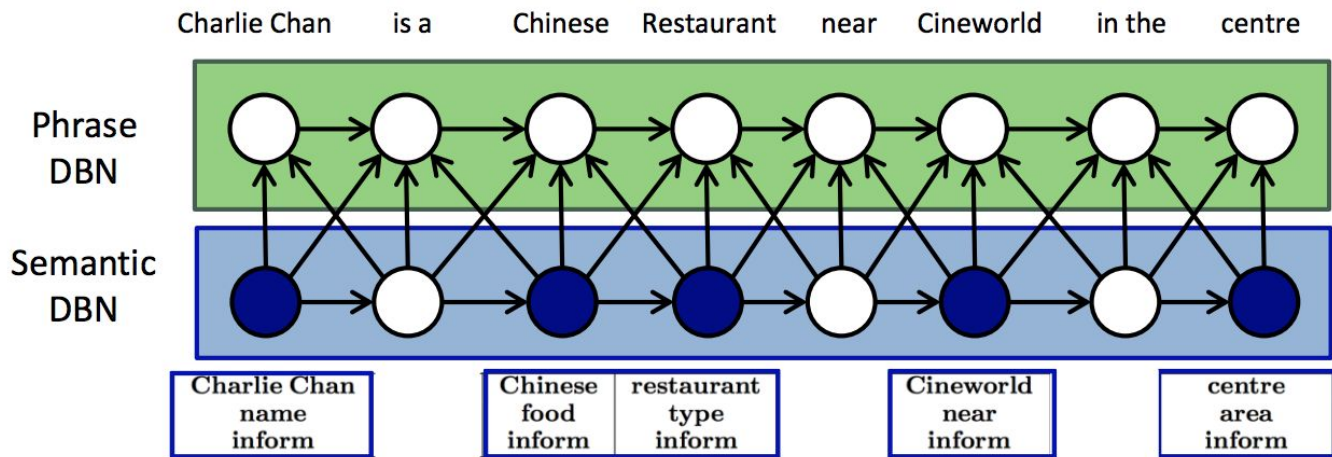
Classes:
inform_area
inform_address
...
request_area
request_postcode



Pros: easy to implement/
understand, simple rules
Cons: computationally inefficient



Phrase-based NLG



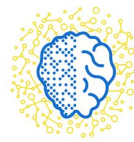
Inform(name=Charlie Chan, food=Chinese, type= restaurant, near=Cineworld, area=centre)

realization phrase semantic stack

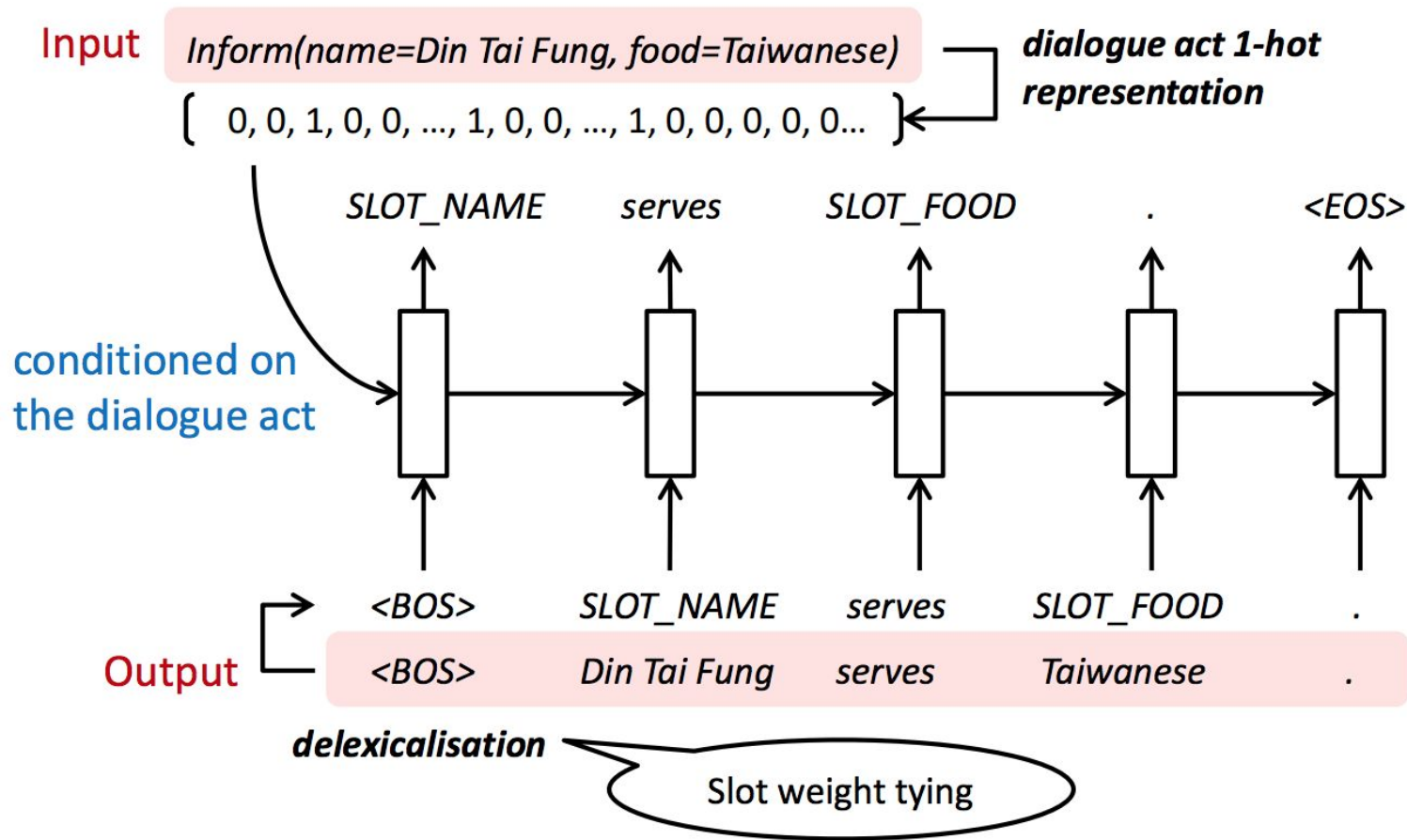
r_t	s_t	h_t	l_t
<s>	START	START	START
The Rice Boat	inform(name(X))	X	inform(name)
is a	inform	inform	EMPTY
restaurant	inform(type(restaurant))	restaurant	inform(type)
in the	inform(area)	area	inform
riverside	inform(area(riverside))	riverside	inform(area)
area	inform(area)	area	inform
that	inform	inform	EMPTY
serves	inform(food)	food	inform
French	inform(food(French))	French	inform(food)
food	inform(food)	food	inform
</s>	END	END	END

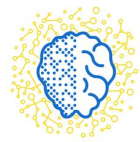
Pros: efficient, good performance

Cons: require semantic alignments



RNN-Based LM NLG





RNN-Based LM NLG: an issue

Issue: semantic repetition

- Din Tai Fung is a great *Taiwanese* restaurant that serves *Taiwanese*.
- Din Tai Fung is a *child friendly* restaurant, and also *allows kids*.

Deficiency in either model or decoding (or both)

Mitigation

- Post-processing rules (Oh & Rudnicky, 2000)
- Gating mechanism (Wen et al., 2015)
- Attention (Mei et al., 2016; Wen et al., 2015)

Semantic Conditioned LSTM (Wen et al., 2015)



Original LSTM cell

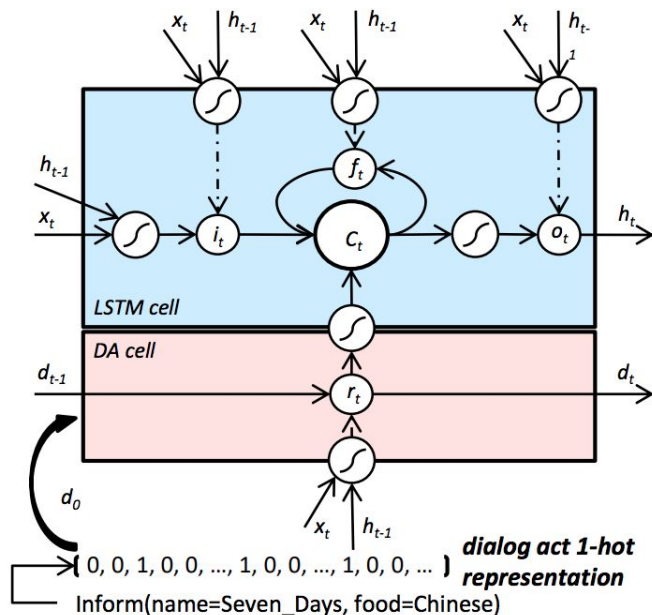
$$\begin{aligned} \mathbf{i}_t &= \sigma(\mathbf{W}_{wi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1}) \\ \mathbf{f}_t &= \sigma(\mathbf{W}_{wf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1}) \\ \mathbf{o}_t &= \sigma(\mathbf{W}_{wo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1}) \\ \hat{\mathbf{c}}_t &= \tanh(\mathbf{W}_{wc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1}) \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t \\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \end{aligned}$$

Dialog Act (DA) cell

$$\begin{aligned} \mathbf{r}_t &= \sigma(\mathbf{W}_{wr}\mathbf{x}_t + \mathbf{W}_{hr}\mathbf{h}_{t-1}) \\ \mathbf{d}_t &= \mathbf{r}_t \odot \mathbf{d}_{t-1} \end{aligned}$$

Modify C

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t + \tanh(\mathbf{W}_{dc}\mathbf{d}_t)$$





Attentive Encoder-Decoder for NLG

Slot & value embedding

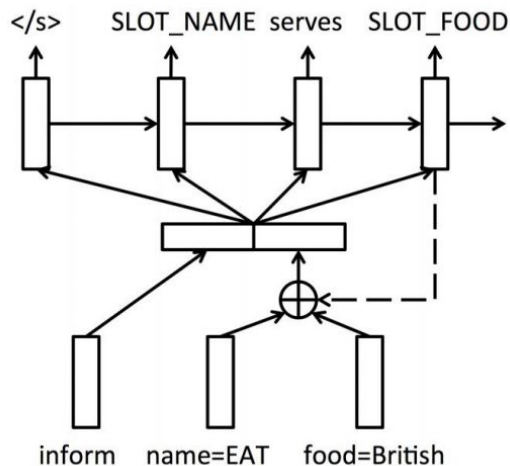
$$\mathbf{z}_i = \mathbf{s}_i + \mathbf{v}_i$$

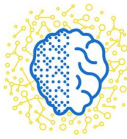
Attentive meaning representation

$$e_{ti} = \mathbf{v}^T \tanh(\mathbf{W}_{hm} \mathbf{h}_{t-1} + \mathbf{W}_{zm} \mathbf{z}_i)$$

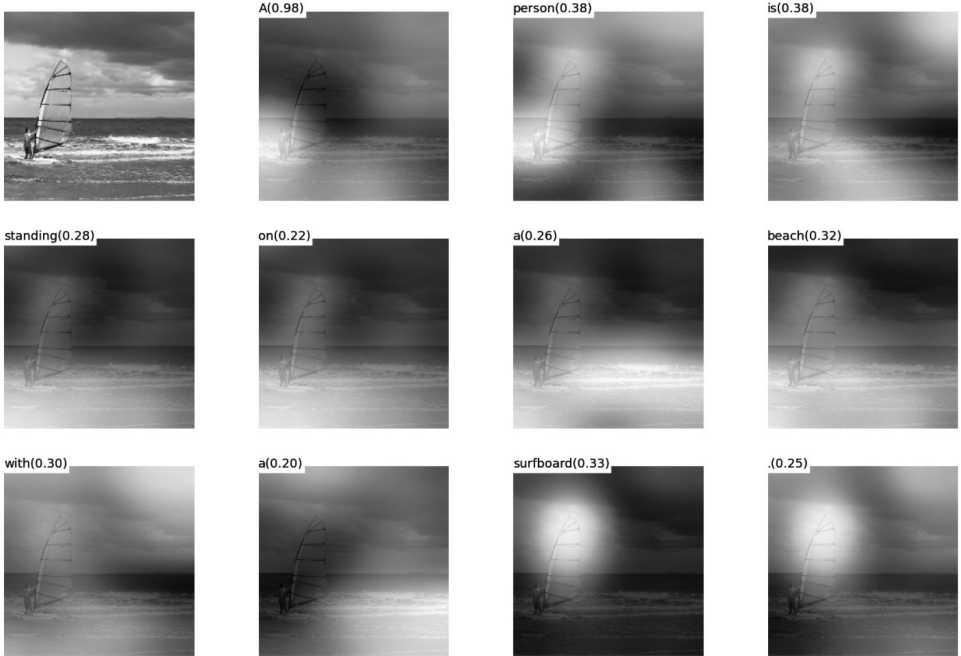
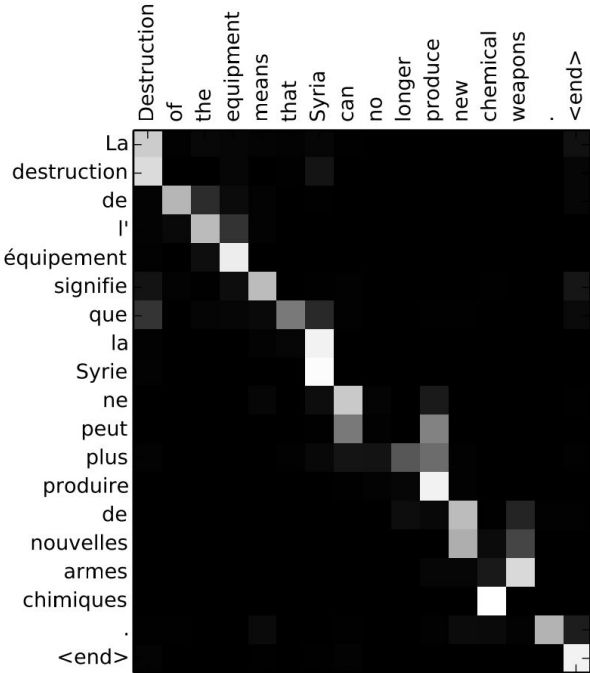
$$\alpha_{ti} = \text{softmax}(e_{ti})$$

$$\mathbf{d}_t = \mathbf{a} \oplus \sum_i \alpha_{ti} \mathbf{z}_i$$





Attention - I



(b) A person is standing on a beach with a surfboard.



Attention - II

Teaching Machine Read and Comprehend (Herman et al. - 2015)

by *ent423* ,*ent261* correspondent updated 9:49 pm et ,thu
march 19 ,2015 (*ent261*) a *ent114* was killed in a parachute
accident in *ent45* ,*ent85* ,near *ent312* ,a *ent119* official told
ent261 on wednesday .he was identified thursday as
special warfare operator 3rd class *ent23* ,29 ,of *ent187* ,
ent265 .` *ent23* distinguished himself consistently
throughout his career .he was the epitome of the quiet
professional in all facets of his life ,and he leaves an
inspiring legacy of natural tenacity and focused

. . .

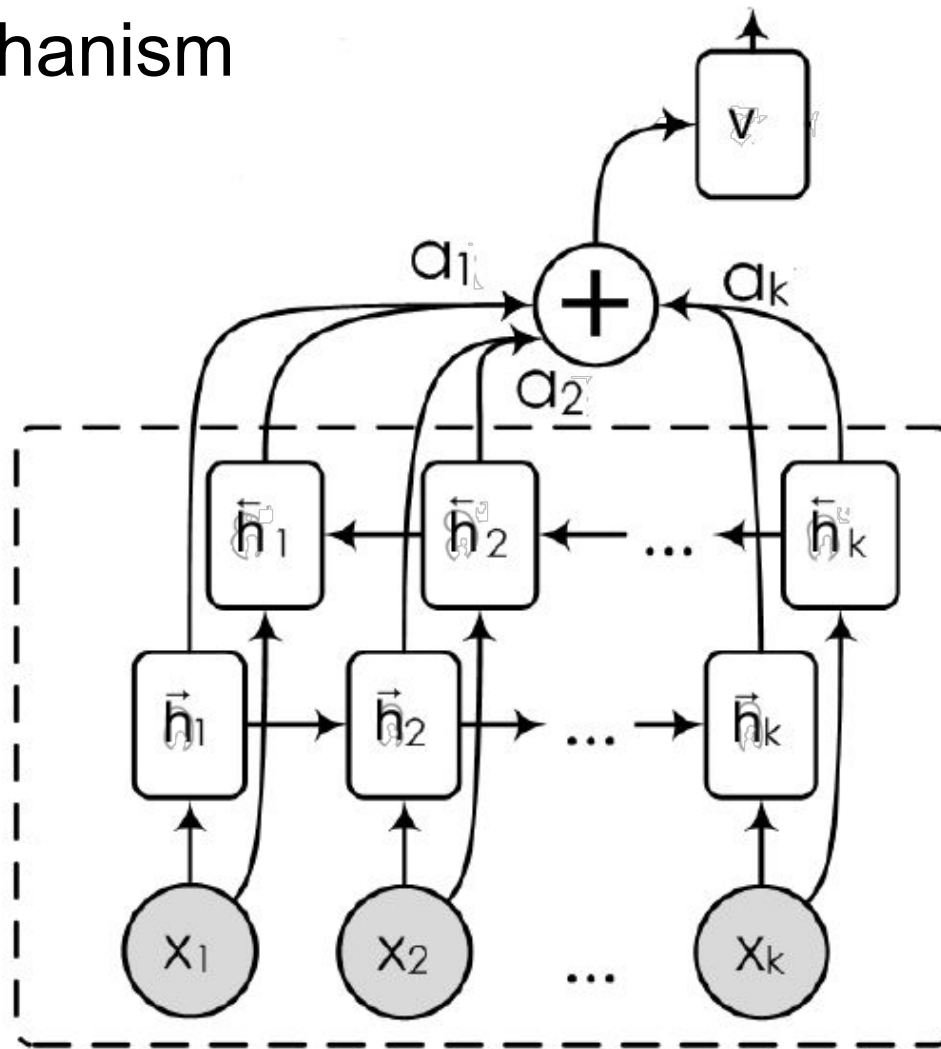
ent119 identifies deceased sailor as **X** ,who leaves behind
a wife

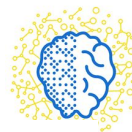
by *ent270* ,*ent223* updated 9:35 am et ,mon march 2 ,2015
(*ent223*) *ent63* went familial for fall at its fashion show in
ent231 on sunday ,dedicating its collection to ``mamma "
with nary a pair of ``mom jeans " in sight .*ent164* and *ent21* ,
who are behind the *ent196* brand ,sent models down the
runway in decidedly feminine dresses and skirts adorned
with roses ,lace and even embroidered doodles by the
designers ' own nieces and nephews .many of the looks
featured saccharine needlework phrases like ``i love you ,

. . .

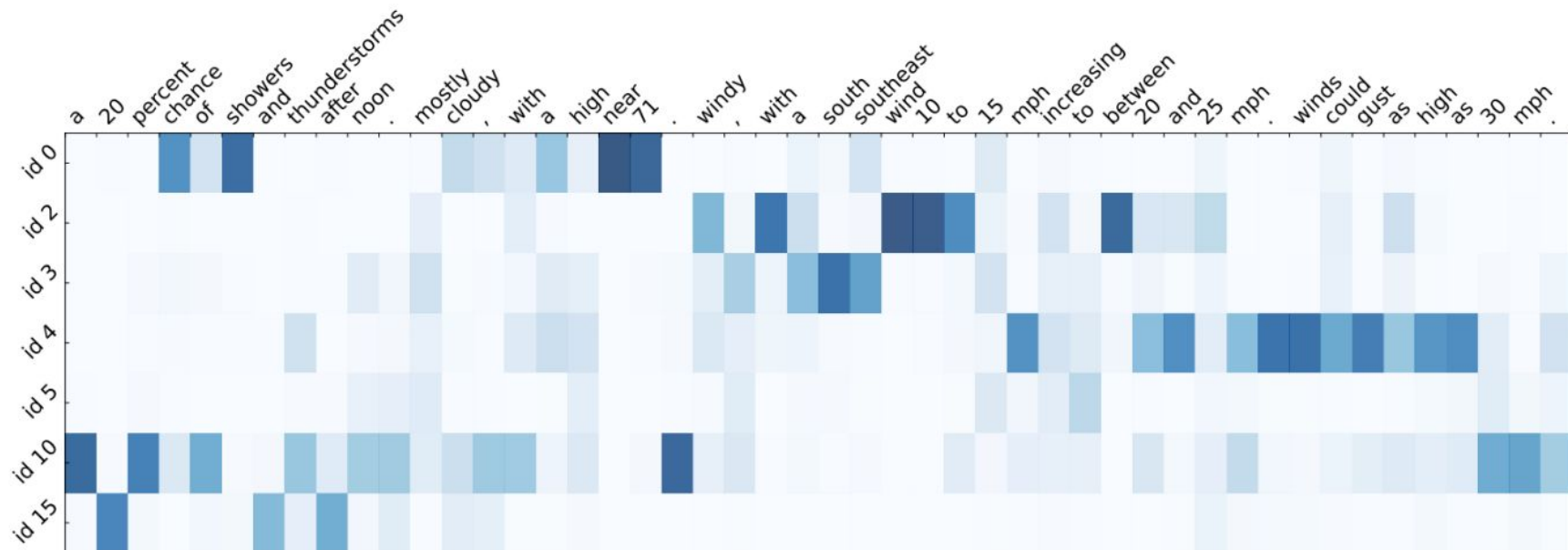
X dedicated their fall fashion show to moms

Attention Mechanism





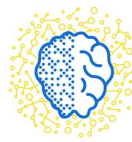
Attention Heat Map



Record details:

id-0: temperature(time=06-21, min=52, mean=63, max=71); id-2: windSpeed(time=06-21, min=8, mean=17, max=23);
id-3: windDir(time=06-21, mode=SSE); id-4: gust(time=06-21, min=0, mean=10, max=30);
id-5: skyCover(time=6-21, mode=50-75); id-10: precipChance(time=06-21, min=19, mean=32, max=73);
id-15: thunderChance(time=13-21, mode=SChc)

Figure 3: An example generation for a set of records from WEATHERGOV.



Structural NLG (Dušek and Jurčiček, 2016)

Goal: NLG based on the syntax tree

Encode trees as sequences

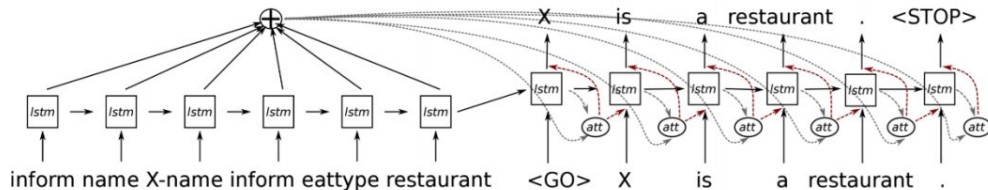
Seq2Seq model for generation

(<root> <root> ((X-name n:subj) be v:fin ((Italian adj:attr) restaurant n:obj (river n:near+X))))

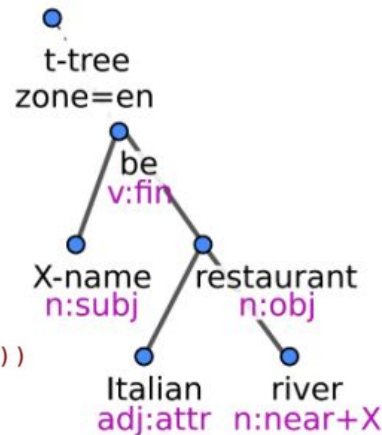
X-name n:subj be v:fin Italian adj:attr restaurant n:obj river n:near+X

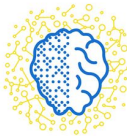


X is an Italian restaurant near the river.



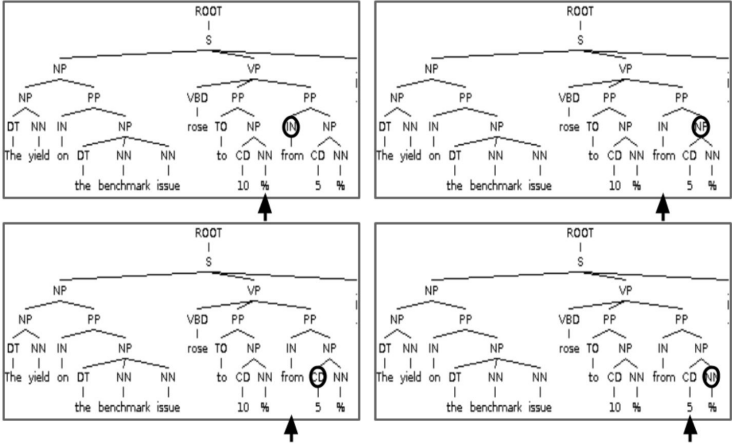
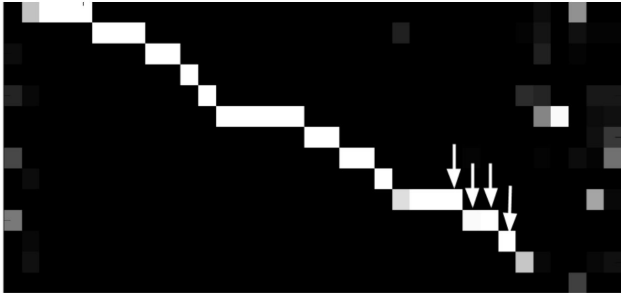
inform(name=X-name,type=placetoeat,eatype=restaurant,
area=riverside,food=Italian)

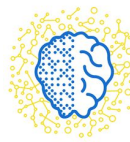




Attentive Tree Generator

Grammar as a Foreign Language
(Vinyals et al. - 2014)





Contextual NLG (Dušek and Jurčiček, 2016)

Goal: adapting users' way of speaking, providing contextaware responses

- Context encoder
- Seq2Seq model

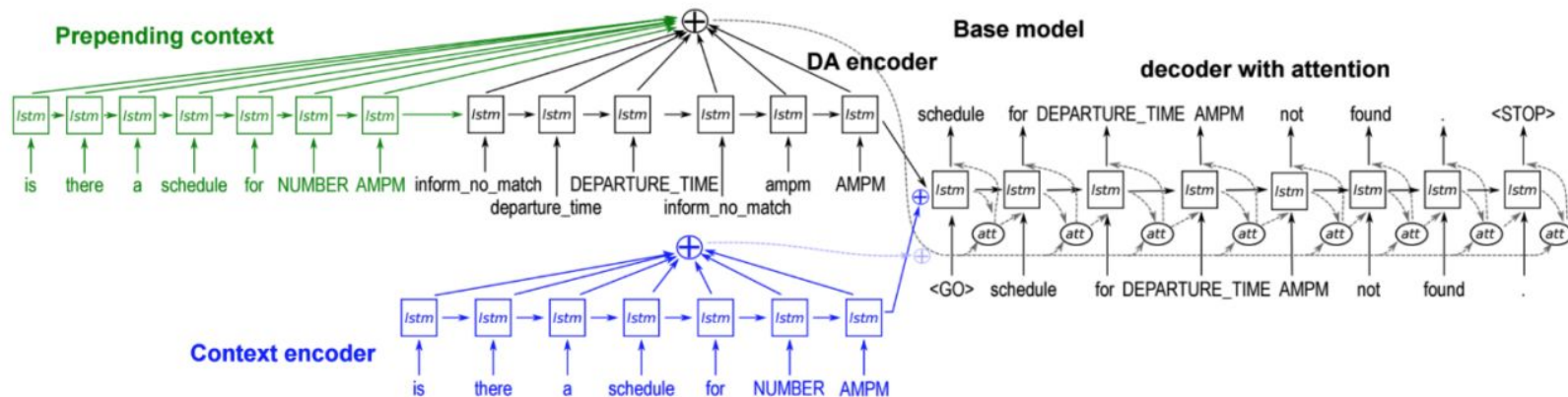
preceding user utterance
is there another option

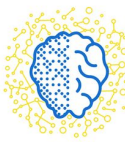
context-aware additions

inform(line=M102, direction=Herald Square, vehicle=bus, departure_time=9:01am, from_stop=Wall Street) **typical NLG**

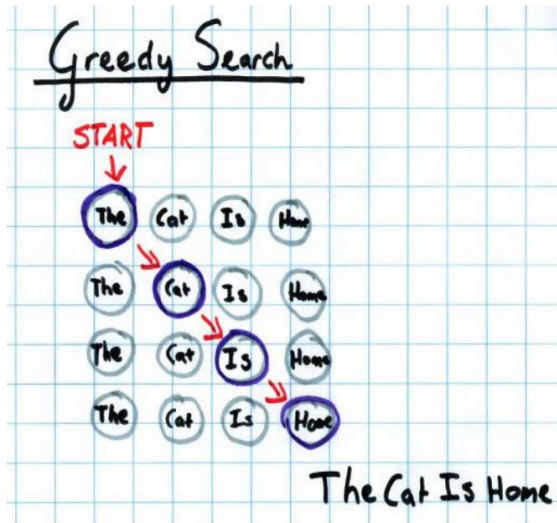
~~Take bus-line M102 from Wall Street to Herald Square at 9:01am.~~

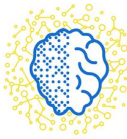
There is a bus at 9:01am from Wall Street to Herald Square using line M102.
contextually bound response





Decoder Sampling Strategy

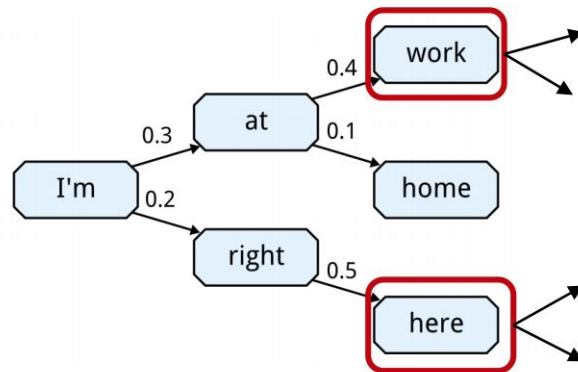
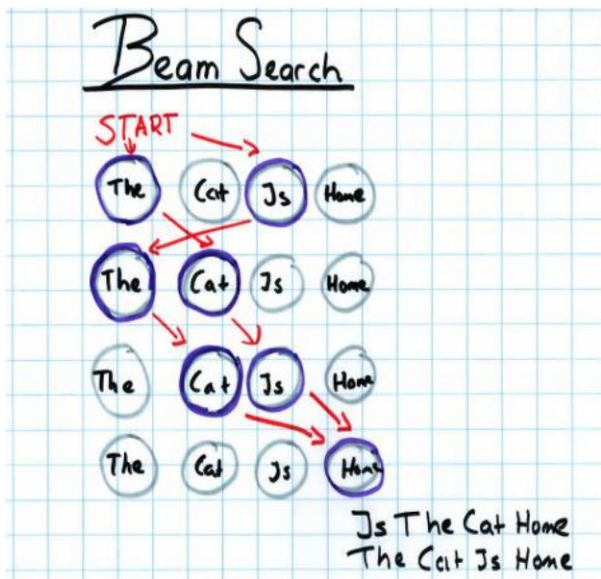


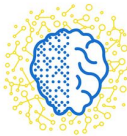


Decoder Sampling Strategy

Beam Search

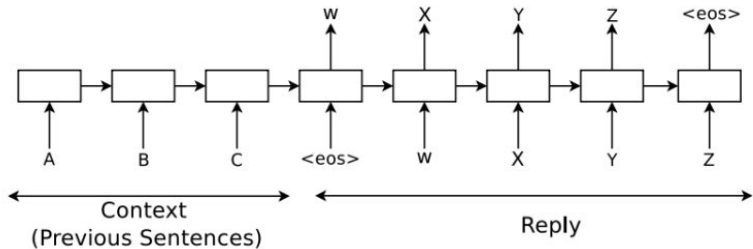
Select the next k-best words and keep a beam with width=k for following decoding





Chit-Chat

Seq2Seq



Human: *what do you think about tesla ?*

Machine: *he 's a good conductor .*

Human: *what do you think about bill gates ?*

Machine: *he 's a good man .*

Human: *What do you think about messi ?*

Machine: *he 's a great player .*

Human: *what do you think about cleopatra ?*

Machine: *oh , she 's very regal .*

Human: *what do you think about england during the reign of elizabeth ?*

Machine: *it was a great place .*

Human: *what is the purpose of life ?*

Machine: *to serve the greater good .*

Human: *what is the purpose of living ?*

Machine: *to live forever .*

Human: *what is the purpose of existence ?*

Machine: *to find out what happens when we get to the planet earth .*

Human: *where are you now ?*

Machine: *i 'm in the middle of nowhere .*

Human: *what is the purpose of dying ?*

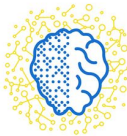
Machine: *to have a life .*

Human: *what is the purpose of being intelligent ?*

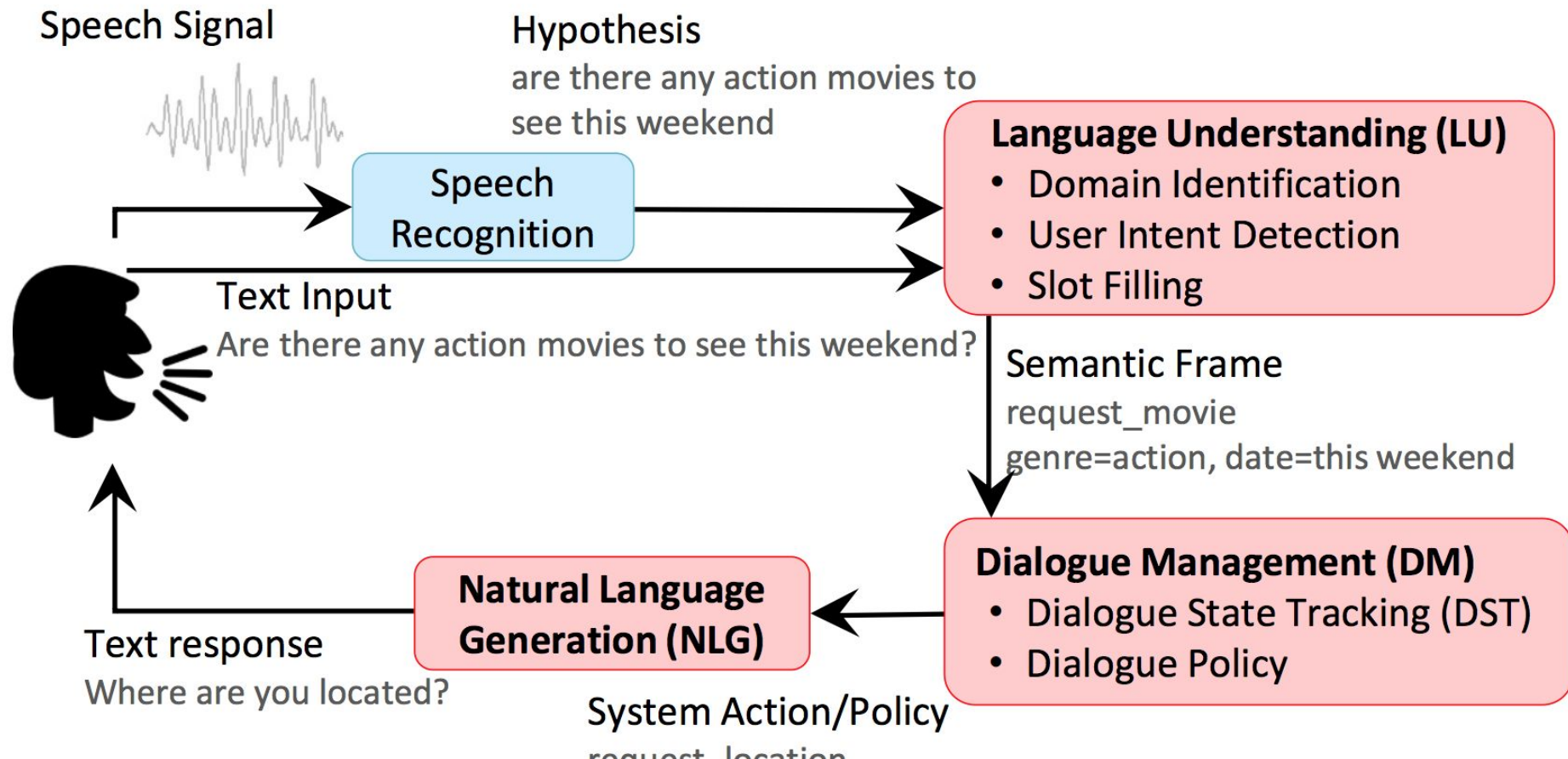
Machine: *to find out what it is .*

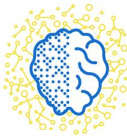
Human: *what is the purpose of emotions ?*

Machine: *i don 't know .*



Task-Oriented Dialogue System





Example Dialog

Hello, how may I help you?

I'm looking for a Thai restaurant.

request (restaurant; foodtype=Thai)

What part of town do you have in mind?

Something in the centre.

inform (area=centre)

Bangkok city is a nice place, it is in the centre of town and it serves Thai food.

What's the address?

request (address)

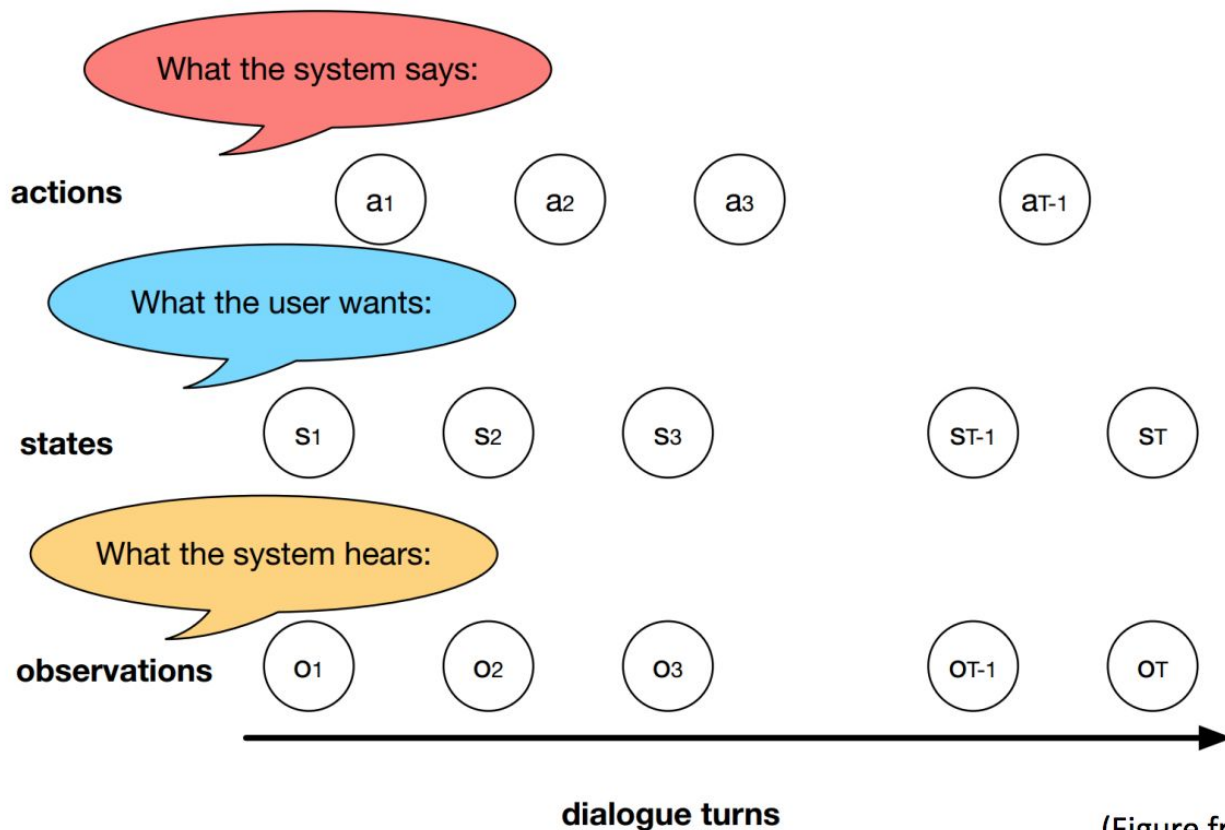
Bangkok city is a nice place, their address is 24 Green street.

Thank you, bye.

bye ()



Dialog Management



(Figure from Gašić)



Dialog State Tracker

Maintain a probabilistic distribution instead of a 1-best prediction for better robustness to recognition errors, ambiguous input, NLU errors

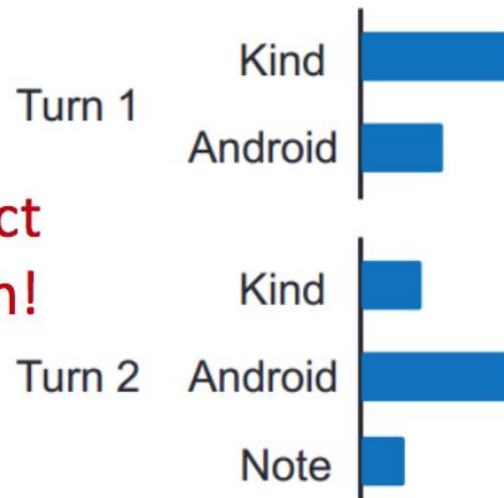
Turn 1
Kind
Android

Turn 2
Note
Android

Turn 1	
Kind	0.5
Android	0.3

Turn 2	
Note	0.4
Android	0.3

**Incorrect
for both!**





Dialog State Tracker

Maintain a probabilistic distribution instead of a 1-best prediction for better robustness to recognition errors, ambiguous input, NLU errors

Slot	Value
# people	5 (0.5)
time	5 (0.5)

Slot	Value
# people	3 (0.8)
time	5 (0.8)





1-Best Input w/o State Tracking

turn

observations

states

actions

1.

I'm looking for a Thai restaurant.

hello(type=restaurant)	0.6
inform(type=restaurant, food=Thai)	0.4

0.6	0.4
R	O

type

food

You are looking for a restaurant right?

2.

Thai.

hello()	0.5
inform(food=Turkish)	0.3
inform(food=Thai)	0.2

0.6	0.4
R	O

type

food

You are looking for a restaurant right?



N-Best Inputs w/o State Tracking

turn

observations

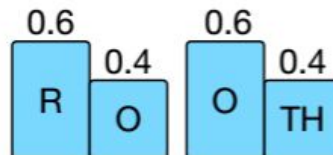
states

actions

1.

I'm looking for a Thai restaurant.

hello(type=restaurant)	0.6
inform(type=restaurant, food=Thai)	0.4

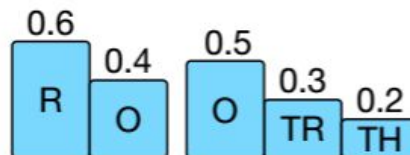


What kind of food would you like?

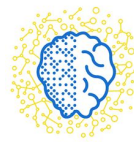
2.

Thai.

hello()	0.5
inform(food=Turkish)	0.3
inform(food=Thai)	0.2



What kind of food would you like?



N-Best Inputs w/ State Tracking

turn

1.

I'm looking for a Thai restaurant.

observations

hello(type=restaurant)	0.6
inform(type=restaurant, food=Thai)	0.4

states

0.6	0.4	0.6	0.4
R	O	O	TH

actions

What kind of food would you like?

type

food

2.

Thai.

hello()	0.5
inform(food=Turkish)	0.3
inform(food=Thai)	0.2

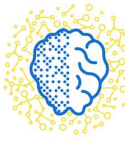
0.6	0.4	0.4	0.3	0.3
R	O	TH	TR	O

type

food

Did you say Thai or Turkish?

Dialog State Tracking Challenge



Definition

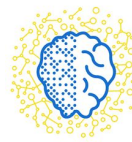
Representation of the system's belief of the user's goal(s) at any time during the dialogue

Challenge

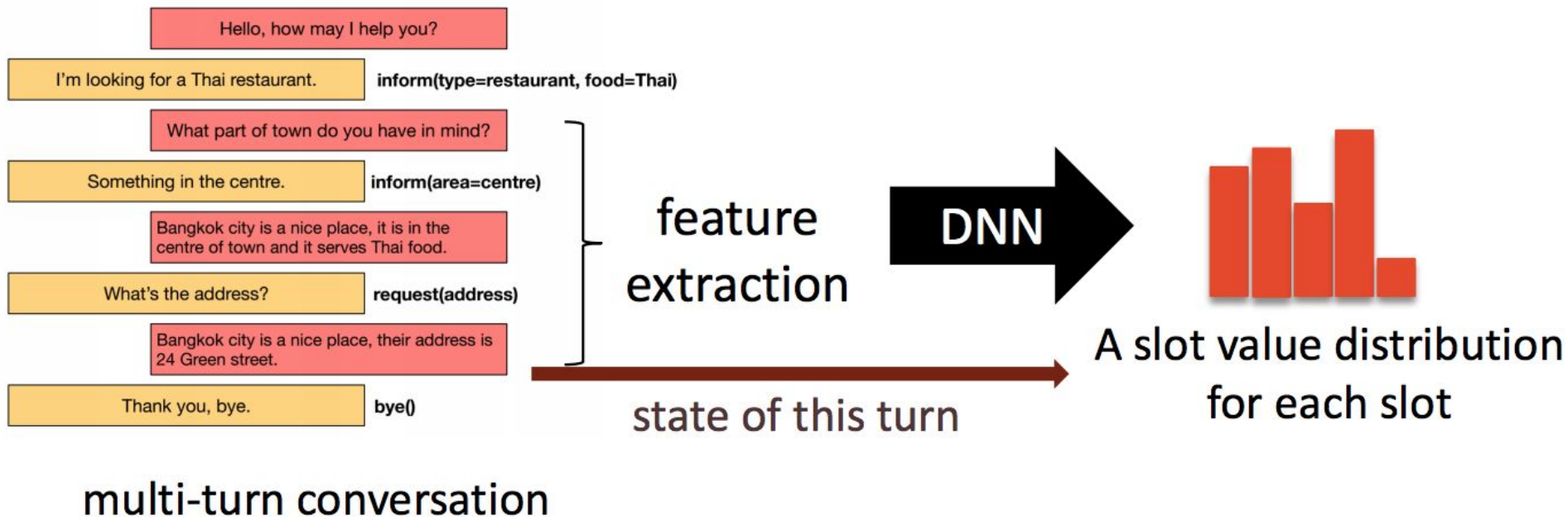
How to define the state space?

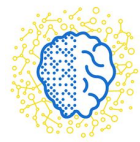
How to tractably maintain the dialogue state?

Which actions to take for each state?



DNN for DST

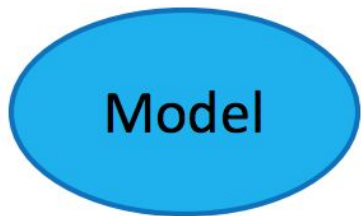




Sequence-based DST



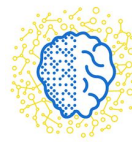
- Sequence of observations labeled w/ dialogue state



- Recurrent neural networks (RNN)



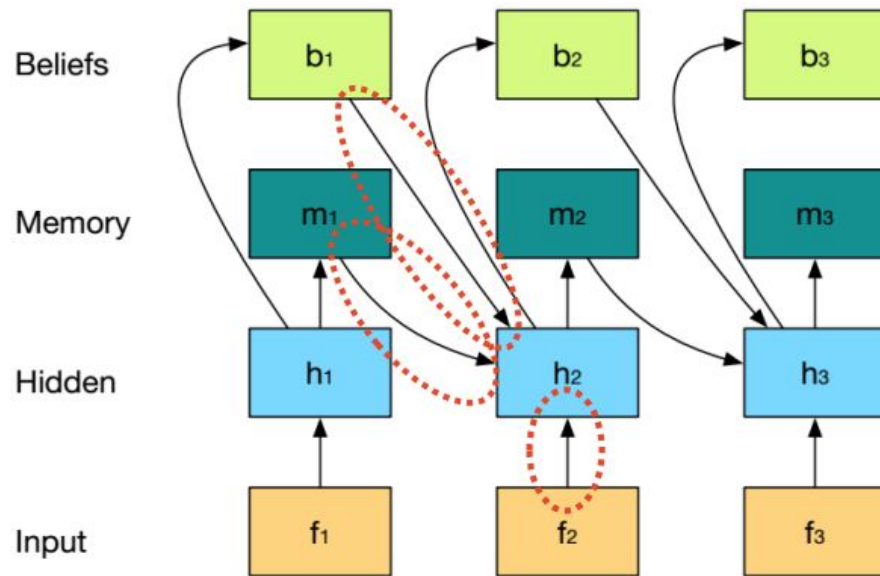
- Distribution over dialogue states
 - Dialogue State Tracking



Sequence-based DST w/ memory

Idea: internal memory for representing dialogue context

- Input
 - most recent dialogue turn
 - last machine dialogue act
 - dialogue state
 - memory layer
- Output
 - update its internal memory
 - distribution over slot values



DST Evaluation



Metrics:

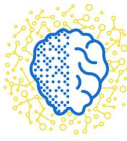
Tracked state accuracy with respect to user goal

L2-norm of the hypothesized dist. and the true label

Machine translation metrics as BLEU, METEOR, etc. do not work.

Recall/Precision/F-measure on individual slots

DeepPavlov installation



```
pip install deeppavlov
```

depends on python 3.6



DeepPavlov installation (Docker)

Install the Docker following the instructions:

<https://docs.docker.com/docker-for-windows/install>

Then go to console and get the container by the following command:

`docker pull altinsky/deeppavlov_jupyter:cuda-9.0`

Run the container with DeepPavlov installation:

`docker run -p 8888:8888 altinsky/deeppavlov_jupyter:cuda-9.0`

Open *<http://127.0.0.1:8888/>* in your browser to access Jupyter Notebook

Upload file with tutorial via Jupyter Notebook

DeepPavlov installation (Docker) (ctd.)



To STOP the container:

docker stop

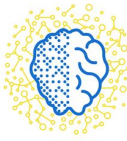
To continue working with your saved container: **docker ps -a** to list saved containers

docker start *_your_container_id_*



Questions?

Acknowledgements



I would like to gratefully thank Vivian Chen from Taiwan National University for permission to use her materials to create this presentation.

Most of architecture pictures are belongs to authors of the papers mentioned. If you do not see any attribution on the picture, most probably I've missed the reference, write me and I add it.