Optimizing Policy via Deep Reinforcement Learning for Dialogue Management

Guanghao Xu, Hyunjung Lee, Myoung-Wan Koo & Jungyun Seo

Sogang Univeristy &Universität Leipzig

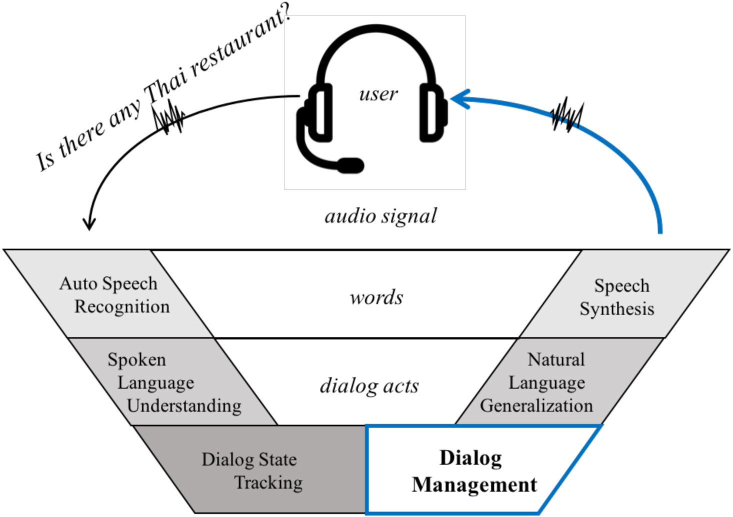
January 17, 2018

서강대학교
SOGANG UNIVERSITY

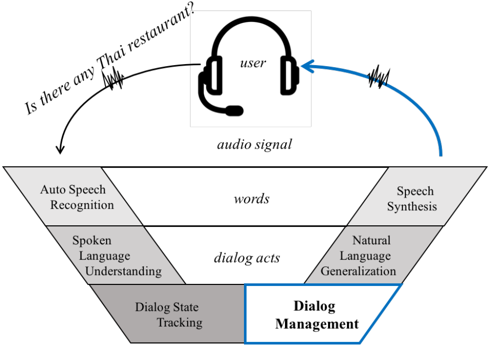
UNIVERSITÄT
LEIPZIG

Overview

Dialogue Manager



Dialogue Manager



Our question 1:

How can Dialog System produce appropriate response in the next turn?

Dialogue Manager

- **Dialogue Manager (DM)** plays a central role in building a successful Spoken Dialog System (SDS)

    1. by apprehending a state of a dialogue in a current turn
    2. by deciding a proper action to take for a next turn
    3. by implementing a human-like agent which interacts with actual users.

Frameworks so far

Rule-based approach

- easy and undemanding to define a set of rules that the system.
- limited flexibility and high maintenance cost.

Reinforcement Learning (RL) framework

- able to learn and train policy over time with experience
- need interventions from a system developer to represent dialogue state, dialogue actions and a reward function which instructs the system on the right track of dialogues.

Goals of this talk

Deep Reinforcement Learning (Deep-RL)

- to learn in an unsupervised way how to control policies in complex environment.
- The agent equipped with deep RL policy surpasses a human expert in several games.
  e.g. *Atari games* [1]

Our question 2:

Which insights of deep RL could be drawn to optimize policy in Dialog Manger without hand-crafted features?

Theoretical Background

Q-function

- Given a policy π :S $\longrightarrow$ A, an RL-agent selects 'best' actions by **maximizing its cumulative discounted reward** R_t,

$$\mathrm{R}_t = r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \ldots + \gamma^{T-1} \cdot r_T$$

where γ is a discount factor and T is a final time step.

- A potential value of actions a in the current state s is estimated by **Q-function** as

$$\mathrm{Q}^*(s, a) = max_\pi E[R_t | s_t = s, a_t = a, \pi]$$

math

Deep-RL

- **Deep Reinforcement Learning** (henceforth, Deep-RL) adopts a function approximator based on deep neural network which is called Q-network.

- Q-network is to estimate the **action-value** function

$$Q(s, a; \theta) \approx Q^*(s, a), \text{ where } \theta \text{ is the parameters}$$

- The Q-network could be constructed in any forms
e.g. *a multi-layer feed forward network, a convolutional neural network, a recurrent neural network.*

Deep RL algorithm

- In deep RL algorithm, the learning agent maintains two Q-networks:
    1. Policy Network
    2. Value Network

Q-Network= *Policy* + *Value* Network

At iteration i

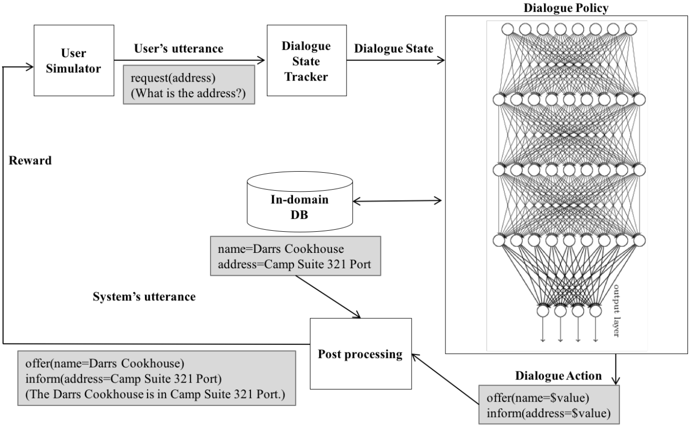
$$\underbrace{L_i(\theta_i) = E[(\underbrace{E[r+\gamma \cdot max_{a'} Q(s', a'; \theta_{i-1}) | s, a]}_{\text{Value Network}} - Q(s, a; \theta_i))^2]}_{\text{Policy Network}}$$

- The **policy network** is trained toward minimizing loss function $L_i(\theta_i)$ that changes at each iteration
- The **value network** estimates value of target action.

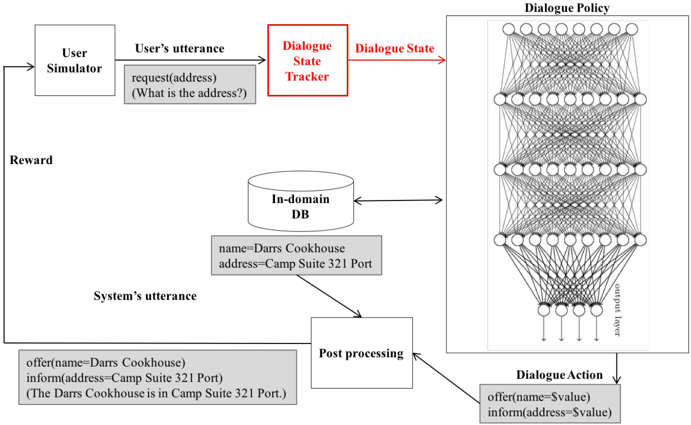
Architecture of Dialogue Manager

Architecture of Dialog Manager

- The architecture of our dialogue manager toward policy optimization.



Dialogue State



Dialogue State

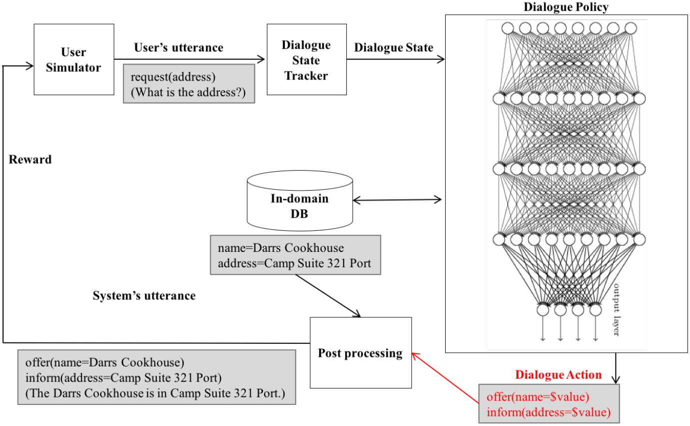
- *Goal*:
  Information that contains what a user wants the system to do should be tracked during entire dialogues to make appropriate response to the user using the SLU results.

- The dialogue state tracker outputs for each turn distributions for each of the three components as follows:
  1. Goal
  2. Method
  3. Requested slots

  ☞ in the form of continuous vector.

- Automatically constructed the dialogue state vector

Dialogue Action



Dialogue Action

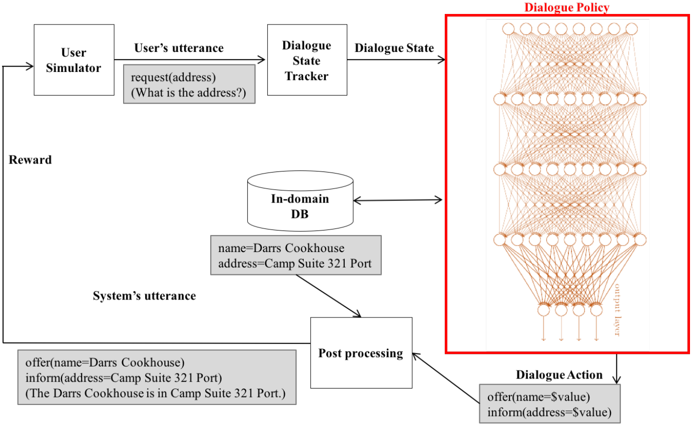
- Agent's responses and user's utterances are converted into **semantic form**

$$\textsc{Act}(\mathit{slot}, \mathit{value})$$

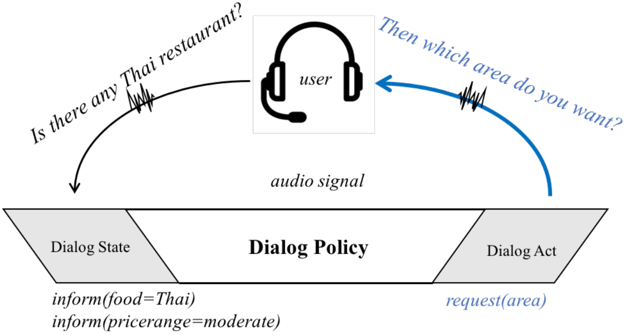
- *Goal*:
  : to have better control over the system's behaviors, rather than directly using raw utterances.

- Due to the sparsity issues, *value* is temporarily left vacant in the level of Q-networks.
- The exact instance of *value* is later added in post-processing step.

Q-network


Dialogue Policy
User Simulator
User's utterance
request(address)
(What is the address?)
Dialogue State Tracker
Dialogue State
Reward
In-domain DB
name=Darrs Cookhouse
address=Camp Suite 321 Port
System's utterance
Post processing
offer(name=Darrs Cookhouse)
inform(address=Camp Suite 321 Port)
(The Darrs Cookhouse is in Camp Suite 321 Port.)
Dialogue Action
offer(name=$value)
inform(address=$value)
output layer

Optimizing Policy



Our question:

Given the input Dialog State s_t, how the **Policy** in DM can derive the optimal output, Dialog Act a_t?

Optimizing Policy

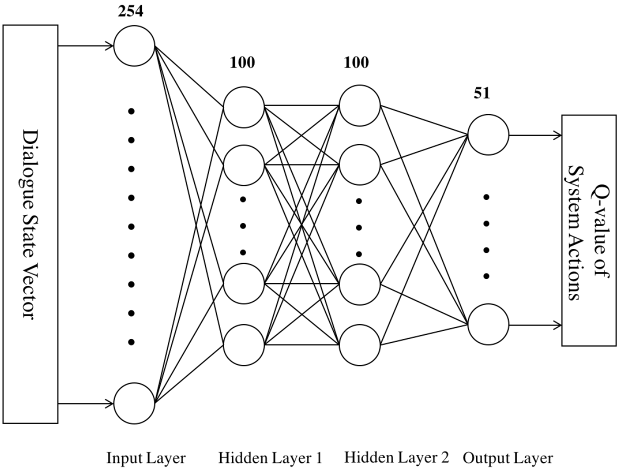
- *Goal*:
  : Q-network should be designed to estimate the **action-value** function

  $$Q(s, a; \theta) \approx Q^*(s, a)$$

  toward optimizing the dialogue policy automatically.
- The Q-network outputs a probability distributions over all agent's actions given the current dialogue state vector

Q-network

- Our Q-network is constructed in the multi-layer feed forward network:



Experimental Setup

Corpora: DSTC2 & 3

- The DSTC2 and 3 dialogue corpora were collected using Amazon Mechanical Turk [6, 7].
- The domain of DSTC2 provides restaurant information, whereas DSTC3 extends to tourist information, including bars, cafes and etc.
- Examples of tagged dialogues in DSTC2 is in Appendix IV.

SLU error rates

- To test the SLU error robustness, we mimic three environments with different levels of noise by using the SLU N-best results stated in the corpora.

Table: SLU Error Rate(DSTC2)

SLU Error Level	Top-1 Error Rate	Top-10 Error Rate
None	0%	0%
Low	29.02%	16.69%
High	36.98%	23.71%

Table: SLU Error Rate(DSTC3)

SLU Error Level	Top-1 Error Rate	Top-10 Error Rate
None	0%	0%
Low	16.17%	6.78%
High	31.22%	19.43%

Baseline model: *Rule-based* Policy

- To compare the performance of deep RL-policy, we build a rule-based dialogue policy as a baseline model.

Table: Algorithm – Rule-based dialogue policy

```
G ← the 'goal' component of the state tracker output.
R ← the 'requested slot' component of the state tracker output.
S ← the DB query result with constrains in G.
A_m: placeholder for output system dialogue acts.
if length(S) = 0 then
    A_m =canthelp(slot=value), fill slot=value using G.
if length(G) < 2 then
    A_m=request(slot), fill slot using slots that not yet included in G.
else:
    venue=random(S)
    A_m=offer(name=venue.name)
    for slot in R do
        A_m=A_m+inform(venue.slot=venue.value)
Output system response A_m.
```

- It issues a query and makes a response to user's utterance using a set of predefined rules.

Exploration Strategy

- During the training of the Q-network, we adopt an ϵ-greedy strategy.
- The probability is initially set to 1.0 and gradually decreased to 0.1 over the first 10k dialogues.
- We set ϵ to 0 and train the policy for another 10k dialogues.

Reward Function

- During scoring the success rate of a dialogue, a reward function is set as follows:
  - **Reward** +20 for successful dialogues
  - **Penalty** -10 for failed dialogues
  - an additional penalty-1 for each dialogue turn

☞ to encourage agent to behaves as fast as possible

Results and Discussion

Results in DSTC2: *deep RL* vs *rule-based* policy

Table: Comparative Results in DSTC2 Domain

SLU 1 Error Level	Policy	Dialogue Success Rate	Average Dialogue Turns
	Rule-based	100%	7.42
	Deep RL	99.38%	5.84
	Rule-based	85.57%	7.47
	Deep-RL	**90.35%**	**7.74**
	Rule-based	77.14%	7.37
	Deep-RL	**89.55%**	**8.16**

- The rule-based policy always achieves a 100% dialogue success rate only if there exists no SLU error.
- Under the *Low* SLU error, the deep RL policy outperforms the rule-based policy 4 ~ 5% in terms of dialogue success rate.
- The Deep RL policy has required much shorter turns than the baseline model with rule-based policy.

Results in DSTC3: *deep RL* vs *rule-based* policy

- The advantageous performance results of deep-RL are more noticeable in the extended dialogue domain, DSTC3.

Table: Comparative Results in DSTC3 Domain

SLU 1 Error Level	Policy	Dialogue Success Rate	Average Dialogue Turns
	Rule-based	100%	8.58
	Deep RL	99.16%	5.84
	Rule-based	91.49%	8.16
	Deep-RL	**95.15%**	**6.86**
	Rule-based	52.49%	11.53
	Deep-RL	**86.85%**	**8.05**

Success Rate under SLU error

- The success rate is converged
  - after 10k dialogues under the *None* SLU error level,
  - after 15k dialogues under the *Low* and *High* case.

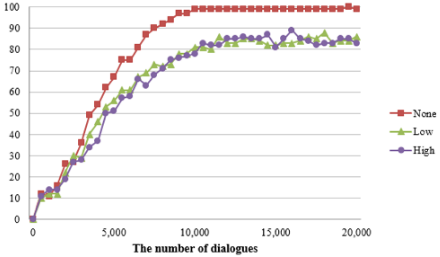


Figure: The Success Rate of Dialogues in SLU Error Levels

- The Deep-RL policy needs approximately 90k ~700k less than traditional MDP-RL policy.

Discussions

- The overall experimental results suggest
    1. Dialogue agent can be trained automatically to successfully complete a dialogue.
    2. It can interact with users within much shorter turns by optimizing the policy in deep RL algorithm.
    3. Deep-RL policy shows more robustness to SLU error than the rule-based policy.
    4. The proposed model requires even smaller size of train data to learn the best action.

Concluding Remarks

Conclusion

- We have proposed the dialogue manager by optimizing the dialogue policy using deep Reinforcement Learning algorithm.
- It shows the deep RL policy is more robust to SLU error and flexible to complex domain of dialogues than the other approaches.
- The deep RL policy interacts with the simulated user more effectively than the rule-based policy.

Implications

Our questions:

- Which insights of deep RL could be drawn to optimize policy in Dialog Manger without hand-crafted features?

- **Deep RL offers a flexible building block for all steps of Dialogue System without any manually stipulated features.**
- **It is expected to overcome a challenge by providing promising apporaches to manage diverse domain conversation.**

Thank you!

- *Hyunjung Lee*: hyunjung.lee@uni-leipzig.de
- *Guanghao Xu*: guanghao412@gmail.com

Acknowledgement

This research was supported by the MISP (Ministry of Science, ICT & Future Planning), Korea, under the National Program for Excellence in SW) (2015-0-00910) supervised by the IITP (Institute for Information & communications Technology Promotion).

References I

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, Ioannis Antonoglou, D. Wierstra, and M. Riedmiller., *Playing atari with deep reinforcement learning*, arXiv preprint arXiv:1312.5602, 2013.

[2] R. Sutton and A. Barto, *Reinforcement learning: An introduction*, Massachusetts, USA, MIT press Cambridge, 1998.

[3] C. Watkins and P. Daya., *Q-learning," Machine learning*, pp. 279-292, 1992.

[4] M.Gašić, *Statistical dialogue modeling*, Diss. PhD thesis, University of Cambridge, 2011.

References II

[5] J. Schatzmann, B. Thomson, K. Weilhammer, and H. Ye S. Young, *Agenda-based user simulation for bootstrapping a POMDP dialogue system.*, In Procedings of Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL): Human Language Technologies, Rochester, NY, pp. 149–152. 2007.

[6] H. Matthew, B. Thomson and J. D. Williams, *Dialog state tracking challenge 2 & 3 handbook*, 2013.

[7] H. Matthew, B. Thomson and J. D. Williams, *The third dialog state tracking challenge.*, in Proceedings of IEEE Spoken Language Technology Workshop (SLT), 2014.

[8] E. Levin, R. Pieraccini and W. Eckert. *Using markov decision process for learning dialogue strategies*, in Proceedings of the IEEE International Conference, vol. 1, pp. 201-204, 1998.

Appendix I:Reinforcement Learning

- *Goal*:
  to learn its behavior by taking actions in an environment in discrete time steps [2, 3].

- An agent in RL selects 'best' actions by **maximizing its cumulative discounted reward** R_t,

$$\mathrm{R}_t = r_t + \gamma \cdot r_{t+1} + \gamma^{2} \cdot r_{t+2} + \ldots + \gamma^{T-1} \cdot r_T$$

where γ is a discount factor and T is a final time step [2].

Appendix I:Reinforcement Learning

- At each time t, the agent
  1. receives a representation of state $s_t \in S$, where S is a state space
  2. selects an action $a_t \in A$, where A is a set of possible actions that the agent can take.
  3. receives a reward r_t
  4. transits to a new state s_{t+1}.

Appendix I:Reinforcement Learning

- Given that the agent follows a policy π :S $\rightarrow$ A,
  an potential value of actions a in the current state s is estimated by **Q-function** as

$$Q^{*}(s, a) = max_{\pi} E[R_t | s_t = s, a_t = a, \pi]$$

- The more accurate the Q-function is, the better policy the agent learns.
- However, they are quite inefficient, especially when the state space becomes large or even infinite.

Appendix I:Reinforcement Learning

- To ensure adequate exploration of state space, the ϵ-**greedy** strategy is applied.
- The agent greedily chooses an action based on the value of agent's action calculated by the policy network,

$$a= \max_a Q(s, a; \theta)\text{ , with probability } 1-\epsilon$$

and selects a random action with probability ϵ

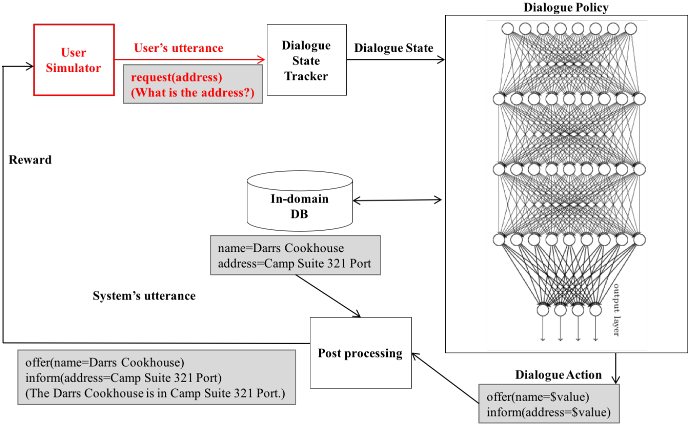
Appendix II: Q-network

- Example of an Input layer of Q-network

	Output of Dialogue State Tracker			SLU N-best results of user's utterance			Results of DB query
Components	Goals	Methods	Requested	SLU 1-best	SLU 2-best	SLU 3-best	Matched count
No. of dimension	5	5	9	78	78	78	1

food	pricerange	name	area	this
0.9458	0.6613	0.0	0.0613	0.0

Appendix III: User Simulator



Appendix III: User Simulator

- Deep RL agent learns over times by experiences.
- The dialogue manager needs a lot of dialogues to be trained, which is impractical to train with real users [4].

- Goal:
  to train Deep RL agent toward optimizing policy automatically by interacting with user-simulator based on agenda-based [5].

Appendix III: User Simulator

- The process of how user simulator operates

1. Initialize the simulator with a certain agenda which consists of
    - CONSTRAINTS
      e.g. *food=korean, price=cheap, area=east...*
    - REQUESTS
      e.g. *address, phone, signature...*
2. During the dialogue, the simulator interacts with the dialog **agent** based on its agenda
3. Evaluate the success rate of dialogues.

Appendix IV: Corpora: DSTC2 & 3

Table: Example Dialogues in DSTC2 Domain

Turn	Speaker	Dialog Act	Real Utterance
0	System	Welcomemsg()	How can I help you?
1	User	inform(area=centre)	Is there any restaurant in the centre area?
1	System	request(pricerange)	What price range do you want?
2	User	inform(pricerange=moderate)	Moderate.
2	System	offer(name=Venue), inform(area=centre)	"Venue" is a restaurant in the centre area.
3	User	request(food)	Which kind of food do they serve?
3	System	offer(name=Venue), inform(food=Thai)	"Venue" is mainly serving Thai food.
hline 7	User	request(address)	What is the address?
7	System	inform(address=9558 ...)	The address is 9558 Ramirez Village Apt.
8	User	thank you()	Thanks!
8	System	bye()	Bye