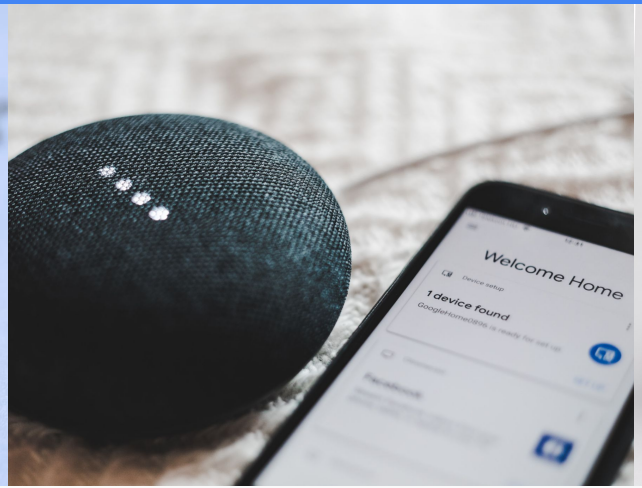
Task-oriented Dialogue System에 대한 넓고 얕은 지식들



집현전 중급반 4조 - 금빛나, 김도은, 김종우

발표자: 김도은, 금빛나 (NLG)

2021.07.04

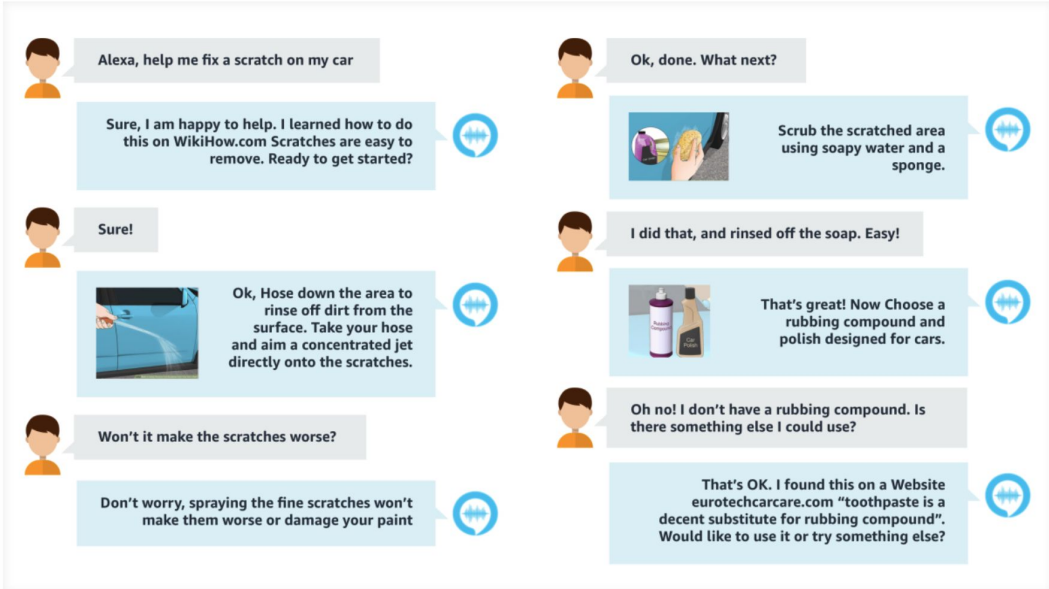
Index

1. What is Task-oriented Dialogue System?

Meena Conversation 1

Human:	*Hi!*
Meena:	*Hey there! What's up?*
Human:	*Same old, Monday blues...what are you up to?*
Meena:	*I'm procrastinating on a paper I'm supposed to be writing.*
Human:	*are you thinking of pulling an all nighter?*
Meena:	*Possibly. What's your paper on?*
Human:	*Nah, i just do the grunge work around here...one day I wish to write a paper that will change the world!*
Meena:	*What's the topic?*
Human:	*not sure, any suggestions?*
Meena:	*Well what's the assignment on?*
Human:	*breathing under water*
Meena:	*How to breathe underwater.*
Human:	*that's what i'm hoping to figure out!*
Meena:	*I hope you figure it out! Because I can't.*
Human:	*i'm sure you can if you try :) you're the best!*
Meena:	*Thank you, but I'm pretty sure you're the best.*
Human:	*how was your weekend?*
Meena:	*Pretty good. Yours?*
Human:	*good, I practiced the violin*
Meena:	*That's awesome! How long have you been playing?*
Human:	*just started learning last month*
Meena:	*That's awesome! What made you want to learn the violin?*

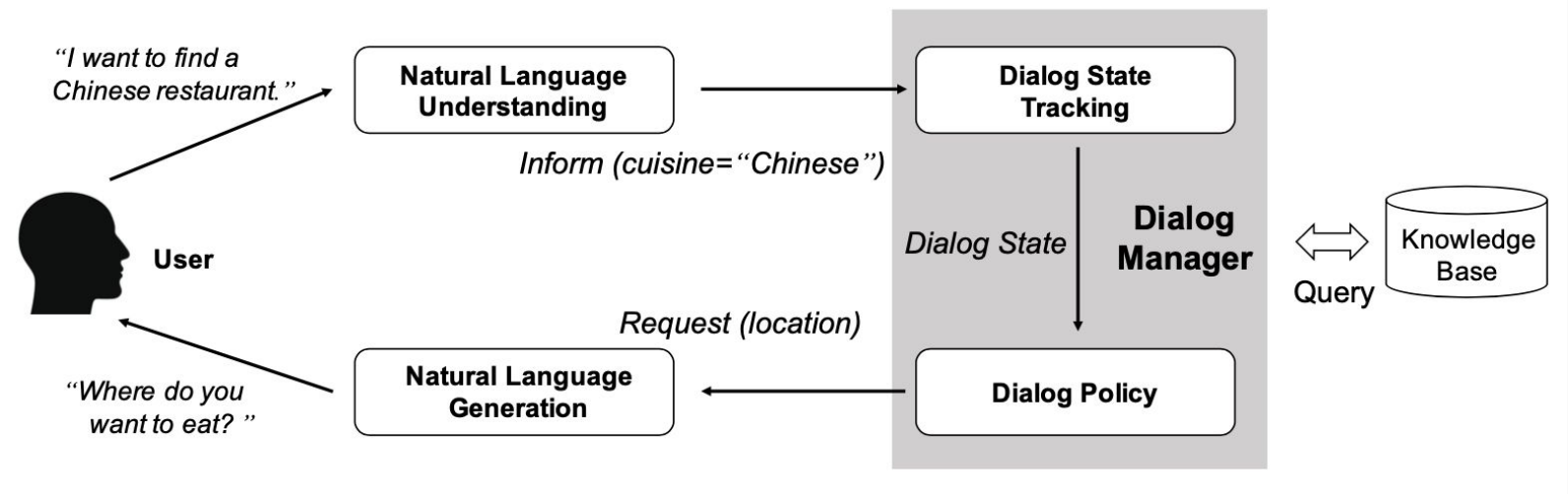
Open domain



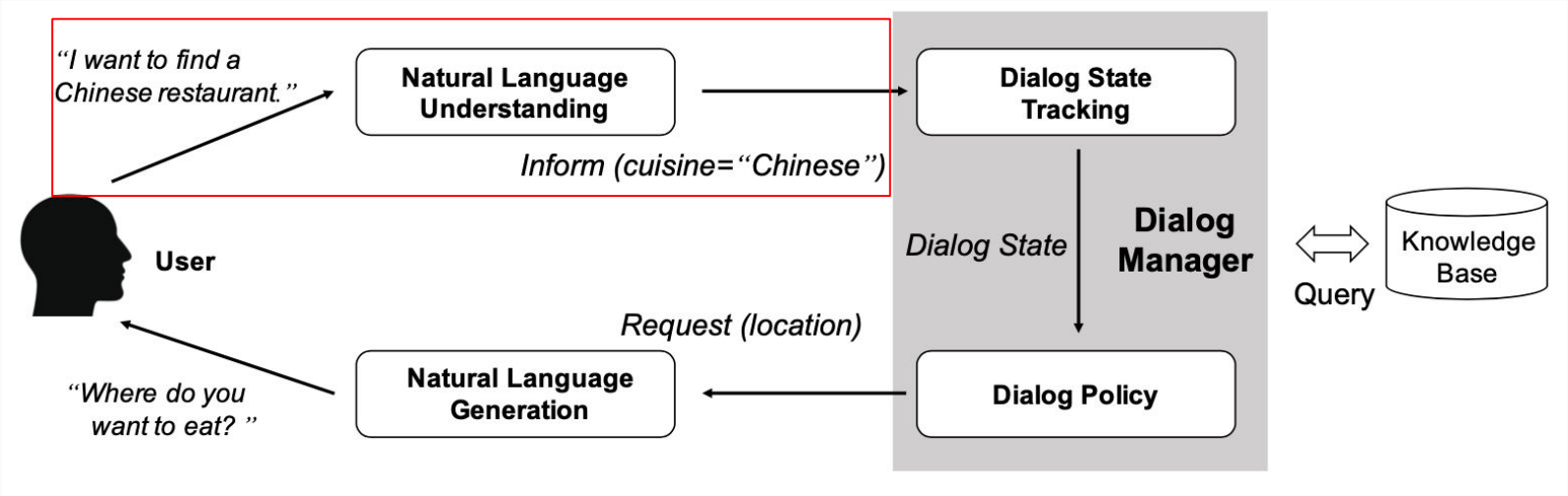
Task Oriented

https://arxiv.org/pdf/2001.09977.pdf
https://www.amazon.science/academic-engagements/amazon-launches-new-alexa-prize-taskbot-challenge

2. System Architecture



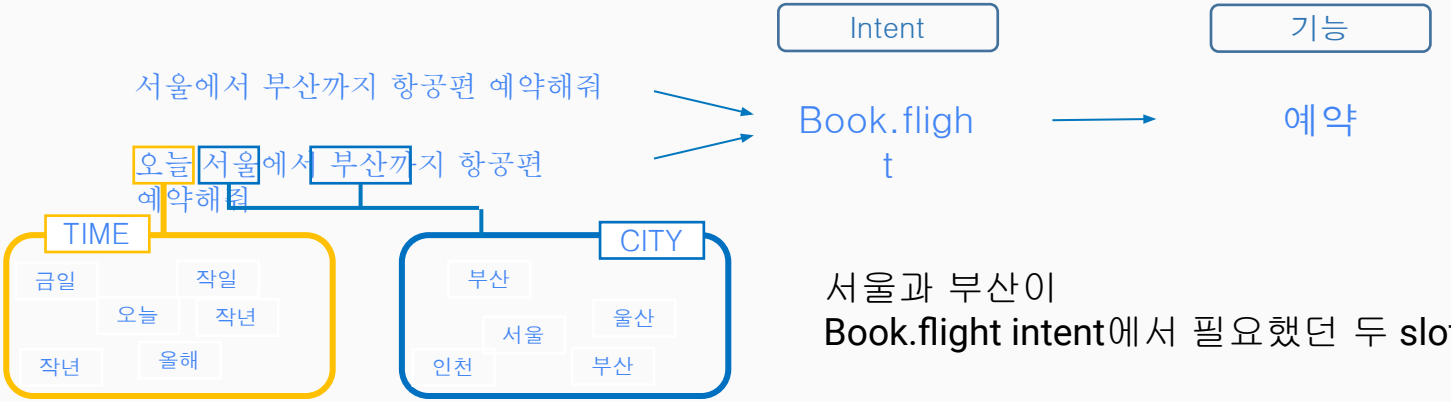
2a. Natural Language Understanding (NLU)



2a. Natural Language Understanding (NLU)

- User Utterance Model

사용자 발화	Intent	Entity Text	Entity Type	Entity Role
서울에서 부산까지 항공편 예약해줘	Book.flight	서울	CITY	DEPARTURE
		부산	CITY	DESTINATION



서울과 부산이
Book.flight intent에서 필요했던 두 slot

2a. Natural Language Understanding (NLU)

- Joint Intent detection and Slot Filling (2016)
  - Intent와 Slot을 동시에 학습시키는 방법 (현재 주류로 채

Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling

Bing Liu[1], *Ian Lane*[1,2]

[1]Electrical and Computer Engineering, Carnegie Mellon University
[2]Language Technologies Institute, Carnegie Mellon University

liubing@cmu.edu, lane@cmu.edu

- ATIS dataset 사용
  - 항공편 정보
  - 4978 context independent 발화
  - 17개 intent, 127 slot 라벨

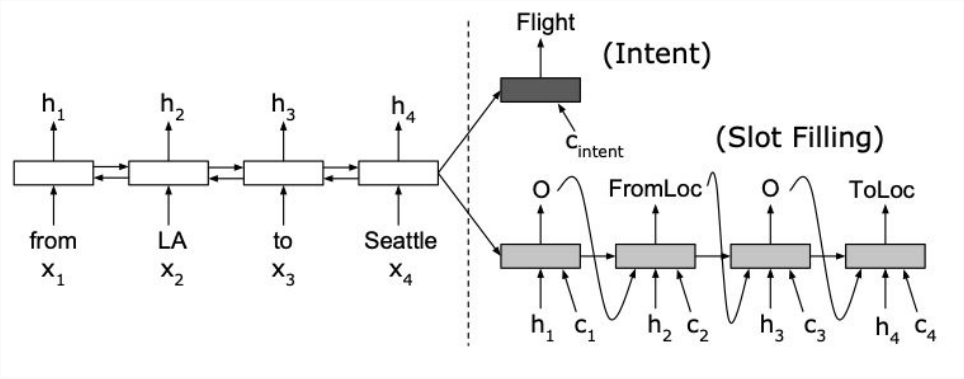
Sentence	first	class	fares	from	boston	to	denver
Slots	B-class_type	I-class_type	O	O	B-fromloc	O	B-toloc
Intent	airfare						

https://www.isca-speech.org/archive/Interspeech_2016/pdfs/1352.PDF

2a. Natural Language Understanding (NLU)

- Joint Intent detection and Slot Filling (2016)
  - Intent와 Slot을 동시에 학습시키는 방법 (현재 주류로 채택)

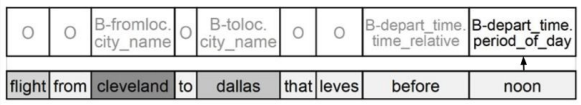
- Intent와 Slot 모두 같은 시퀀스 (문장)를 분석하기 때문에 joint learning이 가능.
- 따로 학습하는 것 보다 더 높은 정확도를 보임.



- Encoder-decoder architecture
- 인코더는 bi-directional LSTM
- h_i (aligned input), c_i (attention)

$$c_i = \sum_{j=1}^{T} \alpha_{i,j} h_j$$

- Attention (weighted sum of hidden vectors from encoder)
- End to End로 학습 가능



- Noon이라는 slot을 예측할 때의 attention 세기

https://www.isca-speech.org/archive/Interspeech_2016/pdfs/1352.PDF

2a. Natural Language Understanding (NLU)

- Joint Intent detection and Slot Filling (2016)
  - Intent와 Slot을 동시에 학습시키는 방법 (현재 주류로 채택)

- Intent와 Slot 모두 같은 시퀀스 (문장)를 분석하기 때문에 joint learning이 가능.
- 따로 학습하는 것 보다 더 높은 정확도를 보임.

Table 4: *Comparison to previous approaches. Joint training model results on ATIS slot filling and intent detection.*

Model	F1 Score	Intent Error (%)
RecNN [8]	93.22	4.60
RecNN+Viterbi [8]	93.96	4.60
Attention Encoder-Decoder NN (with aligned inputs)	**95.87**	**1.57**
Attention BiRNN	**95.98**	**1.79**

https://www.isca-speech.org/archive/Interspeech_2016/pdfs/1352.PDF

2a. Natural Language Understanding (NLU)

- Memory Network (문맥 파악)

User: 부산에서 부터 몇 km 야?

FROM slot은 있지만 TO slot 이 없다.
Intent: NAV.DIRECTION
Slot tagger: @FROM{부산}
Dialog Manager: TO 슬롯을 물어라

Agent: 부산에서 어디까지 가세요?

User: 서울

이때, context 정보가 필요
Intent:NAV.DIRECTION
Slot tagger: @TO{서울}

Agent: 부산에서 서울까지의 거리는 325km 입니다.

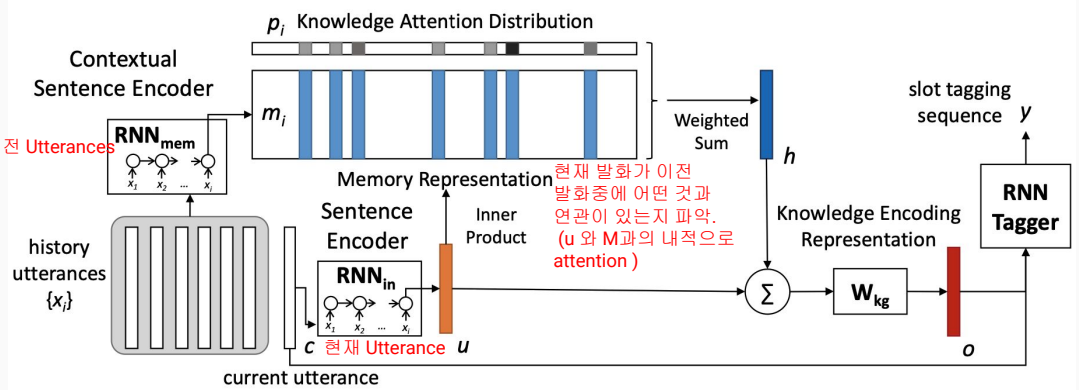


Figure 2: *The illustration of the proposed end-to-end memory network model for multi-turn SLU.*

End-to-End Memory Networks with Knowledge Carryover for Multi-Turn Spoken Language Understanding

Yun-Nung Chen†, Dilek Hakkani-Tür, Gokhan Tur, Jianfeng Gao, and Li Deng

National Taiwan University, Taipei, Taiwan†
Microsoft Research, Redmond, WA, USA
{y.v.chen†, dilek, gokhan.tur}@ieee.org, {jfgao, deng}@microsoft.com

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/06/IS16_ContextualSLU.pdf

2a. Natural Language Understanding (NLU)

- Memory Network (문맥 파악)

User: 부산에서 부터 몇 km 야?

FROM slot은 있지만 TO slot 이 없다.
Intent: NAV.DIRECTION
Slot tagger: @FROM{부산}
Dialog Manager: TO 슬롯을 물어라

Agent: 부산에서 어디까지 가세요?

User: 서울

이때, context 정보가 필요
Intent:NAV.DIRECTION
Slot tagger: @TO{서울}

Agent: 부산에서 서울까지의 거리는 325km 입니다.

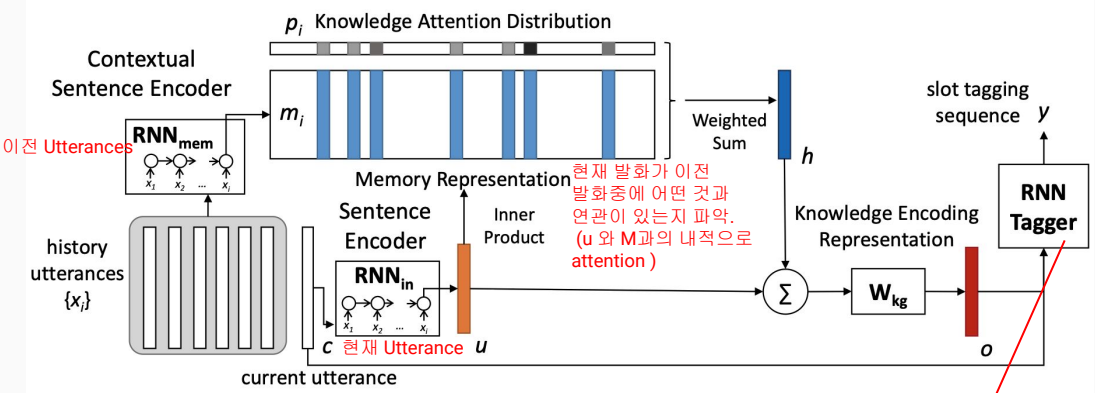


Figure 2: *The illustration of the proposed end-to-end memory network model for multi-turn SLU.*

End-to-End Memory Networks with Knowledge Carryover for Multi-Turn Spoken Language Understanding

Yun-Nung Chen†, Dilek Hakkani-Tür, Gokhan Tur, Jianfeng Gao, and Li Deng

National Taiwan University, Taipei, Taiwan†
Microsoft Research, Redmond, WA, USA
{y.v.chen†, dilek, gokhan.tur}@ieee.org, {jfgao, deng}@microsoft.com

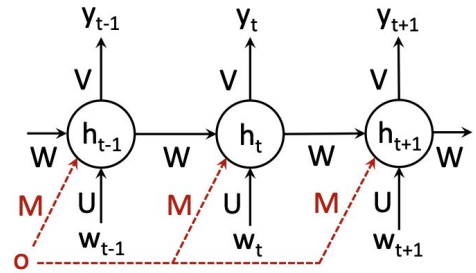


Figure 3: *The RNN model architecture for tagging. The dotted red lines show the encoded knowledge from history in the multi-turn interactions.*

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/06/IS16_ContextualSLU.pdf

(Optional) Importance of NLU in TOD

- NLU error rate을 다르게 했을 때의 Task 성공률과 number of turns

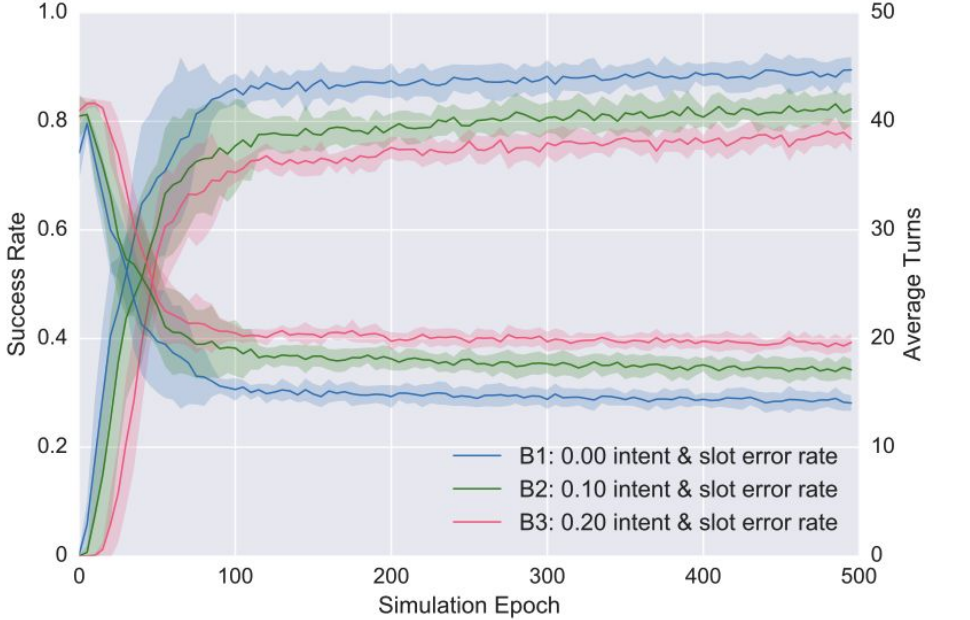


Figure 1: *Learning curves for different NLU error rates.*

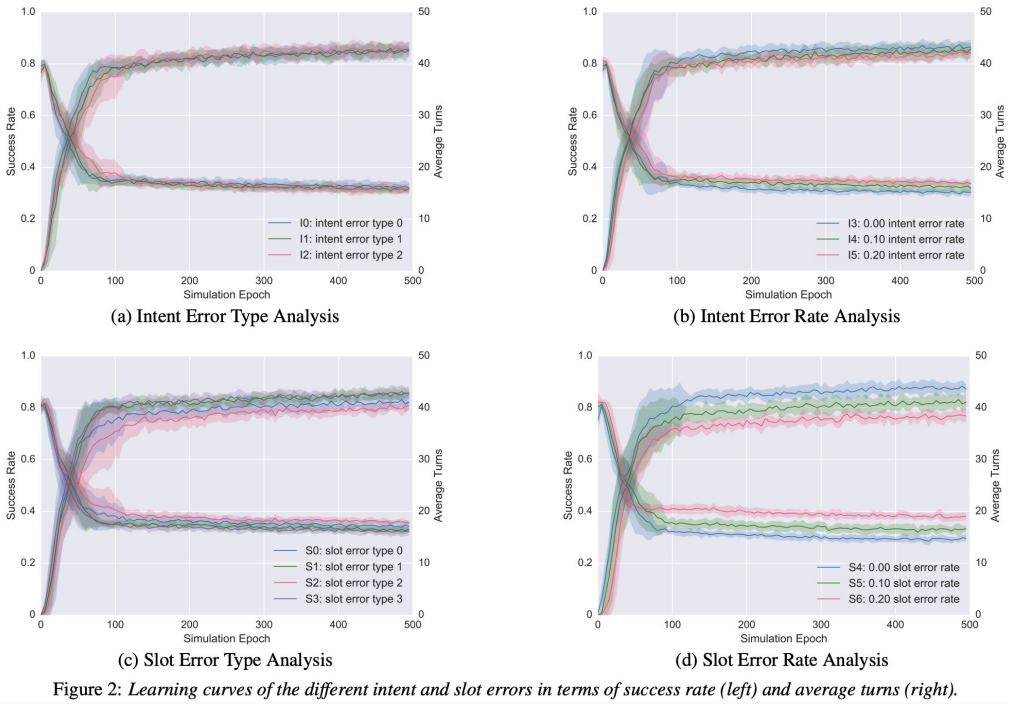
- 파란색 (에러 없음)
- 초록색 (10% 에러)
- 빨간색 (20% 에러)

- NLU 에서의 에러가 커질수록
  - Task 성공률이 낮아지고
  - 대화 지속 turn이 많아진다.
- 당연한 이야기...

https://arxiv.org/pdf/1703.07055.pdf

(Optional) Importance of NLU in TOD

- 그렇다면 NLU에서 Intent와 Slot 중 무엇이 더 중요?



(a) Intent Error Type Analysis (b) Intent Error Rate Analysis

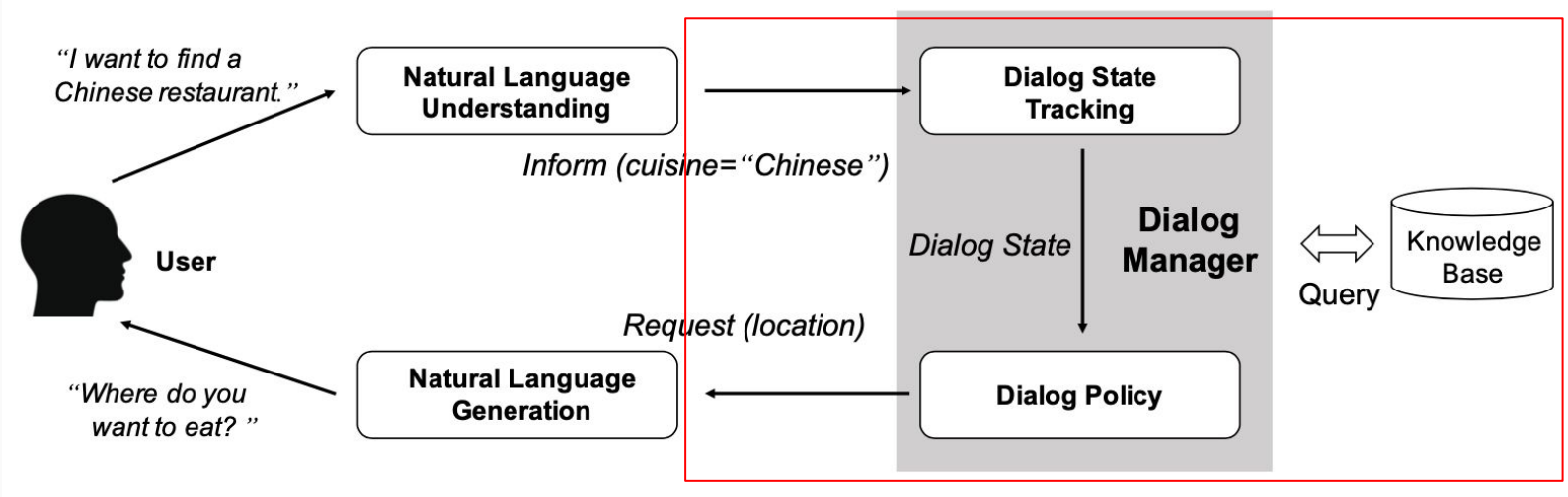
(c) Slot Error Type Analysis (d) Slot Error Rate Analysis

Figure 2: *Learning curves of the different intent and slot errors in terms of success rate (left) and average turns (right).*

- Intent 에러율에 변화를 주었을때 Task 성공률과 turn 개수의 변화가 아주 크지는 않다
- Slot 에러율에 변화를 주었을때 Task 성공률과 turn 개수 변화가 이전 실험보다 훨씬 크다.
- 따라서 Slot이 더 중요하다.
- Joint learning 시에 Slot Tagging에 더 높은 가중치를 두고 학습을 시키자.

https://arxiv.org/pdf/1703.07055.pdf

2b. Dialogue Manager (DM)



2b. Dialogue Manager (DM)

- Dialogue State Tracking
  - 지식 베이스 (Knowledge Base)에 질의를 한다
  - 연속된 user와 agent의 발화를 트랙킹 한다
  - 현재 state를 policy 결정 모듈에 넘겨준다

- Dialogue Policy Learning
  - 현재 state를 입력 받아, Dialogue Action을 결정한다.

2b. Dialogue Manager (DM)

- Dialogue State Tracking

The Second Dialog State Tracking Challenge

Matthew Henderson[1], **Blaise Thomson**[1] **and Jason Williams**[2]
[1]Department of Engineering, University of Cambridge, U.K.
[2]Microsoft Research, Redmond, WA, USA
mh521@eng.cam.ac.uk brmt2@eng.cam.ac.uk jason.williams@microsoft.com

The dialog state at each turn consists of three components:

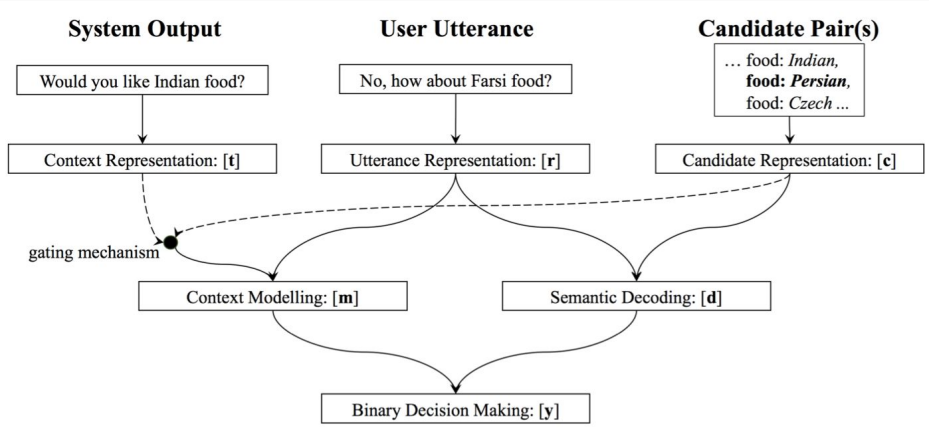
- The **goal constraint** for each *informable* slot. This is either an assignment of a value from the ontology which the user has specified as a constraint, or is a special value — either *Dontcare* which means the user has no preference, or *None* which means the user is yet to specify a valid goal for this slot.
- A set of **requested slots**, i.e. those slots whose values have been requested by the user, and should be informed by the system.
- An assignment of the current dialog **search method**. This is one of
  - *by constraints*, if the user is attempting to issue a constraint,
  - *by alternatives*, if the user is requesting alternative suitable venues,
  - *by name*, if the user is attempting to ask about a specific venue by its name,
  - *finished*, if the user wants to end the call
  - or *none* otherwise.

- Goal Constraint
  - 각 slot들의 distribution
- Requested slots
  - User가 요구하는, 그리고 Agent가 대답해야할 slot이 무엇인지
- Search method
  - By constraints
  - By alternatives
  - By name
  - Finished
  - None

https://www.matthen.com/assets/pdf/The_Second_Dialog_State_Tracking_Challenge.pdf

2b. Dialogue Manager (DM)

- Dialogue State Tracking
  - Rule-based
    - 간단한 if-else로직 이지만, NLU 파트 성능이 아주 좋아야 한다
  - **Neural Belief Tracker**
    - Joint NLU/DST



- 세가지 인풋에 대한 representation learning
- t, r, c 에 대한 context modelling
- r, c 에 대한 Semantic Decoding
- m 과 d 에 대한 Binary decision

- 단점: 미리 정의된 slot value pair들이 있어야 한다

3 Neural Belief Tracker

The Neural Belief Tracker (NBT) is a model designed to detect the slot-value pairs that make up the user's goal at a given turn during the flow of dialogue. Its input consists of the system dialogue acts preceding the user input, the user utterance itself, and a single candidate slot-value pair that it needs to make a decision about. For instance, the model might have to decide whether the goal FOOD=ITALIAN has been expressed in *'I'm looking for good pizza'*. To perform belief tracking, the NBT model *iterates* over all candidate slot-value pairs (defined by the ontology), and decides which ones have just been expressed by the user.

Figure 3 presents the flow of information in the model. The first layer in the NBT hierarchy performs representation learning given the three model inputs, producing vector representations for the user utterance ($\mathbf{r}$), the current candidate slot-value pair ($\mathbf{c}$) and the system dialogue acts ($\mathbf{t_q}, \mathbf{t_s}, \mathbf{t_v}$). Subsequently, the learned vector representations interact through the *context modelling* and *semantic decoding* submodules to obtain the intermediate *interaction summary* vectors $\mathbf{d_r}$, $\mathbf{d_c}$ and $\mathbf{d}$. These are used as input to the final *decision-making* module which decides whether the user expressed the intent represented by the candidate slot-value pair.

https://arxiv.org/pdf/1606.03777.pdf

2b. Dialogue Manager (DM)

- Dialogue State Tracking
  - Rule-based
    - 간단한 if-else로직 이지만, NLU 파트 성능이 아주 좋아야 한다
  - **Neural Belief Tracker**
    - Joint NLU/DST

DST Model	DSTC2		WOZ 2.0	
	Goals	Requests	Goals	Requests
Delexicalisation-Based Model	69.1	95.7	70.8	87.1
Delexicalisation-Based Model + Semantic Dictionary	72.9*	95.7	83.7*	87.6
NEURAL BELIEF TRACKER: NBT-DNN	72.6*	96.4	**84.4***	91.2*
NEURAL BELIEF TRACKER: NBT-CNN	**73.4***	**96.5**	84.2*	**91.6***

  - Joint model의 성능이 월등하다

https://arxiv.org/pdf/1606.03777.pdf

3 Neural Belief Tracker

The Neural Belief Tracker (NBT) is a model designed to detect the slot-value pairs that make up the user's goal at a given turn during the flow of dialogue. Its input consists of the system dialogue acts preceding the user input, the user utterance itself, and a single candidate slot-value pair that it needs to make a decision about. For instance, the model might have to decide whether the goal FOOD=ITALIAN has been expressed in *'I'm looking for good pizza'*. To perform belief tracking, the NBT model *iterates* over all candidate slot-value pairs (defined by the ontology), and decides which ones have just been expressed by the user.

Figure 3 presents the flow of information in the model. The first layer in the NBT hierarchy performs representation learning given the three model inputs, producing vector representations for the user utterance ($\mathbf{r}$), the current candidate slot-value pair ($\mathbf{c}$) and the system dialogue acts ($\mathbf{t_q}, \mathbf{t_s}, \mathbf{t_v}$). Subsequently, the learned vector representations interact through the *context modelling* and *semantic decoding* submodules to obtain the intermediate *interaction summary* vectors $\mathbf{d_r}$, $\mathbf{d_c}$ and $\mathbf{d}$. These are used as input to the final *decision-making* module which decides whether the user expressed the intent represented by the candidate slot-value pair.

2b. Dialogue Manager (DM)

Dialogue Policy Learning

- Rule-based
  - State tracker 로 부터 Requested slot이 무엇인지 주어지면 어떠한 action을 취해야 하는지 룰 베이스로 작성
- 기계 학습
  - 지도 학습
    - 다량의 레이블된 데이터가 필요함 (전문가가 직접 해야함)
    - 데이터가 모든 도메인을 커버할 수 없다
  - 강화 학습
    - 요즘 대세인 연구 :)
    - 강화학습은 특정 환경 안에서 다량의 샘플을 요구하고, 실제 사용자와 대화하면서 엄청난 cold start를 야기한다
    - 따라서 supervised 로 훈련된 Simulated User를 사용

2b. Dialogue Manager (DM)

- Dialogue Policy Learning
  - 기계 학습
    - 지도 학습
      - 다량의 레이블된 데이터가 필요함 (전문가가 직접 해야함)
      - 데이터가 모든 도메인을 커버할 수 없다

END-TO-END JOINT LEARNING OF NATURAL LANGUAGE UNDERSTANDING AND DIALOGUE MANAGER

Xuesong Yang, Yun-Nung Chen§, Dilek Hakkani-Tür†, Paul Crook*, Xiujun Li‡, Jianfeng Gao‡, Li Deng‡*

*University of Illinois at Urbana-Champaign, Urbana, IL, USA
§National Taiwan University, Taipei, Taiwan
†Google Research, Mountain View, CA, USA
‡Microsoft Research, Redmond, WA, USA, *Microsoft Corporation, Redmond, WA, USA

Joint learning again!!

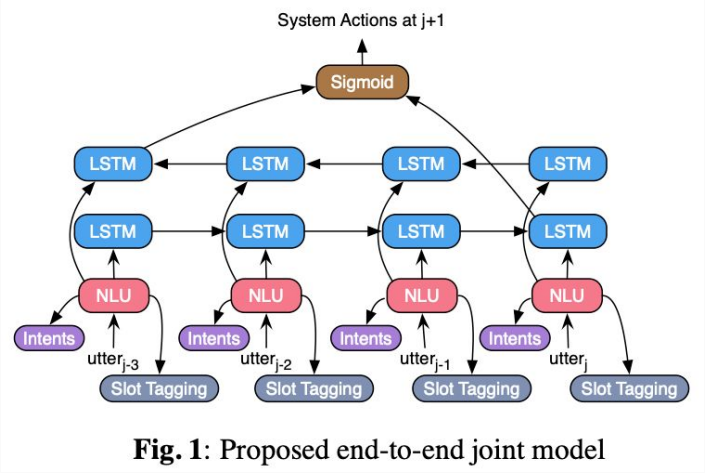
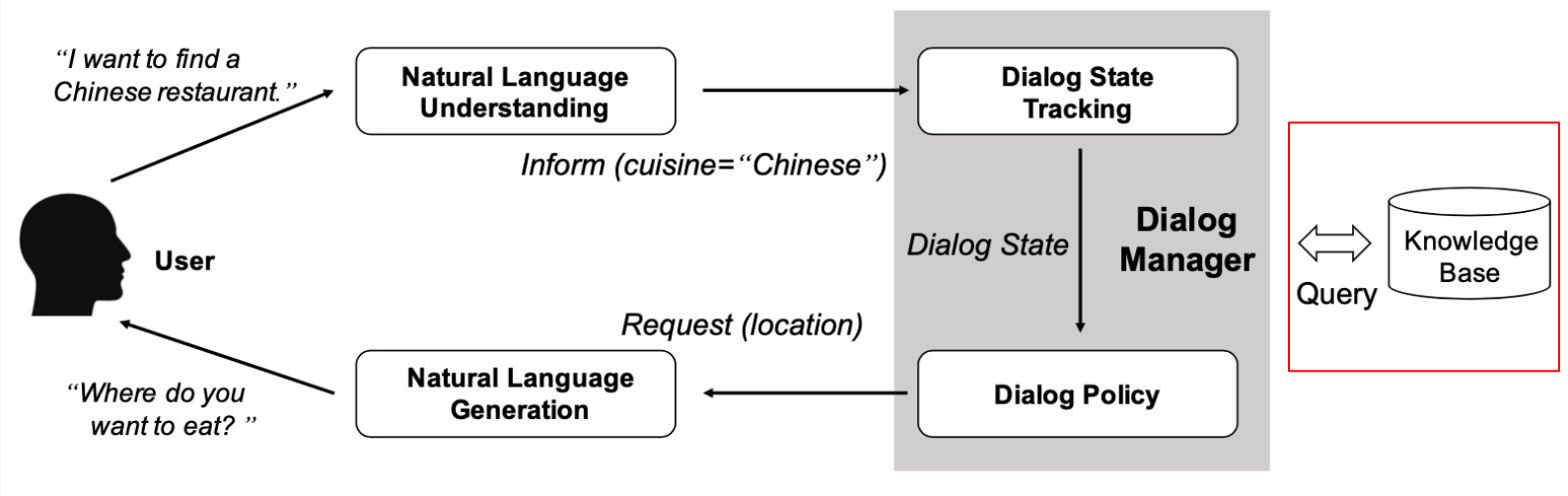


Fig. 1: Proposed end-to-end joint model

- NLU + DM Joint learning
- Dialog state tracker가 따로 필요 없다
- LSTM이 NLU의 아웃풋을 읽고 다음 액션을 출력한다.
- 사실상 dialog state들이 LSTM의 State안에 latent variable 들로 존재하는 꼴
- intent와 slot들은 supervised learning을 위해 입력됨

https://arxiv.org/pdf/1612.00913.pdf

2bi. Knowledge Graph (KG)



https://arxiv.org/pdf/2003.07490.pdf

2bi. Knowledge Graph (KG)

<Knowledge Graph 란>

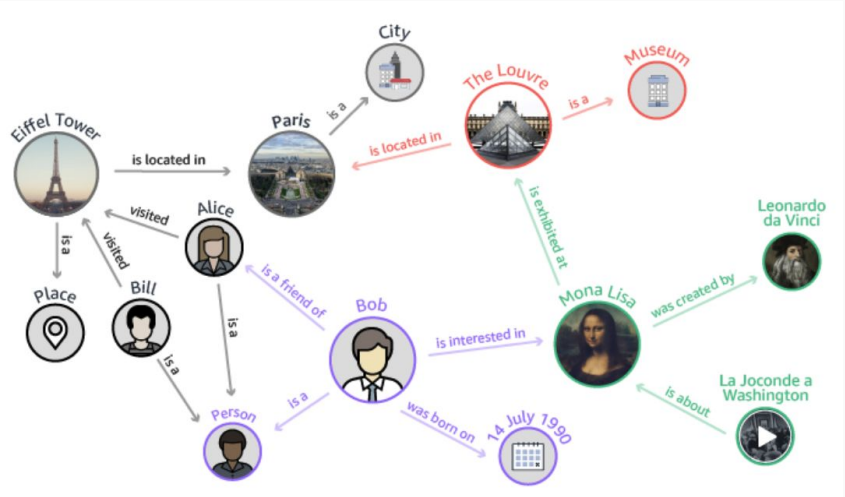
Knowledge graph

From Wikipedia, the free encyclopedia

For other uses, see Knowledge graph (disambiguation).

In knowledge representation and reasoning, **knowledge graph** is a knowledge base that uses a graph-structured data model or topology to integrate data. Knowledge graphs are often used to store interlinked descriptions of entities – objects, events, situations or abstract concepts – with free-form semantics.[1]

Since the development of the Semantic Web, knowledge graphs are often associated with linked open data projects, focusing on the connections between concepts and entities.[2][3] They are also prominently associated with and used by search engines such as Google, Bing, and Yahoo; knowledge-engines and question-answering services such as WolframAlpha, Apple's Siri, and Amazon Alexa; and social networks such as LinkedIn and Facebook.



- 여러 데이터를 그래프 데이터 모델로 합치는 knowledge representation 기법

- RDF (N-triples, Turtle, RDF/XML, JSON-LD)
- Property Graph

https://en.wikipedia.org/wiki/Knowledge_graph ; https://aws.amazon.com/neptune/

2bi. Knowledge Graph (KG)

<Knowledge Graph 란>

Knowledge graph

From Wikipedia, the free encyclopedia

For other uses, see Knowledge graph (disambiguation).

In knowledge representation and reasoning, **knowledge graph** is a knowledge base that uses a graph-structured data model or topology to integrate data. Knowledge graphs are often used to store interlinked descriptions of entities – objects, events, situations or abstract concepts – with free-form semantics.[1]

Since the development of the Semantic Web, knowledge graphs are often associated with linked open data projects, focusing on the connections between concepts and entities.[2][3] They are also prominently associated with and used by search engines such as Google, Bing, and Yahoo; knowledge-engines and question-answering services such as WolframAlpha, Apple's Siri, and Amazon Alexa; and social networks such as LinkedIn and Facebook.

- 여러 데이터를 그래프 데이터 모델로 합치는 knowledge representation 기법

- RDF (N-triples, Turtle, RDF/XML, JSON-LD)
- Property Graph

- <RDF>
  - N-triples (subject - predicate - object)

```
_:alice <http://xmlns.com/foaf/0.1/knows> _:bob .
_:bob <http://xmlns.com/foaf/0.1/knows> _:alice .
```

  - W3C 그룹의 시멘틱 웹 표준을 따름
  - Query Language: SPARQL

Specifically, the following query returns names and emails of every person in the dataset:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
       ?email
WHERE
  {
    ?person  a          foaf:Person .
    ?person  foaf:name  ?name .
    ?person  foaf:mbox  ?email .
  }
```

  - 쿼리 속도가 빠르다

https://en.wikipedia.org/wiki/Knowledge_graph ; https://aws.amazon.com/neptune/

2bi. Knowledge Graph (KG)

<Knowledge Graph 란>

Knowledge graph

From Wikipedia, the free encyclopedia

For other uses, see Knowledge graph (disambiguation).

In knowledge representation and reasoning, **knowledge graph** is a knowledge base that uses a graph-structured data model or topology to integrate data. Knowledge graphs are often used to store interlinked descriptions of entities – objects, events, situations or abstract concepts – with free-form semantics.[1]

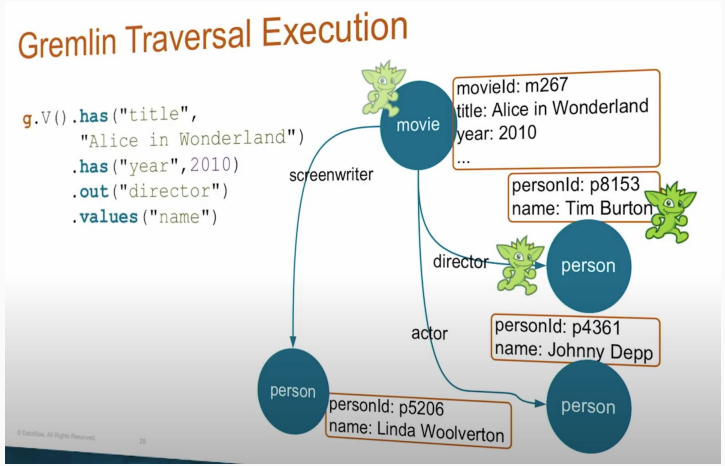
Since the development of the Semantic Web, knowledge graphs are often associated with linked open data projects, focusing on the connections between concepts and entities.[2][3] They are also prominently associated with and used by search engines such as Google, Bing, and Yahoo; knowledge-engines and question-answering services such as WolframAlpha, Apple's Siri, and Amazon Alexa; and social networks such as LinkedIn and Facebook.

- 여러 데이터를 그래프 데이터 모델로 합치는 knowledge representation 기법

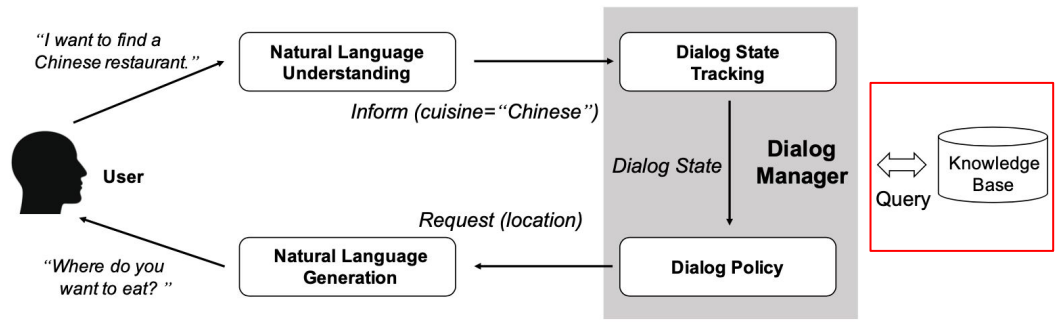
- RDF (N-triples, Turtle, RDF/XML, JSON-LD)
- Property Graph

https://en.wikipedia.org/wiki/Knowledge_graph ; https://www.youtube.com/watch?v=mZmVnEzsDnY&t=1120s

- <Property Graph>
  - Nodes and Edges
  - 노드와 엣지 둘다 property (key-value pair)를 가질 수 있음
  - 정해진 표준 포맷은 없다.
  - Query Language: Cypher, Gremlin, GQL
  - RDF 쿼리에 비해 속도가 느리다.
    - Graph traversal을 해야 하기 때문.



2bi. Knowledge Graph (KG)

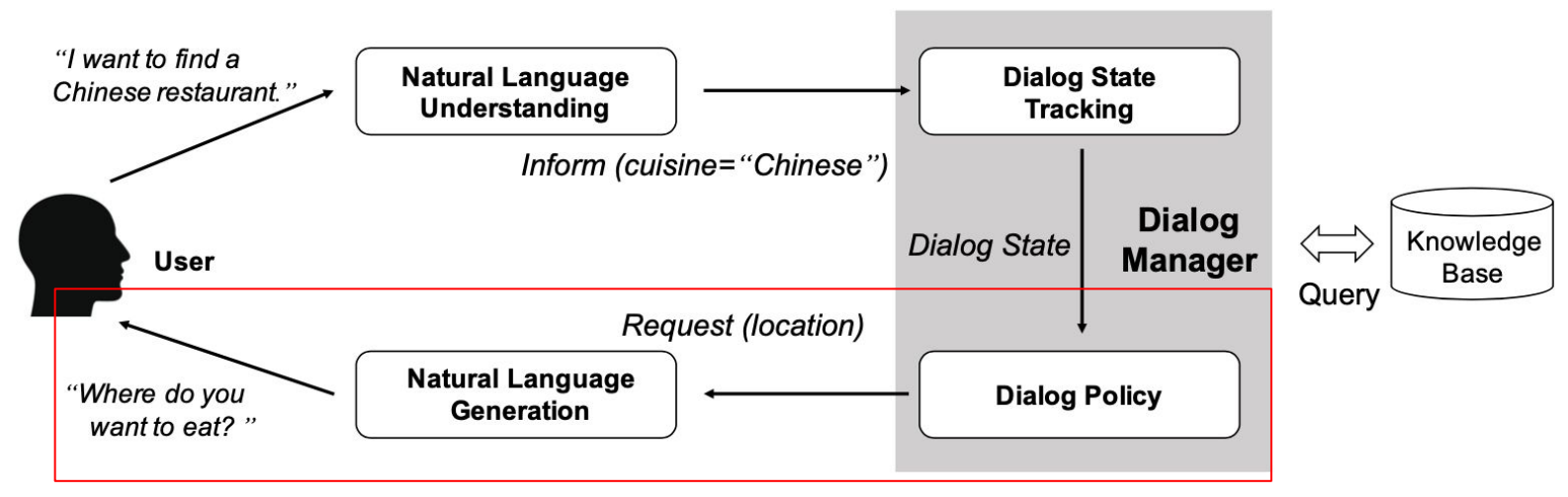


List of Graph Database

- Neo4j.
- ArangoDB.
- Dgraph.
- OrientDB.
- Amazon Neptune.
- Cassandra.

https://arxiv.org/pdf/2003.07490.pdf ; https://neo4j.com ; https://aws.amazon.com/neptune/

2c. Natural Language Generation (NLG)



2c. Natural Language Generation (NLG)

- Conventional tasks of NLG system
  - Content Determination & Text Planning
    - 어떤 정보를 전달할지 결정하는 것 & 말을 다듬어 정보를 정연하게 구성하는 것
  - Sentence Planning
    - 문장들을 더 유창하고 읽기 쉽게 만드는 것
    - 접속사, 대명사, 담화 표지*(ex: so, also, right, okay)* 등을 이용함
  - Realization
    - 문법적으로 옳은 문장을 만드는 것

Reference - https://arxiv.org/pdf/cmp-lg/9605002.pdf

2c. Natural Language Generation (NLG)

Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić,
Pei-Hao Su, David Vandyke and **Steve Young**
Cambridge University Engineering Department,
Trumpington Street, Cambridge, CB2 1PZ, UK
{thw28,mg436,nm480,phs26,djv27,sjy}@cam.ac.uk

Reference - https://arxiv.org/pdf/1508.01745

2c. Natural Language Generation (NLG)

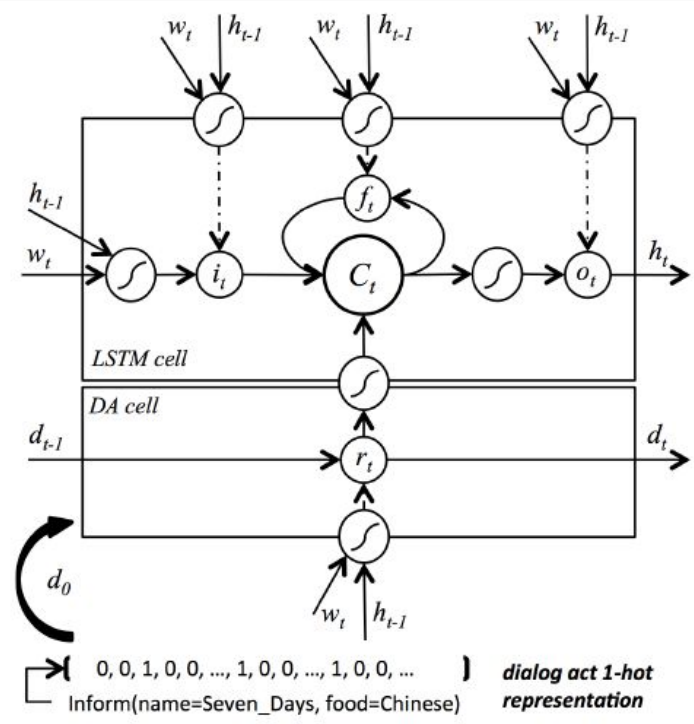
- Rule-base
  - Robustness, Adequacy
  - 동일한, 반복되는, 부자연스러운 답변
  - multiple domains and languages를 다루는 시스템으로 확장하기 어려움
- Corpus-base
  - Corpus로부터 직접적으로 generation하는 법을 배움
    - 인간의 response를 더 자연스럽게 모방
    - 사전정의된 Rule에 대한 의존도 감소
    - 시스템의 확장성 증가
  - Over-generation과 reranking을 통해 최종 response 얻음
  - 학습 데이터의 효율성, 정확도, 자연스러움이 부족

2c. Natural Language Generation (NLG)

- Phrase-base
  - Factored Language Model 기반
  - 의미적으로 정렬된 corpus로부터 generation하는 법을 배움
- Neural network-base
  - RNN based system
    - 긴 종속성을 가진 경우 vanishing gradient 문제가 발생
  - LSTM based system
    - 네트워크를 공간적/시간적으로 deep하게 만듦으로써 더 정확한 결과 얻음

2c. Natural Language Generation (NLG)

- SC-LSTM (Semantically Controlled LSTM)



Realization

Sentence planning

2c. Natural Language Generation (NLG)

- SC-LSTM

Input gate $\mathbf{i}_t = \sigma(\mathbf{W}_{wi}\mathbf{w}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1})$

Forget gate $\mathbf{f}_t = \sigma(\mathbf{W}_{wf}\mathbf{w}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1})$

Output gate $\mathbf{o}_t = \sigma(\mathbf{W}_{wo}\mathbf{w}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1})$

Proposed cell value $\hat{\mathbf{c}}_t = tanh(\mathbf{W}_{wc}\mathbf{w}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1})$

True cell value $\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t$

Hidden state $\mathbf{h}_t = \mathbf{o}_t \odot tanh(\mathbf{c}_t)$

2c. Natural Language Generation (NLG)

- SC-LSTM
  - Slot이 binary slot이거나 don't care 값을 갖는 경우, 명확히 비어휘화*되지 않음
    *비어휘화(delexicalisation) : slot value → slot token

    → control cell 추가

    Reading gate $\mathbf{r}_t = \sigma(\mathbf{W}_{wr}\mathbf{w}_t + \alpha\mathbf{W}_{hr}\mathbf{h}_{t-1})$

    DA vector $\mathbf{d}_t = \mathbf{r}_t \odot \mathbf{d}_{t-1}$

  - Cell state가 DA의 영향을 받기 때문에 식 조정 필요

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t$$

↓

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

2c. Natural Language Generation (NLG)

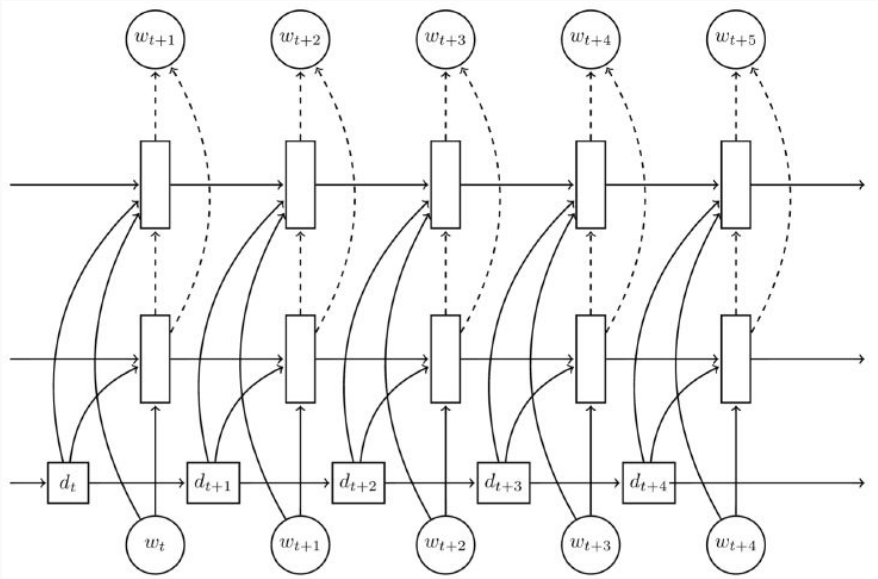
- SC-LSTM system
  - Deep Neural Network (DNN)
    - DA cell의 상단에 LSTM cell을 여러 개 쌓음
    - Neural language generator를 공간적/시간적으로 확장
    - Skip connection과 dropout 적용
    - 모든 hidden layer가 reading gate에 영향을 미침

$$\mathbf{r}_t = \sigma(\mathbf{W}_{wr}\mathbf{w}_t + \alpha\mathbf{W}_{hr}\mathbf{h}_{t-1})$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_{wr}\mathbf{w}_t + \sum_l \alpha_l \mathbf{W}^l_{hr}\mathbf{h}^l_{t-1})$$

2c. Natural Language Generation (NLG)

- SC-LSTM system
  - Deep Neural Network (DNN)



Skip connection : vanishing gradient problem 해결
Dropout : overfitting 및 co-adaptation 방지

2c. Natural Language Generation (NLG)

- SC-LSTM system
  - Cross entropy를 사용하여 Sentence planning과 realization을 공동 최적화

    => 정렬되지 않은 데이터로부터 generation하는 법을 배울 수 있음

  - Output candidates를 random sampling

    => Language variation

  - Stop sign이 생성되거나 일부 제약 조건이 충족될 때까지 output 분포에서 input 토큰을 하나씩 샘플링

    => 필요한 발화를 형성하기 위해 어휘화*될 수 있는 일련의 토큰들이 생성됨

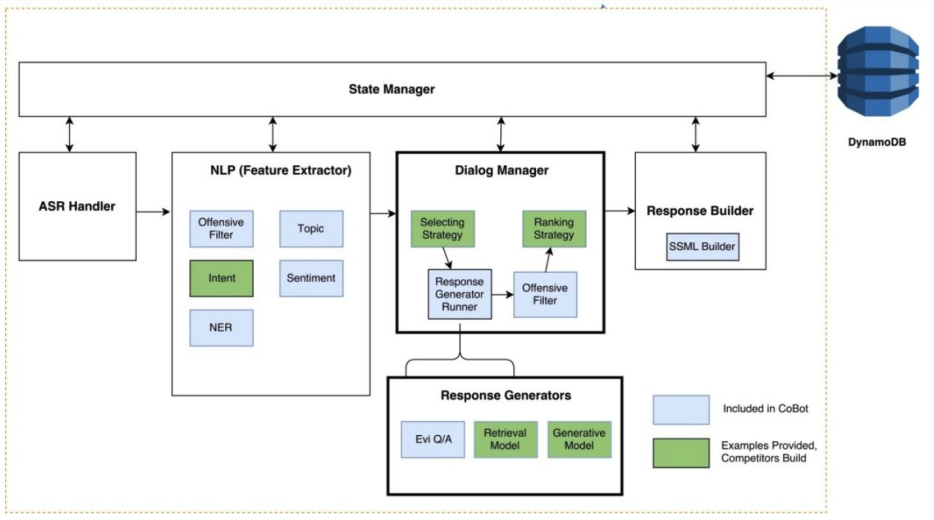
    * 어휘화(lexicalisation) : slot token → slot value

2c. Natural Language Generation (NLG)

- SC-LSTM system
  - LSTM generator는 이전 기록만을 기반으로 단어를 선택하는데, 일부 문장 형식은 backward context에 의존한다는 문제점 존재
  - 해결방법 1 : bidirectional networks
    - Generation 과정이 시간에 따라 순차적이기 때문에 간단하지 않음
  - 해결방법 2 : 또 하나의 SC-LSTM을 backward context로 학습시킴
    - Forward generator로부터 나온 output candidates에서 가장 최상의 것을 선택함

3. Overall design (Summary)

챗봇 == 자연어 처리 기술의 집합체



- ASR Handler
  - 정확한 STT를 위해 N-best 문장 중 word level, sentence level confidence score를 사용하여 걸러낸다.
  - 스코어가 너무 낮을 시, 사용자에게 재질문 or 이전 대화 기록을 state manager로 부터 참고.
- NLP Pipeline
  - Offensive Filter, Topic/감성 분류, NER 등
  - 각 computing instance로 utterance를 전송하여 병렬처리 가능.
- Dialog Manager
  - Selecting: 어느 response generator를 선택할지 결정
  - Ranking: Response generator에 의해 생성된 문장 후보들 중 가장 좋은 답변 채택
- State Manager
  - NLP 결과, each turn 발화 등을 저장. 모든 모듈과 소통.
- Response Generators
  - Context info (intent, slot, asr, state info.)를 입력으로 받고 응답 생성
  - 여러개의 generator를 만들 수 있다.
- Response Builder
  - 생성된 문장의 TTS를 모은다.

https://assets.amazon.science/3c/5c/b65e49ba436d9d5147f95b38e817/advancing-the-state-of-the-art-in-open-domain-dialog-systems-through-the-alexa-prize.pdf