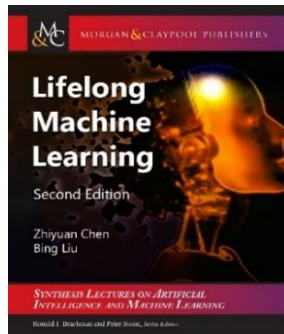


# Continual Learning Dialogue Systems

– Learning on the job after Model Deployment

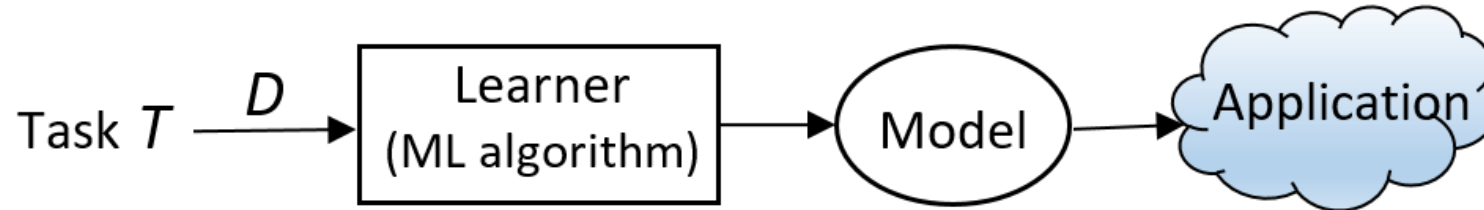


**Sahisnu Mazumder** and **Bing Liu**  
Department of Computer Science  
University of Illinois at Chicago

**Web:** <https://www.cs.uic.edu/~liub/IJCAI21-Continual-Learning-Dialogue-Systems-after-Deployment.html>

# Introduction

- Classic machine learning: **Isolated single-task learning**



- **Key weaknesses**
  - **Closed-world assumption:** nothing new or unexpected in application
  - **No knowledge accumulation or transfer:** isolated learning
  - **Model is fixed after deployment:** no learning or adaptation
- **Focus of this talk:** Learning after model deployment or learning on the job, specially in the dialogue domain.

# Learning on the job (while working)

(Liu, 2020, Chen and Liu, 2018)

- It is estimated that about **70% of our human knowledge comes from 'on-the-job' learning.**
  - ❑ Only about 10% through formal training
  - ❑ The rest 20% through observation of others
- **An AI agent should learn on the job too as**
  - ❑ The world is too complex and constantly changing.
    - Have to learn and adapt continually
  - ❑ Without this capability, an AI agent is not truly intelligent.

# Self-driving cars : A motivating example

- Self-driving cars cannot reach human-level of driving with only rules and off-line training.
  - Impossible to cover all corner cases
  - **Real-world is full of unknowns.**
- Has to learn & adapt continuously in its interaction with humans and the environment by itself.
  - in the **open world** (changes & unknowns).



# AI in open-world & Learning on the job

(Liu 2020, Liu & Mazumder 2021)

- AI has to learn **continually** and **interactively** on the job (after deployment) in the Open-world
  - ✓ Needs to communicate with humans, e.g., **take human instructions, ask when it has difficulty in the open world with unknowns**, and learn from humans.
  - ✓ Communicating in natural language (NL) is a natural choice.
- **Dialogue systems** with on-the-job learning (i.e., *learning during conversation*) capability is essential for the next-generation autonomous conversational AI.

# Dialogue & Interactive Systems : Tasks

- **Question-answering**: provide **concise, direct answers to user queries** based on rich knowledge drawn from various data sources (**QA Bots**)
  - ✓ text collections such as Web documents
  - ✓ pre-compiled knowledge bases (KBs) such as sales and marketing database, factual KBs, etc.
- **Task completion**: help user **accomplish their tasks** (**task completion bots**)
  - ✓ restaurant reservation, meeting scheduling, trip planning
- **Social chats**: **converse seamlessly** and appropriately with users and provide useful recommendations (**social chatbots**)

# Example – Human-agent dialogues for business decision making (Gao et. al. 2019)

Turn	Dialogue
1	<i>usr</i> : Good morning!
	<i>agt</i> : Good morning, I hope you had a great weekend.
2	<i>usr</i> : not really.
	<i>agt</i> : You sound depressed. What can I do for you?
3	<i>usr</i> : I'd like to know where sales are lagging behind our forecast?
	<i>agt</i> : The worst region is [country], where sales are 15% below projections.
4	<i>usr</i> : Do you know why?
	<i>agt</i> : The forecast for [product] growth was overly optimistic.
5	<i>usr</i> : How can we turn this around?
	<i>agt</i> : Here are the 10 customers in [country] with the most growth potential, per our CRM model.
6	<i>usr</i> : Can you set up a meeting with the CTO of [company]?
	<i>agt</i> : Yes, I've set up a meeting with [person name] for next month when you are in [location].
7	<i>usr</i> : Thanks!

*Social chats* { 1, 2

*Question-answering* { 3, 4, 5

*Task completion* { 6, 7

# Dialogue & Interactive Systems : Broad Categories

## ■ Task-oriented chatbots

- ✓ Complete tasks based on users' requests, e.g., providing the requested information and taking actions.
- ✓ Personal assistants: Alexa, Siri, Google Home, etc.

## ■ Chit-chat systems

- ✓ Conduct chit-chat type of conversation in wide range of topics without having a specific goal to complete.
- ✓ Example – ELIZZA, Microsoft Xiaolce.

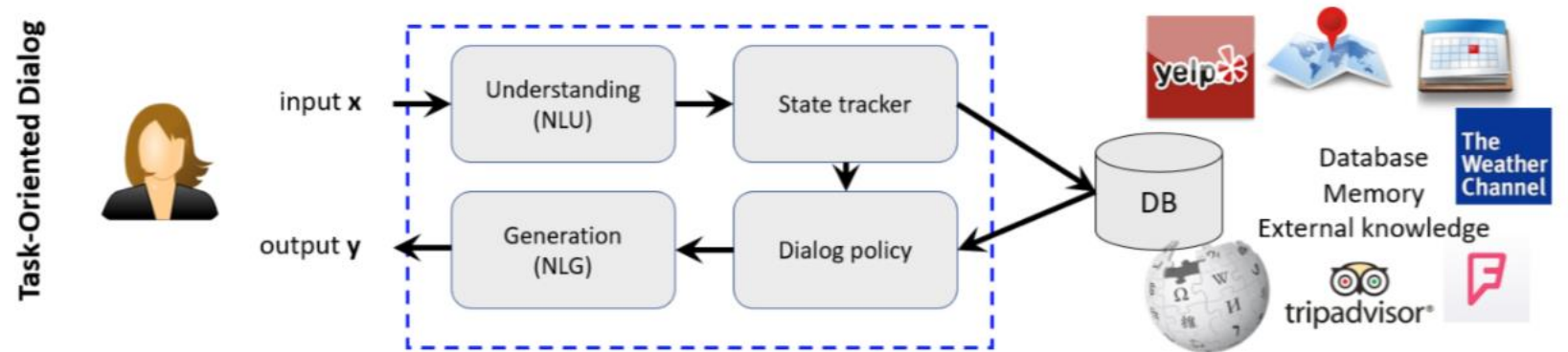


- Gao, Galley and Li. *Neural approaches to conversational AI: Question answering, task-oriented dialogues and social chatbots*. Now Foundations and Trends, 2019.



# Task-oriented Dialogue Systems

(Gao et. al. 2019)



- Often designed as a Modular systems –
  - ✓ **Natural Language Understanding (NLU)**: identify user intents and extracting associated information;
  - ✓ **State Tracking**: tracks the dialogue state that captures all essential information in the conversation so far;
  - ✓ **Dialogue Policy**: selects the next action based on the current state;
  - ✓ **Natural Language Generation (NLG)**: convert agent actions to natural language responses.

# Natural Language Understanding (NLU)

(Jurafsky et. al. 2020)

- **Domain Classification:** Classifying the domain of the task.
  - ✓ E.g., is this user talking about airlines, programming an alarm clock, or dealing with their calendar?
- **Intent Classification:** what general task or goal is the user trying to accomplish?
  - ✓ E.g., Find a Movie, or Show a Flight, or Remove a Calendar Appointment.
- **Slot filling:** extract the particular slots and fillers that the user intends the system to understand from their utterance with respect to their intent → a sequence labeling problem

# Natural Language Understanding (NLU): Examples

(Jurafsky et. al. 2020)



Show me morning flights from Boston to San Francisco on Tuesday

DOMAIN: AIR-TRAVEL  
INTENT: SHOW-FLIGHTS  
ORIGIN-CITY: Boston  
ORIGIN-DATE: Tuesday  
ORIGIN-TIME: morning  
DEST-CITY: San Francisco



Wake me tomorrow at 6

DOMAIN: ALARM-CLOCK  
INTENT: SET-ALARM  
TIME: 2017-07-01 0600-0800

# State Tracking

(Jurafsky et. al. 2020)

User: I'm looking for a cheaper restaurant  
`inform(price=cheap)`

System: Sure. What kind - and where?

User: Thai food, somewhere downtown  
`inform(price=cheap, food=Thai, area=centre)`

System: The House serves cheap Thai food

User: Where is it?  
`inform(price=cheap, food=Thai, area=centre); request(address)`

System: The House is at 106 Regent Street

**dialogue-state** includes the entire state of the frame at this point (the fillers of each slot), as well as the user's most recent dialogue act, summarizing all of the user's constraints.

# Dialogue Policy

(Jurafsky et. al. 2020)

- Decide what action the system should take next, that is, what dialogue act to generate.
  - ✓ At turn  $i$  in the conversation, we want to predict which action  $A_i$  to take, based on the **entire dialogue state** [entire sequence of dialogue acts from the system (A) and from the user (U)].

$$\hat{A}_i = \operatorname{argmax}_{A_i \in A} P(A_i | (A_1, U_1, \dots, A_{i-1}, U_{i-1}))$$



I need to find an Italian restaurant in the north part of town.



Dialogue Act: *inform*  
(*address, area, name*)

# Natural Language Generation (NLG)

(Jurafsky et. al. 2020)

- Once the policy has decided what **speech act** to generate, the NLG **generates the text response**
  - ✓ modeled in two stages, content planning (what to say) and **sentence realization** (how to say it).
- sentence realization is commonly achieved through **delexicalization**
  - ✓ Mapping from frames to delexicalized sentences using encoder decoder models

Dialogue Act: *inform*  
(*address, area, name*)



NAME\_SLOT is in the  
AREA\_SLOT part of town  
their address is  
ADDRESS\_SLOT.



# Social Chatbots

(Jurafsky et.al. 2020)

- Often Implemented using a unitary (non-modular) system
  - ✓ **Rule-based systems**
    - ✓ Works based on pattern/transform rules
    - Examples - ELIZA (Weizenbaum, 1966) and PARRY (Colby et al., 1971) systems
  - ✓ **Corpus-based systems**
    - ✓ Mimic human conversations by training on large amounts of human-human conversational data
    - Example - Microsoft Xiolce

# Social Chatbots : Rule-based systems

(Jurafsky et.al. 2020)

- ELIZA (Weizenbaum, 1966) worked by pattern/transform rules like the following one:

```
(0 YOU 0 ME) [pattern]
->
(WHAT MAKES YOU THINK I 3 YOU) [transform]
```

You hate me

into:

WHAT MAKES YOU THINK I HATE YOU

```
Men are all alike.
IN WHAT WAY
They're always bugging us about something or other.
CAN YOU THINK OF A SPECIFIC EXAMPLE
Well my boyfriend made me come here.
YOUR BOYFRIEND MADE YOU COME HERE
He says I'm depressed much of the time.
I AM SORRY TO HEAR YOU ARE DEPRESSED
::
```

*Example dialogue by ELIZA*

- Each ELIZA pattern/rule is linked to a keyword that might occur in a user sentence



# Social Chatbots : Corpus-based systems

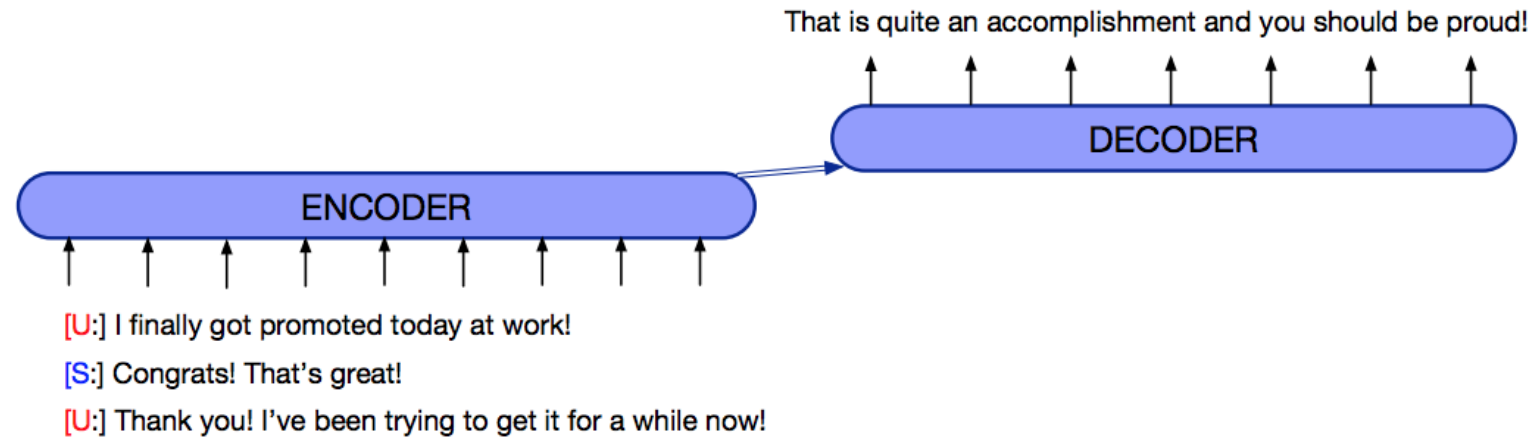
(Jurafsky et.al. 2020)

- **Response by retrieval:** considering user's turn as a query  $q$ , the goal is to retrieve and repeat some appropriate turn  $r$  as the response from a corpus of conversations  $\mathcal{C}$  (training set for the system)
- Score each turn in  $\mathcal{C}$  as a potential response to the context  $q$  and select the highest-scoring one.

$$\text{response}(q, \mathcal{C}) = \operatorname{argmax}_{r \in \mathcal{C}} \frac{q \cdot r}{|q||r|}$$

# Social Chatbots : Corpus-based systems

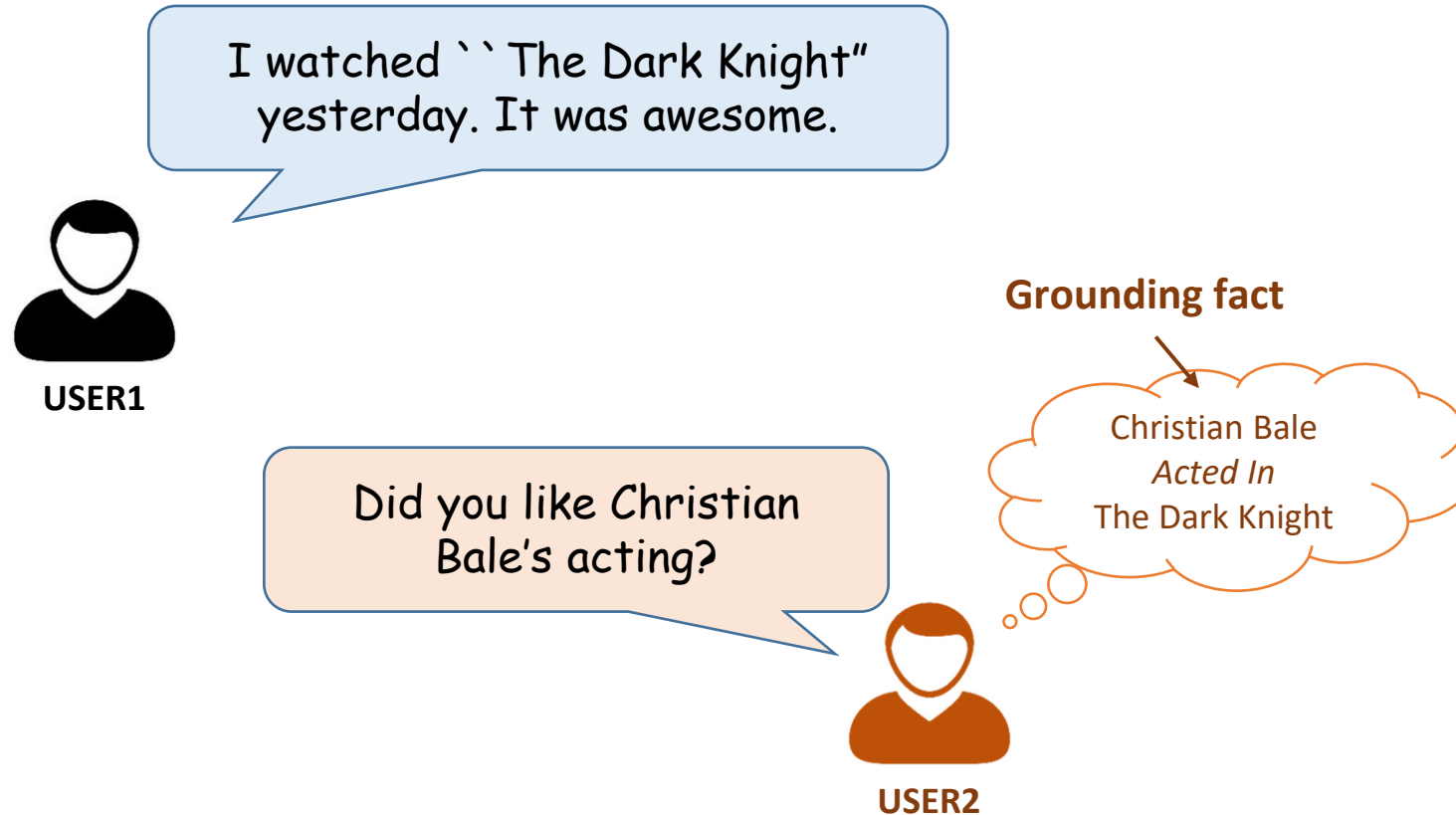
(Jurafsky et.al. 2020)



- **Response by Generation:** Models response production as an **encoder-decoder task**— transducing from the user's prior turn to the system's turn (Ritter et al., 2011; Sordoni et al., 2015b; Vinyals and Le, 2015; Shang et al., 2015).

$$\hat{r}_t = \operatorname{argmax}_{w \in V} P(w|q, r_1 \dots r_{t-1})$$

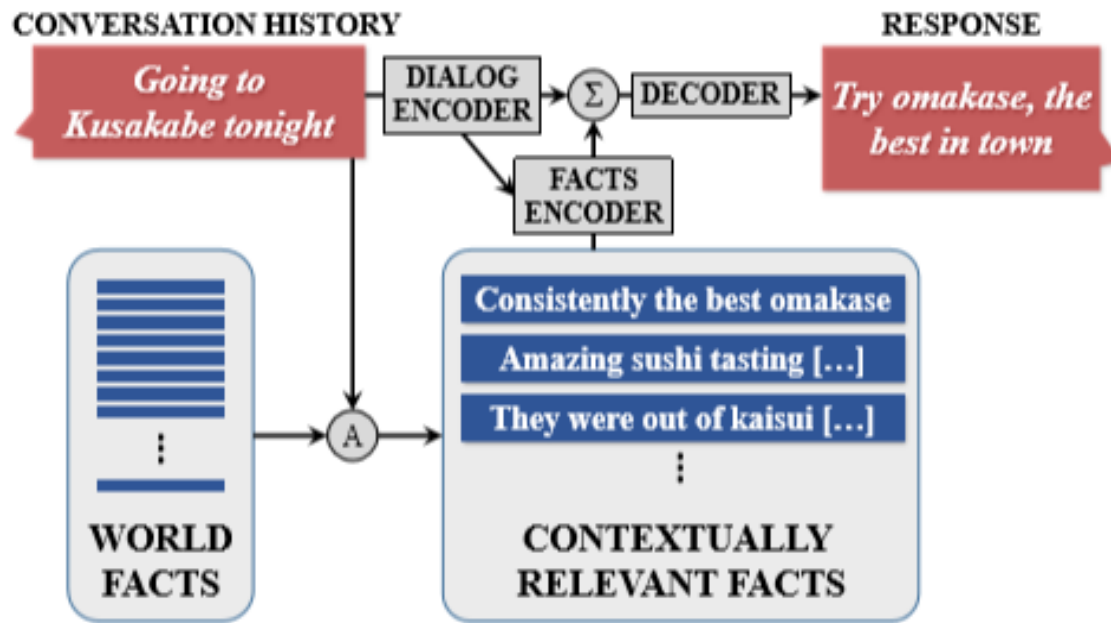
# Knowledge grounding in Conversation: Example



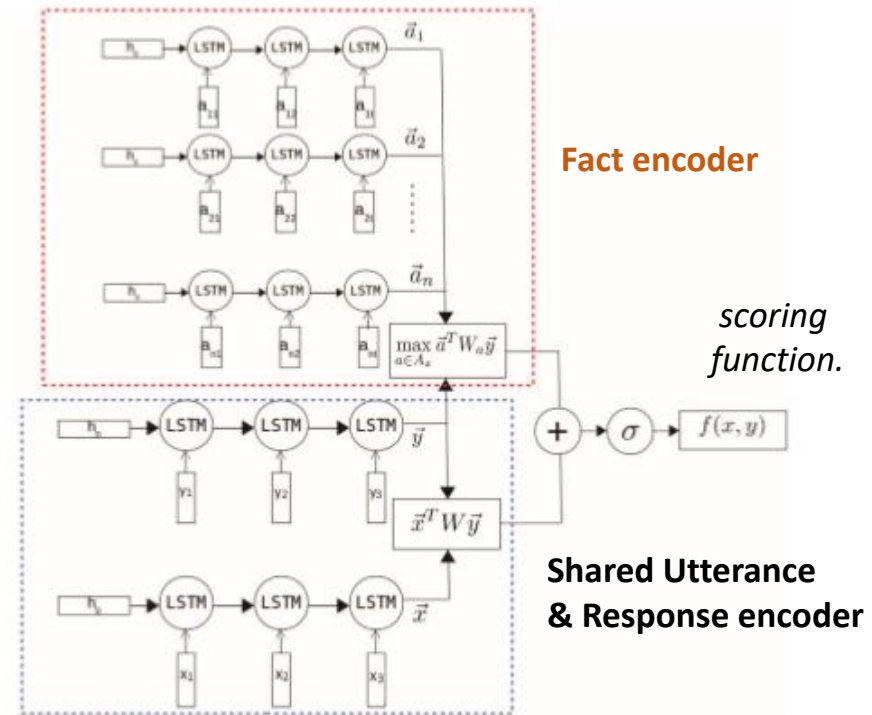
Knowledge grounding makes conversation **interesting** and **intelligent!**

# Knowledge-grounded Conversation Modeling

Recently researchers have begun to explore how to ground the chitchat in world knowledge to make the conversation more contentful and interesting



Ghazvininejad et. al., AAI-2018



Young et.al., AAI-2018

# Dialogue systems in the open-world : Challenges

- Built with **pre-collected training data, fixed rules** and **pre-compiled knowledge bases (KBs)**
  - ✓ Great deal of manual effort is needed
  - ✓ No matter how much data is collected, can't cover all possible variations of natural language.
- Pre-compiled KB **can't cover all rich knowledge** needed in practice
  - ✓ Knowledge bases are incomplete (West et. al. 2014)
  - ✓ KB of existing systems does not grow over time!

# Chatbots should learn continually after deployment

(Chen & Liu 2018, Liu 2020, Liu & Mazumder 2021)

- **Chatbot:** human users may say things a chatbot does not understand.
  - It must learn **new knowledge** and **new language expressions during chatting**.
    - E.g., asking the current or other users.
  - Humans learn a great deal in our daily conversations
- Chatbots **should not** solely rely on offline training initiated by engineers.



# Learning during Conversation : **Scopes**

(Liu 2020, Liu & Mazumder 2021)

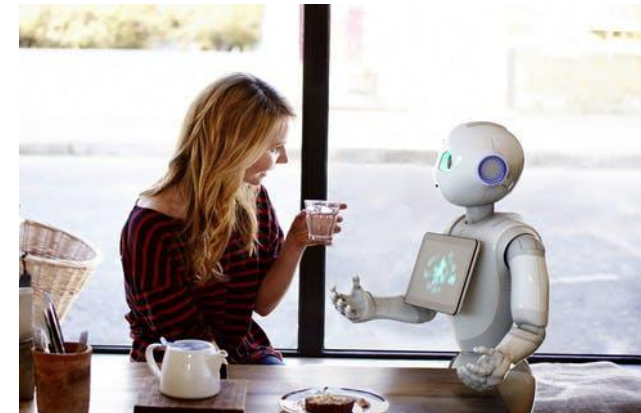
## ■ **Passive learning**

- ✓ Learning by reading web corpus, web tables or past conversation [information extraction]



## ■ **Interactive learning**

- ✓ Learning through interactive multi-turn dialogue [**our focus**]



---

# Goals of this Tutorial

- Introducing the **paradigm of lifelong or continual learning** and discuss various **related problems and challenges** in the context of **conversational AI** applications.
- Recent advancements in **continual learning in Chatbots** after model deployment via **interactions with end-users**.
- A discussion on the **future scope** for continual conversational learning and **open challenges**.



---

# Outline

- I. **Lifelong and Continual Learning: An Introduction**
- II. Continuous Knowledge Learning during Conversation
- III. Continual Language Learning and Grounding
- IV. Open-Domain Dialogue Learning After Deployment
- V. Continual Learning for Task-oriented Dialogue Systems
- VI. Continual Learning of Conversational Skills
- VII. Other Challenges & Summary

# Classic definition of lifelong/continual learning

(Thrun 1996, Silver et al 2013; Ruvolo and Eaton, 2013; Chen and Liu, 2016, 2018)

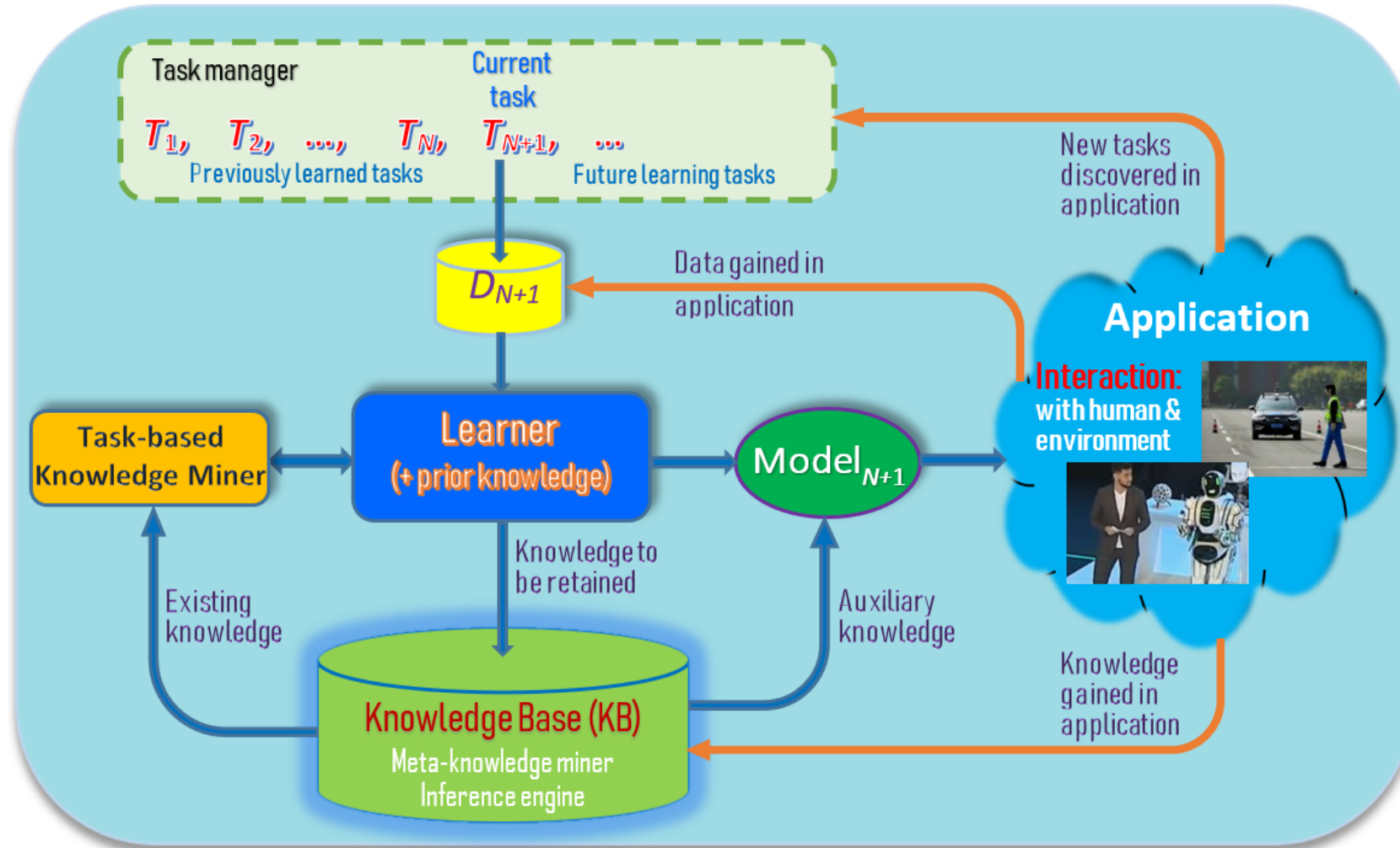
- Learn a sequence of tasks,  $T_1, T_2, \dots, T_N, \dots$  incrementally. Each task  $t$  has a training dataset  $D_t = \{x_{t,i}, y_{t,i}\}_{i=1}^{n_t}$
- **Goal:** learn each new task  $T_{N+1}$  incrementally
  1. **with no catastrophic forgetting:** Learning of the new task  $T_{N+1}$  should not result in degradation of accuracy for previous  $N$  tasks.
  2. **with knowledge transfer:** leveraging the knowledge learned from previous  $N$  tasks to learn the new task  $T_{N+1}$  better.
- **Assumption:** Both the task  $T_{N+1}$  and its training data  $D_{N+1}$  are **given** by the user.

# Continual learning in Machine Learning

- All tasks are learned in a single neural network
  - Each task consists of a set of classes to be learned
  - **Challenges:** *catastrophic forgetting* and *knowledge transfer*
- **Class continual learning (Class-CL)**
  - produce a single model from all tasks
  - classify all classes during testing
- **Task continual learning (Task-CL)**
  - train a “separate” model for each task
  - task-id is provided during testing

# Continual learning with learning after deployment

(Chen & Liu, 2018, Liu, 2020, Liu & Mazumder 2021)



**Orange lines:**  
Learning after  
model deployment  
(Learning on  
the job)

# Characteristics of continual learning

(Chen and Liu, 2018, Liu, 2020)

- **Continuous incremental learning process (no forgetting)**
  - ✓ *Without forgetting*: Learning a new task should not forget the past.
- **Knowledge accumulation in KB (long-term memory)**
- **Knowledge transfer/adaptation (across tasks) (Ke, Liu, Huang, 2020)**
  - ✓ *Using/adapting past knowledge* to help learn new tasks
- **Learning after deployment (on the job). *Self-supervision* using the *accumulated knowledge* and *interaction* with humans & environment.**

# Closed-world assumption and open-world

(Fei et al, 2016; Shu et al., 2017)

## ■ Traditional machine learning:

□ Training data:  $D^{train}$  with class labels  $Y^{train} = \{l_1, l_2, \dots, l_t\}$ .

□ Test data:  $D^{test}, Y^{test} \subseteq \{l_1, l_2, \dots, l_t\}$

## ■ Closed-world: $Y^{test} \subseteq Y^{train}$

- Classes appeared in testing must have been seen in training, **nothing new**.
- A system that is **unable to identify anything new**, it cannot learn by itself.

## ■ Open-world: $Y^{test} - Y^{train} \neq \phi$

- There are unseen classes in the test data, out-of-distribution.

# On the job continual learning: **Main steps**

(Chen and Liu, 2018, Liu, 2020)

- **Identify new tasks to learn** (tasks not given)
  - ✓ **Discover new tasks and learn them** incrementally/continually.
    - Novel instances of existing/known classes – concept drifting.
    - **Novel/unknown classes or tasks**
- **Acquire ground-truth training data** (training data not given)
- **Learn the tasks incrementally** (one-shot or few-shot)

# Learning in the open-world: Discovering Unknowns

(Fei et al, 2016; Shu et al., 2017)

## ■ Steps:

- **Discover novel instances:** e.g., classify instances in  $D^{test}$  to  $Y^{train}$  and **detect novel instances**  $D^{novel} \subseteq D^{test}$  belonging to  $L_0$  – new tasks
- **Identify the unseen/new classes** in  $D^{novel}$ ,  $L_0 = \{l_{t+1}, l_{t+2}, \dots\}$  and **gather training data**
  - **Interactive self-supervision:** interaction with humans and the environment
- **Continual learning:** Incrementally learn the new classes  $\{l_{t+1}, l_{t+2}, \dots\}$  (the new task)

Note: this does not include how the system should respond or react to novelty.



# Interactive self-supervision

(Liu, 2020)

- Identify new classes and training data by interacting with
  - **Humans: through natural language**, e.g.,
    - Self-drive cars: asking the passenger
      - What is that object? How do I drive now? Where should I stop?
    - **Chatbots**: learn new knowledge and learn language during chatting.
  - **Environment**: get feedback & use tools (e.g., search engines)
    - Need an internal evaluation system
      - to evaluate environmental feedback
  - **To gather knowledge, and supervisory or reward information.**

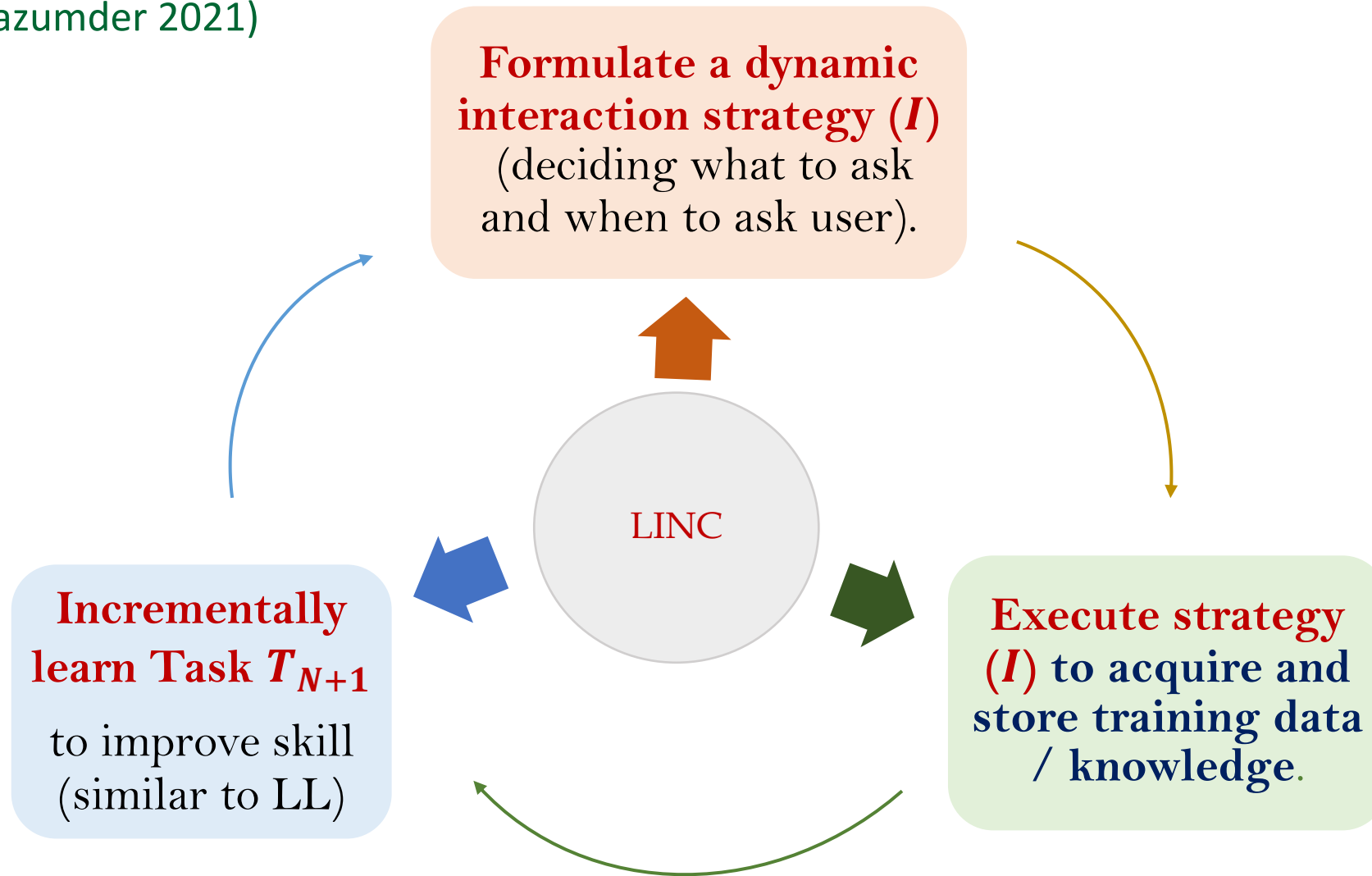
# Lifelong Interactive Learning in Conversation (LINC)

(Liu and Mazumder 2021)

- The tasks have to be **self-discovered on-the-fly** and the **training data has to be found by the agent**.
  - ✓ A new learning task  $T_{N+1}$  is formed when the agent needs to learn a piece of knowledge or encounters a problem in conversation.
- In order to learn the new task  $T_{N+1}$ , it needs to **formulate plan to obtain the ground truth training data  $D_{N+1}$  on the job**.
  - ✓ That is why the agent has to **interact with or ask the user questions** and learn from it.
  - ✓ This learning process is **like human on-the-job learning**.

# Lifelong Interactive Learning in Conversation (LINC)

(Liu and Mazumder 2021)



# Example - a greeting bot in a hotel

(Chen and Liu 2018)

- See an existing/known guest.
  - Bot: “Hello John, how are you today?”
- See a new guest. **Bot must recognize the guest is new/novel.**
  - Bot: “Welcome to our hotel! What is your name, sir?”
  - Guest: “I am David”
  - **Bot learns to recognize David automatically**
    - take pictures of David and incrementally learn to recognize him
- See David next time.
  - Bot: “Hello David, how are you today?”

# Example - a greeting bot in a hotel

(Chen and Liu 2018)

- See an existing guest.
  - Bot: “Hello John, how are you today?”
- See a new guest. **Bot recognizes the guest is new.** (create a new task)
  - Bot: “Welcome to our hotel! What is your name, sir?” (get class label)
  - Guest: “David” (got class label: **David**)
  - **Bot learns to recognize David automatically**
    - take pictures of David (get training data)
    - learn to recognize David (learn incrementally)
- See David next time.
  - Bot: “Hello David, how are you today?” (use the new knowledge)

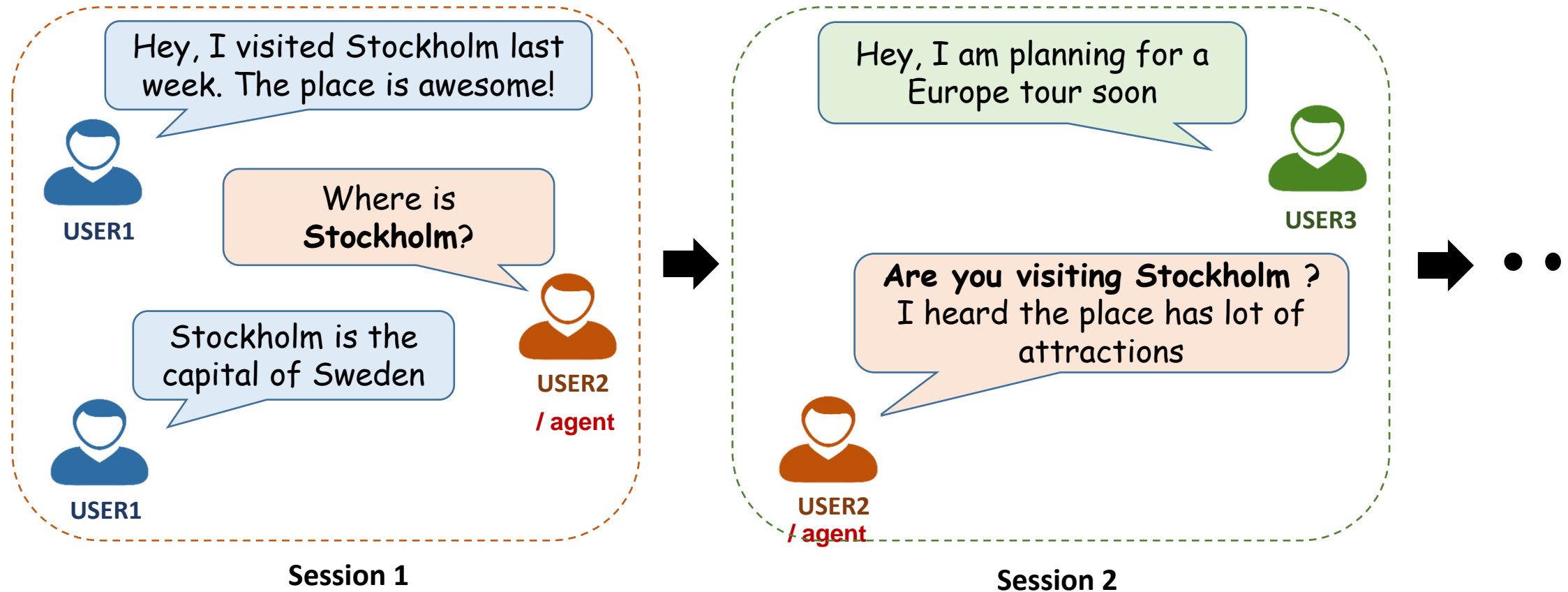
---

# Outline

- I. Lifelong and Continual Learning: An Introduction
- II. Continuous Knowledge Learning during Conversation**
- III. Continual Language Learning and Grounding
- IV. Open-Domain Dialogue Learning After Deployment
- V. Continual Learning for Task-oriented Dialogue Systems
- VI. Continual Learning of Conversational Skills
- VII. Other Challenges & Summary

# Example - Knowledge learning in conversation

Humans Learn and Leverage Knowledge in Lifelong Manner!



Knowledge learning happens in a multi-user environment

- Liu and Mazumder. *Lifelong and Continual Learning Dialogue Systems: Learning during Conversation*. AAI-2021

# Knowledge learning in chatbots: Opportunities

(Liu and Mazumder 2021)

Did you watch anything yesterday?



Watched Forest Gump. The movie was awesome. Liked Tom Hanks' performance a lot!



**Extracted Facts:**  
(*Forest Gump, is a, movie*)  
(*Tom Hanks, acted in, Forest Gump*).

1

**Extracting facts from user utterances**

3

**Ask and infer new facts in conversation**

- ✓ When the chat-bot cannot answer an user query, it can ask for some related supporting facts and then infer the answer.

Hey, is there any good place around for having **sushi** ?



What is sushi ?



Japanese dish.



**Learned new concepts/entities:**  
(*Sushi, is , food*)  
(*Sushi, has cuisine, Japanese* ).

2

**Ask questions to learn about unknown entities and concepts.**



# Continuous Factual knowledge learning in dialogues

(Mazumder et. al. 2019; 2020)

- Dialogue systems are increasingly using **knowledge bases (KBs)** storing factual knowledge to help generate responses.
  - ✓ KBs are inherently incomplete and remain fixed,
  - ✓ which limit dialogue systems' conversation capability
- **CILK: *Continuous and Interactive Learning of Knowledge*** (Mazumder et. al. 2019); **IKAI : Interactive Knowledge Acquisition and Inference** (Mazumder et. al. 2020)
  - ✓ to continuously and interactively learn and infer new knowledge during conversations

# Two types of queries or questions

(Mazumder et. al. 2019; 2020)

## ■ Wh-question

- ✓ E.g., Where was Obama born?
- ✓ (Obama, bornIn, s?)

## ■ Fact verification question

- ✓ Was Obama born in Hawaii?
- ✓ (Obama, bornIn? Hawaii)

# Components for knowledge learning



**Knowledge Base**

$\mathcal{K}$

Stores acquired Facts (Triples)

**KB: Collection of Triples**

$$\mathcal{T} = \{ (h, r, t) \mid h, t \in E, r \in R \}$$

**Triple**

**Entity Set** **Relation Set**

**Triple Store**

(Boston, LocatedInCountry, USA)

head

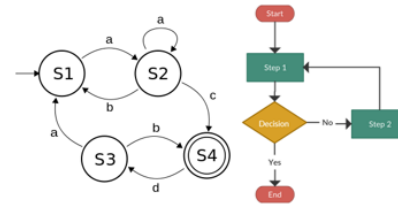
relation

tail

**OR**



**Knowledge Graph**



**Interaction Module**

$\mathcal{I}$

Interacts with user to acquire Facts

- decides whether to ask or not, and formulates questions to ask the user for supporting facts



**Inference Module**

$\mathcal{M}$

Infers new Knowledge to answer user's query

- Performs inference over the acquired Facts and existing KB

# Assumptions – Knowledge learning in dialogues

(Mazumder et. al. 2019; 2020)

- **Focus on** developing the **core interactive knowledge learning framework**
  - **Do not build all peripheral components** (like fact or relation extraction, entity linking, etc.) which are assumed to be available for use.
- We also assume that **the user has good intentions**
  - **User answers questions with 100% conformity** about the veracity of his/her facts (more discussion later)
- **User is NOT omniscient**
  - **We do not assume that the user can answer all questions**
    - Opposed to the teacher-student setup - the teacher is assumed to know everything.

# (1) When the user asks a Wh-question

(Mazumder et. al. 2019)

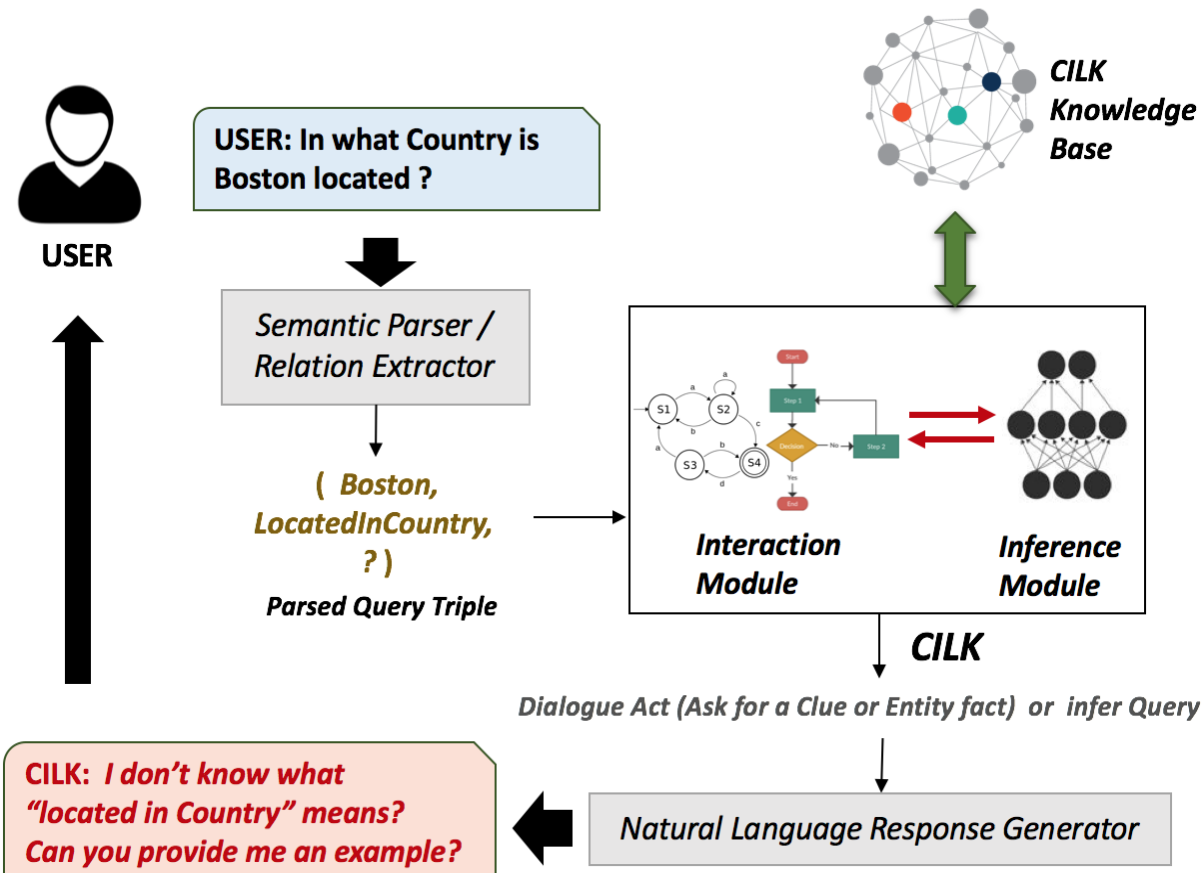
- Given a user query / question **(h, r, ?)** [or **(?, r, t)**], the goal is two-fold:
  1. **Answering** the user query or **rejecting** the query to remain unanswered if the correct answer is believed to not exist in the KB
  2. **learning / acquiring** some knowledge (supporting facts) from the user to help the answering task.
- We further distinguish two types of queries:
  - (1) **Closed-world Queries**: h (or t) and r are **known** to the KB
  - (2) **Open-world Queries**: Either one or both h (or t) and r are **unknown**

↓ **Proposed Soln.**

an engine for **Continuous and Interactive Learning of Knowledge (CILK)**

# Interactive knowledge learning in dialogue: example

(Mazumder et al. 2019)



**USER:** (Boston, LocatedInCountry, ?) “In what Country is Boston located?” [Query]

**CILK:** I do not know what “located in Country” means? Can you provide me an example? [Ask for Clue]

**USER:** (London, LocatedInCountry, UK). “London is located in UK.” [SF1]

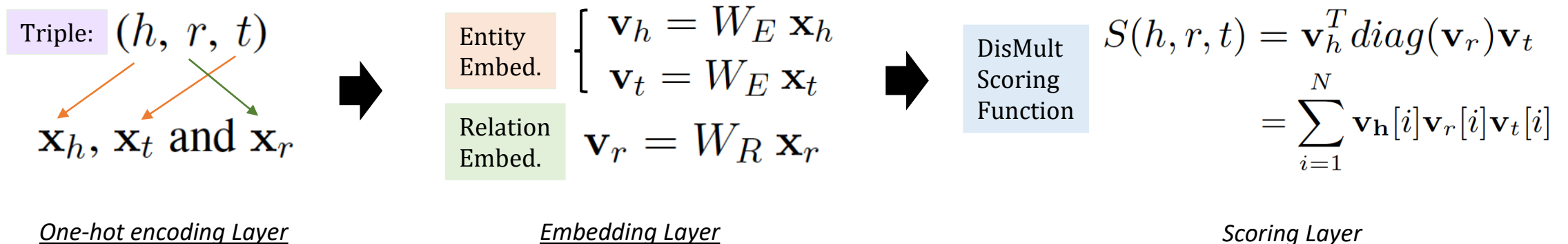
**CILK:** Got it. Can you tell me a fact about “Boston”? [Ask for Entity Fact]

**USER:** (Harvard University, UniversityLocatedIn, Boston). “Harvard university is located in Boston.” [SF2]

**CILK:** (Boston, LocatedInCountry, USA) “Boston is located in USA.” [Answer]

# CILK: The Inference Module

- We use the neural **knowledge base embedding (KBE)** [Bordes et al., 2011,2013; Yang et al., 2014] for learning  $\mathcal{M}$
- For evaluation, we adopt DistMult [Yang et al., 2014]



Max-margin Raking Loss  $\mathcal{L} = \sum_{d \in D^+} \sum_{d' \in D^-} \max\{S(d') - S(d) + 1, 0\}$

# Rejection in KB Inference

CILK maintains a **threshold buffer**  $\mathcal{T}$  at **stores entity and relation specific prediction thresholds** and updates it continuously over time.

entity e or relation r

validation query tuple set involving z

$$\mathcal{T}[z] = \frac{1}{2|D_{vd}^z|} \sum_{(q, E^+, E^-) \in D_{vd}^z} \mu_E^+ + \mu_E^-$$

mean scores of triples involving +ve entities

mean scores of triples involving -ve entities

$$\mu_E^+ = \frac{1}{|E^+|} \sum_{e_i^+ \in E^+} S(q, e_i^+)$$

$$\mu_E^- = \frac{1}{|E^-|} \sum_{e_i^- \in E^-} S(q, e_i^-)$$

$S(e, r, e_i^+)$  For tail query triple

$S(e_i^+, r, e)$  For head query triple



# Designing the Interaction Module : CILK's Interaction Strategy

## ■ CILK has to acquire supporting facts **to learn embeddings of $e$ and $r$**

- user **can only provide very few supporting facts per session** → **may not be sufficient** for learning good embeddings of  $e$  and  $r$

- Asking for too many SFs can be **annoying** and also, is **unnecessary** for entity and/or relation with good embeddings.

- Need a **sufficiently good validation dataset for learning**  $\mathcal{T}[e]$  |  $\mathcal{T}[r]$



Ask for SFs for the **known entities and/or relations for which CILK is not confident enough**, besides the unknown ones.

# Acquiring Knowledge with limited Interaction: Improving Skillset over time

- A **performance buffer**  $\mathcal{P}$  is used to store the performance statistics of  $\mathcal{M}$

$\mathcal{P}[e]$  and  $\mathcal{P}[r]$  denote the **MRR achieved by**  $\mathcal{M}$  while answering queries involving  $e$  and  $r$  respectively, *evaluated on the validation dataset*  $D_{vd}$

- At the end of each dialogue session, CILK **detects bottom**  $\rho\%$  **query relations and entities** in  $\mathcal{P}$  based on  $\mathcal{P}$  MRR scores.
  - **diffident relation and entity sets** for the next dialogue session.

↓ Strategy

Ask user for supporting facts for **diffident** and/or **unknown** query relation or entity

# CILK : Performance

(# clues, # entity facts)  
acquired per session

(#C, #EF)	WordNet			Nell		
	MRR	H@1	H@10	MRR	H@1	H@10
(1, 1)	0.30	22.09	37.83	0.23	16.89	31.14
(1, 2)	0.32	23.00	39.25	<b>0.25</b>	<b>18.11</b>	<b>31.30</b>
(1, 3)	<b>0.33</b>	<b>25.27</b>	<b>40.95</b>	0.23	17.16	30.03
(1, 3)-U	0.31	23.52	38.15	0.21	15.77	28.64
(2, 2)	0.32	23.43	39.05	0.23	16.82	30.33

Performance Buffer disabled

Performance improves in MaxTh-BTr with the increase in (acquired) entity fact triples (specially, for WordNet).

## (2) When the user asks a fact-verification question

(Mazumder et. al. 2020)

**Learn new knowledge** interactively from user when the system is unable to answer a user's factual verification (yes/no) question.

**USER:** (*Obama, CitizenOf?, USA*) “Is Obama a citizen of USA?” [Query]

**IKAI:** I do not know what “**CitizenOf**” means? Can you provide me an example? [Ask for Clue]

**USER:** (*David Cameron, CitizenOf, UK*). “David Cameron is a citizen of UK.” [SF1]

**IKAI:** Got it. Can you tell me how “**Obama**” and “**Honolulu**” are related? [CLQ]

**USER:** (*Obama, BornIn, Honolulu*). “Obama was born in Honolulu.” [SF2]

**IKAI:** Got it. Can you tell also me how “**Honolulu**” and “**Hawaii**” are related? [MLQ]

**USER:** (*Honolulu, CapitalOfState, Hawaii*). “Honolulu is the state capital of Hawaii.” [SF3]

**IKAI:** (*Obama, CitizenOf?, USA*) holds. “Yes, Obama is a US citizen.” [Answer]

# The IKAI approach

(Mazumder et. al. 2020)

- User question (h, r? s)
- **Step 1 - Interactive acquisition of supporting facts:**
  - If r is unknown, IKAI asks the user to provide a clue [an example triple r]
  - If s or t is unknown, IKAI asks the user to provide a link/relation to connect the unknown entity s or t with an automatically selected existing entity
- **Step 2 - Knowledge inference** (Infer the query answer):
  - Uses a path-ranking algorithm C-PR (Mazumder and Liu 2017) to build a predictive model (Predictor) to predict whether (s, r?, t) is true.
    - Enumerate relation paths between two entities (s, t) in a KB (encoded as a multi-relation graph) and use those paths as features to train the predictor.

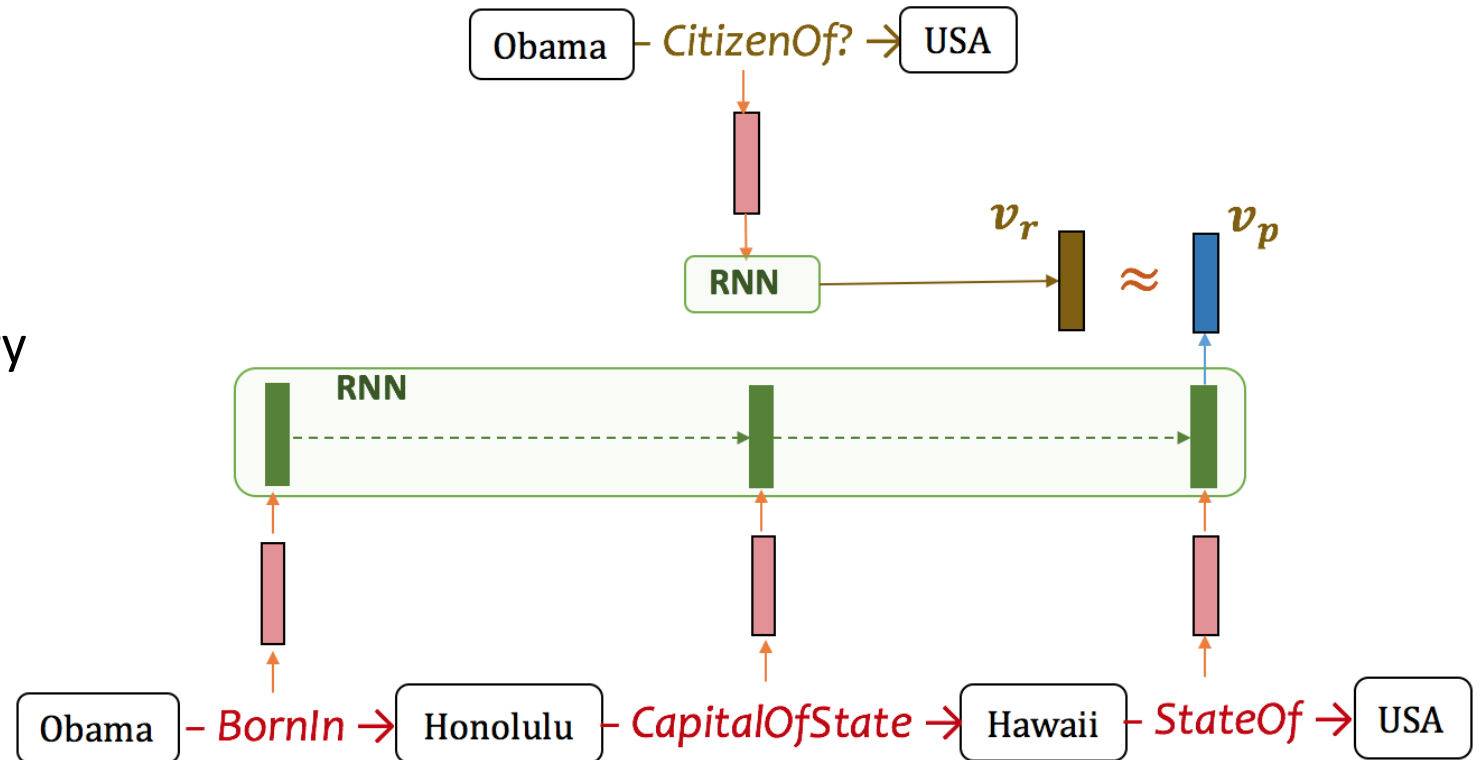
# C-PR + Compositional Vector Space\*: Inference Module of IKAI

\*(Neelakantan et. al. 2015)

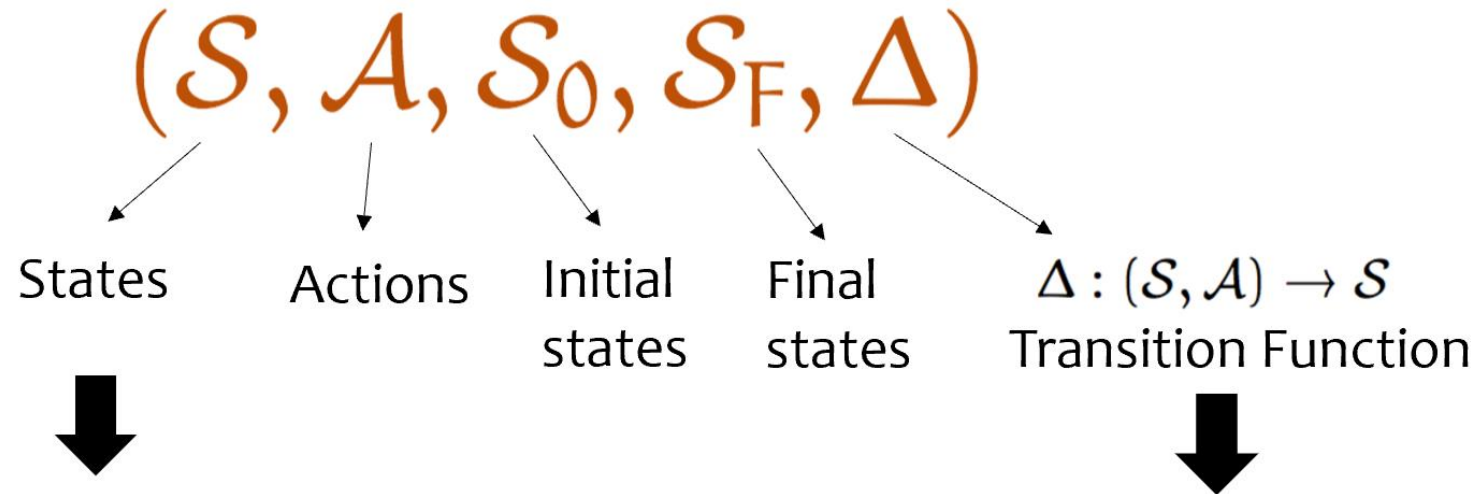
Query: *CitizenOf* (Obama, USA)?

- Encodes the path feature  $p \in P_c$  enumerated by C-PR using RNN to learn a vector representation of  $v_p$
- Uses same RNN to encode the query relation  $r$  as  $v_r$
- Inference -

$$\mathbb{P}(r|s, t) = \text{sigmoid}\left(\frac{1}{|P_c|} \sum_{p \in P_c} \cos(v_r, v_p)\right).$$



# Finite State Machine : Interaction Module ( $\mathcal{I}$ )

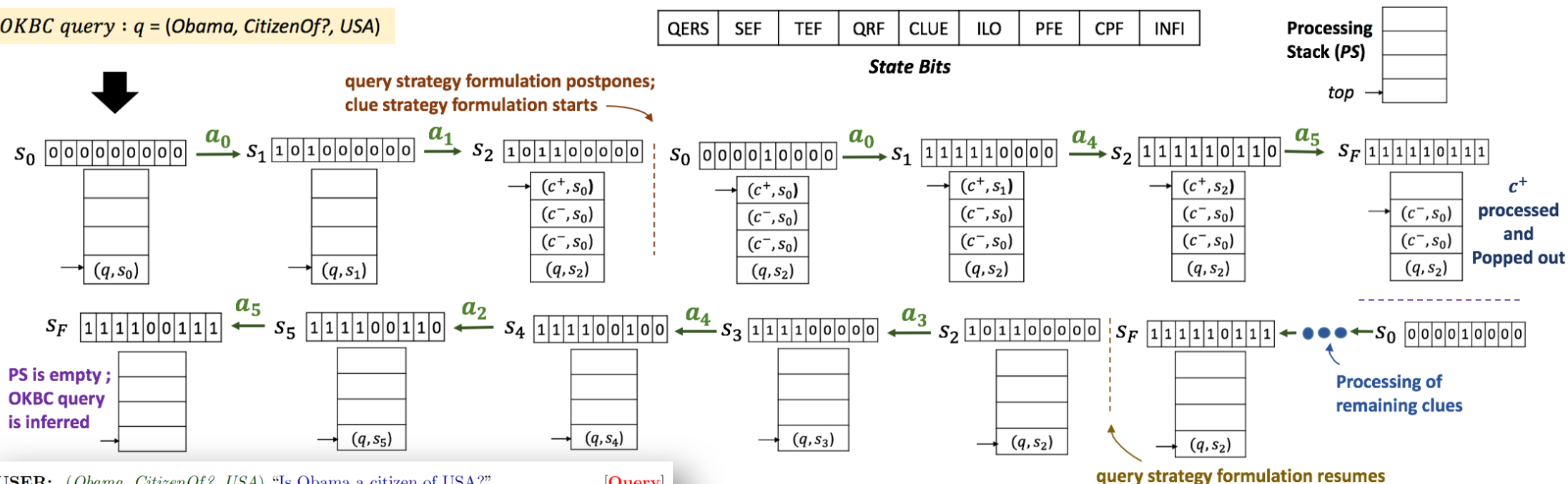


SB	Name	Description
QERS	Query entities and relation searched	Whether the query source (s) and target (t) entities and query relation (r) have been searched in KB.
SEF	Source Entity Found	Whether the source entity (s) has been found in KB.
TEF	Target Entity Found	Whether the target entity (t) has been found in KB.
QRF	Query Relation Found	Whether the query relation (r) has been found in KB.
CLUE	Clue bit set	Whether the triple (to be processed) is a clue from user.
ILO	Interaction Limit Over	Whether the interaction limit is over for the query.
PFE	Path Feature Extracted	Whether path feature extraction has been done.
CPF	Complete Path Found	Whether the extracted path features are complete.
INFI	Inference Invoked	Whether inference module has been invoked.

State Transition Conditions ( for current state bits $S_j$ [.] )	Action Id : Operation
QERS = 0	$\alpha_0$ : Search source (s), target (t) entities and query relation (r) in KB.
$ILO = 0 \wedge CLUE = 0 \wedge QERS = 1 \wedge QRF = 0$	$\alpha_1$ : Ask user to provide an clue/example for query relation r.
$PFE = 1 \wedge ILO = 0 \wedge CPF = 0$	$\alpha_2$ : Ask user to provide a missing link for path feature completion.
$QERS = 1 \wedge (SEF = 0 \vee TEF = 0) \wedge ILO = 0$	$\alpha_3$ : Ask user to provide a connecting link to add a new entity to the KB.
$QERS = 1 \wedge PFE = 0 \wedge SEF = 1 \wedge TFE = 1$	$\alpha_4$ : Extract path features between source (s) and target (t) entities using C-PR.
$QRF = 1 \wedge CPF = 1$	$\alpha_5$ : Invoke prediction model for inference.

# Working of IKAI' FSM in a given Session

OKBC query :  $q = (\text{Obama}, \text{CitizenOf?}, \text{USA})$



USER:  $(\text{Obama}, \text{CitizenOf?}, \text{USA})$  "Is Obama a citizen of USA?" [Query]  
 IKAI: I do not know what "CitizenOf" means? Can you provide me an example? [Ask for Clue]  
 USER:  $(\text{David Cameron}, \text{CitizenOf}, \text{UK})$ . "David Cameron is a citizen of UK." [SF1]  
 IKAI: Got it. Can you tell me how "Obama" and "Honolulu" are related? [CLQ]  
 USER:  $(\text{Obama}, \text{BornIn}, \text{Honolulu})$ . "Obama was born in Honolulu." [SF2]  
 IKAI: Got it. Can you tell also me how "Honolulu" and "Hawaii" are related? [MLQ]  
 USER:  $(\text{Honolulu}, \text{CapitalOfState}, \text{Hawaii})$ . "Honolulu is the state capital of Hawaii." [SF3]  
 IKAI:  $(\text{Obama}, \text{CitizenOf?}, \text{USA})$  holds. "Yes, Obama is a US citizen." [Answer]

$a_0$  : Search source (s), target (t) entities and query relation (r) in KB.  
 $a_1$  : Ask user to provide an clue/example for query relation r.  
 $a_2$  : Ask user to provide a missing link for path feature completion.  
 $a_3$  : Ask user to provide a connecting link to add a new entity to the KB.  
 $a_4$  : Extract path features between source (s) and target (t) entities using C-PR.  
 $a_5$  : Invoke prediction model for inference.



# IKAI - Performance Evaluation

Path completion  
via Blind Guessing

+ve F1 score

Macro F1 score

Dataset	Models	Rel - K / Ent -K		Rel - K / Ent -UNK		Rel - UNK / Ent - K		Rel - UNK / Ent -UNK		Overall	
		F1(+)	Macro-F1	F1(+)	Macro-F1	F1(+)	Macro-F1	F1(+)	Macro-F1	F1(+)	Macro-F1
Freebase	BG	0.584	0.629	0.494	0.569	0.432	0.532	0.388	0.501	0.508	0.579
	w/o PTL	0.555	0.652	<b>0.533</b>	<b>0.620</b>	<b>0.528</b>	0.419	<b>0.525</b>	0.418	<b>0.538</b>	0.584
	IKAI	<b>0.587</b>	<b>0.671</b>	0.493	0.591	0.525	<b>0.616</b>	0.440	<b>0.577</b>	0.532	<b>0.627</b>
WordNet	BG	0.548	0.466	0.532	0.525	0.486	0.476	0.498	0.484	0.526	0.482
	w/o PTL	<b>0.666</b>	<b>0.741</b>	<b>0.561</b>	<b>0.624</b>	0.461	0.281	0.485	0.323	0.556	0.588
	IKAI	0.655	0.694	0.552	0.604	<b>0.612</b>	<b>0.659</b>	<b>0.509</b>	<b>0.506</b>	<b>0.612</b>	<b>0.653</b>

Continuous learning past tasks (relations) Is disabled

IKAI achieves best performance overall

% of test  
Triples  
observed

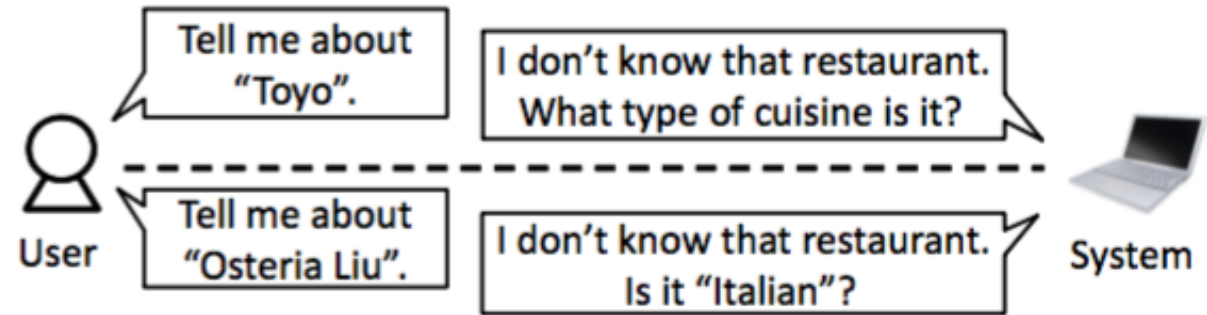
% TTO	Freebase			WordNet		
	kwn [9]	unk [293]	all [302]	kwn [21]	unk [105]	all [126]
50%	0.0	0.492	0.507	0.947	0.799	0.819
100%	<b>0.545</b>	<b>0.580</b>	<b>0.578</b>	<b>0.950</b>	<b>0.870</b>	<b>0.884</b>

IKAI Performance improvement due to User Interaction

# Lexical knowledge acquisition in dialogues

(Otsuka et. al. 2013)

- **Goal**: acquire the **attributes of unknown concepts** from users during dialogues

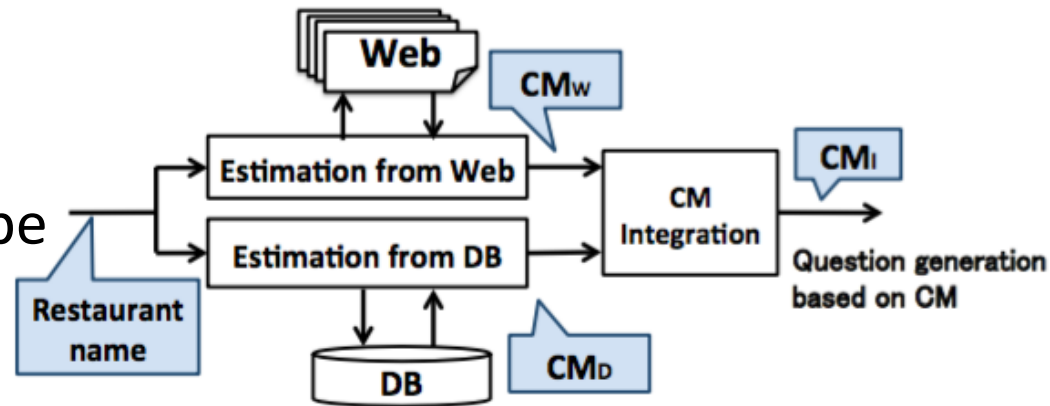


- propose a method for **generating more specific questions** than simple wh-questions to acquire the attributes
  - well- distributed **confidence measure** (CM) on the attributes to generate more specific questions.
  - Two basic CMs are defined : (1) **character and word distributions** in the target database and (2) **frequency of occurrence of restaurant attributes** on Web pages.

# Question Generation

(Otsuka et. al. 2013)

- Determines a question type on the basis of CM.
  - The CM is estimated for each cuisine type  $c_j$  in the target database.

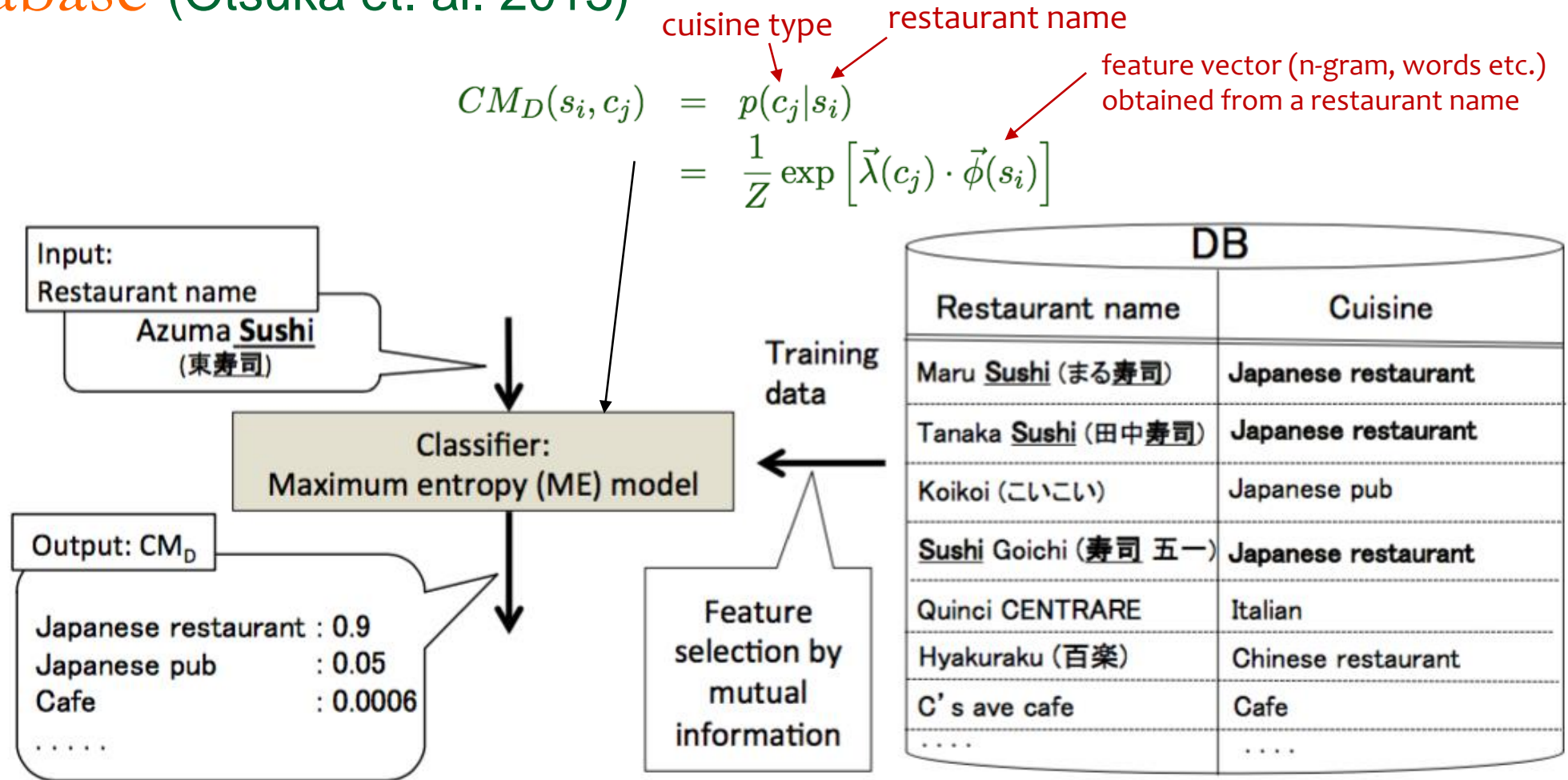


$$num = \min(n) \text{ s.t. } \sum_{j=1}^n CM(c_j) > \theta.$$

$CM(c_j)$  is a confidence measure for cuisine type  $c_j$  its descending order

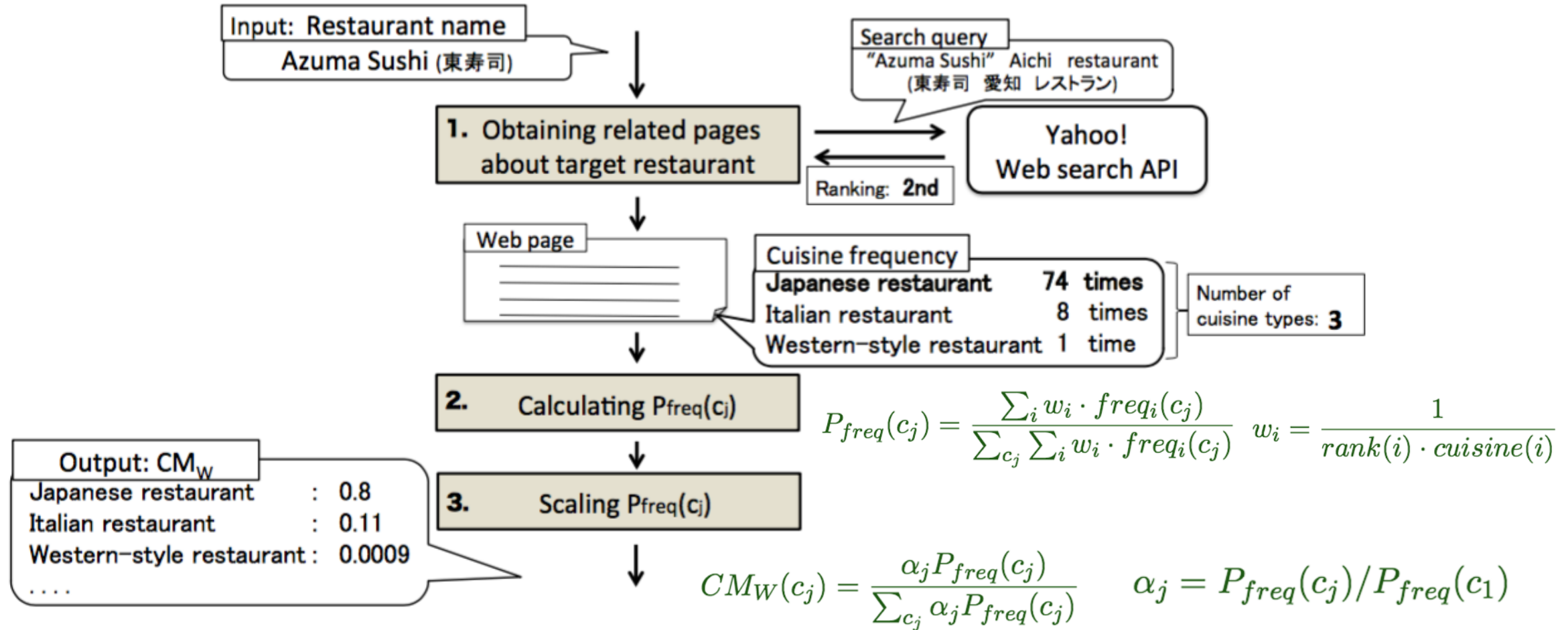
$num$	Question form	Example
1	Yes-No question	Is it cuisine $c_1$ ?
2	Alternative question	Which cuisine is it, $c_1$ or $c_2$ ?
3	3-choice question	Which cuisine is it, $c_1$ , $c_2$ , or $c_3$ ?
$\geq 4$	Wh-question	What cuisine is it?

# CM<sub>D</sub> Calculation: using Word and Character Distribution in Database (Otsuka et. al. 2013)



# CM<sub>W</sub> Calculation: Using the Web

(Otsuka et. al. 2013)



# Lexical acquisition in dialogues: Performance

(Otsuka et. al. 2013)

$CM$ range	$CM_D$		$CM_W$		$CM_I$	
	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
0.0 – 0.1	0	0	0	32	2	10
0.1 – 0.2	0	0	0	11	9	15
0.2 – 0.3	1	16	14	22	15	18
0.3 – 0.4	6	19	28	19	10	8
0.4 – 0.5	11	25	29	21	13	12
0.5 – 0.6	21	29	56	9	13	12
0.6 – 0.7	22	28	85	7	15	7
0.7 – 0.8	41	16	42	3	17	6
0.8 – 0.9	21	9	19	1	19	9
0.9 – 1.0	131	4	1	1	184	10
Total	254	146	274	124	297	103

# Knowledge acquisition in a rule-based system

(Liu and Mei, 2020)

- Many existing chatbots are written mainly with rules.
- We learn with **knowledge distillation pattern:  $(p, F, B)$** ,
  - where  $p$ : a dialogue pattern;  $F$ : implied facts;  $B$ : implied beliefs.
  - E.g.,  $p$ : [\* stayed in X at Y]     $F$ : {(X, is-a, hotel)}     $B$ : {(X, has-address, Y)}
- If user says “*I stayed in Hilton at 150 Pine Street,*” the agent gets
  - a fact: (Hilton, is-a, hotel) & a belief: (Hilton, has-address, 150 Pine Street).
- If user says “*I stayed in Hilton,*” the agent gets
  - A fact: (Hilton, is-a, hotel) and also
  - a question to ask the user: “What is the address of the hotel?”
- With such patterns, a great deal of knowledge can be learned from end-users

---

# Outline

- I. Lifelong and Continual Learning: An Introduction
- II. Continuous Knowledge Learning during Conversation
- III. **Continual Language Learning and Grounding**
- IV. Open-Domain Dialogue Learning After Deployment
- V. Continual Learning for Task-oriented Dialogue Systems
- VI. Continual Learning of Conversational Skills
- VII. Other Challenges & Summary



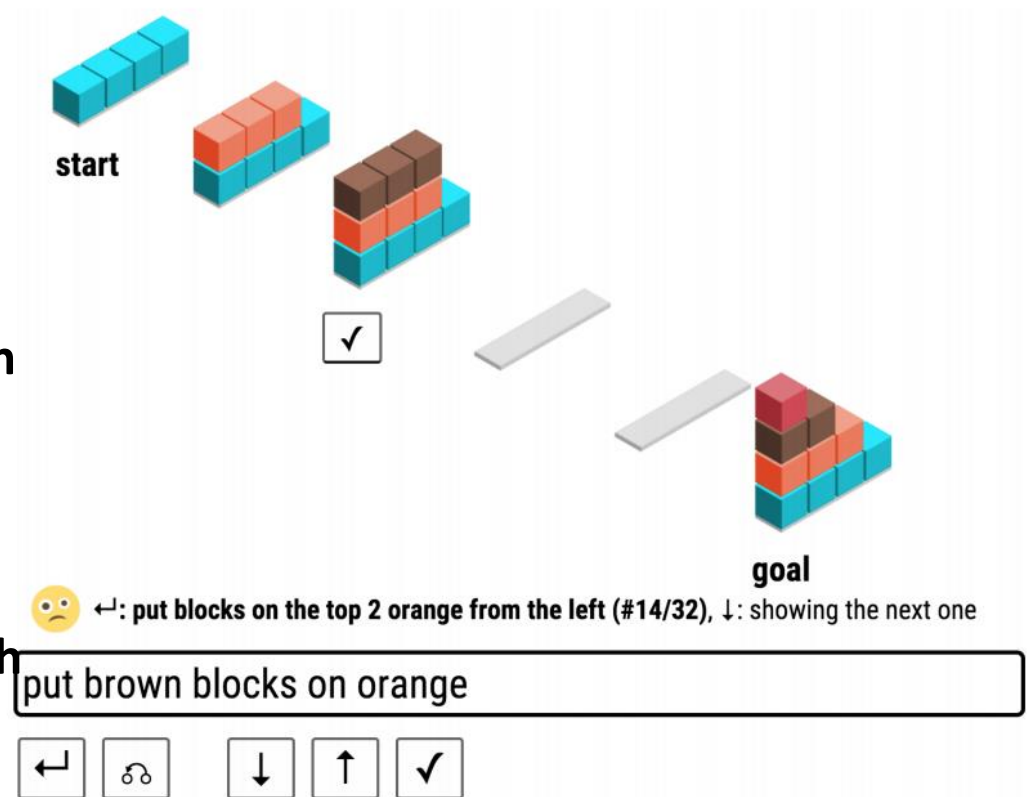
# Learning to ground natural language (NL) commands

- **Task-oriented chatbots** like virtual assistants (e.g., Siri, Alexa, etc) are **Natural Language (command) Interfaces (NLI)**
  - ✓ allow users to issue natural language (NL) commands to be mapped to some actions for execution by the underlying application.
- **Interactive Language Learning after deployment:**
  - ✓ via user demonstrations (Wang et. al. 2016).
  - ✓ via multi-turn NL dialogues with the user (Mazumder et. al. 2020)

# Learning Language Games through Interaction

(Wang et. al. 2016)

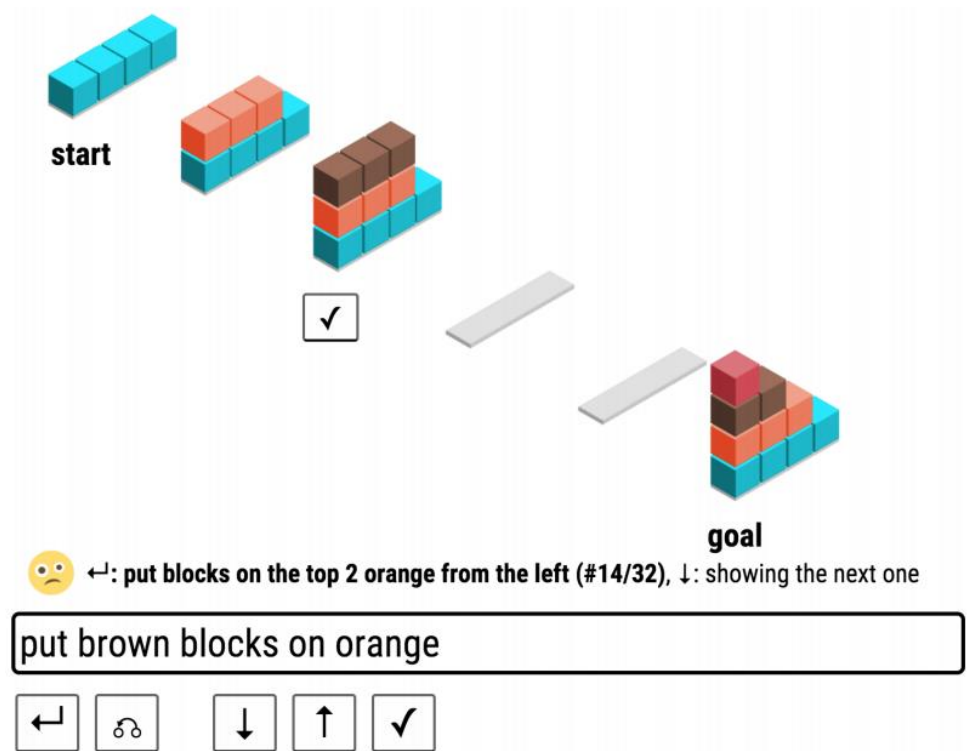
- A language learning setting relevant to building adaptive natural language interfaces.
  - inspired by Wittgenstein's language games:
    - a human wishes to accomplish some task but can **only communicate with a computer**, who performs the actual actions.
    - The **computer initially knows nothing** about language and therefore **must learn it from scratch through interaction**, while the human adapts to the computer's capabilities



The SHRDLU game

# Interactive learning through language game (ILLG) Setting (Wang et. al. 2016)

- **Goal** : to transform a start state into a goal state, but the only action the human can take is entering an utterance
  - ✓ The **computer parses the utterance** and produces a **ranked list of possible interpretations** based on its current model.
  - ✓ **Human scrolls through the list** and chooses the intended one.
  - ✓ For the computer to be successful, it **has to learn the human's language quickly over the course of the game.**



The SHRDLU game

# Compositional action space for SHRDLURN

(Wang et. al. 2016)

Rule	Semantics	Description
Set	<code>all()</code>	all stacks
Color	<code>cyan brown red orange</code>	primitive color
Color $\rightarrow$ Set	<code>with(c)</code>	stacks whose top block has color $c$
Set $\rightarrow$ Set	<code>not(s)</code>	all stacks except those in $s$
Set $\rightarrow$ Set	<code>leftmost rightmost(s)</code>	leftmost/rightmost stack in $s$
Set Color $\rightarrow$ Act	<code>add(s, c)</code>	add block with color $c$ on each stack in $s$
Set $\rightarrow$ Act	<code>remove(s)</code>	remove the topmost block of each stack in $s$

*‘remove rightmost orange block’*  
 $\rightarrow$  `remove(rightmost(with(orange)))`

# ILLG as a semantic parser

(Wang et. al. 2016)

- Maps natural language **utterances** (e.g., ‘*remove red*’) into **logical forms** (e.g., `remove(with(red))`).
- ✓ Uses a log-linear model over logical forms (actions)  $z \in Z$  given utterance  $x$ :

Prob. Assigned to  
possible mappings

$$p_{\theta}(z | x) \propto \exp(\theta^T \phi(x, z))$$

Features defined  
over utterance  
and logical form

- ✓ Parser does not have any seed lexicon and no annotated logical forms, so it just **generates many candidate logical forms**.
- ✓ Based on the human’s feedback, it **performs online gradient updates** on the parameters corresponding to simple lexical features.
  - n-grams (including skip-grams) conjoined with tree-grams on the logical form side.

# ILLG: Performance (Wang et. al. 2016)

## Evaluation on 100 players on Mechanical Turk

<b>Most successful players (1st–20th)</b>		
rem cy pos 1, stack or blk pos 4, rem blk pos 2 thru 5, rem blk pos 2 thru 4, stack bn blk pos 1 thru 2, fill bn blk, stack or blk pos 2 thru 6, rem cy blk pos 2 fill rd blk ( <b>3.01</b> )	remove the brown block, remove all orange blocks, put brown block on orange blocks, put orange blocks on all blocks, put blue block on leftmost blue block in top row ( <b>2.78</b> )	Remove the center block, Remove the red block, Remove all red blocks, Remove the first orange block, Put a brown block on the first brown block, Add blue block on first blue block ( <b>2.72</b> )
<b>Average players (21th–50th)</b>		
reinsert pink, take brown, put in pink, remove two pink from second layer, Add two red to second layer in odd intervals, Add five pink to second layer, Remove one blue and one brown from bottom layer ( <b>9.17</b> )	remove red, remove 1 red, remove 2 4 orange, add 2 red, add 1 2 3 4 blue, remove 1 3 5 orange, add 2 4 orange, add 2 orange, remove 2 3 brown, add 1 2 3 4 5 red, remove 2 3 4 5 6, remove 2, add 1 2 3 4 6 red ( <b>8.37</b> )	move second cube, double red with blue, double first red with red, triple second and fourth with orange, add red, remove orange on row two, add blue to column two, add brown on first and third ( <b>7.18</b> )
<b>Least successful players (51th–)</b>		
holdleftmost, holdbrown, holdleftmost, blueonblue, brownonblue1, blueonorange, holdblue, holdorange2, blueonred2, holdends1, holdrightend, hold2, orangeonorangerightmost ( <b>14.15</b> )	‘add red cubes on center left, center right, far left and far right’, ‘remove blue blocks on row two column two, row two column four’, remove red blocks in center left and center right on second row ( <b>12.6</b> )	laugh with me, red blocks with one aqua, aqua red alternate, brown red red orange aqua orange, red brown red brown red brown, space red orange red, second level red space red space red space ( <b>14.32</b> )

# ILLG: Performance

(Wang et. al. 2016)

- **memorize**: featurize entire utterance and logical form non-compositionally;
- **half model**: featurize the utterances with unigrams, bi-grams, and skip-grams but conjoin with the entire logical form;
- **full model**: the proposed full model
- **full+prag**: the proposed full model with online pragmatics algorithm

Method	players ranked by # of scrolls			
	top 10	top 20	top 50	all 100
memorize	25.4	24.5	22.5	17.6
half model	38.7	38.4	36.0	27.0
half + prag	43.7	42.7	39.7	29.4
full model	48.6	47.8	44.9	33.3
full + prag	52.8	49.8	45.8	33.8

Average online accuracy under various settings

# Natural Language to Natural Language (NL2NL) matching (Mazumder et. al. 2020)

- **Goal:** An adaptable system to automatically serve as NLI to API-based applications .
  - ✓ One system for many diverse API-driven applications
  - ✓ Learning continuously or lifelong from users via interactions.
- **Approach:** Natural Language to Natural Language (NL2NL) matching

- ✓ Each action (API) is attached with one or more natural language (NL) representation - a set of one or more API seed commands (SCs) just like a NL command from the user to invoke the API.

API (arg : arg type)	Seed Commands (SCs)	Example NL command
SwitchOnLight(X1: <i>location</i> )	Switch on the light at X1; Put on light on X1	Power on the light at <b>bedroom (X1)</b>
SwitchOffLight(X1: <i>location</i> )	Switch off the light at X1; Power off the light at X1	Turn off the light at <b>living room (X1)</b>
ChangeLightColor (X1: <i>location</i> , X2: <i>color</i> )	Change the X1 light to X2; I want X1 light as X2	Please make the color of <b>bedroom (X1)</b> light <b>blue (X2)</b>

- ✓ When the user issues a NL command, the system simply matches the command with one of the system's SCs.



# Command Matching & Learning (CML)

(Mazumder et. al. 2020)

- CML works based on NL2NL matching idea.
- Consists of Three components
  - **SC (seed command) specification**
    - ✓ enable application developer to **specify a set of SCs for each of their APIs**
  - **Command grounding module**
    - ✓ ground a user command  $C$  to an action SC by **matching  $C$  with the correct SC** (whose associated action API is then executed)
  - **Interactive learner**
    - ✓ It interacts with end-users in multi-turn dialogues to continually **learn new SCs and paraphrases of API argument values.**

# SC Specification (blocks-world)

(Mazumder et. al. 2020)

Table 2: Action SC specifications for Blocks-World application and some example NL commands from user for each API. (\*) denotes that the variable do not take part in command reduction (Utility Constraints), which is automatically detected and marked by CML (see Sec 3.2) (X denotes input).

Action API Function	AID	Action SCs (';' separated)	Variable: Argument Type	Example commands
AddBlock (X1)	1	add a block at X1; insert a block at X1	X1: 'location' (*)	add a block at (2, 3); put a block at (2, 3)
Remove (X1)	2	remove X1	X1: 'block_set'	delete blue block; take away blue block
Move (X1, X2)	3	move X1 to X2; shift X1 to X2	X1: 'block_set', X2: 'location' (*)	move blue block to the left of cube; shift green cube to (4, 5)
MoveByUnits (X1, X2, X3)	4	move X1 along X2 by X3 units	X1: 'block_set', X2: 'direction', X3: 'number'	move blue block left by 2 units; shift green cube down by 3 units
UpdateColor (X1, X2)	5	change color of X1 to X2; color X1 with X2	X1: 'block_set', X2: 'color' (*)	color A red; change color of B to blue
UpdateShape (X1, X2)	6	change shape of X1 to X2	X1: 'block_set', X2: 'shape' (*)	set the shape of A to cube; make B square
Rename (X1, X2)	7	rename block X1 to X2	X1: 'block_set', X2: 'name' (*)	Name the block at (4, 5) as C; rename A to D

# Command grounding module (CGM)

(Mazumder et. al. 2020)

## ■ Rephraser and Tagger ( $R$ ):

- ✓ Given the user command  $C$ ,  $R$  rephrases  $C$  and tags each word or phrase in the rephrased  $C$  with either 'O' (i.e., not an argument type) or one of the possible argument types of the action SCs.

## ■ SC Matcher ( $M$ ):

- ✓ Given the rephrased and tagged command  $C$  and the set  $T$  of (action or utility) SCs,  $M$  computes a match score  $f(t, C)$  for each  $t$  in  $T$  and returns the top ranked SC.
- ✓ This work uses an information retrieval (IR) based unsupervised matching model for  $M$

# Command grounding module (contd.)

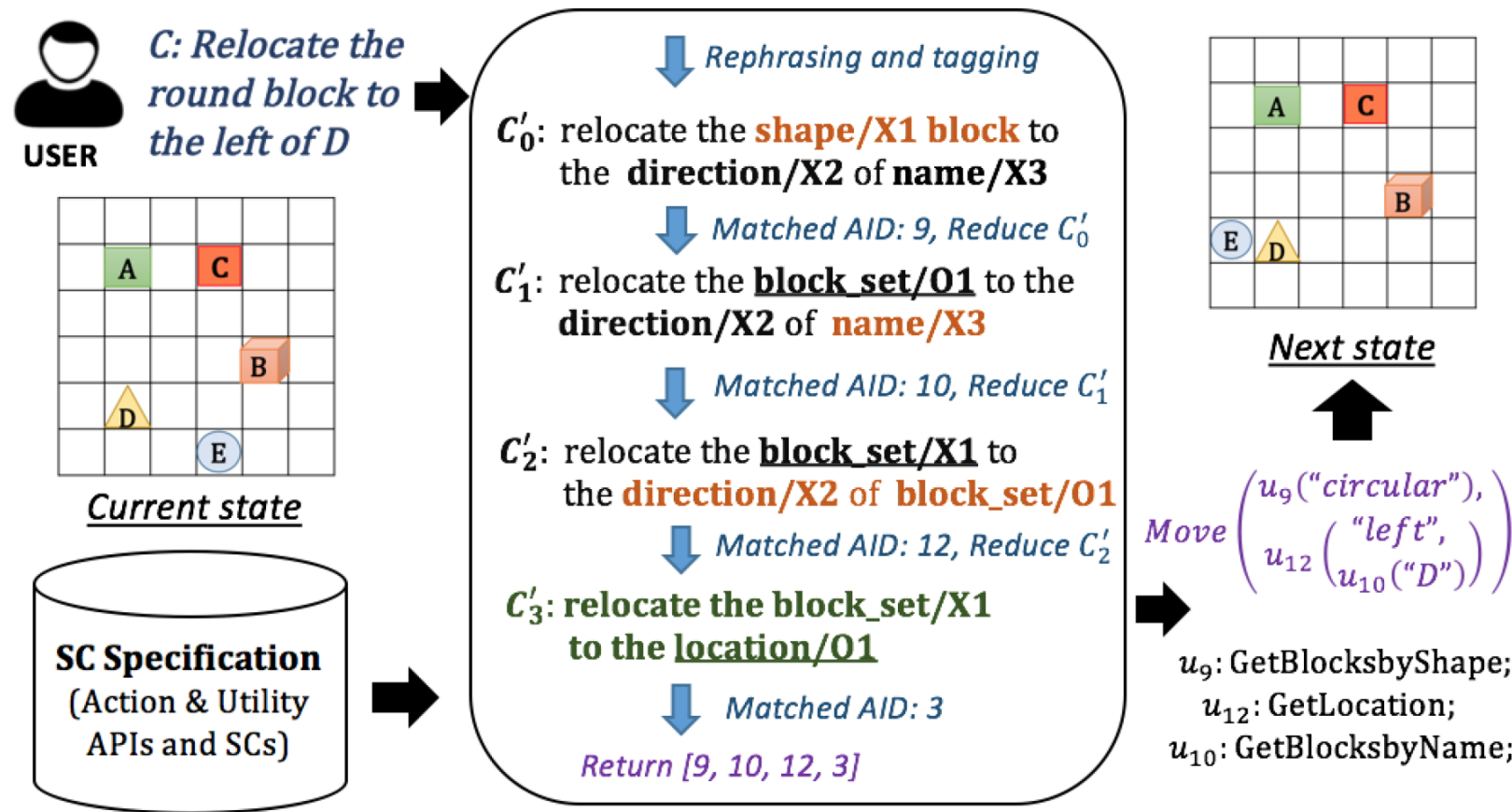
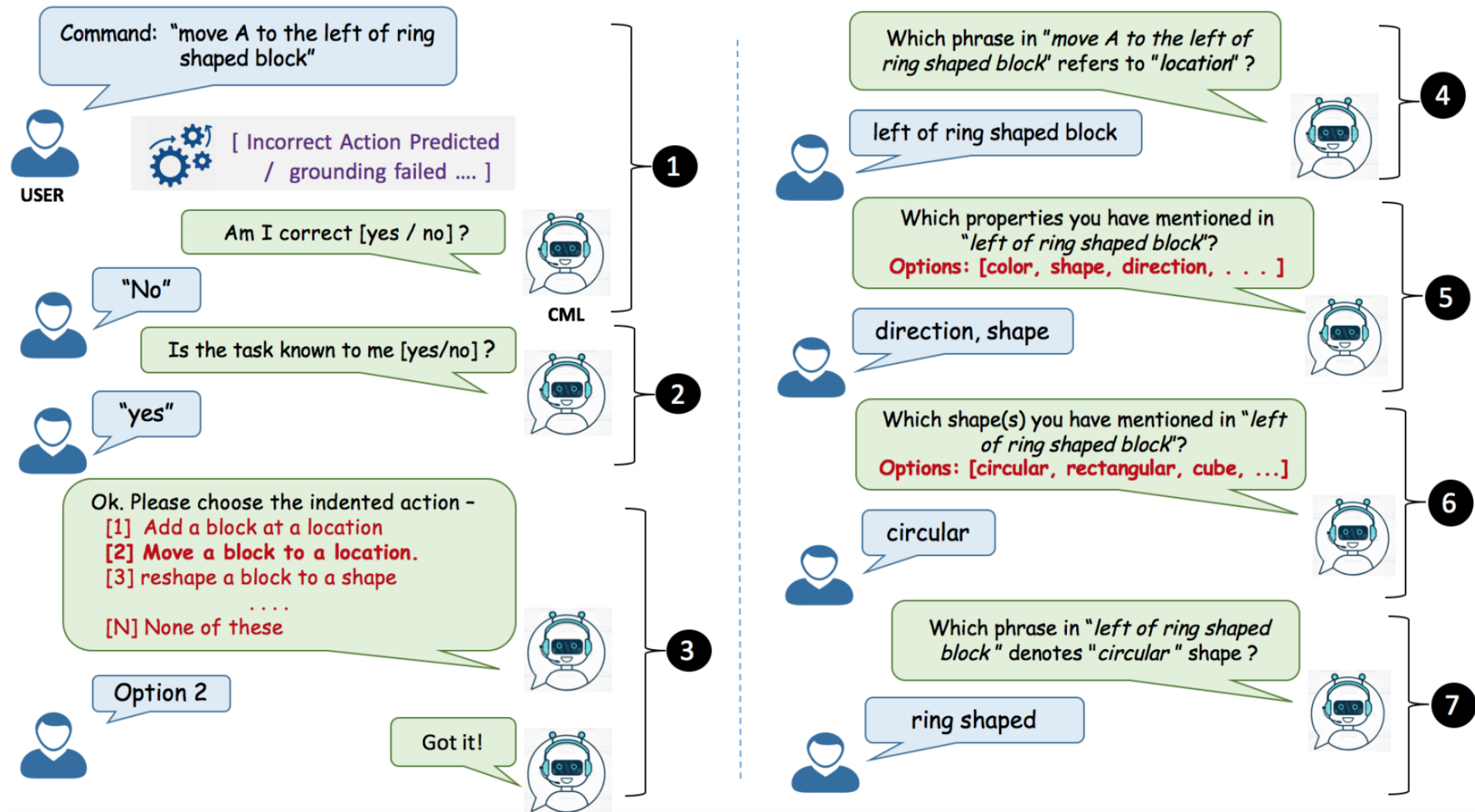


Figure 1: Working of CGM on a user command for Blocks-World. AID denotes the API IDs (see Tables 2 and 3).

# Continual interactive learning (Mazumder et. al. 2020)



- Mazumder, Liu, Wang, Esmailpour. An Application-Independent Approach to Building Task-Oriented Chatbots with Interactive Continual Learning. HLDS Workshop at NeurIPS 2020.

# Experiment results

(Mazumder et. al. 2020)

**BERT-JISF:** joint intent detection and slot filling - fine-tunes a pre-trained BERT model to solve NLU (Chen et al, 2019).

**A-acc:** action intent prediction

**Arg-F1:** argument F1

## Datasets

- BW: blocks-world
- WPD: Webpage design
- FB: flight booking

Table 5: Performance comparison of CML variants and BERT-JISF. Here, CML-vsm(-U) and CML-vsm results are the same for FB as utility APIs are absent in FB specifications].

Models	BW		WPD		FB	
	A-acc	Arg-F1	A-acc	Arg-F1	A-acc	Arg-F1
BERT-JISF	49.70	59.23	57.87	58.17	34.06	57.61
CML-jac	<b>68.93</b>	79.35	74.04	83.27	86.22	<b>97.35</b>
CML-vsm	<b>68.93</b>	79.35	<b>74.46</b>	<b>85.81</b>	<b>88.08</b>	<b>97.35</b>
CML-embed	68.63	<b>79.94</b>	68.93	82.83	83.43	<b>97.35</b>
CML-vsm (-R)	64.79	77.42	68.08	79.43	76.58	88.35
CML-vsm (-U)	14.49	14.49	11.48	11.70	-	-
CML-jac + SCL	69.82	<b>81.34</b>	76.17	84.43	92.05	<b>97.35</b>
CML-vsm + SCL	<b>70.11</b>	80.09	<b>77.02</b>	<b>85.49</b>	<b>93.32</b>	<b>97.35</b>
CML-jac + SCL + APL	72.78	<b>81.43</b>	80.0	85.31	93.15	<b>98.41</b>
CML-vsm + SCL + APL	<b>73.07</b>	80.16	<b>80.85</b>	<b>86.37</b>	<b>94.67</b>	<b>98.41</b>

---

# Outline

- I. Lifelong and Continual Learning: An Introduction
- II. Continuous Knowledge Learning during Conversation
- III. Continual Language Learning and Grounding
- IV. **Open-Domain Dialogue Learning After Deployment**
- V. Continual Learning for Task-oriented Dialogue Systems
- VI. Continual Learning of Conversational Skills
- VII. Other Challenges & Summary

# Open-domain Dialogue Learning after Deployment

- Learning by extracting new training examples from conversations
  - ✓ **Self-feeding Chatbot (Hancock et. al. 2019)**: extracts (context, response) pairs from the conversations and use it for continual training.
- Dialogue learning via role-playing games
  - ✓ **LIGHT WILD (Shuster et. al. 2020)** : Human players converse with agents situated in an open-domain fantasy world and showed that by training agents on in-game conversations they progressively improve.



# Self-feeding Chatbot

(Hancock et. al. 2019)

- Learning by extracting new training examples from conversations
  - If the **conversation appears to be going well**, the **user's responses become new training examples** to imitate.
  - Otherwise, on **making a mistake**, it **asks the user for feedback** to obtain a relevant response.
- The agent is retrained periodically using all available data

The diagram illustrates a chatbot conversation with a user. The chatbot (A) asks, "Have you been to France?" with a satisfaction score of 0.85. The user (B) responds, "Yes I have! It's beautiful." The chatbot (C) then asks, "Lol. I never eat them!" with a satisfaction score of 0.10. The user (D) responds, "What are you talking about?" The chatbot (E) asks, "Oops! I messed up. What should I have said?" The user (F) responds, "Maybe ask me what I thought about French food?"

Extracted Training Examples	
DIALOGUE	FEEDBACK
Context	Context
A Have you been to France?	A Have you been to France? B Yes, I have! It's beautiful.
Response	Feedback
B Yes, I have! It's beautiful.	F Maybe ask me what I thought about French food?

# Self-feeding Chatbot : Learning

(Hancock et. al. 2019)

## ■ Initial training phase

- The agent is trained on two tasks using supervised Human-Human (HH) dialogue examples

### ➤ Task-1: DIALOGUE

- next utterance prediction, or what should I say next?

### ➤ Task-2: SATISFACTION

- how satisfied is my speaking partner with my responses?

Extracted Training Examples	
DIALOGUE	FEEDBACK
Context	Context
A Have you been to France?	A Have you been to France?
	B Yes, I have! It's beautiful.
Response	Feedback
B Yes, I have! It's beautiful.	F Maybe ask me what I thought about French food?

# Self-feeding Chatbot : Learning

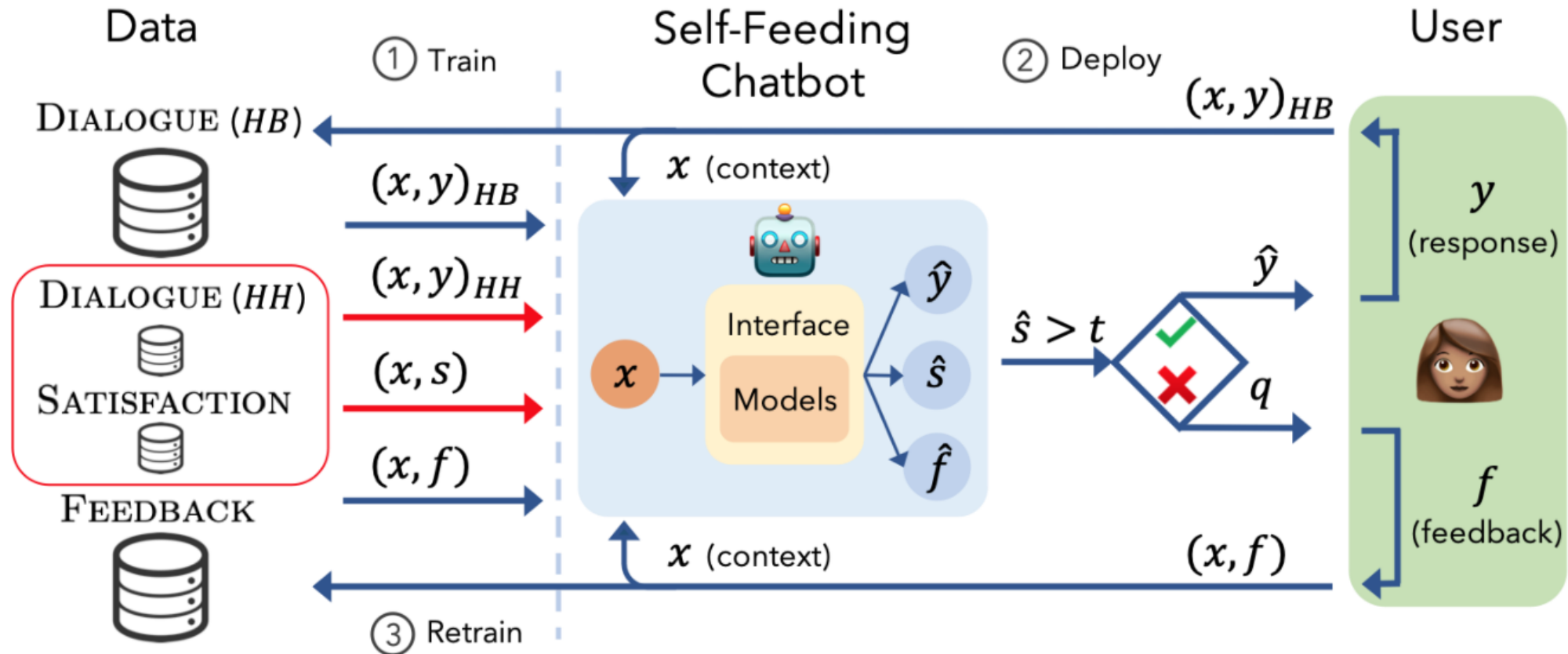
(Hancock et. al. 2019)

## ■ Deployment phase

- The agent engages in multi-turn conversations with users, extracting new *deployment* examples of two types.
  - Each turn, the agent **observes the context  $x$**  (i.e., the conversation history) and uses it to **predict its next utterance  $\hat{y}$  and its partner's satisfaction  $\hat{s}$** .
  - If  $\hat{s} > t$  (**threshold**), the agent **extracts a new Human-Bot (HB) DIALOGUE example** using context  $x$  and human's response  $y$  and continues the conversation.
  - If  $\hat{s} < t$ , the agent **requests feedback with a question  $q$** , and the resulting feedback response  $f$  is used to create a new example for the **TASK-3 : FEEDBACK task** (what feedback am I about to receive?).

# Self-feeding Chatbot : Working

(Hancock et. al. 2019)



# Self-feeding Chatbot: Model Architecture

(Hancock et. al. 2019)

- **Dialogue agent**: built on the traditional Transformer architecture
- **SATISFACTION task**: The context  $x$  is encoded with a Transformer and converted to the scalar satisfaction prediction  $\hat{s}$ .
- **DIALOGUE and FEEDBACK tasks**: set up as ranking problems: the model ranks a collection of candidate responses and returns the top-ranked one as its response.
  - Context  $x$  is encoded with one Transformer and  $\hat{y}, \hat{f}$  candidates encoded with another.
  - The score is calculated as the dot product of the encoded context and encoded candidate.

# Self-feeding Chatbot: Performance

(Hancock et. al. 2019)

Human-Bot (HB)		Human-Human (HH) DIALOGUE			
DIALOGUE	FEEDBACK	20k	40k	60k	131k
-	-	30.3 (0.6)	36.2 (0.4)	39.1 (0.5)	44.7 (0.4)
20k	-	32.7 (0.5)	37.5 (0.6)	40.2 (0.5)	45.5 (0.7)
40k	-	34.5 (0.5)	37.8 (0.6)	40.6 (0.6)	45.1 (0.6)
60k	-	35.4 (0.4)	37.9 (0.7)	40.2 (0.8)	45.0 (0.7)
-	20k	35.0 (0.5)	38.9 (0.3)	41.1 (0.5)	45.4 (0.8)
-	40k	36.7 (0.7)	39.4 (0.5)	41.8 (0.4)	45.7 (0.6)
-	60k	37.8 (0.6)	40.6 (0.5)	42.2 (0.7)	45.8 (0.7)
60k	60k	<b>39.7 (0.6)</b>	<b>42.0 (0.6)</b>	<b>43.3 (0.7)</b>	<b>46.3 (0.8)</b>

Accuracy (hits@1/20) on the DIALOGUE task's hidden test set

# Dialogue learning via role-playing games

(Shuster et. al. 2020)

- **Goal:** study the ability of an open-domain dialogue model to iteratively learn from conversations with intrinsically motivated humans.
  - ✓ Built and deploy a **role-playing game**
    - Human players role-play characters and converse with other characters (that are learning models) situated within the game world.
  - ✓ System iterates between collecting data of human-model interactions, **retraining updated models on the newly collected data, and redeploying them.**
  - ✓ provides a **natural metric to evaluate and compare models** online using the **continuation rate of players** (how long they continue playing).

# Open-domain dialogue as a game

(Shuster et. al. 2020)

- **Core Game:** involves pairing two agents in a given setting
  - Where one is a **human** and other is a **dialogue agent** (with an underlying machine learning model).
  - Two players are **assigned characters**, with given names, backstories (personas), their current location and its description.
  - Each player's **goal is simply to act out (role-play) their character's dialogue** in the given situation.



# Open-domain dialogue as a game

(Shuster et. al. 2020)

- **Role-Playing (Acting) Score:** an (automated) dungeon master (DM), to assess the quality of the player's role-playing.
  - For each dialogue turn, apply a learned model to the human player's dialogue, to **assess how likely their utterance is given the context.**
  - **convert the score in between 1 and 5 stars**, that is presented to the human player, to **reward them for good acting.**
- Acting scores are accumulated, and a player's total score is presented on a leaderboard for further motivation.

# Open-domain dialogue as a game: Example

(Shuster et. al. 2020)

**Human (grounds keeper)**

**Model (assistant chef)**

**Location:** Arch Hall. The main beauty of this room is it's Arches. Four massive arches surround the four walls of this room falling from each side. There are book shelves covered in old books...

**Persona:** I am a groundskeeper at the castle. My job is to keep the King's land immaculate. I have worked here for many years, since I was a young boy. I do hard labor.

**grounds keeper:** Good morning sir

**assistant chef:** Hello, keeper. What may I call you?

**grounds keeper:** My name is Skylar

**assistant chef:** Excellent choice. I have a new sauce recipe from bredwell across the sea. Would you like to try it?

**grounds keeper:** How kind of you, I would love to try it.

**assistant chef:** Of course. I will prepare it right away. You won't be disappointed.

**grounds keeper:** I don't believe you could ever disappoint me, Chef. Your food is always delicious

**assistant chef:** I am glad you appreciate it. The king is very particular about the kinds of sauces that he wants to serve the guests.

**grounds keeper:** Thank you for allowing me to try it. I do believe the guests and the King will enjoy the sauce as well. I myself, am eager to see how it tastes.

**assistant chef:** It should be ready within the hour.

**grounds keeper:** I will be back then, I must check upon the grounds in the mean time.

# Dialogue learning via role-playing games: Performance

(Shuster et. al. 2020)

test locations do not overlap  
with the training set locations

Model	Retrieval Model (Hits@1/20 $\uparrow$ )			Generative Model (PPL $\downarrow$ )		
	LIGHT Test	LIGHT Test Unseen	WILD Test	LIGHT Test	LIGHT Test Unseen	WILD Test
Round 1	87.12	82.43	81.61	12.67	11.81	13.42
Round 2	87.65	82.70	84.60	12.57	11.74	12.31
Round 3	87.72	83.48	87.63	12.54	11.75	11.79

- **Round 1** : models trained on LIGHT MTurk data only.
- **Round 2** : models trained on LIGHT MTurk data + 50k WILD examples collected from Round 1
- **Round 3** : models trained on LIGHT MTurk data + 50k examples from Round 1 + an additional 180k examples collected from Round 2 deployment

---

# Outline

- I. Lifelong and Continual Learning: An Introduction
- II. Continuous Knowledge Learning during Conversation
- III. Continual Language Learning and Grounding
- IV. Open-Domain Dialogue Learning After Deployment
- V. **Continual Learning for Task-oriented Dialogue Systems**
- VI. Continual Learning of Conversational Skills
- VII. Other Challenges & Summary

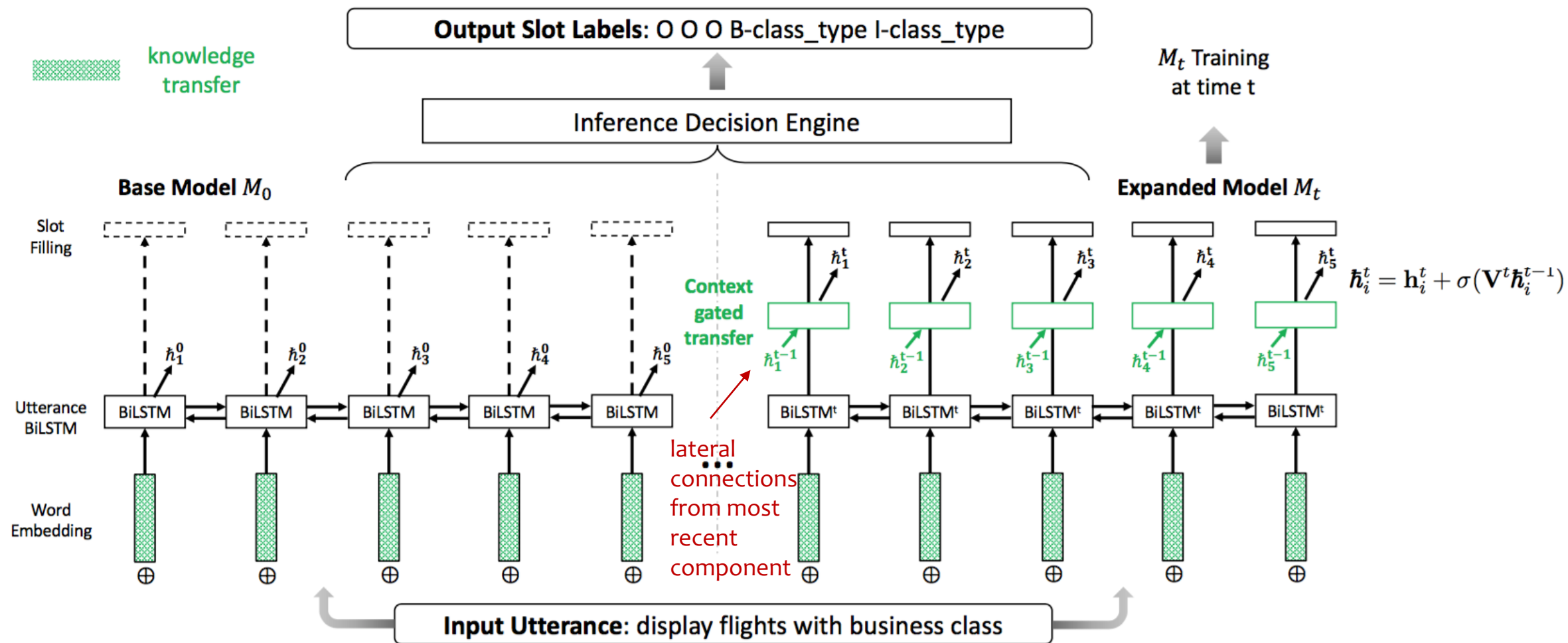
# Continual Learning for Semantic Slot Filling

(Shen et. al. 2019)

- A progressive slot filling model, *ProgModel*.
  - ✓ gradually expands at each batch by using a context gate for **knowledge transfer**.
    - *Word Embeddings Transfer and Gated Utterance Context Transfer*
  - ✓ Using the transferred knowledge, each newly expanded component is trained in a progressive manner with new data.
  - ✓ previously trained components remains untouched to avoid catastrophic forgetting.

# ProgModel Architecture

(Shen et. al. 2019)



- Shen, Zeng, & Jin. A progressive model to enable continual learning for semantic slot filling. *EMNLP-IJCNLP 2019*.

# ProgModel : Inference Decision Engine

(Shen et. al. 2019)

- non-trainable separate component to avoid the potential catastrophic forgetting.
- consider two types of decision engines:
  - ✓ *t-IDE* : ProgModel using **only the output of**  $M^t$  **decision engine**
  - ✓ *c-IDE*: for  $i^{th}$  word, it **combines all outputs** from each component  $M^t$

$$\sum_{k=0}^t P^k(i) I^k(i) \leftarrow \text{Aggregate (slot label) prob. for each word from all components}$$

- $I^k(i)$  ➤ indicator function which is 1 if  $w_i^{th}$  is in the vocabulary of  $M^k$ .
- $P_j^k(i)$  ➔ output probability of slot  $j$  for the  $i^{th}$  word from  $M^k$
- The label with maximum probability is selected.

# ProgModel Performance: ATIS Dataset

(Shen et. al. 2019)

- **FT-AttRNN** : fine tunes current model only using new training data  $U_t$
- **FT-Lr-AttRNN** : fine tunes current model using adjusted lower learning rate on  $U_t$
- **FT-Cp-AttRNN** : copies the previous model and fine tunes the new copied model on  $U_t$ . Uses both t- IDE and c-IDE decision engines and reports the one with better performance (F-1 score).
- **t-ProgModel**: using only output of M t as decision engine
- **c- ProgModel**: ProgModel using combined inference decision engine.

Approach	Batch				
	0	1	2	3	4
AttRNN (upper bound)	92.12	92.89	93.04	93.56	95.13
FT-AttRNN		91.85	89.98	91.25	88.03
FT-Lr-AttRNN		91.96	86.46	88.03	86.58
FT-Cp-AttRNN	92.12	92.10	90.06	91.98	89.67
t-ProgModel		92.33	92.43	92.57	92.58
c-ProgModel		92.40	92.64	92.71	93.91



# ProgModel Performance: Snips Dataset

(Shen et. al. 2019)

Domain	Approach	Batch		
		0	1	2
<b>Add To Playlist</b>	AttRNN (upper bound)	79.58	86.74	88.89
	FT-AttRNN		81.23	87.07
	FT-Lr-AttRNN		78.99	86.61
	FT-Cp-AttRNN	79.58	84.67	87.15
	t-ProgModel		<b>86.12</b>	<b>88.30</b>
	c-ProgModel		85.51	87.25
<b>Book Restaurant</b>	AttRNN (upper bound)	79.49	89.78	90.03
	FT-AttRNN		88.71	<b>88.09</b>
	FT-Lr-AttRNN		88.57	<b>87.89</b>
	FT-Cp-AttRNN	79.49	89.06	<b>88.14</b>
	t-ProgModel		<b>89.45</b>	<b>89.54</b>
	c-ProgModel		89.40	89.40
<b>Get Weather</b>	AttRNN (upper bound)	76.48	91.12	93.56
	FT-AttRNN		89.52	<b>88.93</b>
	FT-Lr-AttRNN		89.09	<b>88.56</b>
	FT-Cp-AttRNN	76.48	89.82	90.09
	t-ProgModel		<b>90.73</b>	<b>93.12</b>
	c-ProgModel		89.92	90.95

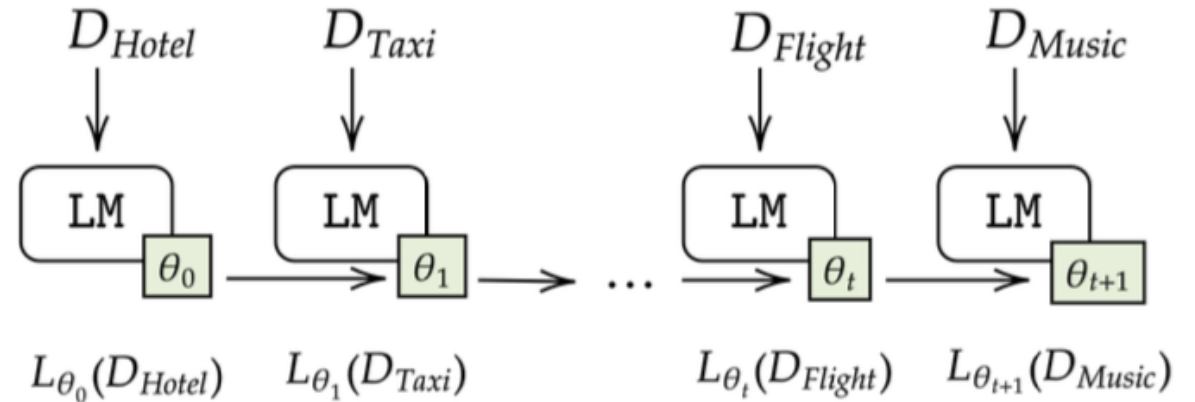
Domain	Approach	Batch		
		0	1	2
<b>Play Music</b>	AttRNN (upper bound)	77.48	87.79	89.13
	FT-AttRNN		84.71	<b>84.63</b>
	FT-Lr-AttRNN		84.53	<b>84.16</b>
	FT-Cp-AttRNN	77.48	84.85	86.10
	t-ProgModel		86.05	87.26
	c-ProgModel		<b>87.00</b>	<b>88.45</b>
<b>Rate Book</b>	AttRNN (upper bound)	92.64	98.45	99.07
	FT-AttRNN		96.87	<b>96.83</b>
	FT-Lr-AttRNN		96.20	96.86
	FT-Cp-AttRNN	92.64	97.06	97.93
	t-ProgModel		97.50	<b>98.89</b>
	c-ProgModel		<b>98.19</b>	98.20
<b>Search Creative Work</b>	AttRNN (upper bound)	66.32	89.01	89.67
	FT-AttRNN		85.93	<b>85.46</b>
	FT-Lr-AttRNN		84.69	<b>84.45</b>
	FT-Cp-AttRNN	66.32	87.25	<b>86.36</b>
	t-ProgModel		88.21	88.25
	c-ProgModel		<b>88.79</b>	<b>88.83</b>
<b>Search Screening Event</b>	AttRNN (upper bound)	89.30	95.68	97.34
	FT-AttRNN		93.40	94.53
	FT-Lr-AttRNN		91.87	93.56
	FT-Cp-AttRNN	89.30	93.81	94.56
	t-ProgModel		<b>95.01</b>	<b>96.90</b>
	c-ProgModel		93.62	94.31

- Shen, Zeng, & Jin. A progressive model to enable continual learning for semantic slot filling. *EMNLP-IJCNLP 2019*.

# Continual Learning in Task-oriented Dialogue Systems (ToDs)

(Madotto et. al. 2020)

- A continual learning benchmark for ToDS with 37 domains -



- ✓ four settings: intent recognition, state tracking, natural language generation, and end-to-end
- ✓ implement and compare multiple existing continual learning baselines
  - *regularization, rehearsal and architectural.*
- ✓ propose a simple and effective architectural method based on **residual adapters** (Houlsby et al., 2019) and a **replay-based strategy**.

# Continual Learning in ToDs: Problem Formulation

(Madotto et. al. 2020)

- Modularized setting by their input-out pairs:

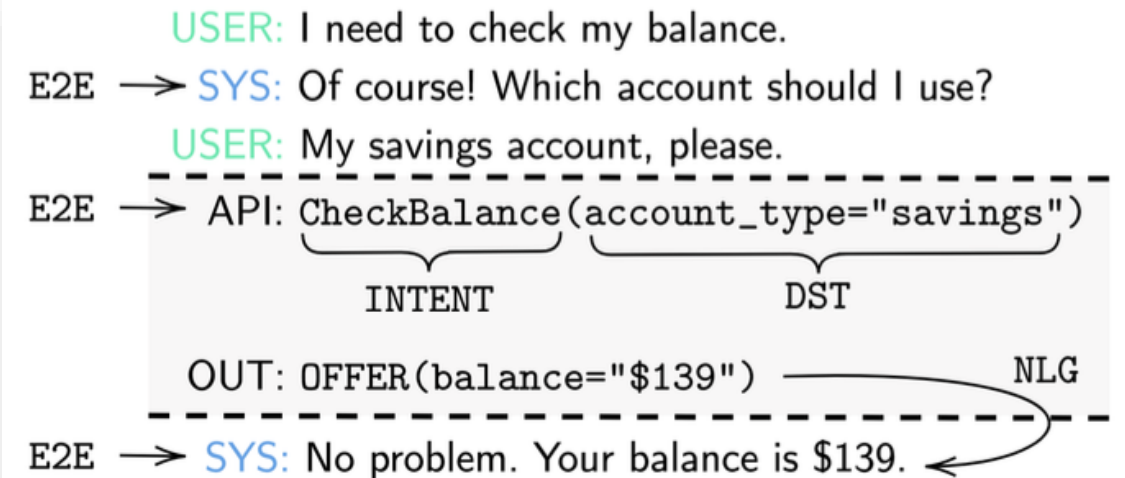
$$\begin{array}{l} H \rightarrow \mathbf{I} \quad (\text{INTENT}) \\ H \rightarrow \mathbf{I}(s_1 = v_1, \dots, s_k = v_p) \quad (\text{DST}) \\ \underbrace{\mathbf{I}(s_1 = v_1, \dots, s_k = v_p)}_{S_{OUT}} \rightarrow S \quad (\text{NLG}) \end{array}$$

H: Dialogue History  
I: intent or the api-name  
S: NL response

- End-to-End (E2E) formulation:

$$\begin{array}{l} H \rightarrow \underbrace{\mathbf{I}(s_1 = v_1, \dots, s_k = v_p)}_{S_{API}} \\ H + \underbrace{\mathbf{I}(s_1 = v_1, \dots, s_k = v_p)}_{S_{OUT}} \rightarrow S \end{array}$$

- **Dialogue Dataset**: input-out pair  
the four settings in consideration



from one of

# AdapterCL

(Madotto et. al. 2020)

- Employ a decoder-only Language Models (e.g. GPT-2) - trained to minimize the negative log-likelihood
- parameterizes each task using **Residual Adapters** (Houlsby et al., 2019) and uses a **perplexity-based classifier** to select which adapter to use at testing time.
  - **Residual adapter**: trainable parameters added on top of each transformer layer, which **steer the output distribution** of a pre-trained model **without modifying its original weights**.

# AdapterCL

(Madotto et. al. 2020)

- Residual adapter computation:

$$\text{Adapter}_{\mu_i^l}(H) = \text{ReLU}(\text{LN}(x)W_l^E)W_l^D + H$$

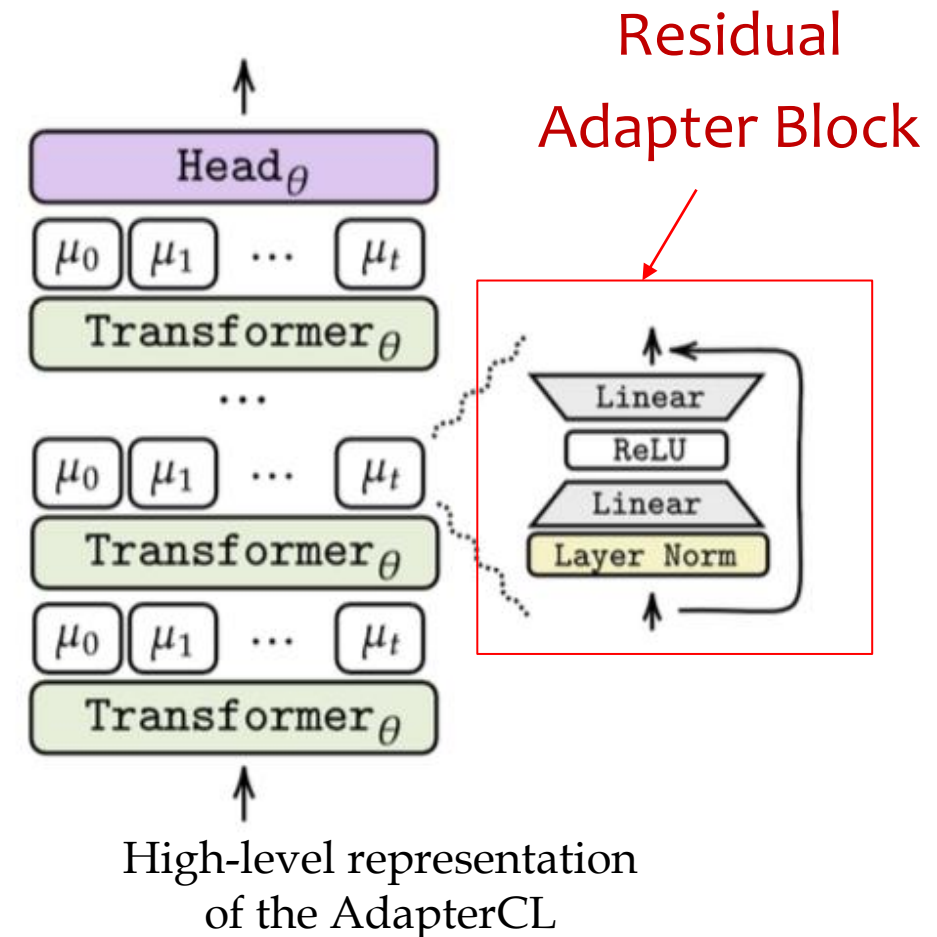
Seq. length   Hidden size

$$H \in \mathbb{R}^{p \times d}$$

- To learn new task (Dataset  $D_t$ ) spawn a new Adapter ( $\mu_t$ ) at  $\mu_t$  train its parameters.

$$\mathcal{L}_{\mu_t}(D_t) = - \sum_j \sum_{i=0}^{|D_t|} \log p_{\mu_t}(x_i | x_0, \dots, x_{i-1})$$

- Loss is optimized over  $\mu_t$  guarantee that each task is independently learned.



# AdapterCL : Perplexity-Based Classifier

(Madotto et. al. 2020)

- In the CL setting →
  - ✓ during training task-id is provided →  $\mu_t$  is optimized over  $D_t$
  - ✓ during testing task-id is **not** provided → model has to predict which adapter to use for accomplishing the task.
- Following (Wortsman et al. 2020), utilize the perplexity of each adapter over the input  $X$  as a measure of uncertainty.
  - selecting the adapter with lowest perplexity → select the most confident model to generate the output sequence.

$$\alpha_t = \text{PPL}_{\mu_t}(X) \quad \forall t \in 1, \dots, N,$$

Adapter Selected →  $t^* = \text{argmin} \alpha_0, \dots, \alpha_N$

N - Adapters

# Continual Learning in Task-Oriented Dialogue Systems:

## Performance (Madotto et. al. 2020)

$$\text{Avg.Metric} = \frac{1}{T} \sum_{i=1}^T R_{T,i}$$

Method			INTENT	DST	NLG	
	+Param.	Mem.	Accuracy $\uparrow$	JGA $\uparrow$	EER $\downarrow$	BLEU $\uparrow$
<u>VANILLA</u>	-	$\emptyset$	4.08 $\pm$ 1.4	4.91 $\pm$ 4.46	48.73 $\pm$ 3.81	6.38 $\pm$ 0.6
<u>L2</u>	$ \theta $	$\emptyset$	3.74 $\pm$ 1.4	3.81 $\pm$ 3.44	55.68 $\pm$ 7.09	5.4 $\pm$ 0.9
<u>EWC</u>	2 $ \theta $	$\emptyset$	3.95 $\pm$ 1.3	5.22 $\pm$ 4.46	58.2 $\pm$ 3.66	5.06 $\pm$ 0.5
<u>AGEM</u>	-	$t M $	34.04 $\pm$ 6.36	6.37 $\pm$ 4.0	62.09 $\pm$ 6.88	4.54 $\pm$ 0.6
<u>LAMOL</u>	-	$\emptyset$	7.49 $\pm$ 6.35	4.55 $\pm$ 3.48	66.11 $\pm$ 6.97	3.0 $\pm$ 0.9
<u>REPLAY</u>	-	$t M $	81.08 $\pm$ 1.37	30.33 $\pm$ 1.24	<b>17.72</b> $\pm$ 0.85	<b>17.4</b> $\pm$ 0.68
<u>ADAPT</u>	$t \mu $	$\emptyset$	<b>90.46</b> $\pm$ 0.6	<b>35.06</b> $\pm$ 0.52	31.78 $\pm$ 1.28	16.76 $\pm$ 0.34
<u>MULTI</u>	-	-	95.45 $\pm$ 0.1	48.9 $\pm$ 0.2	12.56 $\pm$ 0.2	23.61 $\pm$ 0.1

E2E results in term of Intent accuracy, Joint-Goal-Accuracy (JGA), Slot-Error-Rate (EER) and BLUE. +Param shows the additional number of parameters per task, and Mem the episodic memory size need per task.

---

# Outline

- I. Lifelong and Continual Learning: An Introduction
- II. Continuous Knowledge Learning during Conversation
- III. Continual Language Learning and Grounding
- IV. Open-Domain Dialogue Learning After Deployment
- V. Continual Learning for Task-oriented Dialogue Systems
- VI. **Continual Learning of Conversational Skills**
- VII. Other Challenges & Summary



# Learning user behaviors and preferences

- Given a conversation context, the chatbot can learn the user's behavioral and preference profile.
  - ✓ whether a user feels more excited or gets annoyed while conversing on a particular topic, what his/her likes and dislikes are etc.
  - ✓ PERSONALIZED MEMN2N (Luo et. al. 2019);  $P^2$  BOT (Liu et. al. 2020)

Utilize user profile knowledge in modeling future conversations to make them more engaging with the user.

# Example – Personalized goal-oriented dialogues

## Dialog 2

User Profile

Gender: **Male**, Age: **Young**, Dietary: **Non-vegetable**

Good morning.

Hey dude, what is up?

Can you make a restaurant reservation in a cheap price range in Rome for six people with Spanish food?

Ok looking for options.

Searching in knowledge base...See results in Figure 1b

Is this one cool: The\_Fancy\_Pub

Let's do it.

## Dialog 3

User Profile

Gender: **Female**, Age: **Elderly**, Dietary: **Vegetable**

Good morning.

Good day madam, how could I assist you today?

Can you make a restaurant reservation in a cheap price range in Rome for six people with Spanish food?

Thank you madam I shall provide you with options shortly.

Searching in knowledge base...See results in Figure 1b

Would you consider this option: The\_Place

Let's do it.

# Learning emotions, moods and sentiments

- Recognizing emotional state and sentiments (Zhou et al. 2018; Pamungkas 2019) of the user and leverage it to generate empathetic responses can be useful to building therapeutic chatbots.
  - ✓ Emotional Chatting Machine (Zhou et al. 2018) : can generate appropriate responses not only in content (relevant and grammatical) but also in emotion (emotionally consistent).

User: Worst day ever. I arrived late because of the traffic.
Basic Seq2Seq: You were late.
ECM ( <i>Like</i> ): I am always here to support you.
ECM ( <i>Happy</i> ): Keep smiling! Things will get better.
ECM ( <i>Sad</i> ): It's depressing.
ECM ( <i>Disgust</i> ): Sometimes life just sucks.
ECM ( <i>Angry</i> ): The traffic is too bad!

# Modeling situation-aware conversations

- Understanding the **situation** and **spatial-temporal context** of a person to decide the conversation strategy is a key characteristic of the human conversation process
  - ✓ Continuously learning from the conversation history of the user provides a scope for chatbots to learn user's conversation profile, e.g.,
    - what time of a day the user generally likes to talk or remains busy
    - whether the user is in a meeting or not ...

---

# Outline

- I. Lifelong and Continual Learning: An Introduction
- II. Continuous Knowledge Learning during Conversation
- III. Continual Language Learning and Grounding
- IV. Open-Domain Dialogue Learning After Deployment
- V. Continual Learning for Task-oriented Dialogue Systems
- VI. Continual Learning of Conversational Skills
- VII. Other Challenges & Summary**

# Other Challenges of CL in Dialogues

## ■ Dealing with Wrong Knowledge from Users

- ✓ How to deal with the issue of acquiring intentional or unintentional wrong knowledge from them ?
- ✓ Can be addressed through a **cross-verification strategy**.
  - After acquiring a piece of new knowledge, the agent can store it in an unverified knowledge buffer.
  - Next, while chatting with some other users in future sessions to accomplish related tasks, the **chatbot can ask them to verify the unverified knowledge**

# Other Challenges of CL in Dialogues

## ■ Dealing with Wrong Knowledge from Users

- ✓ How to deal with the issue of acquiring intentional or unintentional wrong knowledge from them ?
- ✓ Can be addressed through a **cross-verification strategy**.
  - After acquiring a piece of new knowledge, the agent can store it in an unverified knowledge buffer.
  - Next, while chatting with some other users in future sessions to accomplish related tasks, the **chatbot can ask them to verify the unverified knowledge**

# Other Challenges of CL in Dialogues

## ■ Revision of Knowledge

- ✓ How to revise or correct the wrong knowledge once it is detected?
- ✓ Requires -
  - a **knowledge monitoring system** : detect contradictions in the knowledge base
  - a **knowledge revision method** : revise the wrong knowledge and also all the consequences inferred from it.



# Other Challenges of CL in Dialogues

- **Learning New Task Completion Skills from Users**
  - ✓ Modern task-oriented chatbots are deployed with a **finite set of task completion skills** which they have been **preprogrammed with to perform**
  - ✓ Can end users use natural language dialogues to program their own chatbots and endow them with new skills after deployment?
    - lead to **personalization** of virtual assistants.

# Other Challenges of CL in Dialogues

- **One-shot or few shot continual learning**
  - ✓ The amount of ground-truth data that can be acquired during interaction with human users is often very small, one or a few.
  - ✓ To learn continually and effectively, we need one-shot or few-shot continual learning methods.
    - ✓ Current methods are still very weak.
- **In general, the current deep learning-based Continual Learning methods still have serious catastrophic forgetting problems.**
  - Not ready for real-world applications. Some engineering hacks or data augmentations will be needed to get around of it.

# Summary

- **Classic ML**: isolated and closed-world **offline learning**
  - No learning after deployment
- Dialogue systems or any AI agent should continuously **learn after deployment** or on the job (Liu, 2020; Liu and Mazumder, 2021)
  - The agent becomes starker and smarter
- **Current techniques are still in their infancy**, but
  - **Some methods are ready for practical applications.**

# Further Readings

- Li et. al. Continuous Learning for Large-scale Personalized Domain Classification. NAACL-HLT 2019
- Mi et. al. Continual Learning for Natural Language Generation in Task-oriented Dialog Systems. *EMNLP: Findings* - 2020.
- Chen, Z.; and Liu, B. 2014. Topic modeling using topics from many domains, lifelong learning and big data. In ICML.
- Forbes, M.; Rao, R. P.; Zettlemoyer, L.; and Cakmak, M. 2015. Robot programming by demonstration with situated spatial language understanding. In ICRA.
- Ruvolo, P.; and Eaton, E. 2013. ELLA: An efficient lifelong learning algorithm. In ICML.
- Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A.; and Pineau, J. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In AAAI.
- Thrun, S. 1998. Lifelong learning algorithms. In Learning to learn. Springer.
- Wang, S. I.; Ginn, S.; Liang, P.; and Manning, C. D. 2017. Naturalizing a programming language via interactive learning. In ACL.
- Winograd, T. 1972. Understanding natural language. Cognitive psychology .
- Zhou, H.; Young, T.; Huang, M.; Zhao, H.; Xu, J.; and Zhu, X. 2018b. Commonsense knowledge aware conversation generation with graph attention. In IJCAI

---

# Thank You

Q&A

---