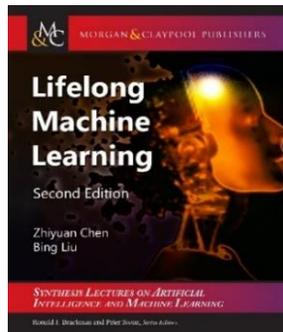

Continual Learning Dialogue Systems

– Learning after Model Deployment



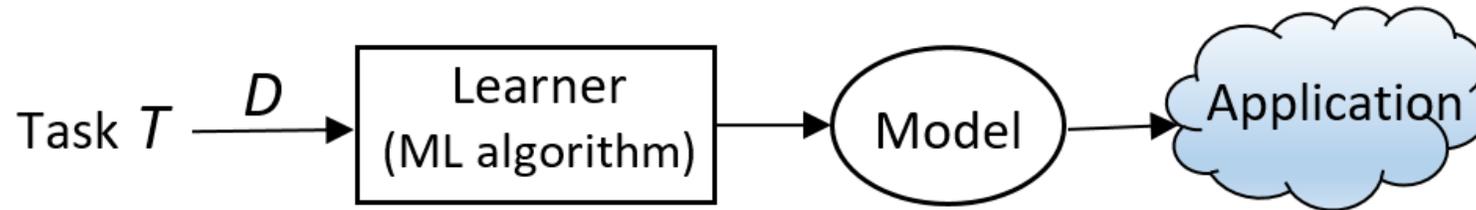
Bing Liu

Department of Computer Science

University of Illinois at Chicago

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**
Learning on the job or during model application.

Chatbots should learn continually after deployment

(Chen & Liu, 2018, Liu, 2020)

- **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.



Outline

- **Continual learning with learning after deployment**
- Continual learning to ground new language expressions
- Continual learning of factual knowledge
- Dealing with wrong knowledge from users
- Summary

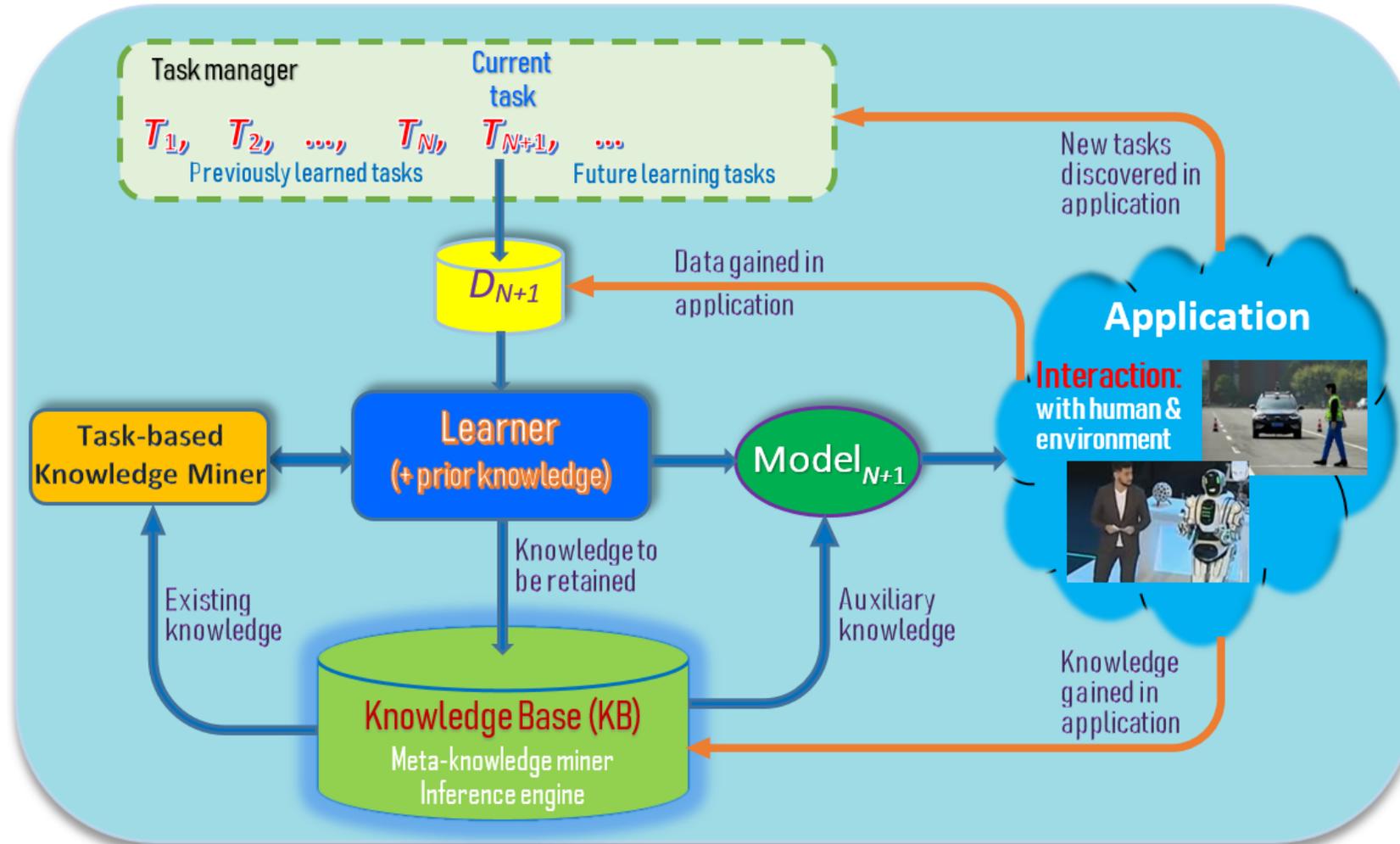
Classic definition of lifelong/continual learning

(Thrun 1996, Silver et al 2013; Ruvolo and Eaton, 2013; Chen and Liu, 2014, 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 with learning after deployment

(Fei et al 2016, Shu et al 2017a, 2017, Chen & Liu, 2018, Liu, 2020)



Orange lines:
Learning after model deployment
- Learning on the job

Liu. Learning on the Job: Online Lifelong and Continual Learning. AAI-2020

Characteristics of continual learning

(Chen and Liu, 2018, Liu, 2020)

- **Continuous incremental learning process** (no forgetting)
- **Knowledge accumulation in KB** (long-term memory)
- **Knowledge transfer/adaptation** (across tasks) (Ke, Liu, Huang, 2020)
- **Learning after deployment (on the job)**. *Self-supervision* using the *accumulated knowledge* and *interaction* with **humans & environment**.

Main steps:

- **Identify new tasks to learn** (tasks not given)
- **Acquire ground-truth training data** (training data not given)
- **Learn the tasks incrementally** (one-shot or few-shot)

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 continually and adapt
 - Without this capability, an AI agent is not truly intelligent.

(1) Chen and Liu Lifelong machine learning, 2015, 2018. (2) Liu. Learning on the Job: Online Lifelong and Continual Learning. AAIL-2020

Example 1 – a chatbot system

(Liu and Mazumder, 2021)

■ Session 1

- **User-1:** Hey, I visited Stockholm last week. The place is awesome!
- **Chatbot:** Where is **Stockholm**?
- **User-1:** Stockholm is the capital of Sweden.

■ Session 2

- **User-2:** I am planning a tour to Europe next month.
- **Chatbot:** Are you visiting **Stockholm**? I heard it is an awesome place.

Example 2 - 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.** (creat 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)

Learning new knowledge during dialogue

- **Learning to ground language expressions** (Mazumder et al, 2020b)
 - Learning via multi-turn dialogues with the user
 - Learning via user demonstrations
- **Learning factual knowledge** (Mazumder et al, 2019, 2020a)
 - Extracting new knowledge from user utterances via dialogues (Liu and Mei, 2020)
 - Asking and inferring new facts when the bot cannot answer a user query.
 - Asking questions to learn about unknown entities and concepts (Ono et al. 2017).
- **Learning conversation skills** (Hancock et al. 2019; Shuster et al. 2020)
 - Learning user behaviors and preferences
 - Modeling situation-aware conversations
 - Learning the user's emotion and mood state

Outline

- Continual learning with learning after deployment
- **Continual learning to ground new language expressions**
- Continual learning of factual knowledge
- Dealing with wrong knowledge from users
- Summary

Learning to ground 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.
- Existing approaches to building such chatbots:
 - Train an end-to-end deep learning model.
 - Semantic parsing -> logical forms -> translated to executable actions
- We discuss CML (Command Matching and Learning)

Natural language to natural language matching

- An **application-independent approach** to building task-oriented chatbots with interactive continual learning.
 - Based on *natural language to natural language* matching (**NL2NL**)
 - CML to automatically build NLIs for any API-driven applications.
- **To build a new NLI** (or add a new skill to an existing NLI),
 - the application developer only needs to **write a set S_i of seed commands** (SCs) in NL to represent each action i .
 - **SCs in S_i are just like paraphrased NL commands from users to invoke i** , but the objects to be acted upon in each SC are replaced with variables, the arguments of i .
 - An interactive learning mechanism to enable CML to **continually learn new (paraphrased) SCs from users**.

An example

- **Microsoft Paint tool:** The API action
drawCircle(X1, X2)
 - drawing a circle having color X1 at coordinate X2.
- Let a SC be “draw a X1 circle at X2” for this API,
 - where X1 and X2 are variables representing the arguments of the API.
- User command: “draw a blue circle at (20, 40)”
 - It can be matched or grounded to this SC, where the grounded API arguments are X1 = ‘blue’ and X2 = (20, 40).

CML has three components

- **SC (seed command) specification**
 - to enable the 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 to learn new SCs and paraphrases of API argument values.

SC Specification (blocks-world)

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)

■ 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, **Matcher 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.)

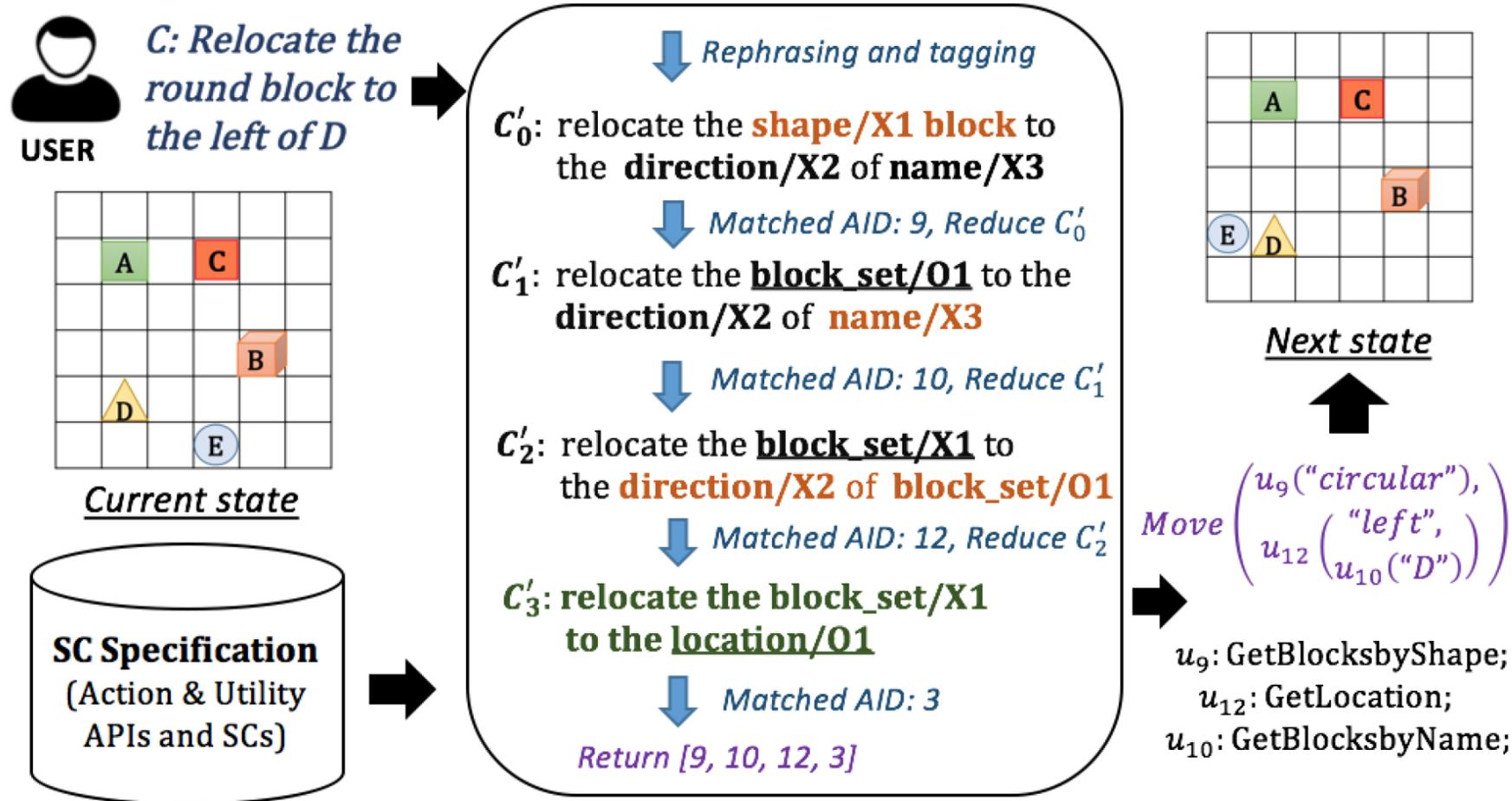


Figure 1: Working of CGM on a user command for Blocks-World. AID denotes the API IDs (see Tables 2 and 3).

Continual interactive learning

- If CML does not understand a user command C .
- CML learns a new SC from the user command C through interactive dialogue
- It also learn new paraphrased argument values from in C to improve repheaser R over time.

Algorithm 2 Interactive Knowledge Learning

Input: C' : Reduced user command by Algorithm 1; \mathcal{T} : action and utility SC Store; \mathcal{Q} : Question Template Store; \mathcal{M} : SC Matcher;

```
1:  $r_1 \leftarrow \text{Verify\_Pred\_SC}(\mathcal{Q}, C')$  { $r_i$  is user's response}
2: if  $r_1 = \text{"no"}$  then
3:    $r_2 \leftarrow \text{ShowSC\_List}(C_{rnk})$  { $C_{rnk}$  is the action SC rank list returned by  $\mathcal{M}$ }
4: end if
5: for all variable  $x_i$  in  $r_2$  do
6:    $r_{expr} \leftarrow \text{Ask\_Ref\_Expr}(x_i, C)$ 
7:    $r_{prop} \leftarrow \text{Ask\_Ref\_Prop}(r_{expr})$ 
8:    $r_{val} \leftarrow \text{Choose\_Prop\_Val}(r_{prop}, r_{expr})$ 
9:    $r_{para} \leftarrow \text{Ask\_Para\_Expr}(r_{val}, r_{expr})$ 
10:  Update  $\mathcal{R}$  with all  $(r_{val}, r_{para})$  pairs
11: end for
12: Rephrase  $C$  to get a new SC and update  $\mathcal{T}$ 
```

Experiment results

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	68.93	79.35	74.04	83.27	86.22	97.35
CML-vsm	68.93	79.35	74.46	85.81	88.08	97.35
CML-embed	68.63	79.94	68.93	82.83	83.43	97.35
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	81.34	76.17	84.43	92.05	97.35
CML-vsm + SCL	70.11	80.09	77.02	85.49	93.32	97.35
CML-jac + SCL + APL	72.78	81.43	80.0	85.31	93.15	98.41
CML-vsm + SCL + APL	73.07	80.16	80.85	86.37	94.67	98.41

Outline

- Continual learning with learning after deployment
- Continual learning to ground new language expressions
- **Continual learning of factual knowledge**
- Dealing with wrong knowledge from users
- Summary

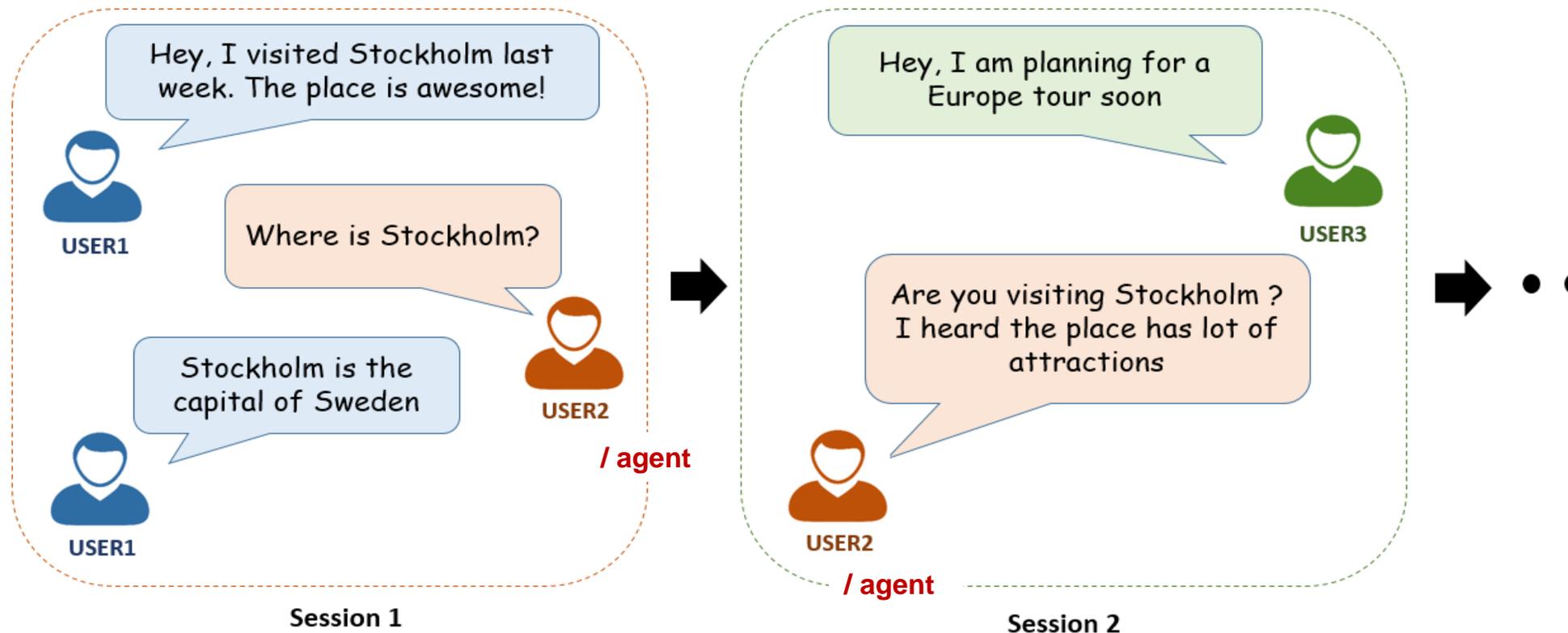
Continuous knowledge learning in dialogues

(Mazumder et al. 2019)

- 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*** for dialogue systems
 - to continuously and interactively learn and infer new knowledge during conversations

Knowledge learning in conversation

Humans Learn and Leverage Knowledge in Lifelong Manner!



Knowledge learning happens in a multi-user environment

Opportunities to learn in conversations

1. Extracting knowledge directly from user utterances. E.g.,
 - **User:** Obama was born in Hawaii.
 - **Agent extracts:** (Obama, BornIn, Hawaii) – expressed in triples **(h, r, t)**
2. Asking user questions & expecting correct answers, e.g.,
 - **Agent:** Where was Obama born?
 - **User:** Hawaii => (Obama, BornIn, Hawaii)
3. **When the agent cannot answer user questions**, it asks the user for some supporting facts and then infers the answers.
 - **We focus on this setting** (which covers 1 and 2)

Two types of queries or questions

■ 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

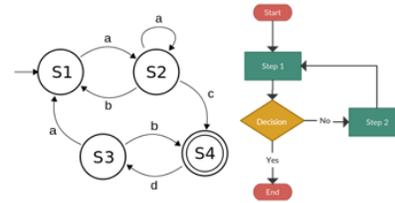
OR

Boston

LocatedInCountry

USA

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

- **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** as opposed to the teacher-student setup - the teacher is assumed to know everything.

(1) When the user asks a Wh-question

- Given a user query / question **(h, r, ?)** [or **(?, r, t)**], our 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**

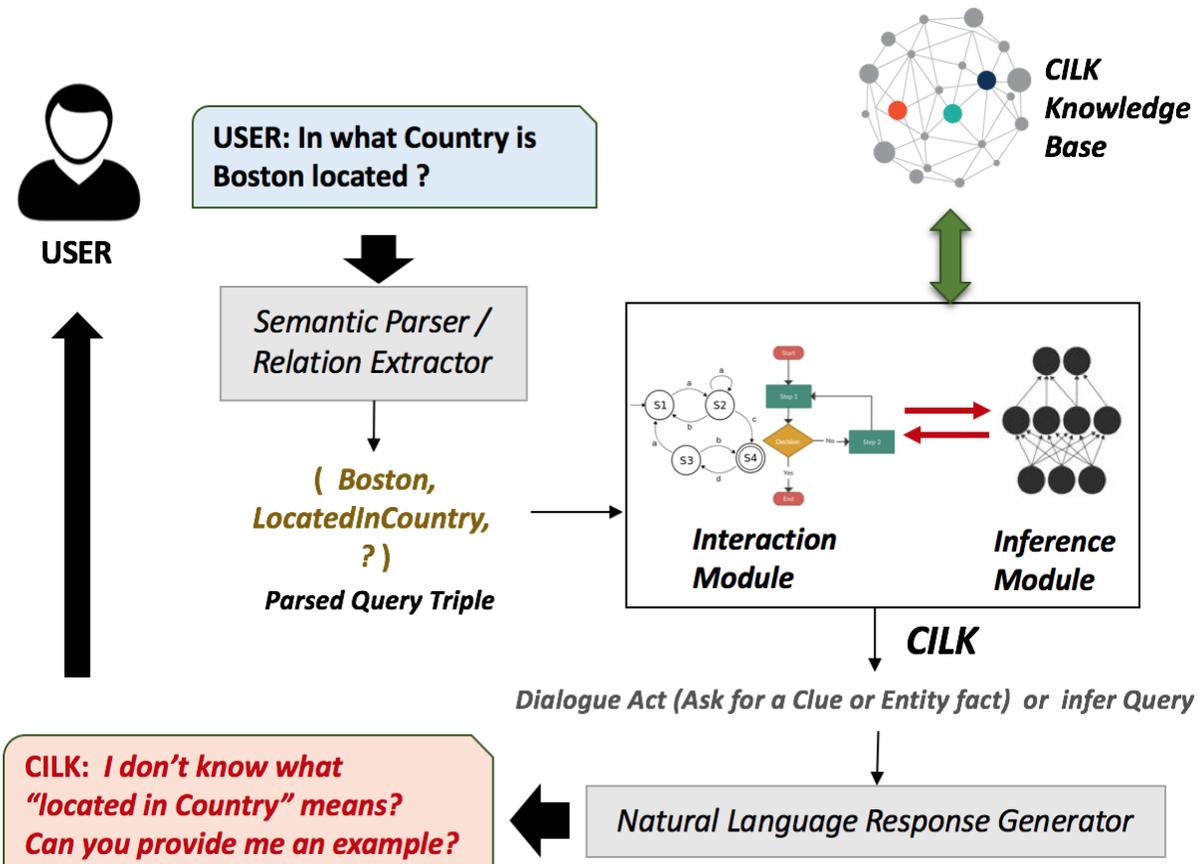


Proposed Soln.

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]

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

- **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]

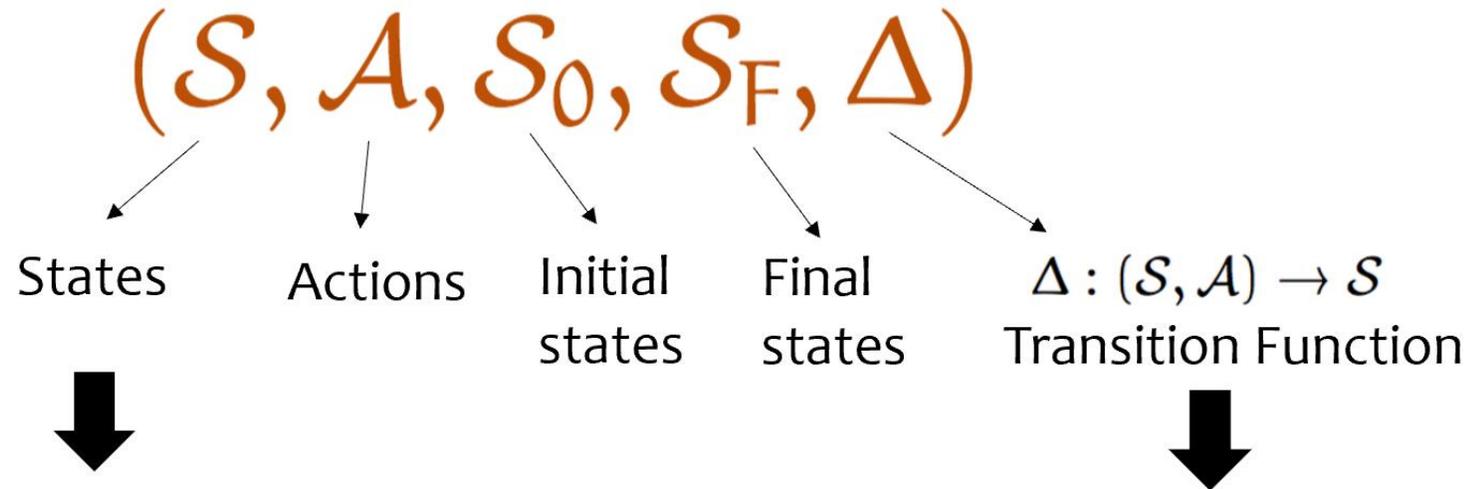
USER: (*Honolulu, CapitalOfState, Hawaii*). “Honolulu is the state capital of Hawaii.” [SF3]

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

The IKAI approach

- 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.

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

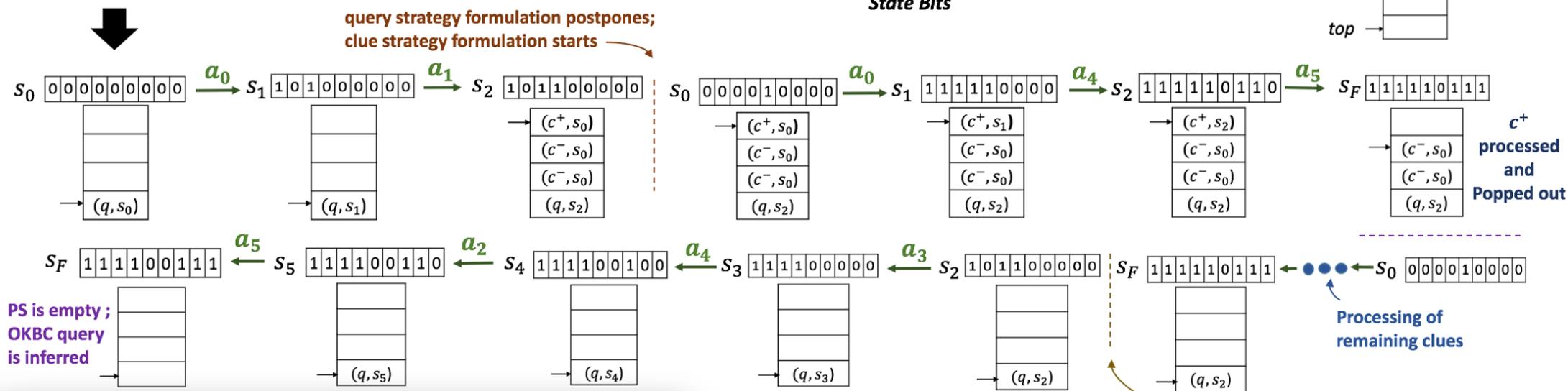


SB	Name	Description
QERS	Query entities and relation searched	Whether the query source (s) and target (t) entities and query relation (r) have been searched in KB.
SEF	Source Entity Found	Whether the source entity (s) has been found in KB.
TEF	Target Entity Found	Whether the target entity (t) has been 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
$QERS = 0$	a_0 : Search source (s), target (t) entities and query relation (r) in KB.
$ILO = 0 \wedge CLUE = 0 \wedge QERS = 1 \wedge QRF = 0$	a_1 : Ask user to provide an clue/example for query relation r.
$PFE = 1 \wedge ILO = 0 \wedge CPF = 0$	a_2 : Ask user to provide a missing link for path feature completion.
$QERS = 1 \wedge (SEF = 0 \vee TEF = 0) \wedge ILO = 0$	a_3 : Ask user to provide a connecting link to add a new entity to the KB.
$QERS = 1 \wedge PFE = 0 \wedge SEF = 1 \wedge TFE = 1$	a_4 : Extract path features between source (s) and target (t) entities using C-PR.
$QRF = 1 \wedge CPF = 1$	a_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.

(3) 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

Liu and Mei. Lifelong Knowledge Learning in Rule-based Dialogue Systems. arXiv:2011.09811, 2020.

Outline

- Continual learning with learning after deployment
- Continual learning to ground new language expressions
- Continual learning of factual knowledge
- **Dealing with wrong knowledge from users**
- Summary

Dealing with wrong knowledge from users

- Learn from end-users comes with **a major challenge**:
 - Acquiring intentional or unintentional wrong knowledge from users.
- Since chatbots normally **work in multi-user environments**
 - The issue may be addressed through **cross-verification**.
 - 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**

Outline

- Continual learning with learning after deployment
- Continual learning to ground new language expressions
- Continual learning of factual knowledge
- Dealing with wrong knowledge from users
- **Summary**

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 (Chen and Liu, 2018; Liu, 2020)
 - The agent becomes starter and smarter
- **Current techniques are still in their infancy**, but
 - **Some methods are ready for practical applications.**

Thank You

Q&A
