Continual and Interactive 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/novel in application
  - **No lifelong/continuous learning**:
    - learning each task separately in isolation
    - No knowledge accumulation or transfer
  - **No learning after deployment**: model fixed after deployment
Self-driving cars need to learn continuously too

- 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.

- Have to learn & adapt continuously in its interaction with humans and the environment by itself.
  - in the open world (changes & unknowns).

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

- **Definition of lifelong/continual learning**

- **Continual Learning**
  - Class continual learning
  - Task continual learning

- **Continuous learning after model deployment**
  - Continual learning to ground new language expressions
  - Continual learning of factual knowledge

- **Summary**
Classic definition of lifelong/continual learning

- Learn a sequence of tasks, $T_1$, $T_2$, ..., $T_N$, ... 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. **without catastrophic forgetting:** Learning of the new task $T_{N+1}$ should not result in degradation of accuracy for any of previous $N$ tasks.
  2. **with knowledge transfer:** leveraging the knowledge learned from the previous tasks to learn the new task $T_{N+1}$ better.

- **Assumption:** Once a task is learned its data is no longer accessible, at least a majority of it, and both the task $T_{N+1}$ and its training data $D_{N+1}$ are **given** by the user, and
Continual learning with learning after deployment


Orange lines:
Learning after model deployment
- Learning on the job
Characteristics of continual learning
(Chen and Liu, 2018, Liu, 2020)

- **Continuous and incremental learning process** (no forgetting)
- **Knowledge accumulation in KB** (long-term memory)
- **Knowledge transfer/adaptation** (across tasks)

**Learning after deployment (on the job).** *Self-supervision* using the accumulated knowledge and interaction with humans & environment.

**Main steps:**

- Identify new tasks to learn (open-world learning or OOD detection)
- Acquire ground-truth training data (collecting training data interactively)
- Learn the tasks incrementally (continual learning)
Closed-world assumption and open-world learning

(Fei et al, 2016; Shu et al., 2017, Hu et al., 2020)

- **Traditional machine learning:**
  - Training data: $D^{train} = \{D_1, D_2, \ldots, D_t\}$ of classes $Y^{train} = \{l_1, l_2, \ldots, l_t\}$.
  - Test data: $D^{test}, Y^{test} \in \{l_1, l_2, \ldots, l_t\}$

- **Closed-world assumption:** $Y^{test} \subseteq Y^{train}$
  - 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 learning:** $Y^{test} - Y^{train} \neq \emptyset$
  - Training data: $D^{train} = \{D_1, D_2, \ldots, D_t\}$, $Y^{train} = \{l_1, l_2, \ldots, l_t\}$.
  - Test data: $D^{test}, Y^{test} \in \{l_1, l_2, \ldots, l_t, L_0\}$
Learning on the job (after model deployment)
(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.
Example 1 - 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 - recognize he/she is new
- Bot: “Welcome to our hotel! What is your name, sir?”
- Guest: “David”
- Bot learns to recognize David automatically
  - take pictures of David
  - learn to recognize David

See David next time.
- Bot: “Hello David, how are you today?”
Example 2 – a chatbot system
(Liu and Mei, 2020; 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 a nice place.
Outline

- Definition of lifelong/continual learning
- Continual Learning
  - Class continual learning
  - Task continual learning
- Continuous learning after model deployment
  - Continual learning to ground new language expressions
  - Continual learning of factual knowledge
- Summary
Continual learning (of a sequence of tasks)

- 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
Main cause of catastrophic forgetting (CF):
- discriminative features for an early task may not be discriminative for a new task
- to learn the new task well, features learned need to be updated, causing CF

**PCL (Per class Continual Learning)**
- Each task can have one or more classes
- PCL enables the learning algorithm to holistically consider the features of each class
- Proposed a new regularization - *holistic regularization (h-reg).*

\[
\mathcal{L} = \mathbb{E}_{x \sim \mathbb{P}_x^{C_i}} [-\log(S(f_{C_i}(x)))] + \lambda \cdot \mathbb{E}_{x \sim \mathbb{P}_x^{C_i}} \left\| \nabla_x f_{C_i}(x) \right\|_2^n
\]

(NLL) \hspace{1cm} (H-reg)
## PCL architecture and results

**Feature Extractor:**
- WRN for Cifar10; Bert for DBPedia

### Table: Architecture and Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>w/o PTF</th>
<th>EWC</th>
<th>LwF</th>
<th>IMM</th>
<th>PGMA</th>
<th>RPSnet</th>
<th>OWM</th>
<th>PCL-L2</th>
<th>PCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST (10 tasks)</td>
<td>no</td>
<td>9.91</td>
<td>19.96</td>
<td>29.16</td>
<td>71.36</td>
<td>40.29</td>
<td>94.46</td>
<td>83.85</td>
<td>97.00</td>
</tr>
<tr>
<td>EMNIST-47 (47 tasks)</td>
<td>no</td>
<td>2.13</td>
<td>4.59</td>
<td>18.69</td>
<td>10.13</td>
<td>10.08</td>
<td>77.45</td>
<td>51.38</td>
<td>80.05</td>
</tr>
<tr>
<td>CIFAR10 (10 tasks)</td>
<td>yes</td>
<td>10.21</td>
<td>19.39</td>
<td>51.22</td>
<td>56.22</td>
<td>55.54</td>
<td>83.03</td>
<td>77.95</td>
<td>84.93</td>
</tr>
<tr>
<td>CIFAR100 (100 tasks)</td>
<td>yes</td>
<td>2.93</td>
<td>6.25</td>
<td>12.58</td>
<td>12.37</td>
<td>4.13</td>
<td>63.26</td>
<td>54.83</td>
<td>63.61</td>
</tr>
<tr>
<td>20news (20 tasks)</td>
<td>no</td>
<td>4.98</td>
<td>5.61</td>
<td>5.00</td>
<td>11.37</td>
<td>8.32</td>
<td>52.02</td>
<td>49.01</td>
<td>54.37</td>
</tr>
<tr>
<td>DBPedia (14 tasks)</td>
<td>yes</td>
<td>7.14</td>
<td>7.14</td>
<td>7.14</td>
<td>66.40</td>
<td>50.58</td>
<td>95.37</td>
<td>68.12</td>
<td>96.23</td>
</tr>
<tr>
<td>CIFAR10 (10 tasks)</td>
<td>no</td>
<td>10.01</td>
<td>10.05</td>
<td>10.25</td>
<td>20.08</td>
<td>16.31</td>
<td>19.63</td>
<td>10.00</td>
<td>31.58</td>
</tr>
<tr>
<td>CIFAR100 (100 tasks)</td>
<td>no</td>
<td>1.03</td>
<td>2.13</td>
<td>1.21</td>
<td>1.86</td>
<td>1.96</td>
<td>3.67</td>
<td>1.87</td>
<td>5.58</td>
</tr>
<tr>
<td>DBPedia (14 tasks)</td>
<td>no</td>
<td>7.14</td>
<td>7.14</td>
<td>7.14</td>
<td>9.58</td>
<td>36.70</td>
<td>92.23</td>
<td>64.96</td>
<td>93.51</td>
</tr>
</tbody>
</table>
Task continual learning
(e.g., Ke, Liu and Huang, 2020)

- Existing research mainly focused on overcoming CF.
- Some works also addressed *knowledge transfer*
  - Leveraged the past knowledge to help learn the new task when tasks are similar and have shared knowledge.
- **Our goal:** Achieve both *Forgetting Avoidance* and *Knowledge Transfer* at the same time
- **CAT** (*Continual learning with forgetting Avoidance and knowledge Transfer*)
  - learn a mixed sequence of similar and dissimilar tasks
  - achieve both objectives.
Main idea of CAT

- For dissimilar tasks, **Task Masks** are trained to block all used/important neurons of the tasks.
- Encourage **knowledge transfer** among **similar** tasks by training attention.
  - **Forward** knowledge transfer (past knowledge helps new task)
  - **Backward** knowledge transfer (new data helps improve old tasks)
- Automatically detect task similarity

\[
\mathbb{E}_{(\mathcal{X}(t), \mathcal{Y}(t))} [\mathcal{L}(f_{k\rightarrow t}(\mathcal{X}(t); \theta_{k\rightarrow t}), \mathcal{Y}(t))] > \mathbb{E}_{(\mathcal{X}(t), \mathcal{Y}(t))} [\mathcal{L}(f_{\emptyset}(\mathcal{X}(t); \theta_{\emptyset}), \mathcal{Y}(t))]
\]
Model and results

In all 4 mixed datasets, CAT outperforms all baselines.
Outline

- Definition of lifelong/continual learning
- **Continual Learning**
  - Class continual learning
  - Task continual learning
- **Continuous learning after model deployment**
  - Continual learning to ground new language expressions
  - Continual learning of factual knowledge
- Summary
Learning to ground NL commands on the job
(Mazumder et al., 2020)

- **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 automatically builds 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 $X_1$ at coordinate $X_2$.

- Let a SC be "draw a $X_1$ circle at $X_2$" for this API,
  - where $X_1$ and $X_2$ 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 $X_1 = \text{‘blue’}$ and $X_2 = (20, 40)$. 
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.
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).

<table>
<thead>
<tr>
<th>Action API Function</th>
<th>AID</th>
<th>Action SCs (’;’ separated)</th>
<th>Variable: Argument Type</th>
<th>Example commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddBlock (X1)</td>
<td>1</td>
<td>add a block at X1; insert a block at X1</td>
<td>X1: ‘location’ (*)</td>
<td>add a block at (2, 3); put a block at (2, 3)</td>
</tr>
<tr>
<td>Remove (X1)</td>
<td>2</td>
<td>remove X1</td>
<td>X1: ‘block_set’</td>
<td>delete blue block; take away blue block</td>
</tr>
<tr>
<td>Move (X1, X2)</td>
<td>3</td>
<td>move X1 to X2; shift X1 to X2</td>
<td>X1: ‘block_set’, X2: ‘location’ (*)</td>
<td>move blue block to the left of cube; shift green cube to (4, 5)</td>
</tr>
<tr>
<td>MoveByUnits (X1, X2, X3)</td>
<td>4</td>
<td>move X1 along X2 by X3 units</td>
<td>X1: ‘block_set’, X2: ‘direction’, X3: ‘number’</td>
<td>move blue block left by 2 units; shift green cube down by 3 units</td>
</tr>
<tr>
<td>UpdateColor (X1, X2)</td>
<td>5</td>
<td>change color of X1 to X2; color X1 with X2</td>
<td>X1: ‘block_set’, X2: ‘color’ (*)</td>
<td>color A red; change color of B to blue</td>
</tr>
<tr>
<td>UpdateShape (X1, X2)</td>
<td>6</td>
<td>change shape of X1 to X2</td>
<td>X1: ‘block_set’, X2: ‘shape’ (*)</td>
<td>set the shape of A to cube; make B square</td>
</tr>
<tr>
<td>Rename (X1, X2)</td>
<td>7</td>
<td>rename block X1 to X2</td>
<td>X1: ‘block_set’, X2: ‘name’ (*)</td>
<td>Name the block at (4, 5) as C; rename A to D</td>
</tr>
</tbody>
</table>
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 $M$.
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 commend C.
- CML learns a new SC from the user command C through interactive dialogue
- It also learns 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; $T$: action and utility SC Store; $Q$: Question Template Store; $M$: SC Matcher;

1: $r_1 \leftarrow \text{Verify_Pred_SC}(Q, C')\{r_i$ is user’s response$\}$
2: if $r_1$ = “no” then
3: $r_2 \leftarrow \text{ShowSC_List}(C_{rnk})\{C_{rnk}$ is the action SC rank list returned by $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 $R$ with all $(r_{val}, r_{para})$ pairs
11: end for
12: Rephrase $C'$ to get a new SC and update $T$
```
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.

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

Continual Learning
- Class continual learning
- Task continual learning

Continuous learning after model deployment
- Continual learning to ground new language expressions
- Continual learning of factual knowledge

Summary
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

- to continuously and interactively learn and infer new knowledge during conversations.
Knowledge learning in conversation

Humans Learn and Leverage Knowledge in a Lifelong Manner!

Hey, I visited Stockholm last week. The place is awesome!

Where is Stockholm?

Stockholm is the capital of Sweden

Hey, I am planning for a Europe tour soon

Are you visiting Stockholm? I heard the place has lot of attractions

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

- Stores acquired Facts (Triples)
  - KB: Collection of Triples
  - $\mathcal{T} = \{ (h, r, t) \mid h, t \in E, r \in R \}$

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

- Infers new Knowledge to answer user’s query
  - Performs inference over the acquired Facts and existing KB


Mazumder, Liu, Wang, and Ma. Lifelong and Interactive Learning of Factual Knowledge in Dialogues. SIGDIAL-2019
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 (cross-verification can be used to deal with wrong knowledge)

- **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**

\[\downarrow\] Proposed Soln.

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

Mazumder, Liu, Wang, and Ma. Lifelong and Interactive Learning of Factual Knowledge in Dialogues. SIGDIAL-2019
Interactive knowledge learning in dialogue: example
(Mazumder et al. 2019)

USER: In what Country is Boston located?

CILK: I do not know what “located in Country” means? Can you provide me an example?

USER: (London, LocatedInCountry, UK). “London is located in UK.”

CILK: Got it. Can you tell me a fact about “Boston”?

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

CILK: (Boston, LocatedInCountry, USA) “Boston is located in USA.”

Mazumder, Liu, Wang, and Ma. Lifelong and Interactive Learning of Factual Knowledge in Dialogues. SIGDIAL-2019
(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?”

**IKAI:** I do not know what “CitizenOf” means? Can you provide me an example?

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

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

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

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

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

**IKAI:** (Obama, CitizenOf?, USA) holds. “Yes, Obama is a US citizen.”
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 ($\mathcal{I}$)

\[ (S, A, S_0, S_F, \Delta) \]

- **States**
- **Actions**
- **Initial states**
- **Final states**
- **Transition Function** $\Delta: (S, A) \rightarrow S$

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

**State Transition Conditions** (for current state bits $S_i$)

<table>
<thead>
<tr>
<th>QERS</th>
<th>ILO = 0 ∧ CLUE = 0 ∧ QERS = 1 ∧ QRF = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFE</td>
<td>1 ∧ ILO = 0 ∧ CPF = 0</td>
</tr>
<tr>
<td>QERS</td>
<td>1 ∧ (SEF = 0 ∨ TEF = 0) ∧ ILO = 0</td>
</tr>
<tr>
<td>QERS</td>
<td>1 ∧ PFE = 0 ∧ SEF = 1 ∧ TFE = 1</td>
</tr>
<tr>
<td>QRF</td>
<td>1 ∧ CPF = 1</td>
</tr>
</tbody>
</table>

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

Working of IKAI’ FSM in a given Session


@Allen Institute for AI (AI2), June 18, 2021
Outline

- Definition of lifelong/continual learning
- Continual Learning
  - Class continual learning
  - Task continual learning
- Continuous learning after model deployment
  - Continual learning to ground new language expressions
  - Continual learning of factual knowledge
- Summary
Classic ML: isolated, closed-world, single-task learning

AI agents must learn continuously in the open world on the job or after deployment (Liu, 2020, Mazumder et al. 2020)

- Discover new tasks: open-world novelty (OOD) detection
- Obtain ground-truth data: via self-supervised interactions with humans and the environment.
- Continual learning: learning the new task incrementally and online

Current techniques are still in their infancy.

- All tasks are very challenging.
- Continual learning is still not ready for applications
Thank You

Q&A