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-Discussion-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).

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

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

Social chats

Question-answering

Task completion

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.

---

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)].

$$\hat{A}_i = \arg\max_{A_i \in A} P(A_i | (A_1, U_1, ..., A_{i-1}, U_{i-1})$$

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 XioIce

Social Chatbots: Rule-based systems
(Jurafsky et.al. 2020)

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

  (Ø YOU Ø ME) [pattern]

  ->

  (WHAT MAKES YOU THINK I 3 YOU) [transform]

  You hate me
  into:

  WHAT MAKES YOU THINK I HATE YOU

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

Example dialogue by ELIZA

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
..
..
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 $C$ (training set for the system)

- Score each turn in $C$ as a potential response to the context $q$ and select the highest-scoring one.

$$\text{response}(q, C) = \arg\max_{r \in C} \frac{q \cdot r}{|q||r|}$$
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).

\[ \hat{r}_t = \arg\max_{w \in V} P(w|q, r_1...r_{t-1}) \]
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.

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!

- West et. al. Knowledge base completion via search-based question answering. WWW-2014
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

- Learn a sequence of tasks, $T_1, T_2, \ldots, T_N, \ldots$ incrementally. Each task $t$ has a training dataset $D_t = \{x_{t,i}, y_{t,i}\}^{n_t}_{i=1}$

- **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: \( D^{\text{train}} \) with class labels \( Y^{\text{train}} = \{l_1, l_2, ..., l_t\} \).
  - Test data: \( D^{\text{test}}, Y^{\text{test}} \subseteq \{l_1, l_2, ..., l_t\} \)

- Closed-world: \( Y^{\text{test}} \subseteq Y^{\text{train}} \)
  - 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: \( Y^{\text{test}} - Y^{\text{train}} \neq \emptyset \)
  - 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 $D^{novel}$, $L_0 = \{l_{t+1}, l_{t+2}, \ldots\}$ and gather training data.
  - **Interactive self-supervision:** interaction with humans and the environment.

- **Continual learning:** Incrementally learn the new classes $\{l_{t+1}, l_{t+2}, \ldots\}$ (the new task).

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

---

Fei, Wang, and Liu. Learning Cumulatively to Become More Knowledgeable. KDD-2016

Shu, Hu and Liu. DOC: Deep Open Classification of Text Documents. EMNLP 2017
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.
Formulate a dynamic interaction strategy ($I$) (deciding what to ask and when to ask user).

Execute strategy ($I$) to acquire and store training data / knowledge.

Incrementally learn Task $T_{N+1}$ to improve skill (similar to LL)
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?”
  - Guest: “David”
  - Bot learns to recognize David automatically
    - take pictures of David
    - learn to recognize David (get training data) (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
Humans Learn and Leverage Knowledge in 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

Knowledge learning in chatbots: Opportunities
(Liu and Mazumder 2021)

1. Extracting facts from user utterances

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).

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

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).

3. Ask and infer new facts in conversation

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

Ask and infer new facts in conversation

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).

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).

✓ When the chatbot cannot answer an user query, it can ask for some related supporting facts and then infer the answer.
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)*

---

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

**Mazumder, Liu, Ma and Wang, Continuous and Interactive Factual Knowledge Learning in Verification Dialogues. HAMLETS - NeurIPS-2020 Workshop, 2020.**
Components for knowledge learning

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

- **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

- **Triple Store**
  - (Boston, LocatedInCountry, USA)
  - head: Boston
  - relation: LocatedInCountry
  - tail: USA

- Mazumder, Liu, Wang, and Ma. Lifelong and Interactive Learning of Factual Knowledge in Dialogues. SIGDIAL-2019
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 \((h, r, ?) \text{ [or (?)}, \text{ r, t}]\), 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**: \(h \text{ (or t) and r are known to the KB}\)

(2) **Open-world Queries**: Either one or both \(h \text{ (or t) and r are unknown}\)

Proposed Soln.

an engine for **Continuous and Interactive Learning of Knowledge (CILK)**
Interactive knowledge learning in dialogue: example
(Mazumder et al. 2019)

USER: In what Country is Boston located?

Semantic Parser / Relation Extractor

(Country, LocatedInCountry, ?)

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].

\[
\begin{align*}
\text{Triple: } (h, r, t) & & \text{Entity Embed.} & \begin{cases} 
v_h = W_E x_h \\
v_t = W_E x_t \\
v_r = W_R x_r 
\end{cases} & \text{Relation Embed.} \\
\text{DisMult Scoring Function} & S(h, r, t) = v_h^T \text{diag}(v_r) v_t \\
& = \sum_{i=1}^{N} v_h[i] v_r[i] v_t[i]
\end{align*}
\]

\[
\mathcal{L} = \sum_{d \in D^+} \sum_{d' \in D^-} \max\{S(d') - S(d) + 1, 0\}
\]

- Yang, Yih, He, Gao, Deng. *Embedding entities and relations for learning and inference in knowledge bases*. ICLR 2014
CILK maintains a **threshold buffer** $\mathcal{T}$ at stores entity and relation specific prediction thresholds and updates it continuously over time.

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

**mean scores of triples involving +ve entities**

$$\mu^+_E + \mu^-_E$$

**mean scores of triples involving -ve entities**

$$\mu^-_E = \frac{1}{|E^-|} \sum_{e^-_i \in E^-} S(q, e^-_i)$$

$$\mu^+_E = \frac{1}{|E^+|} \sum_{e^+_i \in E^+} S(q, e^+_i)$$

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

For head query triple $S(e^-_i, r, e)$

---

Mazumder, Liu, Wang, and Ma. Lifelong and Interactive Learning of Factual Knowledge in Dialogues. SIGDIAL-2019
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 $\tau[e], \tau[r]$

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

Mazumder, Liu, Wang, and Ma. Lifelong and Interactive Learning of Factual Knowledge in Dialogues. SIGDIAL-2019
Acquiring Knowledge with limited Interaction: Improving Skillset over time

A **performance buffer** \( \mathcal{P} \) is used to store the performance statistics of \( M \):

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

At the end of each dialogue session, CILK **detects bottom** \( \rho \% \) **query relations** and **entities** in **based on** \( \mathcal{P} \text{MRR scores} \),

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

<table>
<thead>
<tr>
<th>(#C, #EF)</th>
<th>WordNet</th>
<th>Nell</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRR</td>
<td>H@1</td>
</tr>
<tr>
<td>(1, 1)</td>
<td>0.30</td>
<td>22.09</td>
</tr>
<tr>
<td>(1, 2)</td>
<td>0.32</td>
<td>23.00</td>
</tr>
<tr>
<td>(1, 3)</td>
<td>0.33</td>
<td>25.27</td>
</tr>
<tr>
<td>(1, 3)-U</td>
<td>0.31</td>
<td>23.52</td>
</tr>
<tr>
<td>(2, 2)</td>
<td>0.32</td>
<td>23.43</td>
</tr>
</tbody>
</table>

- 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 (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 -

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

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. *Compositional vector space models for knowledge base completion*. In ACL, 2015.
Finite State Machine: Interaction Module ($\mathcal{I}$)

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

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

<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 $S_i$ [\])

- $QERS = 0$
- $ILO = 0 \land CLUE = 0 \land QERS = 1 \land QRF = 0$
- $PFE = 1 \land ILO = 0 \land CPF = 0$
- $QERS = 1 \land (SEF = 0 \lor TEF = 0) \land ILO = 0$
- $QERS = 1 \land PFE = 0 \land SEF = 1 \land TFE = 1$
- $QRF = 1 \land CPF = 1$

Action Id : Operation

- $a_0$ : 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.

Working of IKAI’ FSM in a given Session

**OKBC query** : \( q = (Obama, CitizenOf, USA) \)

**Processing Stack (PS)**

**Processing of remaining clues**

**Query strategy formulation resumes**

**State Bits**

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

**Query**

**IAKI**: 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**

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

**CLQ**

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

**SF2**

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

**MLQ**

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

**SF3**

**IAKI**: (Obama, CitizenOf, USA) holds. “Yes, Obama is a US citizen.”

**Answer**

- \( \alpha_0 \) : Search source (s), target (t) entities and query relation (r) in KB.
- \( \alpha_1 \) : Ask user to provide a connecting link to add a new entity to the KB.
- \( \alpha_2 \) : Ask user to provide an clue/example for query relation r.
- \( \alpha_3 \) : Ask user to provide a missing link for path feature completion.
- \( \alpha_4 \) : Extract path features between source (s) and target (t) entities using C-PR.
- \( \alpha_5 \) : Invoke prediction model for inference.
IKAI - Performance Evaluation

IKAI achieves best performance overall

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Models</th>
<th>Rel - K / Ent-K (F1(+))</th>
<th>Rel - K / Ent-UNK (F1(+))</th>
<th>Rel - UNK / Ent - K (F1(+))</th>
<th>Rel - UNK / Ent - UNK (F1(+))</th>
<th>Overall (F1(+))</th>
<th>Overall (Macro-F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freebase</td>
<td>BG</td>
<td>0.584</td>
<td>0.629</td>
<td>0.494</td>
<td>0.569</td>
<td>0.432</td>
<td>0.532</td>
</tr>
<tr>
<td></td>
<td>w/o PTL</td>
<td>0.555</td>
<td>0.652</td>
<td>0.533</td>
<td>0.620</td>
<td>0.528</td>
<td>0.419</td>
</tr>
<tr>
<td></td>
<td>IKAI</td>
<td>0.587</td>
<td>0.671</td>
<td>0.493</td>
<td>0.591</td>
<td>0.525</td>
<td>0.616</td>
</tr>
<tr>
<td>WordNet</td>
<td>BG</td>
<td>0.548</td>
<td>0.466</td>
<td>0.532</td>
<td>0.525</td>
<td>0.486</td>
<td>0.476</td>
</tr>
<tr>
<td></td>
<td>w/o PTL</td>
<td>0.666</td>
<td>0.741</td>
<td>0.561</td>
<td>0.624</td>
<td>0.461</td>
<td>0.281</td>
</tr>
<tr>
<td></td>
<td>IKAI</td>
<td>0.655</td>
<td>0.694</td>
<td>0.552</td>
<td>0.604</td>
<td>0.612</td>
<td>0.659</td>
</tr>
</tbody>
</table>

+ve F1 score
Macro F1 score

Path completion via Blind Guessing
Continuous learning past tasks (relations) is disabled

IKAI Performance improvement due to User Interaction

<table>
<thead>
<tr>
<th>% TTO</th>
<th>Freebase</th>
<th>WordNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>0.0</td>
<td>0.492</td>
</tr>
<tr>
<td>100%</td>
<td>0.545</td>
<td>0.580</td>
</tr>
</tbody>
</table>

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.
Determines a question type on the basis of CM.

- The CM is estimated for each cuisine type $c_j$ in the target database.

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

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

<table>
<thead>
<tr>
<th>num</th>
<th>Question form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes-No question</td>
<td>Is it cuisine $c_1$?</td>
</tr>
<tr>
<td>2</td>
<td>Alternative question</td>
<td>Which cuisine is it, $c_1$ or $c_2$?</td>
</tr>
<tr>
<td>3</td>
<td>3-choice question</td>
<td>Which cuisine is it, $c_1$, $c_2$, or $c_3$?</td>
</tr>
<tr>
<td>$\geq 4$</td>
<td>Wh-question</td>
<td>What cuisine is it?</td>
</tr>
</tbody>
</table>
**CM_D Calculation:** using Word and Character Distribution in Database (Otsuka et. al. 2013)

\[ CM_D(s_i, c_j) = \frac{1}{Z} \exp \left[ \Lambda(c_j) \cdot \Phi(s_i) \right] \]

- **cuisine type**
- **restaurant name**
- feature vector (n-gram, words etc.) obtained from a restaurant name

**Input:**
Restaurant name

**Output:**
CM_D

- Japanese restaurant: 0.9
- Japanese pub: 0.05
- Cafe: 0.0006

**DB**

<table>
<thead>
<tr>
<th>Restaurant name</th>
<th>Cuisine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maru Sushi (まる寿司)</td>
<td>Japanese restaurant</td>
</tr>
<tr>
<td>Tanaka Sushi (田中寿司)</td>
<td>Japanese restaurant</td>
</tr>
<tr>
<td>Koikoi (こいこい)</td>
<td>Japanese pub</td>
</tr>
<tr>
<td>Sushi Goichi (寿司 五一)</td>
<td>Japanese restaurant</td>
</tr>
<tr>
<td>Quinci CENTRARE</td>
<td>Italian</td>
</tr>
<tr>
<td>Hyakuraku (百楽)</td>
<td>Chinese restaurant</td>
</tr>
<tr>
<td>C's ave cafe</td>
<td>Cafe</td>
</tr>
</tbody>
</table>

**Classifier:**
Maximum entropy (ME) model

**Training data**
CM_W Calculation: Using the Web
(Otsuka et. al. 2013)

\[
P_{\text{freq}}(c_j) = \frac{\sum_i w_i \cdot f_{\text{freq}}(c_j)}{\sum_{c_j} \sum_i w_i \cdot f_{\text{freq}}(c_j)} \quad w_i = \frac{1}{\text{rank}(i) \cdot \text{cuisine}(i)}
\]

\[
CM_W(c_j) = \frac{\alpha_j P_{\text{freq}}(c_j)}{\sum_{c_j} \alpha_j P_{\text{freq}}(c_j)} \quad \alpha_j = \frac{P_{\text{freq}}(c_j)}{P_{\text{freq}}(c_1)}
\]
Lexical acquisition in dialogues: Performance (Otsuka et. al. 2013)

<table>
<thead>
<tr>
<th>CM range</th>
<th>CMD Correct</th>
<th>CMD Incorrect</th>
<th>CMW Correct</th>
<th>CMW Incorrect</th>
<th>CMI Correct</th>
<th>CMI Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0 – 0.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>32</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>0.1 – 0.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>9</td>
<td>15</td>
</tr>
<tr>
<td>0.2 – 0.3</td>
<td>1</td>
<td>16</td>
<td>14</td>
<td>22</td>
<td>15</td>
<td>18</td>
</tr>
<tr>
<td>0.3 – 0.4</td>
<td>6</td>
<td>19</td>
<td>28</td>
<td>19</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>0.4 – 0.5</td>
<td>11</td>
<td>25</td>
<td>29</td>
<td>21</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>0.5 – 0.6</td>
<td>21</td>
<td>29</td>
<td>56</td>
<td>9</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>0.6 – 0.7</td>
<td>22</td>
<td>28</td>
<td>85</td>
<td>7</td>
<td>15</td>
<td>7</td>
</tr>
<tr>
<td>0.7 – 0.8</td>
<td>41</td>
<td>16</td>
<td>42</td>
<td>3</td>
<td>17</td>
<td>6</td>
</tr>
<tr>
<td>0.8 – 0.9</td>
<td>21</td>
<td>9</td>
<td>19</td>
<td>1</td>
<td>19</td>
<td>9</td>
</tr>
<tr>
<td>0.9 – 1.0</td>
<td>131</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>184</td>
<td>10</td>
</tr>
<tr>
<td>Total</td>
<td>254</td>
<td>146</td>
<td>274</td>
<td>124</td>
<td>297</td>
<td>103</td>
</tr>
</tbody>
</table>
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 SHRDLURN 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 SHRDLURN game
Compositional action space for SHRDLURN
(Wang et. al. 2016)

<table>
<thead>
<tr>
<th>Rule</th>
<th>Semantics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set</td>
<td>all()</td>
<td>all stacks</td>
</tr>
<tr>
<td>Color</td>
<td>cyan</td>
<td>brown</td>
</tr>
<tr>
<td>Color → Set</td>
<td>with(c)</td>
<td>stacks whose top block has color $c$</td>
</tr>
<tr>
<td>Set → Set</td>
<td>not(s)</td>
<td>all stacks except those in $s$</td>
</tr>
<tr>
<td>Set → Set</td>
<td>leftmost</td>
<td>rightmost(s)</td>
</tr>
<tr>
<td>Set Color → Act</td>
<td>add(s,c)</td>
<td>add block with color $c$ on each stack in $s$</td>
</tr>
<tr>
<td>Set → Act</td>
<td>remove(s)</td>
<td>remove the topmost block of each stack in $s$</td>
</tr>
</tbody>
</table>

remove rightmost orange block
→ 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 \( x \):
  \[
  p_\theta(z \mid x) \propto \exp(\theta^T \phi(x, z))
  \]
  
  ✓ 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.
### Evaluation on 100 players on Mechanical Turk

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

**Most successful players (1st–20th)**

- 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 (3.01)
- 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 (2.78)
- 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 (2.72)

**Average players (21th–50th)**

- 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 (9.17)
- remove red, remove 1 red, remove 2 4 orange, add 2 red, add 1 2 3 4 blue, emove 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 (8.37)
- 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 (7.18)

**Least successful players (51th–)**

- holdleftmost, holdbrown, holdleftmost, blueonblue, brownonblue1, blueonorange, holdblue, holdorange2, blueonred2, holdends1, holdrightend, hold2, orangeonorangerrightmost (14.15)
- ‘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 (12.6)
- 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 (14.32)
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

<table>
<thead>
<tr>
<th>Method</th>
<th>players ranked by # of scrolls</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>top 10</td>
</tr>
<tr>
<td>memorize</td>
<td>25.4</td>
</tr>
<tr>
<td>half model</td>
<td>38.7</td>
</tr>
<tr>
<td>half + prag</td>
<td>43.7</td>
</tr>
<tr>
<td>full model</td>
<td>48.6</td>
</tr>
<tr>
<td>full + prag</td>
<td>52.8</td>
</tr>
</tbody>
</table>

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

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

Mazumder, Liu, Wang, Esmaeilpour. An Application-Independent Approach to Building Task-Oriented
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).

<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)  
(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, 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 (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.

<table>
<thead>
<tr>
<th>Models</th>
<th>BW</th>
<th>WPD</th>
<th>FB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A-acc</td>
<td>Arg-F1</td>
<td>A-acc</td>
</tr>
<tr>
<td>BERT-JISF</td>
<td>49.70</td>
<td>59.23</td>
<td>57.87</td>
</tr>
<tr>
<td>CML-jac</td>
<td>68.93</td>
<td>79.35</td>
<td>74.04</td>
</tr>
<tr>
<td>CML-vsm</td>
<td>68.93</td>
<td>79.35</td>
<td>74.46</td>
</tr>
<tr>
<td>CML-embed</td>
<td>68.63</td>
<td>79.94</td>
<td>68.93</td>
</tr>
<tr>
<td>CML-vsm (-R)</td>
<td>64.79</td>
<td>77.42</td>
<td>68.08</td>
</tr>
<tr>
<td>CML-vsm (-U)</td>
<td>14.49</td>
<td>14.49</td>
<td>11.48</td>
</tr>
<tr>
<td>CML-jac + SCL</td>
<td>69.82</td>
<td>81.34</td>
<td>76.17</td>
</tr>
<tr>
<td>CML-vsm + SCL</td>
<td>70.11</td>
<td>80.09</td>
<td>77.02</td>
</tr>
<tr>
<td>CML-jac + SCL + APL</td>
<td>72.78</td>
<td>81.43</td>
<td>80.00</td>
</tr>
<tr>
<td>CML-vsm + SCL + APL</td>
<td>73.07</td>
<td>80.16</td>
<td>80.85</td>
</tr>
</tbody>
</table>
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.

---

Hancock, Bordes, Mazare, and Weston. Learning from Dialogue after Deployment: Feed Yourself, Chatbot!. ACL 2019.

### 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

---

Hancock, Bordes, Mazare, and Weston. Learning from Dialogue after Deployment: Feed Yourself, Chatbot!. ACL 2019.
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?

Hancock, Bordes, Mazare, and Weston. Learning from Dialogue after Deployment: Feed Yourself, Chatbot!. ACL 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)

Hancock, Bordes, Mazare, and Weston. Learning from Dialogue after Deployment: Feed Yourself, Chatbot!. ACL 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.

---

Hancock, Bordes, Mazare, and Weston. Learning from Dialogue after Deployment: Feed Yourself, Chatbot!. ACL 2019.
# Self-feeding Chatbot: Performance
(Hancock et. al. 2019)

<table>
<thead>
<tr>
<th>Human-Bot (HB)</th>
<th>Human-Human (HH) DIALOGUE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DIALOGUE</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>20k</td>
<td>-</td>
</tr>
<tr>
<td>40k</td>
<td>-</td>
</tr>
<tr>
<td>60k</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>20k</td>
</tr>
<tr>
<td>-</td>
<td>40k</td>
</tr>
<tr>
<td>-</td>
<td>60k</td>
</tr>
<tr>
<td>60k</td>
<td>60k</td>
</tr>
</tbody>
</table>

**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)

<table>
<thead>
<tr>
<th>Human (grounds keeper)</th>
<th>Model (assistant chef)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Location:</strong> 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...</td>
<td></td>
</tr>
<tr>
<td><strong>Persona:</strong> 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.</td>
<td></td>
</tr>
<tr>
<td><strong>grounds keeper:</strong> Good morning sir</td>
<td></td>
</tr>
<tr>
<td><strong>assistant chef:</strong> Hello, keeper. What may I call you?</td>
<td></td>
</tr>
<tr>
<td><strong>grounds keeper:</strong> My name is Skylar</td>
<td></td>
</tr>
<tr>
<td><strong>assistant chef:</strong> Excellent choice. I have a new sauce recipe from bredwell across the sea. Would you like to try it?</td>
<td></td>
</tr>
</tbody>
</table>

| **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)

<table>
<thead>
<tr>
<th>Model</th>
<th>LIGHT Test</th>
<th>LIGHT Test Unseen</th>
<th>WILD Test</th>
<th>LIGHT Test</th>
<th>LIGHT Test Unseen</th>
<th>WILD Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round 1</td>
<td>87.12</td>
<td>82.43</td>
<td>81.61</td>
<td>12.67</td>
<td>11.81</td>
<td>13.42</td>
</tr>
<tr>
<td>Round 2</td>
<td>87.65</td>
<td>82.70</td>
<td>84.60</td>
<td>12.57</td>
<td>11.74</td>
<td>12.31</td>
</tr>
<tr>
<td>Round 3</td>
<td>87.72</td>
<td>83.48</td>
<td>87.63</td>
<td>12.54</td>
<td>11.75</td>
<td>11.79</td>
</tr>
</tbody>
</table>

- **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

test locations do not overlap with the training set locations
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
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)

**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 $M^t$

\[
\sum_{k=0}^{t} P^k(i) I^k(i)
\]

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

Aggregate (slot label) prob. for each word from all components.
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.

<table>
<thead>
<tr>
<th>Approach</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttRNN (upper bound)</td>
<td>92.12</td>
<td>92.89</td>
<td>93.04</td>
<td>93.56</td>
<td>95.13</td>
</tr>
<tr>
<td>FT-AttRNN</td>
<td>91.85</td>
<td>89.98</td>
<td>91.25</td>
<td>88.03</td>
<td></td>
</tr>
<tr>
<td>FT-Lr-AttRNN</td>
<td>91.96</td>
<td>86.46</td>
<td>88.03</td>
<td>86.58</td>
<td></td>
</tr>
<tr>
<td>FT-Cp-AttRNN</td>
<td>92.12</td>
<td>90.06</td>
<td>91.98</td>
<td>89.67</td>
<td></td>
</tr>
<tr>
<td>t-ProgModel</td>
<td>92.33</td>
<td>92.43</td>
<td>92.57</td>
<td>92.58</td>
<td></td>
</tr>
<tr>
<td>c-ProgModel</td>
<td>92.40</td>
<td>92.64</td>
<td>92.71</td>
<td>93.91</td>
<td></td>
</tr>
</tbody>
</table>
## ProgModel Performance: Snips Dataset
(Shen et. al. 2019)

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

  \[
  H \rightarrow I \quad \text{(INTENT)}
  
  \begin{align*}
  H \rightarrow I(s_1 = v_1, \ldots, s_k = v_p) \quad \text{(DST)}
  
  I(s_1 = v_1, \ldots, s_k = v_p) \rightarrow S \quad \text{(NLG)}
  
  S_{OUT}
  
  \end{align*}

- End-to-End (E2E) formulation:

  \[
  H \rightarrow I(s_1 = v_1, \ldots, s_k = v_p)
  
  H + I(s_1 = v_1, \ldots, s_k = v_p) \rightarrow S
  
  S_{API}
  
  S_{OUT}
  
  \]

- Dialogue Dataset: input-out pair

  the four settings in consideration

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

USER: I need to check my balance.
E2E \rightarrow SYS: Of course! Which account should I use?
USER: My savings account, please.
E2E \rightarrow API: CheckBalance(account_type="savings")
\hspace{10em}
\begin{align*}
\text{INTENT} & \\
\text{DST} & \\
\text{NLG} & \\
\end{align*}
OUT: OFFER(balance="$139")
E2E \rightarrow SYS: No problem. Your balance is $139.
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:
  \[ \text{Adapter}_{\mu_t}(H) = \text{ReLU}(\text{LN}(x)W_l^E)W_l^D + H \]
  \[ H \in \mathbb{R}^{p \times d} \]

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

- 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.

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

$$t^* = \arg\min \alpha_0, \cdots, \alpha_N$$
Continual Learning in Task-Oriented Dialogue Systems: Performance (Madotto et. al. 2020)

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

<table>
<thead>
<tr>
<th>Method</th>
<th>+Param.</th>
<th>Mem.</th>
<th><strong>INTENT</strong></th>
<th><strong>DST</strong></th>
<th><strong>NLG</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>( \text{Accuracy}^\uparrow )</td>
<td>( \text{JGA}^\uparrow )</td>
<td>( \text{EER}^\downarrow )</td>
</tr>
<tr>
<td>VANNILLA</td>
<td>-</td>
<td>( \emptyset )</td>
<td>4.08 ( \pm ) 1.4</td>
<td>4.91 ( \pm ) 4.46</td>
<td>48.73 ( \pm ) 3.81</td>
</tr>
<tr>
<td>L2</td>
<td>(</td>
<td>\theta</td>
<td>)</td>
<td>( \emptyset )</td>
<td>3.74 ( \pm ) 1.4</td>
</tr>
<tr>
<td>EWC</td>
<td>( 2</td>
<td>\theta</td>
<td>)</td>
<td>( \emptyset )</td>
<td>3.95 ( \pm ) 1.3</td>
</tr>
<tr>
<td>AGEM</td>
<td>-</td>
<td>( t</td>
<td>M</td>
<td>)</td>
<td>34.04 ( \pm ) 6.36</td>
</tr>
<tr>
<td>LAMOL</td>
<td>-</td>
<td>( \emptyset )</td>
<td>7.49 ( \pm ) 6.35</td>
<td>4.55 ( \pm ) 3.48</td>
<td>66.11 ( \pm ) 6.97</td>
</tr>
<tr>
<td>REPLAY</td>
<td>-</td>
<td>( t</td>
<td>M</td>
<td>)</td>
<td>81.08 ( \pm ) 1.37</td>
</tr>
<tr>
<td>ADAPT</td>
<td>( t</td>
<td>\mu</td>
<td>)</td>
<td>( \emptyset )</td>
<td>90.46 ( \pm ) 0.6</td>
</tr>
<tr>
<td>MULTI</td>
<td>-</td>
<td>-</td>
<td>95.45 ( \pm ) 0.1</td>
<td>48.9 ( \pm ) 0.2</td>
<td>12.56 ( \pm ) 0.2</td>
</tr>
</tbody>
</table>

E2E results in terms 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 needed 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 provie you with options shortly.
Searching in knowledge base...See results in Figure 1b
Would you consider this option: The_Place
Let's do it.

---

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).

<table>
<thead>
<tr>
<th>User: Worst day ever. I arrived late because of the traffic.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Seq2Seq: You were late.</td>
</tr>
<tr>
<td>ECM (Like): I am always here to support you.</td>
</tr>
<tr>
<td>ECM (Happy): Keep smiling! Things will get better.</td>
</tr>
<tr>
<td>ECM (Sad): It’s depressing.</td>
</tr>
<tr>
<td>ECM (Disgust): Sometimes life just sucks.</td>
</tr>
<tr>
<td>ECM (Angry): The traffic is too bad!</td>
</tr>
</tbody>
</table>
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 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
- Chen, Z.; and Liu, B. 2014. Topic modeling using topics from many domains, lifelong learning and big data. In ICML.
Thank You

Q&A