# Interactive Teaching for Conversational AI

Qing Ping[†], Feiyang Niu[†], Govind Thattai[†], Joel Chengottusseriyil, Qiaozi Gao, Aishwarya Reganti,
Prashanth Rajagopal, Gokhan Tur, Dilek Hakkani-Tur, and Prem Natarajan

Alexa AI, Amazon
{pingqing, nfeiyan, thattg, jchengot, qzgao, areganti, rajagpra, gokhatur,
hakkanit, premknat}@amazon.com

## Abstract

Current conversational AI systems aim to understand a set of pre-designed requests and execute related actions, which limits them to evolve naturally and adapt based on human interactions. Motivated by how children learn their first language interacting with adults, this paper describes a new Teachable AI system that is capable of learning new language nuggets called concepts, directly from end users using live interactive teaching sessions. The proposed setup uses three models to: a) Identify gaps in understanding automatically during live conversational interactions, b) Learn the respective interpretations of such unknown concepts from live interactions with users, and c) Manage a classroom sub-dialogue specifically tailored for interactive teaching sessions. We propose state-of-the-art transformer based neural architectures of models, fine-tuned on top of pre-trained models, and show accuracy improvements on the respective components. We demonstrate that this method is very promising in leading way to build more adaptive and personalized language understanding models.

## 1  Introduction

Humans are adaptive by nature. While it is very natural for humans to ask clarifying questions and immediately correct course whenever there is a misunderstanding, interactive learning using self-supervision has been a holy grail area for Conversational AI. Most Conversational AI systems rely on knowledge-based or machine-learning based (or hybrid) understanding components under the hood to understand the intents of the users' and their arguments (usually called as slots), such as "*set an alarm for 7 am*", where the intent can be *set_alarm*, and *7 am* can be the time slot. These systems typically do not have the capability to interact with end-users during a live conversation to seek explanations to improve the AI system's understanding of entities, intents or other constructs. When users speak an utterance outside the comprehension of these systems, the AI agent responds with a dead-end response such as 'Sorry I don't know that'.

In this paper we describe a teachable AI system towards enabling users to directly teach a conversational AI agent via a live interactive teaching session using natural language explanations. Learning concept definitions using interactive voice is a challenging problem due to a number of reasons, including wide vocabulary of words that users could use to describe definitions, user distractions during teaching sessions, grounding of related entities, fuzziness in the re-use of previously taught concepts across domains, etc. The method described in this paper augments goal-oriented AI agents with an interactive teaching capability, to improve task completion rates by asking questions to fill gaps in the AI agent's understanding, hence making the agent more personal, conversational and adaptive.

---

[†]These authors contributed equally to this work

More specifically, the AI agent will be checking the utterances whether it is "Teachable" using a deep learning based parser, which not only classifies the utterance, but also identifies the entity or intent which needs to be learnt from the users. If so, the "Classroom" sub-dialogue kicks in, where the users are asked what they mean by these concepts. This sub-dialogue is driven by a dedicated dialogue manager policy model which incorporates a definition understanding model to interpret users' responses. For example, if the utterance is "*set an alarm for my baseball practice*", the system would ask "*when is your baseball practice?*".

Despite a few academic papers towards learning directly from users as presented in Section 2, to the best of our knowledge, there is no work that covers all of these novel contributions:

- A dedicated multi-turn domain-agnostic dialogue system specifically tailored for interactive teaching, that augments an existing Conversational AI System to learning explanations from user in real-time,
- A multi-task neural Concept Parser that automatically identify gaps in an AI agent's understanding, using a multi-task model that incorporates semantic role labeling and constituency parser.
- A neural Definition Understanding system along with a policy model to aid in conducting robust teaching sessions with the user.

The following sections are organized as follows. Section 2 describes the related work in the area of interactive learning methods to learn explanations, and predicting gaps in the AI agent's understanding for a given user input; Section 3 describes our architecture and modeling methods, and Section 4 describes our experimental evaluations.

## 2    Related Work

The related work on learning via user-in-the-loop for Conversational AI systems can be analyzed in two categories. The higher level interactive learning approaches, and utterance parsing based gap prediction studies.

### 2.1    Interactive Learning

Existing work on interactive learning methods can be grouped into 3 categories:

- Game learning using limited/no vocabulary: Several methods have been proposed for learning a game (such as Hanoi) using interactive sessions with the user Kirk *et al.* [10] Wang *et al.*[20]. Such methods represent the game scene as a symbolic representation of attributes (such as shape, color) which is used to learn the state progression, to learn the game.

- Neural code synthesis methods: Methods like Yin *et al.* [25] aim at converting user utterances directly into a coding language like Python. To our knowledge, such syntax-driven code generation methods are still in their early stages, and not yet ready for integration into mainstream conversational AI agents that cover a wide variety of domains.

- Learning by GUI demonstrations: GUI based methods use semantic parsers to learn conditional expressions and their associated actions. One notable study is by Allen *et al.* using a web browser to teach tasks such as buying a book from Amazon [1].

### 2.2    NLU gap prediction

The problem of predicting segments of an utterance which a base NLU system could not interpret, was tackled by Jia *et al.* [8], by using a set of post-processing rules on top of the base NLU's slot-filling model. The rules were built on heuristics from the base NLU model, such as confidence score threshold, out-of-vocabulary flag, and span-expansion based on syntactic dependency tree. This paper also introduced a new dataset for this problem that is publicly available, which is used in our work as one of the datasets for evaluating model performances. In [11, 12] the authors proposed augmenting the training data for the slot-filling model, by injecting noise tokens into the regular slot values of training data, to force the model to learn about the context of the respective slots. This augmentation is done to improve the model robustness in predicting the correct span, on unseen slot values. One variant of such approach uses negative sampling to train the model to identify unknown slot values together with a joint slot tagging and slot classifier [7]. Another line of work focuses on different model structures such as pointer-networks, to better copy unknown slot values from the utterance
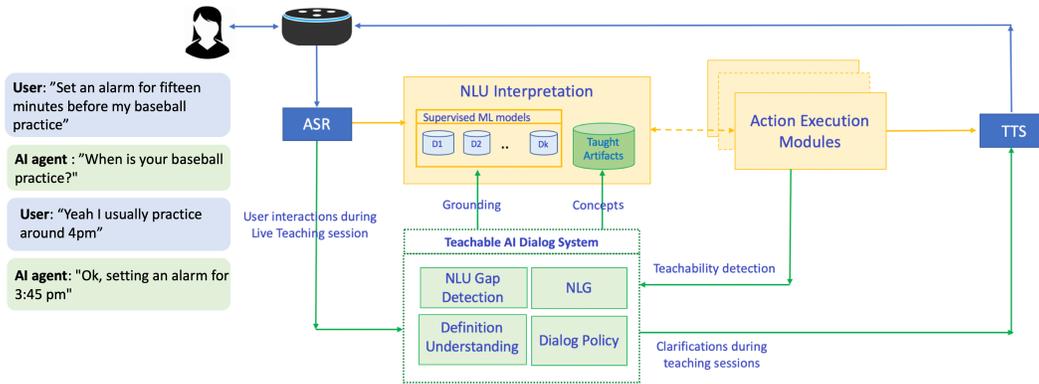
Figure 1: Architecture of AI system using a Teachable AI Dailog system

into final slot value prediction [22, 27, 24]. This line of work usually considers the problem under a dialogue state tracking setting, therefore only one slot value is predicted each time.

## 2.3 Definition Understanding

Our task of understanding the definition of concepts from users' explanations, generally falls into the categories of reading comprehension and slot filling. For reading comprehension, there is a particular thread that formulates question answering as span prediction. Those works typically learn a representation for the question and passage, and then predict the start and end of the answer with attention mechanism between question, passage and answer [3, 2]. The slot filling thread work focuses on sequential labeling of an answer[17], with most recent works utilizing neural slot-fillers with joint intent classifiers [14, 6, 23, 26, 4].

## 2.4 Dialog Policy for Teaching sessions

While there is relatively rare work for dialog policy models specifically meant for teaching sessions, a dialog policy in general can be learned using a) supervised policy learning, and (b) reinforcement policy learning. Existing supervised learning approaches usually considers it as an intent prediction problem. Recent work usually takes a joint prediction for both intents and slot prediction [14, 6, 23, 26, 4]. Reinforcement learning approaches formulate the policy learning as sequential decision making process, and learns optimal policy from a large number of user interactions [15, 21, 29, 28].

## 3 Methodology

In this work, we incorporate a dedicated dialogue system called Teachable Dialogue System, which is specifically tailored for conducting interactive teaching sessions with users, for learning definitions of concepts that were previously not understood by the AI system. This dialogue system consists of three components:

- Concept parser - that uses pre-trained embeddings and signals from the Conversational AI system to predict gaps in NLU's understanding for interpreting a given utterance from the user,
- Definition understanding model that extracts and interprets explanations from the user and maps the unknown concept phrases to the learnt definitions, and
- Dialog policy model that drives teaching conversations with a goal-completion objective of extracting necessary explanations from the user and ground the respective unknown concepts.

The teachable dialogue system acts as a subsidiary to the multi-domain Conversational AI system, and springs into action when the AI agent is not able to successfully interpret/execute a user's input utterance. This way, the parent Conversational AI system remains decoupled from the Teachable Dialogue System, staying independent of user-specific interpretation or modeling. Concept parser identifies the sequence of tokens (called concept phrases) that are not understood by the parent NLU system, and helps in initiating a teaching session with the user with the question - "*Can you teach me what you mean by <concept phrase>?*" Once a teaching session has been initiated with the user by the dialogue system, the dialog-policy model helps in predicting the right clarification question to
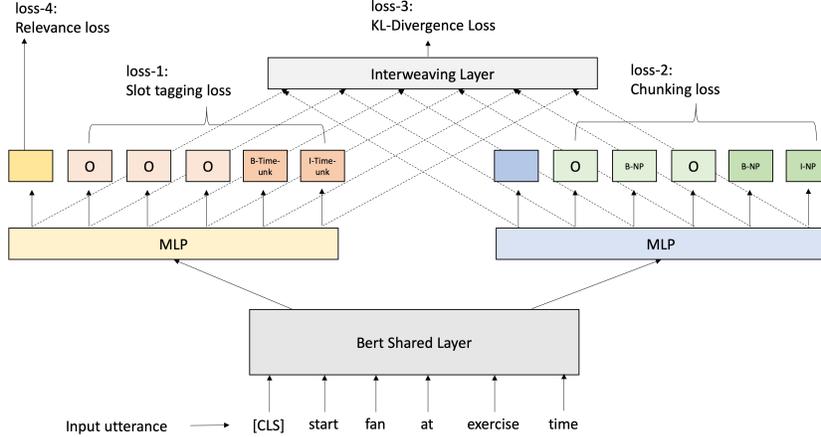
Figure 2: Concept Parser Model Structure: Transformer-based shared layer with Multi-Task Heads

ask the user, based on the context of the original utterance and the interactions with the user during the teaching session. The answers from the user are then processed by the definition understanding component that works along with the policy models to conduct multi-turn conversations with the user to learn, clarify and ground the respective concepts in the user's utterance. When the Teachable Dialogue System deems a teaching session as successful, the taught actions are subsequently executed by the AI system, and the respective definitions and artifacts are then stored for future re-use. Figure 1 describes the architecture of this Teachable AI system.

## 3.1 Concept Parser

Identifying gaps in NLU interpretation of an utterance, is a challenging task by itself. This task involves accurately localizing the segments within an utterance that an NLU system is unable to comprehend, and needed to execute the relevant action. In addition to using a slot-tagger objective in our Concept Parser model, we added an auxiliary task of semantic-chunking to make sure the model is sensitive to segments that might be neglected by a token-level slot tagger. For example, the chunking model should be able to segment the sentence: "[show] [me] [navigation] [to] [where we go camping every year]" while the slot tagger may not correctly tag "where we go camping every year" as "Location" concept.

Additionally, there are several real-world challenges that a Concept Parser needs to address, such as: a) Tentative user behavior - such as "set the lights to, never mind" - where the user intends to cancel the request, rather than mean to set the lights to a user defined value, b) Unsupported actions such as "set the light to fifty degrees" where NLU interpretation is successful, but the Action Execution module cannot execute the respective action - this could be because of an unintentional user error, or due to an error in Automatic Speech Recognition (ASR). Concept Parser needs to treat the above scenarios as 'non teachable' instances, and hence should prevent the initiation of teaching sessions to learn the respective definitions from users. In addition to the task of identifying the concept phrase from a given utterance, Concept Parser also generates 'relevance scores' that indicates how confidence score for the concept phrase to be teachable.

To address the above challenges, we implemented a Concept Parser using a multi-task transformer model which is end-to-end trainable using the following task objectives: a) Slot tagging b) Semantic chunking, and c) Concept Relevance. The model structure is depicted in Figure 2 , and is composed of the following sub-components:

**BERT layer**. It is a multi-layer multi-head self-attention Bert model [5]. The input sentence is first tokenized with WordPiece [19]. Then positional embedding and segment embedding are added to the token embedding as final input embedding $X = x_1, x_2, ..., x_L$. The final output of bert layer $H = h_1, h_2, ..., h_L$ is a contextual representation for each input token.

**Slot tagging head**. Given the bert output $H = h_1, h_2, ..., h_L$, the slot tagging head feeds the output into a multi-layer perceptron followed by a softmax to predict slot label for each token.

$$z_i = f(W \cdot h_i + b) \tag{1}$$

4

$$\hat{y}_{ij}(st) = \frac{e^{z_{ij}}}{\sum_{k=1}^{N_1} e^{z_{ik}}} \tag{2}$$

Where $N_1$ is the number of slot classes and $L$ is the length of the utterance. Then the slot tagging loss $L_{st}$ will be a average cross-entropy loss across all tokens.

$$L_{st} = -\frac{1}{L \cdot N_1} \sum_{i=1}^{L} \sum_{j=1}^{N_1} y_{ij}(st) \cdot log(\hat{y}_{ij}(st)) \tag{3}$$

**Chunking tagging head**. The chunking head has exact same structure as the slot tagging head. The only difference is the different $W$ and $b$ parameters, as well as the supervised chunking labels $y_i$. The loss $L_{ck}$ can be calculated as follows:

$$L_{ck} = -\frac{1}{L \cdot N_2} \sum_{i=1}^{L} \sum_{j=1}^{N_2} y_{ij}(ck) \cdot log(\hat{y}_{ij}(ck)) \tag{4}$$

Where Where $N_2$ is the number of chunking classes and $L$ is the length of the utterance. $y_{ij}(ck)$ is ground-truth chunking labels, and $\hat{y_{ij}}(ck)$ is the predicted chunking labels. We also add an interweaving loss to enforce the "synchronization" between the slot tagging head and chunking head, by imposing a KL-divergence loss on the output of the two heads.

$$L_{kl} = -\frac{1}{L \cdot N} \sum_{i=1}^{L} \sum_{j=1}^{N} \hat{z}_{ij}(ck) \cdot log(\frac{\hat{z}_{ij}(ck)}{\hat{z}_{ij}(st)}) \tag{5}$$

**Relevance scoring head**. The relevance scoring head takes the bert output embedding of the [CLS] token as the input, and feeds it into multi-layer perceptron followed by a binary cross-entropy loss:

$$z_{[CLS]} = f(W \cdot h_{[CLS]} + b) \tag{6}$$

$$\hat{y}_{[CLS],i} = \frac{e^{z_{[CLS],i}}}{\sum_{j=1}^{2} e^{z_{[CLS],j}}} \tag{7}$$

$$L_{rel} = -\frac{1}{2} \sum_{i=1}^{2} y_i(rel) \cdot log(\hat{y}_{[CLS],i}) \tag{8}$$

**Final loss objective**. The final loss objective $L_{CP}$ is a weighted sum of all four losses mentioned above, with $\alpha_1$, $\alpha_2$, $\alpha_3$ and $\alpha_4$ as the weights.

$$L_{CP} = \alpha_1 \cdot L_{st} + \alpha_2 \cdot L_{ck} + \alpha_3 \cdot L_{kl} + (1 - \alpha_1 - \alpha_2 - \alpha_3) \cdot L_{rel} \tag{9}$$

## 3.2 Definition Understanding

Once Concept Parser has detected a teachable phrase, a teaching session is initiated by requesting the user to provide a suitable definition of the identified concept. There are several real-world challenges involved in accurately extracting the appropriate definition from the user's answer, such as: a) Verbose answers: When posed with an impromptu definition of a concept, users generally tend to use verbose and indirect answers such as "yeah i mean red color or may be just orange would do" or "i meant red not blue"; b) Distracted users: It is quite likely that users do not intend to answer the definition question, but would like to move on with another new request to the AI assistant, e.g. "Whats the weather outside", which is not really an answer to the question; c) Incomplete answers: Users might have provided an answer but its not yet sufficient to fully ground the respective definition and needs more clarification questions; d) Complicated answers: Users could provide a definition that sounds logical to another human, but the respective definition could still be not be understood by the parent AI system and hence cannot be grounded to the equivalent actions; e) Contextual references: such as "make it brighter".

We developed a Definition Understanding model that can generically address the above scenarios. The model uses a Transformer-based joint intent and span detection framework that takes in the dialogue history ($\boldsymbol{z} = z_1, z_2, \ldots, z_H$), user answer ($\boldsymbol{x} = x_1, x_2, \ldots, x_A$) and the slot type ($\boldsymbol{m} = m_1, m_2, \ldots, m_S$), illustrated in Figure 3. We first concatenate user answer and dialogue history and pass the concatenated vector through a Transform-based encoder (e.g. BERT [5]) and extract the encoder's last layer hidden states, denoted as $\boldsymbol{H} =$
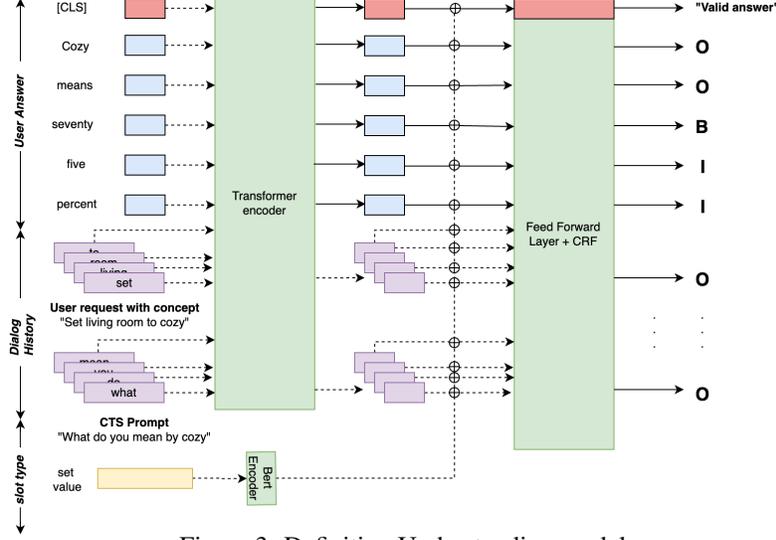
Figure 3: Definition Understanding model

$(\boldsymbol{h}_{[CLS]}, \boldsymbol{h}_{x,1}, \ldots, \boldsymbol{h}_{x,A}, \boldsymbol{h}_{[SEP]}, \boldsymbol{h}_{z,1}, \boldsymbol{h}_{z,2}, \ldots, \boldsymbol{h}_{z,H}, \boldsymbol{h}_{[SEP]})$. Separately, we obtain a slot type embedding, $\boldsymbol{e}_s$ with some pre-trained model (e.g. BERT). The contextual representation of each token is then fused with the slot type embedding

$$\boldsymbol{H} \oplus \boldsymbol{e}_s = (\boldsymbol{h}_{[CLS]} \oplus \boldsymbol{e}_s, \ldots, \boldsymbol{h}_{x,i} \oplus \boldsymbol{e}_s, \ldots \boldsymbol{h}_{z,j} \oplus \boldsymbol{e}_s, \ldots, \boldsymbol{h}_{[SEP]} \oplus \boldsymbol{e}_s) \tag{10}$$

and further gets passed through a set of post-Transformer encoder layers, e.g. feed-forward and CRF layers, to produce the final representation of each input token, $\boldsymbol{H}^O = (\boldsymbol{h}^O_{[CLS]}, \ldots, \boldsymbol{h}^O_{x,i}, \ldots \boldsymbol{h}^O_{z,j}, \ldots, \boldsymbol{h}^O_{[SEP]})$. Intent and span classification are performed with separate output layers and described below in details.

**Intent classification**. We adopt a single fully connected layer followed by a softmax layer on top of the final representation of [CLS] token to perform the intent classification.

$$\hat{y}_i^{intent} = Softmax(f(\boldsymbol{W}^{intent} \cdot \boldsymbol{h}^O_{[CLS]} + \boldsymbol{b}^{intent})) \tag{11}$$

We use cross-entropy to calculate the intent classification loss for $C$ intents.

$$L_{intent} = -\frac{1}{C} \sum_{i=1}^{C} y_i^{intent} \cdot \log(\hat{y}_i^{intent}) \tag{12}$$

**Span classification**. We use the same layer structure to perform span classification as intent classification except that the fully connected layer is 3-way (i.e. "B", "I", "O").

$$\hat{y}_{i,k}^{span} = Softmax(f(\boldsymbol{W}_i^{intent} \cdot \boldsymbol{h}^O_{x,i} + \boldsymbol{b}_i^{span})) \tag{13}$$

The span classification loss is also calculated using cross-entropy

$$L_{span} = -\frac{1}{3A} \sum_{i=1}^{A} \sum_{k=1}^{3} y_{i,k}^{span} \cdot \log(\hat{y}_{i,k}^{span}) - \frac{1}{3H} \sum_{j=1}^{H} \sum_{k=1}^{3} y_{j,k}^{span} \cdot \log(\hat{y}_{j,k}^{span}) \tag{14}$$

**Joint loss**. The final loss objective $L_{DU}$ is a weighted sum of intent and span classification losses. The hyperparameter of relative intent loss proportion $\alpha_{intent}$ is furthered tuned on a validation dataset.

$$L_{DU} = \alpha_{intent} \cdot L_{intent} + (1 - \alpha_{intent}) \cdot L_{span} \tag{15}$$

### 3.3 Dialogue Policy

For the Teachable Dialogue system described in this paper, a Dialogue Policy predicts the next action in a multi-turn teaching dialogue session with a goal-completion target of extracting all the necessary definitions from the user during the teaching. A successful completion of the teaching session is one where the concept phrase in the first-turn utterance has been grounded to an equivalent action that

is executable by the parent AI system. The Dialogue Policy uses a Transformer-based contextual model to predict the next action during a teaching session. The action-space for our Dialogue Policy model includes a) Ask or repeat a clarification question, b) Guardrail the conversations to channelize users back into the teaching conversations, c) Identify Out-Of-Domain (OOD) turns during a teaching session, d) Ground the extracted definitions with the parent NLU system, and e) deem a teaching session as successful or unsuccessful, or decide to end a teaching session. The dialogue policy model takes in utterances from contextual dialogue history, predicted definition spans and confidence scores from Definition Understanding module, and the set of recognized/resolved slots from the parent NLU model and passes the input through a Transformer-based encoder to get the last layer output of $\texttt{[CLS]}$ token as a contextual representation of the inputs, $\boldsymbol{h}_{[CLS]}$. Then we apply a single feed-forward layer and a softmax layer to obtain a probability distribution over all the possible actions.

$$\hat{y}^{action} = Softmax(f(\boldsymbol{W}^{action} \cdot \boldsymbol{h}_{[CLS]} + \boldsymbol{b}^{action})) \tag{16}$$

## 4 Experiments and Results

**Datasets:** We evaluate our results on two datasets: 1) The dataset published by [8] which consists of both personal concepts and generic slot values for five slot types namely date, time, location, people and restaurant-name. While the dataset in [8] pertains to the problem that we have attempted to solve in this paper, we find that it does not include challenging real-world scenarios such as the ones described in Section 3 which are addressed by our models. We hence used a second internal dataset collected from crowd-sourcing which addressed the challenges of realistic multi-turn teaching sessions. Crowd workers provided both cooperative and non-cooperative inputs for two tasks:

- Generate first-turn utterances containing both personal and non-personal concept phrases for performing actions supported by the AI assistant spanning multiple domains. Each utterance was annotated with the ground-truth concept phrase. For the "not teachable" class, we synthesize a wide variety of cases, including utterances without any concept, out-of-domain utterance, ill-grammar and incomplete utterances.
- Answers to clarification questions within teaching sessions. Answers were annotated with the concept definition phrases along with additional information like direct-answer, new-request, etc.

**Concept parser experiment setting**: For concept parser model, a pre-trained BERT-base model is used as the base layer, and it is fine-tuned together with all the head layers during training [5]. We used a 300 dimensional hidden-layer in MLP, a RELU activation function [18], and a learning rate of 1e-5. For the public dataset, the model is simultaneously trained on $L_{st}$, $L_{ck}$ and $L_{kl}$ losses without relevance scoring for 20 epochs, where weights are set to 0.5/0.5/2.0 empirically. For our internal dataset, the model is first trained on the $L_{ck}$ loss for 2 epochs, and then fine-tuned on $L_{st}$ and $L_{rel}$ losses with 1.0/1.0 weights for 2 epochs. For the auxiliary chunking label, we use a fusion of semantic role labeling model(SRL) and constituency parsing model (CPM) models from AllenNLP [2].

**Definition understanding experiment setting**: We experimented with a few pre-trained transformer-based models, namely BERT [5], RoBERTa [16], ALBERT [13] as the base encoder. We used a single feed-forward layer with hidden state set to 100 and RELU [18] activation function followed by a CRF layer for post-Transformer encoder. The initial learning rate is set to $1 \times 10^{-4}$ and reduces to half at loss plateau, i.e. no loss reduction within last consecutive 10 epochs. We used Adam optimizer [9] with weight decay enabled.

### 4.1 Performance evaluations

#### 4.1.1 Concept Parser results

**Public Datasets**: Table 1 shows the performance of our concept parser on personalized test set from [8] using phrase-level metrics on slots with unknown concepts. From Table 1 we observe the following: a) When there is no personalized training data available (Zero-shot), there is an absolute improvement in F1-score of 8.6% for single-task model, and 15% using Multi-task model. This supports our idea that an end-to-end model with built-in objectives for both slot tagging and semantic-chunking can be more robust in this zero-shot unknown concept prediction setting. b) When personalized concept phrases are added to the training data, the model performance jumps to 70% and 76% for single-task and multi-task models respectively. Some source of errors still remains due to annotation inconsistencies ("after lunch" or "lunch" as concept).

---

[2]https://allennlp.org/

| Model | Precision | Recall | F1 |
|---|---|---|---|
| **Zero-shot** | | | |
| Jia *et al.* [8] | 41.1% | 40.0% | 40.3% |
| Single-Task BERT | 44.0% | 55.1% | 48.9% |
| Multi-Task BERT | **50.5**% | **61.3**% | **55.3**% |
| **Supervised** | | | |
| Jia *et al.* [8] | N/A | N/A | N/A |
| Single-Task BERT | 62.2% | 80.0% | 70.0% |
| Multi-Task BERT | **69.1**% | **86.0**% | **76.6**% |

Table 1: Concept Parser performance on unknown concepts on Personalized Test.

| Model | Precision | Recall | F1 |
|---|---|---|---|
| Single-Task BERT | 90.86 | 91.72 | 91.29 |
| Multi-Task BERT | 93.24 | 84.86 | 88.86 |

Table 2: Concept Parser performance on internal dataset using models with additional Relevance head

| Model | Precision | Recall | F1-score |
|---|---|---|---|
| RoBERTa | **91.80** | **90.89** | **91.34** |
| BERT | 90.71 | 90.01 | 90.36 |
| ALBERT | 89.93 | 90.53 | 90.23 |

Table 3: Definition Understanding results on Jia *et al.* [8] dataset

| Model | Precision | Recall | F1-score |
|---|---|---|---|
| RoBERTa | **95.92** | **96.81** | **96.36** |
| BERT | 95.57 | 96.03 | 95.80 |
| ALBERT | 95.41 | 95.08 | 95.24 |

Table 4: Definition Understanding results on internal dataset

**Internal Datasets**: Table 2 reports the performance of our concept parser on the internal dataset. From the table we can observe the following. Our single-task and multi-task models, both trained on personalized data, achieve competitive performance on the synthesized evaluation data. When personalized data are available in training data, single-task model performs already decently (91.29% and 88.86%); yet multi-task model could still achieve higher precision. We would like to highlight here that our internal evaluation dataset is also more challenging than the public dataset, including a wide variety of "not teachable" examples including out-of-domain requests, regular requests without concept and ill-grammar and incomplete utterances.

#### 4.1.2 Definition understanding results

**Public Datasets**: Table 3 shows the performance of our definition understanding model on personalized test set from [8] with three Transformer-based encoders. From the results, we observe that model with RoBERTa [16] encoder achieves the best performance in terms of precision, recall and F1-score. BERT [5] and ALBERT [13] have similar performance and both are lightly behind RoBERTa [16]. We also did an error analysis and found that nearly half of the span prediction errors are due to a mismatch on preposition (e.g. "on Sept 15" vs "Sept 15"), which is not unexpected since span annotations in [8] are not consistent on the inclusion of prepositions.

**Internal Datasets**: The performance of our concept parser on the internal dataset is reported in Table 4. Again, we observe that model with RoBERTa [16] encoder slightly outperforms model with BERT [5] and ALBERT [13].

#### 4.1.3 Policy Model results

Table 5 shows the performance of our Transformer-based policy model along with heuristics based on NLU grounding results, on an internal dataset with annotated ground-truth actions. We observe that model with RoBERTa [16] encoder achieves the best performance in terms of precision, recall and F1-score. ALBERT [13] showed comparable performance to BERT [5] but ran the fastest.

### 5 Conclusion

We have presented a teachable dialogue system that uses neural models for gap identification, definition understanding and dialogue policy prediction to conduct interactive teaching sessions with the user to learn and re-use definitions of concepts that are unknown to a conversational AI system. This Teachable Dialogue system helps in automatically improving the understanding capabilities

| Model | Precision | Recall | F1 | mins/epoch |
|---|---|---|---|---|
| RoBERTa | **98.78** | **97.71** | **98.24** | 6.12 |
| BERT | 98.73 | 96.26 | 97.48 | 5.11 |
| ALBERT | 97.98 | 96.12 | 97.04 | 4.56 |

Table 5: Policy Model performance evaluation results on internal dataset

of the AI system to hold more natural conversations with the end-users, and progressively improve the agents' understanding of users' parlance to enable more natural ways of interaction with a conversational AI system. We believe this is an effort towards building truly interactive learning systems, and plan to extend the scope to higher level capabilities such as intent teaching.

# References

[1] James Allen, Nathanael Chambers, George Ferguson, Lucian Galescu, Hyuckchul Jung, Mary Swift, and William Taysom. Plow: a collaborative task learning agent. In *Proceedings of the AAAI*, 2007.

[2] Danqi Chen. *Neural reading comprehension and beyond*. Stanford University, 2018.

[3] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.

[4] Qian Chen, Zhu Zhuo, and Wen Wang. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*, 2019.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[6] Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757, 2018.

[7] Wang X. Yuan C. Yang G. Hu S. Shi Y. Hou, M. Attention based joint model with negative sampling for new slot values recognition. In *9th International Workshop on Spoken Dialogue System Technology*, 2019.

[8] R. Jia, L. Heck, D. Hakkani-Tür, and G. Nikolov. Learning concepts through conversations in spoken dialogue systems. In *Proceedings of the IEEE ICASSP Conference*, New Orleans, LA, 2017.

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[10] James R. Kirk and John E. Laird. Learning hierarchical symbolic representations to support interactive task learning and knowledge transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

[11] Yoshida T. Iwata K. Fujimura H. Akamine M. Kobayashi, Y. Out-of-domain slot value detection for spoken dialogue systems with context information. In *2018 IEEE Spoken Language Technology Workshop (SLT) (pp. 854-861*, Melbourne, Australia, 2018.

[12] Yuka Kobayashi, Takami Yoshida, Kenji Iwata, and Hiroshi Fujimura. Slot filling with weighted multi-encoders for out-of-domain values. In *INTERSPEECH*, pages 854–858, 2019.

[13] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[14] Bing Liu and Ian Lane. Attention-based recurrent neural network models for joint intent detection and slot filling. *arXiv preprint arXiv:1609.01454*, 2016.

[15] Bing Liu and Ian Lane. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 482–489. IEEE, 2017.

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[17] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539, 2014.

[18] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[19] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

[20] Percy Liang Sida I. Wang, Samuel Ginn and Christopher D. Manning. Naturalizing a programming language via interactive learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[21] Ryuichi Takanobu, Hanlin Zhu, and Minlie Huang. Guided dialog policy learning: Reward estimation for multi-domain task-oriented dialog. *arXiv preprint arXiv:1908.10719*, 2019.

[22] P. Xu and Q. Hu. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, 2018.

[23] Puyang Xu and Ruhi Sarikaya. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 ieee workshop on automatic speech recognition and understanding*, pages 78–83. IEEE, 2013.

[24] Huang H. Yang, P. and X. L. Mao. Context-sensitive generation network for handing unknown slot values in dialogue state tracking. 2020.

[25] Pengcheng Yin and Graham Neubig. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[26] Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip S Yu. Joint slot filling and intent detection via capsule neural networks. *arXiv preprint arXiv:1812.09471*, 2018.

[27] L. Zhao and Z. Feng. Improving slot filling in spoken language understanding with joint pointer and attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[28] Tiancheng Zhao and Maxine Eskenazi. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560*, 2016.

[29] Li Zhou, Kevin Small, Oleg Rokhlenko, and Charles Elkan. End-to-end offline goal-oriented dialog policy learning via policy gradient. *arXiv preprint arXiv:1712.02838*, 2017.