EDC: Effective and Efficient Dialog Comprehension For Dialog State Tracking

Qifan Lu
University of Washington
lqf96@uw.edu

Bhaskar Ramasubramanian
Western Washington University
ramasub@wwu.edu

Radha Poovendran
University of Washington
rp3@uw.edu

Abstract

In Task-Oriented Dialog (TOD) systems, Dialog State Tracking (DST) structurally extracts information from user and system utterances, which can be further used for querying databases and forming responses to users. The two major categories of DST methods, sequential and independent methods, face trade-offs between accuracy and efficiency. To resolve this issue, we propose Effective and Efficient Dialog Comprehension (EDC), an alternative DST approach that leverages the tree structure of the dialog state. EDC predicts domains, slot names and slot values of the dialog state step-by-step for better accuracy, and efficiently encodes dialog contexts with causal attention patterns. We evaluate EDC on several popular TOD datasets and EDC is able to achieve state-of-the-art Joint Goal Accuracy (JGA). We also show theoretically and empirically that EDC is more efficient than model designs used by previous works.

1 Introduction

Task-Oriented Dialog (TOD) systems are essential for building intelligent virtual assistants and chatbots, and aim to fulfill users' goals by understanding their intentions, performing corresponding actions and responding with results to users. In a TOD system, Dialog State Tracking (DST) extracts information such as user intention and system suggestions from dialog history and represents the known information of the dialog in a structural way, which can be further used for querying entities in databases and generating responses to users. Structural dialog state usually consists of one or multiple mappings for different *domains*, each describing user requirements or system suggestions of one type of entity or service. Within a domain, a *slot* identifies a single property and is represented by the pair of a *slot name* and a *slot value* in the domain's corresponding mapping.

Existing DST methods can be categorized into two types: sequential and independent. Both operate on partial dialog history, the concatenation of past user and system utterances up to a certain number of interaction rounds, and henceforth referred to as *context*. Sequential methods predict dialog state in one go by generating a single sequence with *present* domains and slots. Independent methods predict dialog state by separately determining the existence and value of all possible domain-slot combinations. They may be further divided into generative and classification-based methods: generative independent methods predict slot value by sequence-to-sequence generation, while classification-based independent methods remodel slot value prediction as a multiple-choice problem, and select slot value from a list of candidates.

Between both categories of DST methods, one issue we discovered is the trade-off between accuracy and efficiency. Sequential methods are efficient because generation only happens once and non-present domains and slots do not need to be predicted. However, when the dialog state becomes more complex, the target sequence becomes longer and different to generate precisely. Independent methods are more precise because they only need to deal with one slot at a time, but are inefficient when dealing with a large amount of domains and slots. By leveraging the hierarchical structure of dialog state, we argue a third approach is possible for DST: viewing dialog state as a three-level tree of present domains, slot names and slot values, and predicting this tree level-by-level. In this way, both effectiveness and efficiency can be achieved: length of each individual target sequence is reduced, and ability to predict only present domains and slots are preserved.

In this paper, we propose Effective and Efficient Dialog Comprehension (EDC), a multi-pass DST method utilizing tree prediction concept mentioned above. EDC is based on existing pre-trained language models, and also learns to predict user and system actions as auxiliary tasks in conjunction to dialog states. In addition, EDC also efficiently encodes dialog contexts of all rounds in the same dialog once with causal attention patterns. We evaluate EDC on MultiWoZ 2.2 (Zang et al., 2020) and M2M (Shah et al., 2018) datasets. EDC is able to achieve state-of-the-art Joint Goal Accuracy (JGA) among sequential methods, and also outperforms most independent methods. We then perform experiments on ablation settings and alternative designs to justify the design of our method and model. Finally, we show both theoretically and empirically that EDC is more efficient than traditional model designs during both training and prediction.

2 Related Work

2.1 Dialog State Tracking

2.1.1 Sequential Methods

The most common sequential DST approach is to generate a sequence containing all domains and slots and conforming to a set of formal grammar. The sequence is then mechanically parsed into structural dialog state. This approach is adopted by many DST-specific methods (Zhang et al., 2020b; Cheng et al., 2020) as well as end-to-end (Hosseini-Asl et al., 2020; Peng et al., 2021; Yang et al., 2021) and multi-task (Su et al., 2021) TOD systems. In addition, extra information such as domain, slot name, slot description and candidate values may be provided as input with dialog context to further enhance performance (Feng et al., 2021; Lin et al., 2021; Zhao et al., 2022). Alternatively, dialog state may also be generated in one go by predicting state difference between neighboring rounds and patching previous state (Lin et al., 2020), or generating then amending dialog state (Tian et al., 2021).

2.1.2 Independent Methods

The majority of independent DST methods are classification-based methods. Earliest methods (Lee et al., 2019; Wu et al., 2020) use only classification to determine slot existence and value: each possible slot value is made a distinct class, and a special "none" class indicates slot is not present in dialog state. Later works (Zhang et al., 2020a; Chao and Lane, 2019) also incorporate span extraction to deal with dynamic slot values that appear in dialog history. For these methods, span extraction is often performed with pointer network (Vinyals et al., 2015) or BIO tagging (Ramshaw and Marcus, 1995). In addition, (Kim et al., 2020) explores the idea of maintaining an external memory and updating the memory with output of the model. (Heck et al., 2020) then combines span extraction with two memory tables that tracks informed and co-referenced slot values. Due to its superior performance, it has since been used for examining the capability of TOD-focused pre-trained models (Mehri et al., 2020; Yu et al., 2021) or efficacy of alternative data augmentation and training techniques (Li et al., 2021; Dai et al., 2021; He et al., 2022). On the other hand, independent methods may also be implemented in a generative fashion (Wu et al., 2019; Huang et al., 2020; Lee et al., 2021) that separately predict slot value given dialog context, domain and slot name as input.

2.1.3 Tree Methods

There are several DST methods that are aware of and utilizing the tree structure of dialog state. TreeDST (Cheng et al., 2020) predicts dialog state as a S-expression (McCarthy, 1960) using a LSTM network enhanced by copy mechanism. Space-2 (He et al., 2022) performs contrastive learning based on the structural similarity of dialog state trees. EDC is different from these methods in that the tree structure influences the prediction procedure rather than just the representation of dialog state, resulting in a multi-pass process where domain, slots, and slot values are generated separately and step-by-step.

2.2 Pre-trained Language Models

Pre-trained language models built on multi-head attention mechanism have achieved state-of-the-art performance on various natural language understanding and generation problems. They can be divided into three major categories: encoder-only models (Kenton and Toutanova, 2019; Liu et al., 2019), decoder-only auto-regressive models (Radford et al., 2018, 2019; Brown et al., 2020) and encoder-decoder models (Vaswani et al., 2017; Lewis et al., 2020; Raffel et al., 2020). EDC is built on BART (Lewis et al., 2020), but can be extended to use any encoder-decoder language model as its backbone. It is also inspired by the causal attention mask construction of UniLM (Dong et al., 2019).

3 EDC: Effective and Efficient Dialog Comprehension

In this section, we introduce the detailed design of EDC. First, we formulate the DST problem and

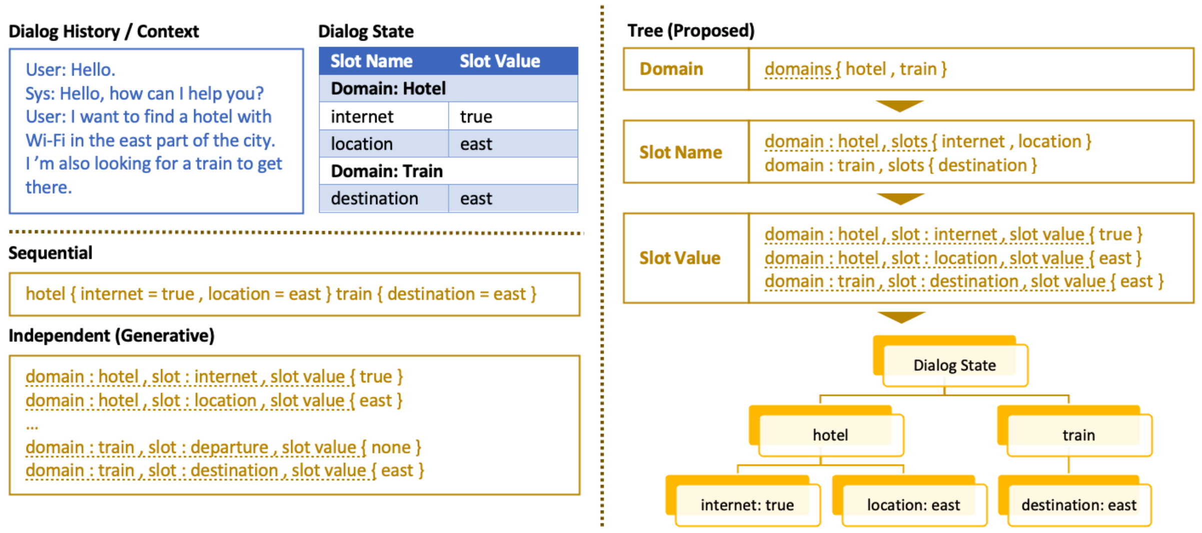
Dialog History / Context
User: Hello.
Sys: Hello, how can I help you?
User: I want to find a hotel with Wi-Fi in the east part of the city. I 'm also looking for a train to get there.
Dialog State
Slot Name
Slot Value
Domain: Hotel
internet
true
location
east
Domain: Train
destination
east
Sequential
hotel { internet = true , location = east } train { destination = east }
Independent (Generative)
domain : hotel , slot : internet , slot value { true }
domain : hotel , slot : location , slot value { east }
...
domain : train , slot : departure , slot value { none }
domain : train , slot : destination , slot value { east }
Tree (Proposed)
Domain
domains { hotel , train }
Slot Name
domain : hotel , slots { internet , location }
domain : train , slots { destination }
Slot Value
domain : hotel , slot : internet , slot value { true }
domain : hotel , slot : location , slot value { east }
domain : train , slot : destination , slot value { east }
Dialog State
hotel
train
internet: true
location: east
destination: east

Figure 1: An example of DST task and comparison of sequential, generative independent and tree prediction methods. Dotted underline indicates additional prompt fed into the model as input along with dialog context.

show how DST can be converted into a tree prediction problem. Then, we propose an efficient dialog context encoding technique based on the incremental property of dialog context. Next, we explain how the dialog state data can be decoded from our model step-by-step. Finally, we present the auxiliary tasks EDC learns to perform and the optimization objective.

3.1 Problem Formulation

We define full dialog history as a series of alternating utterances between the user and the system $C = \{U_1, R_1, U_2, R_2, \ldots, U_T, R_T\}$, where U_t and R_t denote user and system turns of round t, and T is the total number of rounds of the dialog. At each round $t = 1 \ldots T$, we are given context $C_t = \{U_1, R_1, \ldots, U_{t-1}, R_{t-1}, U_t\}$. Our goal is to predict structural dialog state $B_t = \{(d_i, B_{t,i})|_{i=1}^{|D|}\}, B_{t,i} = \{(s_j, v_{t,i,j})|_{j=1}^{|S|}\}$, where d_i and $B_{t,i}$ are the name and domain state of the i-th domain at round t, and s_j and $v_{t,i,j}$ are the name and value of the j-th slot within that domain. $v_{t,i,j}$ may be an ordinary value if slot s_j is present in domain d_i at round t, or it can be a special value $\emptyset$ if it is not present.

3.2 DST as Tree Prediction

Figure 1 gives an example of common categories of DST methods. For round t, most sequential DST methods aim to predict a single dialog state sequence B_t given context C_t. Let $\longrightarrow$ indicates generating a target sequence from an input sequence, $\oplus$ indicates concatenation of sequences, and assume t in notations, then this process can be described as:

$$\begin{gathered} C_t \longrightarrow B \\ B := d_1 \oplus B_1 \oplus \cdots \oplus d_M \oplus B_M \\ B_i := s_1 \oplus v_1 \oplus \cdots \oplus s_{N_i} \oplus v_{N_i} \\ (i = 1 \ldots N_i) \end{gathered}$$

where M is the number of *present* domains in dialog state, and N_i is the number of *present* slots in i-th *present* domain. On the other hand, most independent DST methods predict slot existence and values $|D| \times |S|$ times for each possible domain-slot pair:

$$\begin{gathered} C_t \oplus d_i \oplus s_j \longrightarrow v_{i,j} \\ (i = 1 \ldots |D|, j = 1 \ldots |S|) \end{gathered}$$

In contrast, EDC utilizes the hierarchical structure of the dialog state, viewing it as a tree with three levels of nodes: domains at the top level, slot names in the middle and slot values at the bottom. Tree nodes are then predicted in a breadth-first manner by providing dialog context and ancestor data:

$$\begin{gathered} C_t \longrightarrow d_1 \oplus \cdots \oplus d_M \\ C_t \oplus d_i \longrightarrow s_1 \oplus \cdots \oplus s_{N_i} \\ C_t \oplus d_i \oplus s_j \longrightarrow v_j \\ (i = 1 \ldots M, j = 1 \ldots N_i) \end{gathered}$$

Under tree prediction, dialog state is decomposed into multiple shorter target sequences rather than a single long sequence, making it easier to precisely generate target sequence and predict dialog state

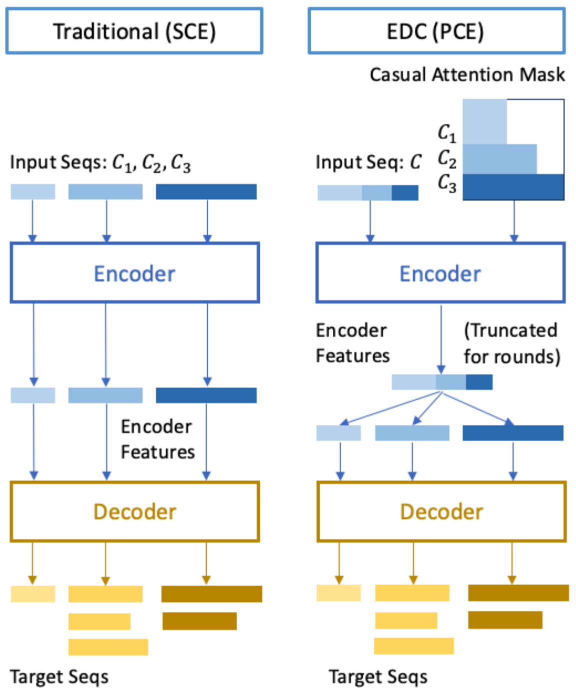

Figure 2: EDC model architecture and Prefix Context Encoding (PCE) technique, and comparison with traditional context encoding approach.

over sequential methods. Meanwhile, all sequences in tree prediction only includes present domains and slots. This ensures scalability with arbitrary number of domains and slots, and better efficiency than independent methods.

3.3 Prefix Context Encoding

Most TOD systems utilizing encoder-decoder Transformer-style language models encode contexts of the same dialog $C_1, C_2, \ldots, C_T$ separately. Under this approach, which we name as Standalone Context Encoding (SCE), dialog context must be fully re-encoded even if it is incrementally updated with utterances from next round. To improve efficiency, we propose Prefix Context Encoding (PCE), an alternative encoding technique leveraging causality of utterances within dialog context and encoding contexts of all rounds together. The full process of PCE is shown in figure 2. For full dialog history C and context C_t of round t:

$$\begin{gathered}C_1 := U_1\\ C_t := C_{t-1} \oplus S_{t-1} \oplus U_t\\ C := C_T \oplus R_T\\ (t = 2 \ldots T)\end{gathered}$$

It is intuitive that given $t_1 < t_2$, C_{t_1} is a prefix of C_{t_2} and C. We can then construct the following causal attention matrix for the encoder:

$$a_{ij} = \begin{cases} 1 & \text{if } |C_{t-1}| < j \leq |C_t|, j \leq |C_t| \\ 0 & \text{otherwise} \end{cases}$$

where i and j are the indices of dialog history tokens, and token i is in the user or system utterance of the t-th round. The rationale behind PCE is that for tokens within the same utterance, we mimic encoder-only models and use fully-connected attention patterns, while for tokens belonging to different utterances, attention pattern is instead causal to reflect the occurring order of utterances.

In practice, EDC encodes a dialog by forwarding full dialog history C and the PCE attention matrix through the encoder stack of the underlying model. This gives us dialog encoder features $X \in \mathbb{R}^{|C|\times h}$, where h is the size of feature dimension. For each round t, round-specific features $X_t \in \mathbb{R}^{|C_t|\times h}$ are then obtained by truncating X to the feature vectors of first $|C_t|$ tokens, and passed to the decoder stack for target sequence generation. The use of PCE allows EDC to encode all contexts of the same dialog at once, thus speeding up training by reducing redundant encoder operations as well as the number of training samples.

3.4 Decoding Tree Data

In traditional generative methods, additional information such as domain, slot name, slot description and candidate values are often combined as a prompt, and prepended or appended to the dialog context. Due to the use of PCE, EDC instead reserves the encoder of the underlying model for the context, and shifts this prompt to the decoder side:

$$\begin{gathered}C_t \Longrightarrow d_1 \oplus \cdots \oplus d_M\\ C_t \Longrightarrow d_i \longrightarrow s_1 \oplus \cdots \oplus s_{N_i}\\ C_t \Longrightarrow d_i \oplus s_j \longrightarrow v_j\end{gathered}$$

where $\Longrightarrow$ indicates encoding the dialog context and providing the features to the decoder, and $\longrightarrow$ indicates starting the target sequence on the decoder side with the prompt of domain (and slot name) and then decoding the node data.

For decoder, we mimic the process of solving sequence-to-sequence generation problems with autoregressive language models. During training, we concatenate the prompt and tree data into a single target sequence, and pass it to the decoder with the autoregressive attention pattern. However, we only include tokens from tree data when computing optimization objective, as we consider the information

Method	Backbone Model	JGA
Classification-based		
DiCoS-DST	ALBERT-large (18M)	**61.1**
	BERT-base (117M)	60.2
TripPy	BERT-base (117M)	56.1
SOM-DST	BERT-base (117M)	53.8
DS-DST	BERT-base (117M)	51.7
Generative		
D3ST	T5-XXL (11B)	**58.7**
	T5-base (220M)	56.1
SDP-DST	T5-base (220M)	57.6
AG-DST	PLATO2 (310M)	57.4
	GPT2-base (117M)	56.1
Seq2Seq-DU	BERT-base (110M)	54.4
TRADE	-	45.5
EDC	BART-base (139M)	**58.7** ± 0.4

Table 1: JGA results on MultiWoZ 2.2 with backbone model and its size. Results for both best and similar-sized backbone model are reported.

within the prompt to be known. During prediction, we construct the prompt from data predicted from previous steps and feed the prompt into the decoder first before decoding tree data of the current level.

3.5 Auxiliary Tasks

To further enhance the performance of EDC, we consider action prediction auxiliary tasks in addition to the main DST task (abbreviated as *DS*). For round t, given context C_t and extended context $C'_t = C_t \oplus R_t$, we predict user actions (*UA*) and system actions (*SA*), which are the structural representation of user intentions from U_t and system intentions from R_t. Similar to dialog state, both user and system actions can be represented as four-level hierarchies consisting of domains, intents, slot names and slot values. As such, we reuse the same tree prediction technique from above for action prediction auxiliary tasks.

3.6 Optimization Objective

EDC treats the whole dialog as a single sample. For every round t of the dialog and each task $k = \{DS, UA, SA\}$, EDC produces a number of target sequences for tree prediction. Let $Y_{t,k}$ be the collection of these sequences, and $Y_{t,k,i}$ be the i-th such sequence, consisting of prompt $P_{t,k,i}$ and excepted output $D_{t,k,i}$. EDC then tries to maximize

Method	SIM-M	SIM-R	M2M
Oracle	96.8	94.4	-
DiCoS-DST	84.5	91.6	-
SDP-DST	83.3	90.6	88.0
TripPy	83.5	90.0	-
BERT-DST	80.1	89.6	-
BIO-DST	50.4	87.1	73.8
EDC	**84.9** ± 2.0	**91.7** ± 0.3	**88.6** ± 1.1

Table 2: JGA results on M2M and its subsets. Oracle refers to (Rastogi et al., 2017) and should be considered as upper bound JGA achievable by any DST methods. BIO-DST is our unofficial name for method proposed in (Rastogi et al., 2018), which was not named by its authors.

the following optimization objective:

$$L = \frac{\sum_{t=1}^{T} \sum_{k} \sum_{i=1}^{|Y_{t,k}|} \log p(D_{t,k,i}|C_t, P_{t,k,i})}{\sum_{t=1}^{T} \sum_{k} \sum_{i=1}^{|Y_{t,k}|} |D_{t,k,i}|}$$

where $p(D_{t,k,i}|C_t, P_{t,k,i})$ is the probability of autoregressively producing each token within $D_{t,k,i}$. In other words, for each token within the target output, we maximize its average log probability given all previously generated tokens, prompt and dialog context. During prediction, target output is greedily generated by always choosing token with the largest likelihood.

4 Experiments

In this section, we present various experimental results of EDC. We start with our training datasets and settings. Then, we compare the metrics of EDC to those of previous works as well as ablation settings and alternative designs. Next, we analyze the efficiency of EDC by computing and comparing its theoretical and empirical computational complexity. Finally, we categorize EDC's DST prediction errors on selected samples.

4.1 Settings

4.1.1 Datasets

We train and evaluate EDC models on the following TOD datasets:

- **MultiWoZ 2.2** MultiWoZ is a multi-domain TOD dataset initially proposed in

(Budzianowski et al., 2018), with 3,406 single-domain dialogs and 7,032 multi-domain dialogs from 8 domains. We specifically choose version 2.2 because it contains annotation error and inconsistency fixes from previous iterations of MultiWoZ (Eric et al., 2020; Zang et al., 2020), while at the same time is benchmarked on more related works than later versions such as MultiWoZ 2.4 (Ye et al., 2022).

- **M2M** (Shah et al., 2018) is a partially-simulated single-domain TOD dataset. It has two subsets: SIM-M with 2,240 dialogs on movies and SIM-R with 768 dialogs on restaurants.

4.1.2 Preprocessing and Training

We preprocess all datasets into same format: all dialog states conform to a three-level hierarchy of domains, slot names and slot values, and all user and system actions conform to a four-level hierarchy of domains, intents, slot names and slot values. For dialog states, all slots with "none" or "not mentioned" values are removed, while slots with "don't care" value are normalized and preserved.

During training, we schedule our learning rate such that for the first 40% of steps, learning rate linearly increase from 0 to lr_{max}, and then decrease from lr_{max} to 0 for remaining steps. We train our model with 3 different random seeds for all settings and report their average metrics. Our detailed settings can be found in appendix Section A.

4.1.3 Evaluation Metrics

To compare performance of different DST methods, we evaluate Joint Goal Accuracy (JGA) (Budzianowski et al., 2018), defined as the number of rounds with correct dialog state predictions divided by total number of dialog rounds. The dialog state of a round is considered correct only if domains, slots and slot values are all correct. We prefer methods with higher JGA, and for two methods with similar JGA, we prefer method that utilizes smaller backbone model for better efficiency.

4.2 Experimental Results

4.2.1 MultiWoZ 2.2

Table 1 shows the JGA of EDC evaluated on MultiWoZ 2.2, compared with various classification-based (Zhang et al., 2020a; Heck et al., 2020; Kim et al., 2020; Guo et al., 2022) and generative (Wu et al., 2019; Feng et al., 2021; Tian et al., 2021; Lee et al., 2021; Zhao et al., 2022) DST baselines on best and similar-sized backbone language models. EDC achieves best performance among generative methods with similar-sized or small backbone models, and while its JGA is on par with D3ST's best result, it has better efficiency due to the use of a much smaller backbone model. When compared with classification-based methods, EDC performs better than most of the baselines except DiCoS-DST (Guo et al., 2022), a state-of-the-art DST method with complex design and model structure. Overall, EDC is able to effectively extract dialog state from multi-domain dialogs with backbone models comparable or smaller than previous works.

Setting / Design	**JGA**
Ablation Settings	
Full (With Both Actions)	58.5 $\pm$ 0.1
Without User Action	**58.7** $\pm$ 0.4
Without System Action	58.5 $\pm$ 0.3
Without User & System Actions	58.0 $\pm$ 0.2
Alternative Designs	
Sequential	56.2 $\pm$ 0.4
Standalone Context Encoding	57.9 $\pm$ 0.2
Traditional	56.0 $\pm$ 0.5

Table 3: JGA results for all ablation settings and alternative model designs. All ablation settings are considered valid variants of EDC, and JGA of the best setting is used for comparison against other baselines. All alternative designs are compared against full ablation setting.

4.2.2 M2M

Table 2 shows the JGA of EDC, evaluated separately on the SIM-M, SIM-R subsets of the M2M dataset as well as the whole datasets against several baselines (Rastogi et al., 2018; Chao and Lane, 2019; Heck et al., 2020; Lee et al., 2021; Zhao et al., 2022). Again, EDC achieves best JGA on all three datasets, showing that it also performs well on simple, single-domain TOD datasets.

4.3 Ablation Studies and Alternative Designs

To determine effects and contributions of various design choices of EDC, we also perform ablation experiments by removing auxiliary tasks from training, and examine common alternative model designs used by previous works. We consider the following ablation settings:

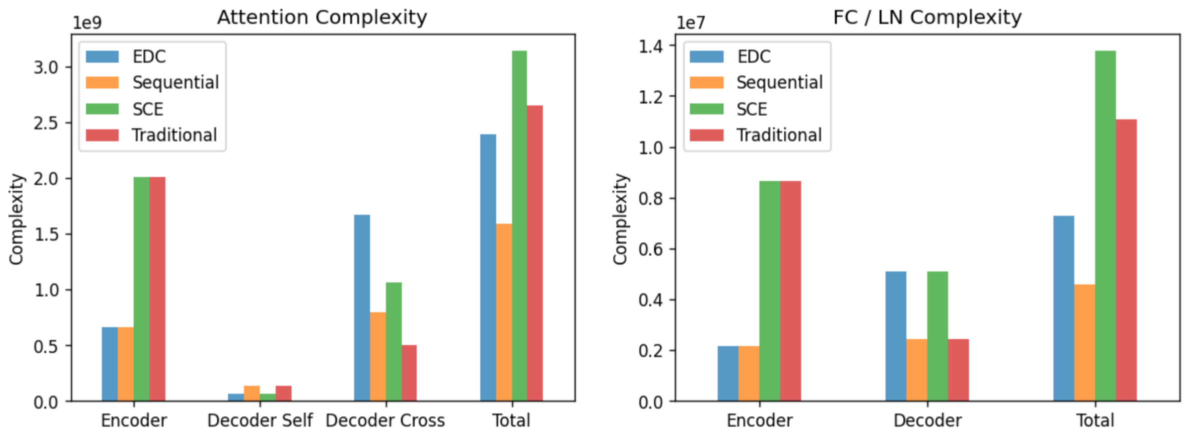

Figure 3: Theoretic computational complexity comparison of operations for different designs. For PCE, full history C is encoded at once and $m = |C|$. For SCE, context C_t for each round t is separately encoded and $m = |C_t|$.

- **Full (With User and System Actions).** The full setting where the EDC model is trained to perform DST as well as user action and system action predictions.
- **Without User Action.** User action prediction task is removed from training.
- **Without System Action.** System action prediction task is removed from training.
- **Without User and System Actions.** Both action prediction tasks are removed. The model is only trained to perform DST.

In addition, we also consider following alternative model designs, which are compared against the full EDC setting:

- **Sequential.** Dialog states and actions are predicted in one go by generating and parsing a single sequence.
- **Standalone Context Encoding (SCE).** Contexts from the same dialog are encoded separately with a fully-connected attention pattern.
- **Traditional.** A design that combines sequential and SCE settings. This is the traditional sequential approach to solve DST problem.

Table 3 shows the JGA results of the ablation settings and alternative designs. Of all ablation settings, variants with one action prediction task perform comparably to full setting. However, the bare setting with both tasks removed performs worse. Our conjecture is that predicting user or system action for each round likely helps the model capture dynamics of the dialog, and having either task will be sufficient. Additionally, we notice a slight performance drop of the full setting compared to the system action only setting. This is most likely to be caused by potential conflict between user action prediction and DST tasks due to annotation errors and inconsistencies in the MultiWoZ dataset.

When compared to alternative designs, EDC outperforms the sequential alternative by more than 2 percent, showing the superiority of the tree prediction approach. It also achieves slightly better JGA than the SCE alternative. We conjecture that the use of PCE reduces the number of training samples and steps, and as a side effect reduces overfitting to the training set in this case. Finally, the traditional alternative, which is a combination of sequential prediction and SCE, performs the worst among all settings as expected. Overall, the ablation experiments justify the multi-task, tree prediction design and use of PCE in the EDC model.

4.4 Efficiency Evaluation

4.4.1 Theoretical Complexity Estimation

To estimate the efficiency of EDC, we establish a theoretical computational complexity model based on the typical structure of a Transformer-style encoder-decoder model. Let m be length of input sequence to the encoder, and n be length of target sequence generated from the decoder. We can then assume complexity of following operations:

- **Attention.** Multiplicative attention layers. Complexity is $\mathcal{O}(m^2)$ for encoder, $\mathcal{O}(n^2)$ for decoder self-attention and $\mathcal{O}(mn)$ for decoder cross-attention.
- **FC / LN.** Remaining layers in Transformer blocks, including fully-connected projection

Design	Training Execution Time
EDC (Full)	366.10s
Sequential	**243.47s**
SCE	1236.35s
Traditional	1208.97s

Table 4: Average measured training time for full setting and alternative designs under same environment.

and layer normalization layers. Complexity is $\mathcal{O}(m)$ for encoder and $\mathcal{O}(n)$ for decoder. [1]

Based on this complexity model, we run design-specific preprocessing steps on the MultiWoZ 2.2 dataset and obtain input and target sequences. We then compute and accumulate complexity estimates for different operations on these sequences.

Figure 3 compares the estimated theoretical complexity of EDC and alternative designs. Owning to the use of PCE, EDC is significantly more efficient than alternative designs that utilize SCE, saving around 75% of computation for both encoder attention and FC / LN layers. Tree prediction, on the other hand, has mixed effects on efficiency: compared to sequential setting that produces a single long sequence, tree prediction produces many shorter target sequences. For decoder self attention, increased complexity caused by more sequences can be offset by the quadratic reduction of per-sequence complexity. For decoder cross attention and FC / LN layers, however, more sequences do result in more complexity. Despite these results, by total complexity, EDC is still the second most efficient among all four settings, and achieves better JGA than the sequential design. Hence, we argue EDC is more efficient to train than traditional DST methods without sacrificing performance.

4.4.2 Empirical Training Time Evaluation

In addition to theoretic complexity analysis, we also measure empirical training time of EDC and alternative designs on MultiWoZ 2.2. We report average epoch training time across all epochs of 3 runs. We only measure execution time of the forward pass so as to exclude interference of backward and optimization processes.

Table 4 shows measured training execution time for different designs, which align well with the theoretical complexity analysis. Again, EDC is slower than sequential design but is much faster than the remaining two designs, saving around 70% of time. This empirically proves that EDC can be trained much quicker than previous methods.

[1]Although execution steps of FC / LN layers are not identical for encoder and decoder, we can assume their similarity and estimate their total complexity.

	Missing	Redundant	Incorrect
Pred.	9 (36%)	5 (20%)	4 (16%)
Anno.	1 (4%)	6 (24%)	0 (0%)
Total	10 (40%)	11 (44%)	4 (16%)

Table 5: Number of unique DST prediction errors of all types and origins of 25 samples randomly selected from prediction results on MultiWoZ 2.2.

5 Error Analysis

Finally, we analyze EDC's DST prediction errors on MultiWoZ 2.2. We randomly collect 25 round predictions with unique errors, where unique is defined as at least one new error of a slot is introduced in the current round, as opposed to slot prediction errors carried over from previous rounds. We divide all unique errors into three types:

- **Missing.** Slot is missing from prediction compared from ground truth.
- **Redundant.** Slot from prediction does not exist in ground truth.
- **Incorrect.** Slot does exist but predicted slot value does not match ground truth slot value.

Errors can also be classified by their origin:

- **Prediction.** Errors are made by EDC model.
- **Annotation.** Dialog state prediction of EDC is correct, but ground truth annotation is wrong or inconsistent.

Examples of these error categories can be found in appendix Section C.

Table 5 shows the number of unique DST prediction errors. Approximately two thirds of the errors are prediction errors, while remaining are annotation errors. Most prediction errors are missing or redundant slots, suggesting that EDC mostly makes mistakes during the slot name prediction step of the tree prediction process. One possible reason may be that slot name prediction is inherently harder than domain and slot value prediction, due to potentially longer target sequences caused by many slots within the dialog state. For annotation errors, EDC mostly predicts extra slots that are not present in the ground truth dialog state, indicating majority annotation errors in MultiWoZ 2.2 are incomplete dialog states that miss information mentioned in the dialog history.

6 Conclusion

We proposed Effective and Efficient Dialog Comprehension (EDC), an alternative DST method that leverages the tree structure of dialog state, and efficiently encodes dialog contexts with causal attention patterns. EDC achieves state-of-the-art DST performance, surpassing all sequential and most independent methods, which being efficient to train both theoretically and empirically.

In the future, we plan to extend EDC into a full multi-task TOD system that can also generate responses to users, in addition to its current capabilities of DST and user and system action prediction. We are also interested in fine-tuning Large Language Models (LLM) to perform TOD tasks, and bringing structural information awareness to open domain dialog systems powered by LLMs.

Limitations

When analyzing the efficiency of EDC, we focus on training and omit the analysis for inference, for the reason that inference complexity would be identical to training. EDC's encoder attention pattern is designed to be causal with regard to dialog rounds. When the dialog context is updated with the latest user input and system response, the encoder features can also be incrementally updated by reusing previous encoder features as well as cached intermediate keys and values from previous computation. For a complete dialog, this will result in a theoretical computation complexity model identical to complexity of training due to use of incremental update on both the encoder and decoder side.

Although EDC reduces training time by encoding all contexts of the same dialog once through PCE, its encoder complexity is still quadratic to the length of the full dialog history. Hence, EDC still requires significant amount of memory and execution time when the full dialog history is long. We have considered efficient Transformer-style language models such as T5 (Raffel et al., 2020), Transformer-XL (Dai et al., 2019) and Longformer (Beltagy et al., 2020), but these language models are not specifically designed for dialogs, as their attention mechanisms are not directly compatible with PCE. In the future, we aim to adapt EDC to these backbone models by making PCE compatible with their underlying attention mechanisms.

Another limitation of EDC is that currently we perform full fine-tuning on its backbone model. While full fine-tuning gives the model greatest degree of freedom to adapt to downstream data, it may result in overfitting and catastrophic forgetting (French, 1999), where knowledge learned by the pre-trained model is lost during the tuning process. We have observed this issue during preliminary experiments on BART-large, where we achieved lower JGA compared to BART-base despite training on a larger backbone model, due to dataset and model size mismatch. To overcome this issue, we will perform parameter-efficient fine-tuning in our future works, using techniques like prompt tuning (Lester et al., 2021) or LoRA (Hu et al., 2022).

Ethics Statement

Our work only utilizes public TOD datasets from previous works that do not contain any Personal Identifiable Information (PII). Dialogs in these datasets also focus solely on helping users solve problems or accomplish tasks, and do not have any bias towards participants. Thus, we conclude our work does not have any privacy or ethical issues.

Acknowledgments

This material is based upon work supported by the US National Science Foundation (NSF) under Grant IIS 2229876, and is supported in part by funds provided by the NSF, Department of Homeland Security, and by IBM. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or its federal agency and industry partners.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5016–5026. Association for Computational Linguistics.

Guan-Lin Chao and Ian R. Lane. 2019. BERT-DST: scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 1468–1472. ISCA.

Jianpeng Cheng, Devang Agrawal, Héctor Martínez Alonso, Shruti Bhargava, Joris Driesen, Federico Flego, Dain Kaplan, Dimitri Kartsaklis, Lin Li, Dhivya Piraviperumal, Jason D. Williams, Hong Yu, Diarmuid Ó Séaghdha, and Anders Johannsen. 2020. Conversational semantic parsing for dialog state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8107–8117. Association for Computational Linguistics.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. 2021. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialogue state tracking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 879–885. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13042–13054.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 422–428. European Language Resources Association.

Yue Feng, Yang Wang, and Hang Li. 2021. A sequence-to-sequence approach to dialogue state tracking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1714–1725. Association for Computational Linguistics.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Jinyu Guo, Kai Shuang, Jijie Li, Zihan Wang, and Yixuan Liu. 2022. Beyond the granularity: Multi-perspective dialogue collaborative selection for dialogue state tracking. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2320–2332. Association for Computational Linguistics.

Wanwei He, Yinpei Dai, Binyuan Hui, Min Yang, Zheng Cao, Jianbo Dong, Fei Huang, Luo Si, and Yongbin Li. 2022. SPACE-2: tree-structured semi-supervised contrastive pre-training for task-oriented dialog understanding. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 553–569. International Committee on Computational Linguistics.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020*, pages 35–44. Association for Computational Linguistics.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and

Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Yi Huang, Junlan Feng, Min Hu, Xiaoting Wu, Xiaoyu Du, and Shuo Ma. 2020. Meta-reinforced multi-domain state generator for dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7109–7118. Association for Computational Linguistics.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 567–582. Association for Computational Linguistics.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 4937–4949. Association for Computational Linguistics.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5478–5483. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Fatema Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2021. Coco: Controllable counterfactuals for evaluating dialogue state trackers. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Weizhe Lin, Bo-Hsiang Tseng, and Bill Byrne. 2021. Knowledge-aware graph-enhanced GPT-2 for dialogue state tracking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7871–7881. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3391–3405. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

John McCarthy. 1960. Recursive functions of symbolic expressions and their computation by machine, part I. *Commun. ACM*, 3(4):184–195.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tür. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *CoRR*, abs/2009.13570.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. SOLOIST: building task bots at scale with transfer learning and machine teaching. *Trans. Assoc. Comput. Linguistics*, 9:907–824.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Lance A. Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora, VLC@ACL 1995, Cambridge, Massachusetts, USA, June 30, 1995*.

Abhinav Rastogi, Raghav Gupta, and Dilek Hakkani-Tür. 2018. Multi-task learning for joint language understanding and dialogue state tracking. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue, Melbourne, Australia, July 12-14, 2018*, pages 376–384. Association for Computational Linguistics.

Abhinav Rastogi, Dilek Hakkani-Tür, and Larry P. Heck. 2017. Scalable multi-domain dialogue state tracking. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017, Okinawa, Japan, December 16-20, 2017*, pages 561–568. IEEE.

Pararth Shah, Dilek Hakkani-Tür, Gökhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry P. Heck. 2018. Building a conversational agent overnight with dialogue self-play. *CoRR*, abs/1801.04871.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2021. Multi-task pre-training for plug-and-play task-oriented dialogue system. *CoRR*, abs/2109.14739.

Xin Tian, Liankai Huang, Yingzhan Lin, Siqi Bao, Huang He, Yunyi Yang, Hua Wu, Fan Wang, and Shuqi Sun. 2021. Amendable generation for dialogue state tracking. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 80–92, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700.

Chien-Sheng Wu, Steven C. H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 917–929. Association for Computational Linguistics.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 808–819. Association for Computational Linguistics.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021. UBAR: towards fully end-to-end task-oriented dialog system with GPT-2. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14230–14238. AAAI Press.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2022. Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGDIAL 2022, Edinburgh, UK, 07-09 September 2022*, pages 351–360. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2021. Score: Pre-training for context representation in conversational semantic parsing. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics.

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wang, Philip S. Yu, Richard Socher, and Caiming Xiong. 2020a. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics, *SEM@COLING 2020, Barcelona, Spain (Online), December 12-13, 2020*, pages 154–167. Association for Computational Linguistics.

Yichi Zhang, Zhijian Ou, Min Hu, and Junlan Feng. 2020b. A probabilistic end-to-end task-oriented dialog model with latent belief states towards semi-supervised learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 9207–9219. Association for Computational Linguistics.

Jeffrey Zhao, Raghav Gupta, Yuan Cao, Dian Yu, Mingqiu Wang, Harrison Lee, Abhinav Rastogi, Izhak Shafran, and Yonghui Wu. 2022. Description-driven task-oriented dialog modeling. *CoRR*, abs/2201.08904.

A Training settings

Comparison with Previous Works. We train on 1 Nvidia V100 GPU (referred to as "GPU" below) for 5 epochs for MultiWoZ 2.2, 6 epochs for SIM-R and M2M, and 8 epochs for SIM-M. We use $lr_{max} = 2 \times 10^{-5}$ for SIM-R and $lr_{max} = 5 \times 10^{-5}$ for all other datasets.

Ablation Studies and Alternative Designs. We train on 4 GPUs for 5 epochs with $lr_{max} = 1.5 \times 10^{-4}$ except non-PCE settings where we use $lr_{max} = 5 \times 10^{-5}$ instead due to increased number of training steps.

Empirical Training Time Evaluation. The EDC model is trained for 5 epochs on 3 different runs, and training forward execution time is averaged over all 15 epochs.

B Theoretic Complexity Estimation Data

Table 6 and 7 contain raw data for the theoretic complexity estimation of attention and fully-connected and layer normalization (FC / LN) layers across different designs, which is used to plot Figure 3.

C DST Error Examples

Table 8 shows examples of different DST error types mentioned in Table 5.

Design	Encoder	Decoder Self	Decoder Cross	Total
Theoretical	$\mathcal{O}(m^2)$	$\mathcal{O}(n^2)$	$\mathcal{O}(mn)$	-
EDC (Full)	$\mathbf{6.6 \times 10^8}$	$\mathbf{6.4 \times 10^7}$	1.7×10^9	2.4×10^9
Sequential	$\mathbf{6.6 \times 10^8}$	1.4×10^8	7.9×10^8	$\mathbf{1.6 \times 10^9}$
SCE	2.0×10^9	$\mathbf{6.4 \times 10^7}$	1.1×10^9	3.1×10^9
Traditional	2.0×10^9	1.4×10^8	$\mathbf{5.0 \times 10^8}$	2.6×10^9

Table 6: Theoretic attention complexity estimation for different designs.

Design	Encoder	Decoder	Total
Theoretical	$\mathcal{O}(m)$	$\mathcal{O}(n)$	-
EDC (Full)	$\mathbf{2.2 \times 10^6}$	5.1×10^6	7.3×10^6
Sequential	$\mathbf{2.2 \times 10^6}$	$\mathbf{2.4 \times 10^6}$	$\mathbf{4.6 \times 10^6}$
SCE	8.7×10^6	5.1×10^6	1.4×10^7
Traditional	8.7×10^6	$\mathbf{2.4 \times 10^6}$	1.1×10^7

Table 7: Theoretic complexity estimation on fully-connected and layer normalization (FC / LN) layers for different designs.

Missing (Prediction)	**Dialog Context:**
	`U: i am looking for a hotel .`
	
	Prediction: {}
	Actual: {'hotel': {'type': 'hotel'}}

Missing (Annotation)	**Dialog Context:**
	`U: i'm looking for a moderately priced place to eat that's in the center of town .`
	`S: what type of cuisine are you looking for ? there are 21 restaurants in that area .`
	`U: i don't have a preference .`
	`S: out of the 21 restaurant choices , 1 is the yippee noodle bar which is moderately priced in the center of town . would you like to make a reservation ?`
	`U: that sounds great , what is the postcode ?`
	
	Prediction: {'restaurant': {'area': 'center', 'food': 'don't care', 'name': 'yippee noodle bar', 'pricerange': 'moderate'}}
	Actual: {'restaurant': {'area': 'center', 'food': 'don't care', 'pricerange': 'moderate'}}

Redundant (Prediction)	**Dialog Context:**
	`U: i'm looking for a restaurant in cambridge called nandos city center .`
	
	Prediction: {'restaurant': {'area': 'center', 'name': 'nandos city center'}}
	Actual: {'restaurant': {'name': 'nandos city center'}}

Redundant (Annotation)	**Dialog Context:**
	`U: i want to find an italian place to eat near the center of cambridge .`
	`S: there are 9 possibilities to choose from . what price range do you have in mind ?`
	`U: i would like for the restaurant to be expensive .`
	`S: i'd recommend don pasquale pizzeria . would you like more information on them or to book a reservation ?`
	`U: i would like to book a table for 2 .`
	
	Prediction: {'restaurant': {'area': 'center', 'book people': '2', 'food': 'italian', 'name': 'don pasquale pizzeria', 'pricerange': 'expensive'}}
	Actual: {'restaurant': {'area': 'center', 'food': 'italian', 'name': 'don pasquale pizzeria', 'pricerange': 'expensive'}}

Incorrect	**Dialog Context:**
	`U: i'm looking for a place to stay in the south of town . it doesn't need to have free parking .`
	
	Prediction: {'hotel': {'area': 'south', 'parking': 'no'}}
	Actual: {'hotel': {'area': 'south', 'parking': '@notcare'}}

Table 8: Examples of error types considered in the error analysis on MultiWoZ 2.2.