CoCo: Controllable Counterfactuals for Evaluating Dialogue State Trackers

Anonymous authors
Paper under double-blind review

Abstract

Dialogue state trackers have made significant progress on benchmark datasets, but their generalization capability to novel and realistic scenarios beyond the held-out conversations is less understood. We propose controllable counterfactuals (CoCo) to bridge this gap and evaluate dialogue state tracking (DST) models on novel scenarios, i.e., would the system successfully tackle the request if the user responded differently but still consistently with the dialogue flow? CoCo leverages turn-level belief states as counterfactual conditionals to produce novel conversation scenarios in two steps: (i) *counterfactual goal* generation at turn-level by dropping and adding slots followed by replacing slot values, (ii) *counterfactual conversation* generation that is conditioned on (i) and consistent with the dialogue flow. Evaluating state-of-the-art DST models on MultiWOZ dataset with CoCo-generated counterfactuals results in a significant performance drop of up to 30.8% (from 49.4% to 18.6%) in absolute joint goal accuracy. In comparison, widely used techniques like paraphrasing only affect the accuracy by at most 2%. Human evaluations show that COCO-generated conversations perfectly reflect the underlying user goal with more than 95% accuracy and are as human-like as the original conversations, further strengthening its reliability and promise to be adopted as part of the robustness evaluation of DST models.

1 Introduction

Task-oriented dialogue (TOD) systems have recently attracted growing attention and achieved substantial progress (Zhang et al., 2019b; Neelakantan et al., 2019; Peng et al., 2020; Wang et al., 2020b;a), partly made possible by the construction of large-scale datasets (Budzianowski et al., 2018; Byrne et al., 2019; Rastogi et al., 2019). Dialogue state tracking (DST) is a backbone of TOD systems, where it is responsible for extracting the user's goal represented as a set of slot-value pairs (e.g., (*area*, *center*), (*food*, *British*)), as illustrated in the upper part of Figure 1. The DST module's output is treated as the summary of the user's goal so far in the dialogue and directly consumed by the subsequent dialogue policy component to determine the system's next action and response. Hence, the accuracy of the DST module is critical to prevent downstream error propagation (Liu and Lane, 2018), affecting the success of the whole system.

With the advent of representation learning in NLP (Pennington et al., 2014; Devlin et al., 2019; Radford et al., 2019), the accuracy of DST models has increased from 15.8% (in 2018) to 55.7% (in 2020). While measuring the held-out accuracy is often useful, practitioners consistently overestimate their model's generalization (Ribeiro et al., 2020; Patel et al., 2008) since test data is usually collected in the same way as training data. In line with this hypothesis, Table 1 demonstrates that there is a substantial overlap of the slot values between training and evaluation sets of the MultiWOZ DST benchmark (Budzianowski et al., 2018). In Table 2, we observe that the slot co-occurrence distributions for evaluation sets tightly align with that of train split, hinting towards the potential limitation of the held-out accuracy in reflecting the actual generalization capability of DST models. Inspired by this phenomenon, we aim to address and provide insights into the following question: *how well do state-of-the-art DST models generalize to the novel but realistic scenarios that are not captured well enough by the held-out evaluation set?*

Most prior work (Iyyer et al., 2018; Jin et al., 2019) focus on adversarial example generation for robustness evaluation. They often rely on perturbations made directly on test examples in the held-

data	attraction-name	hotel-name	restaurant-name	taxi-departure	taxi-destination	train-departure	train-destination
dev	94.5	96.4	97.3	98.6	98.2	99.6	99.6
test	96.2	98.4	96.8	95.6	99.5	99.4	99.4

Table 1: The percentage (%) of domain-slot values in evaluation sets that occur in training data.

slot name	data	area	book day	book time	food	name	price range
book people	train	1.9	38.8	39.2	2.1	16.4	1.5
	dev	1.9	38.9	38.9	1.9	16.3	2.2
	test	2.7	36.9	37.7	1.6	18.7	2.4

Table 2: Co-occurrence distribution(%) of *book people* slot with other slots in *restaurant* domain within the same user utterance. It rarely co-occurs with particulars slots (e.g., *price range*), which hinders the evaluation of DST models on realistic user utterances such as "*I want to book an expensive restaurant for 8 people.*"

out set and assume direct access to evaluated models' gradients or outputs. Adversarial examples generated by these methods are often unnatural or obtained to hurt target models deliberately. It is imperative to emphasize here that both our primary goal and approach significantly differ from the previous line of work: (i) Our goal is to evaluate DST models beyond held-out accuracy, (ii) We leverage turn-level structured meaning representation (belief state) along with its dialogue history as conditions to generate user response without relying on the original user utterance, (iii) Our approach is entirely model-agnostic, assuming no access to evaluated DST models, (iv) Perhaps most importantly, we aim to produce novel but realistic and meaningful conversation scenarios rather than intentionally adversarial ones.

We propose *controllable counterfactuals* (CoCo) as a principled, model-agnostic approach to generate novel scenarios beyond the held-out conversations. Our approach is inspired by the combination of two natural questions: how would DST systems react to (1) unseen slot values and (2) rare but realistic slot combinations? CoCo first encapsulates these two aspects under a unified concept called a *counterfactual goal* obtained by a stochastic policy of dropping and adding slots to the original turn-level belief state followed by replacing slot values. In the second step, CoCo conditions on the dialogue history and the counterfactual goal to generate a *counterfactual conversation*. We cast the actual utterance generation as a conditional language modeling objective. This formulation allows us to plug-in a pretrained encoder-decoder architecture (Raffel et al., 2020) as the backbone that powers the counterfactual conversation generation. We also propose a strategy to filter utterances that fail to reflect the counterfactual goal exactly. We consider *value substitution* (VS), as presented in Figure 1, as a special CoCo case that only replaces the slot values in the original utterance without adding or dropping slots. When we use VS as a fall-back strategy for CoCo (i.e., apply VS when CoCo fails to generate valid user responses after filtering), we call it CoCo+.

Evaluating three strong DST models (Wu et al., 2019; Heck et al., 2020; Hosseini-Asl et al., 2020) with our proposed controllable counterfactuals generated by CoCo and CoCo+ shows that the performance of each significantly drops (up to 30.8%) compared to their joint goal accuracy on the original MultiWOZ held-out evaluation set. On the other hand, we find that these models are, in fact, quite robust to paraphrasing with back-translation, where their performance drops only 2%. Analyzing the effect of training data augmentation with CoCo+ shows that it consistently improves the robustness of the investigated DST models on counterfactual conversations generated by each of VS, CoCo and CoCo+. More interestingly, the same data augmentation strategy improves the joint goal accuracy of the best of these strong DST models by 1.3% on the original MultiWOZ evaluation set. Human evaluations show that CoCo-generated counterfactual conversations perfectly reflect the underlying user goal with more than 95% accuracy and are found to be quite close to original conversations in terms of their human-like scoring. This further proves our proposed approach's reliability and potential to be adopted as part of DST models' robustness evaluation. We plan to publicly release CoCo-generated and human-verified set of examples (MultiWOZ-CoCo) as additional evaluation set to be used by future research.

2 Related Work

Dialogue State Tracking. DST has been a core component in current state-of-the-art TOD systems. Traditional approaches usually rely on hand-crafted features or domain-specific lexicon (Henderson

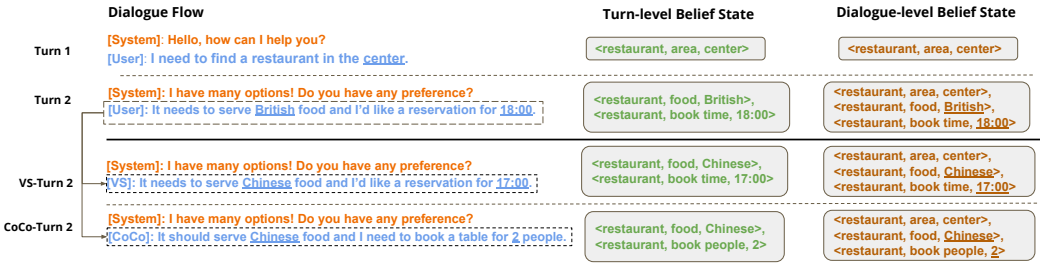

Figure 1: The upper left is a dialogue example between *user* and *system* with its turn-level and dialogue-level belief states on the upper right. The lower left are valid user utterance variations generated by VS and CoCo with their corresponding belief states derived from the original ones on the right.

et al., 2014; Wen et al., 2017) and require a predefined ontology, making them hard to extend to unseen value. To tackle this issue, various methods have been proposed. Gao et al. (2019) treats DST as a reading comprehension problem and predicts slot values with start and end positions in the dialogue context. Zhang et al. (2019a) proposes DS-DST, a dual-strategy model that predicts values in domains with a few possible candidates from classifiers and others from span extractors. Furthermore, Heck et al. (2020) proposes TripPy, a triple copy strategy model, which allows it to copy values from the context, previous turns' predictions and system informs.

An alternative to classification and span prediction is value generation. Wu et al. (2019) generates slot values with a pointer generator network See et al. (2017) without relying on fixed vocabularies and spans. (Hosseini-Asl et al., 2020) models DST as a conditional generation problem and directly finetunes GPT2 (Radford et al., 2019) on DST task and achieves state-of-the-art on the MultiWOZ.

Adversarial Example Generation. Adversarial example generation has been commonly studied in computer vision (Szegedy et al., 2014; Goodfellow et al., 2015). Recently, it has received growing attention in NLP domain as well. Papernot et al. (2016) finds adversarial examples in the embedding space, and then remapped them to the discrete space. Alzantot et al. (2018) proposes a population-based word replacing method and aims to generate fluent adversarial sentences. These methods often edit the original data greedily assuming access to the model's gradients or outputs besides querying the underlying model many times (Jin et al., 2019). Alternative line of work investigates generating adversarial examples in a model-agnostic way. Iyyer et al. (2018) proposes to generate adversarial paraphrases of original data with different syntactic structure. Jia and Liang (2017) automatically generates sentences with key word overlappings of questions in SQuAD (Rajpurkar et al., 2016) to distract computer systems without changing the correct answer or misleading humans.

Although different methods have been proposed to evaluate the robustness of NLP models, majority of the prior work in this line focus either on text classification, neural machine translation or reading comprehension problems. Perhaps the most similar existing works with ours are (Einolghozati et al., 2019; Cheng et al., 2019). Einolghozati et al. (2019) focuses on intent classification and slot tagging in TOD while Cheng et al. (2019) targets at synthetic competitive negotiation dialogues (Lewis et al., 2017) without DST component. In this work, however, we focus on evaluating a core component of state-of-the-art TOD, DST, on the widely used benchmark, MultiWOZ. To the best of our knowledge, ours is the first work to systematically evaluate the robustness of DST models.

3 Background

Multi-domain DST task definition. Let $X_t = \{(U_1^{\text{sys}}, U_1^{\text{usr}}), ..., (U_t^{\text{sys}}, U_t^{\text{usr}})\}$ denote a sequence of turns of a dialogue until the t-th turn, where U_i^{sys} and U_i^{usr} ($1 \leq i \leq t$) denote system and user utterance at the i-th turn, respectively. In multi-domain DST, each turn $(U_i^{\text{sys}}, U_i^{\text{usr}})$ talks about a specific domain (e.g., *hotel*), and a certain number of slots (e.g., *price range*) in that domain. We denote all N possible domain-slot pairs as $S = \{S_1, ...S_N\}$. The task is to track the value for each S_j ($1 \leq j \leq N$) over X_t (e.g., *hotel-price range*, *cheap*). Belief states can be considered at two granularities: turn-level (L_t) and dialog-level (B_t). L_t tracks the information introduced in the last turn while B_t tracks the accumulated state from the first turn to the last. As illustrated in the upper part of Figure 1, when the dialogue flow arrives at the second turn, B_2 becomes {(*restaurant-area*,

center), (*restaurant-food*, *British*), (*restaurant-book time*, *18:00*)}, while L_2 is {(*restaurant-food*, *British*), (*restaurant-book time*, *18:00*)}, essentially tracking the update to B_t by the last turn.

Problem definition. Given a tuple $< X_t, L_t, B_t >$, our goal is to generate a new user utterance $\hat{U}_t^{\text{usr}}$ to form a novel conversation scenario $\hat{X}_t = \{(U_1^{\text{sys}}, U_1^{\text{usr}}), ..., (U_t^{\text{sys}}, \hat{U}_t^{\text{usr}})\}$ by replacing the original user utterance U_t^{usr} with $\hat{U}_t^{\text{usr}}$. To preserve the coherence of dialogue flow, we cast the problem as generating an alternative user utterance $\hat{U}_t^{\text{usr}}$ conditioned on a modified $\hat{L}_t$ derived from original turn-level belief state L_t in a way that is consistent with the global belief state B_t. This formulation naturally allows for producing a new tuple $< \hat{X}_t, \hat{L}_t, \hat{B}_t >$ controllable by $\hat{L}_t$, where $\hat{B}_t$ is induced by B_t based on the difference between L_t and $\hat{L}_t$. As illustrated in the lower part of Figure 1, U_2^{usr} is replaced with the two alternative utterances that are natural and coherent with the dialogue history. We propose to use the resulting set of $< \hat{X}_t, \hat{L}_t, \hat{B}_t >$ to probe the DST models.

Paraphrase baseline with back-translation. Paraphrasing the original utterance U_t^{usr} is a natural way to generate $\hat{U}_t^{\text{usr}}$. With the availability of advanced neural machine translation (NMT) models, round-trip translation between two languages (i.e., back-translation (BT)) has become a widely used method to obtain paraphrases for downstream applications (Yu et al., 2018). We use publicly available pretrained *English→German* ($\log(g|e)$) and *German→English* ($\log(e|g)$) NMT models.[1] We translate U_t^{usr} from English to German with a beam size K, and then translate each of the K hypotheses back to English with the beam size K. Consequently, we generate K^2 paraphrase candidates of $\hat{U}_t^{\text{usr}}$ and then rank them according to their round-trip confidence score $\log(g|e) + \log(e|g)$. As paraphrases are expected to preserve the meaning of U_t^{usr}, we set $\hat{L}_t = L_t$ and $\hat{B}_t = B_t$.

4 CoCo

As illustrated in Figure 2, CoCo consists of three main pillars. We first train a conditional user utterance generation model $p_\theta(U_t^{\text{usr}}|U_t^{\text{sys}}, L_t)$ using original dialogues. Secondly, we modify L_t into a possibly arbitrary $\hat{L}_t$ by our counterfactual goal generator. Given $\hat{L}_t$ and U_t^{sys}, we sample $\hat{U}_t^{\text{usr}} \sim p_\theta(\hat{U}_t^{\text{usr}}|U_t^{\text{sys}}, \hat{L}_t)$ with beam search followed by two orthogonal filtering mechanisms to further eliminate user utterances that fail to reflect the counterfactual goal $\hat{L}_t$.

4.1 Value Substitution

A robust DST model should correctly reflect value changes in user utterances when tracking user's goal. However, slot-value combinations, e.g. (*restaurant-book time*, *18:00*), in evaluation sets are limited and even have significant overlaps with training data as shown in Table 1. To evaluate DST models with more diverse patterns, we propose a Value Substitution (VS) method to generate $\hat{U}_t^{\text{usr}}$. Specifically, for each value of S_j in L_t, if the value only appears in U_t^{usr} rather than U_t^{sys}, we allow it to be substituted. Otherwise, we keep it as is. This heuristic is based on the following three observations: (1) if the value comes from U_t^{sys}, e.g. TOD system's recommendation of restaurant food, changing it may make the dialogue flow less natural and coherent (2) if it never appears in the dialogue flow, e.g. *yes* of *hotel-parking*, changing it may cause belief state label errors (3) if it only appears in U_t^{usr}, it is expected that changing the value won't cause issues in (1) and (2).

For values that can be substituted, new values are sampled from a *Slot-Value Dictionary*, a predefined value set for each domain-slot. These new values are then used to update their counterparts in U_t^{usr}, L_t and B_t. We defer the details of slot-value dictionary to section 4.2. After the update, we get $\hat{U}_t^{\text{usr}}$, $\hat{L}_t$ and $\hat{B}_t$, and can use $< \hat{X}_t, \hat{L}_t, \hat{B}_t >$ to evaluate the performance of DST models. An example of how VS works is illustrated in the lower part of Figure 1. At the second turn, as *British* and *18:00* are in L_2 and only appear in U_2^{usr} rather than U_2^{sys}, we can replace them with *Chinese* and *17:00* that are sampled from a slot-value dictionary, respectively, to get $\hat{U}_2^{\text{usr}}$, $\hat{L}_2$ and $\hat{X}_2$ without interrupting the naturalness of the dialogue flow.

[1] https://pytorch.org/hub/pytorch_fairseq_translation

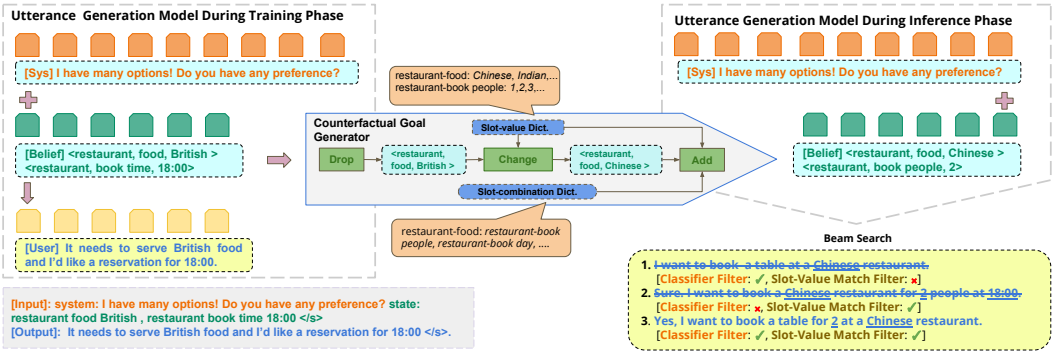

Figure 2: The overall pipeline of CoCo. The very left part represents the training phase of utterance generation model, where the concatenation of U_t^{sys} and L_t is processed by the encoder, which the decoder then conditions on to generate the user utterance U_t^{usr}. The input and output of this model is shown within the box at the lower-left. The right part depicts the inference phase, where the counterfactual goal generator first modifies the original belief L_t fed from the left part into a new one $\hat{L}_t$, which is then fed to the trained utterance generator along with the same conversation history to generate $\hat{U}_t^{\text{usr}}$ by beam search followed by filtering undesired utterances. Note that conversational turns in inference phase don't have to originate from training phase.

4.2 Controllable Counterfactual Generation

Back-translation (BT) and value-substitution (VS) provide controllability at different granularities. BT only provides syntactic variety while preserving the meaning, hence the belief state. VS can only replace the values of the existing slots in an utterance while still having to exactly retain all the slots. However, neither of them are able to explore conversations with even slightly modified set of slots. We propose a principled approach to unlock the capability of conversation generation that generalizes beyond just transformation of existing utterances. We cast it as a task of generating novel user utterances (U_t^{usr}) from a given conversation history (U_t^{sys}) and a turn-level user goal (L_t).

We propose to tackle this problem with a conditional generation model that utilizes a pre-trained encoder-decoder architecture (Raffel et al., 2020; Lewis et al., 2020) to approximate $p(U_t^{\text{usr}}|U_t^{\text{sys}}, L_t)$, where the concatenation of U_t^{sys} and L_t is used as input to the encoder and U_t^{usr} is set to be the target sequence to be generated by the decoder, as illustrated in the lower-left of Fig. 2. To learn this distribution, we factorize it by chain rule (Bengio et al., 2003) and train a neural network with parameters θ to minimize the aggregated negative log-likelihood $\mathcal{J}_{\text{gen}}$ over each dialogue turn tuple $(U_t^{\text{sys}}, L_t, U_t^{\text{usr}})$ where $U_t^{\text{usr}} = (U_{t,1}^{\text{usr}}, U_{t,2}^{\text{usr}}, \ldots, U_{t,n_t}^{\text{usr}})$ and $U_{t,k}^{\text{usr}}$ is its k-th token: [2]

$$p_\theta(U_t^{\text{usr}}|U_t^{\text{sys}}, L_t) = \prod_{k=1}^{n_t} p_\theta(U_{t,k}^{\text{usr}}|U_{t,<k}^{\text{usr}}, U_t^{\text{sys}}, L_t), \quad \mathcal{J}_{\text{gen}} = -\sum_{k=1}^{n_t} \log p_\theta(U_{t,k}^{\text{usr}}|U_{t,<k}^{\text{usr}}, U_t^{\text{sys}}, L_t) \quad (1)$$

Once the parameters θ of the goal-conditioned utterance generation model p_θ are learned from these tuples, it gives us the unique ability to generate novel conversation turns by plugging in an arbitrary but consistent counterfactual goal $\hat{L}_t$ derived from L_t. An example of how the counterfactual goal generator operates is shown in the middle part of Figure 2. The counterfactual goal generator has three components, namely operation, slot-value dictionary and slot-combination dictionary.

Operation decides to apply which combination of the following three meta-operations, namely *drop*, *change* and *add* on L_t. *Drop* is used to remove values from a non-empty slot in L_t. *Change* borrows the same operation in VS, to substitute existing values. *Add* allows us to add new domain-slot values into L_t, giving us the power of generating valid but more complicated $\hat{U}_t^{\text{usr}}$.

Slot-Value Dictionary has a pre-defined value set S_j^{val} for each S_j. Once *change* and/or *add* meta-operation is activated for S_j, counterfactual goal generator will randomly sample a value from S_j^{val}.

Slot-Combination Dictionary has a predefined domain-slot set S_j^{add} for each S_j. When *add* meta-operation is activated, counterfactual goal generator will sample a domain-slot from the intersection

[2] More training details can be found in Appendix D.1.

among all S_j^{add}, where S_j has non-empty values within L_t. Once a new domains-lot is sampled, its value will then be sampled from its corresponding value set as defined in slot-value dictionary.

Given L_t, the counterfactual goal generator first takes L_t as its input, and sequentially applies *drop*, *change* and *add* to output $\hat{L}_t$. Given $\hat{L}_t$ and U_t^{sys}, we can sample $\hat{U}_t^{\text{usr}} \sim p_\theta(\hat{U}_t^{\text{usr}}|U_t^{\text{sys}}, \hat{L}_t)$ with beam search. We use a rule-based method to get $\hat{B}_t$ of $\hat{X}_t$ following (Chao and Lane, 2019). Given B_{t-1} and $\hat{L}_t$, we update the domain-slot in B_{t-1} if its value in $\hat{L}_t$ is not *none*. Otherwise, we keep its value as it is in B_{t-1}. After the update, we use $\hat{B}_t$ as the dialogue-level label of $\hat{X}_t$.

4.3 Filtering

We have presented methods to generate $\hat{U}_t^{\text{usr}}$, but how do we make sure that the generated utterance correctly reflects the user goal represented by $\hat{L}_t$? To motivate our methods, we take an example generated by beam search located at the lower right of Figure 2 for illustration. In this example, the first hypothesis doesn't include value *2* for *restaurant-book people* that is within $\hat{L}_t$. On the contrary, the second hypothesis includes a value *18:00* for *restaurant-book time* that is not part of $\hat{L}_t$. We call these two phenomenons *de-generation* and *over-generation*, respectively. Filtering candidates with these issues is thus an important step to make sure (U_t^{sys}, $\hat{U}_t^{\text{usr}}$) perfectly expresses the user goals in $\hat{L}_t$. We propose two filtering methods, namely *slot-value match filter* and *classifier filter*, to alleviate *de-generation* and *over-generation* issues, respectively.

Slot-Value Match Filter. To tackle with *de-generation* issue, we choose a subset of values in $\hat{L}_t$ (values that should only appear in $\hat{U}_t^{\text{usr}}$ rather than U_t^{sys}) to eliminate candidates that fail to contain all the values in the subset.[3] In Fig. 2, the first hypothesis from the beam search output will be eliminated by this filter because it does not include the value *2* for *restaurant-book people* in $\hat{L}_t$.

Classifier Filter. As shown in Table 2, the slot *restaurant-book people* frequently appears together with *restaurant-book time* in the data we use to train our generation model $p_\theta(\hat{U}_t^{\text{usr}}|U_t^{\text{sys}}, \hat{L}_t)$. The imbalanced training data may cause that during inference, although $\hat{L}_t$ may not include *restaurant-book time*, the model may still generate a user utterance $\hat{U}_t^{\text{usr}}$ containing information about this slot. To deal with this *over-generation* problem, we propose to use a N-way multi-label classifier to eliminate such candidates. We employ BERT-base (Devlin et al., 2019) as its backbone:

$$H_t^{\texttt{CLS}} = \texttt{BERT}([\texttt{CLS}] \oplus [X_{t-1}] \oplus [\texttt{SEP}] \oplus [U_t^{\text{sys}}] \oplus [U_t^{\text{usr}}]) \in \mathbb{R}^{d_{\text{emb}}} \tag{2}$$

where $H_t^{\texttt{CLS}} \in \mathbb{R}^{d_{\text{emb}}}$ is the representations of `CLS` token of BERT with dimension d_{emb}. We then feed $H_t^{\texttt{CLS}}$ into a linear projection layer followed by `Sigmoid` function:

$$P = \texttt{Sigmoid}(W(H_t^{\texttt{CLS}})) \in \mathbb{R}^N, \;\; \mathcal{J}_{\text{cls}} = -\frac{1}{N}\sum_{j=1}^{N}(Y_j \cdot \log P_j + (1 - Y_j) \cdot \log(1 - P_j)) \tag{3}$$

where $W \in \mathbb{R}^{N \times d_{\text{emb}}}$ is the trainable weight of the linear projection layer and P_j is probability that slot S_j appears at t-th turn of X_t with Y_j as its label. The classifier is trained with $\mathcal{J}_{\text{cls}}$, i.e. the mean binary cross entropy loss of every slot S_j and achieves a precision of 92.3% and a recall of 93.5% on the development set [4]. During inference, the classifier takes $\hat{X}_t$ as input and predicts whether a slot S_i appears at t-th turn or not with threshold 0.5. We use this filter to eliminate generated candidates for which the classifier predicts at least one slot S_j mentioned in ($U_t^{\text{sys}}, \hat{U}_t^{\text{usr}}$) while $S_j \notin \hat{L}_t$. In Fig. 2, our classifier filter eliminates the second hypothesis from the output of beam search because $\hat{L}_t$ does not contain the slot *restaurant-book time* while it is mentioned in the generated utterance.

5 Experiments

5.1 Experimental Setup

We consider three strong multi-domain DST models to evaluate the effect of CoCo-generated counterfactual conversations in several scenarios. TRADE (Wu et al., 2019) builds upon pointer gener-

[3] For *hotel-parking* and *hotel-internet*, we use *parking* and *wifi* as their corresponding values for filtering.

[4] We defer further details of the classifier training to Appendix D.2.

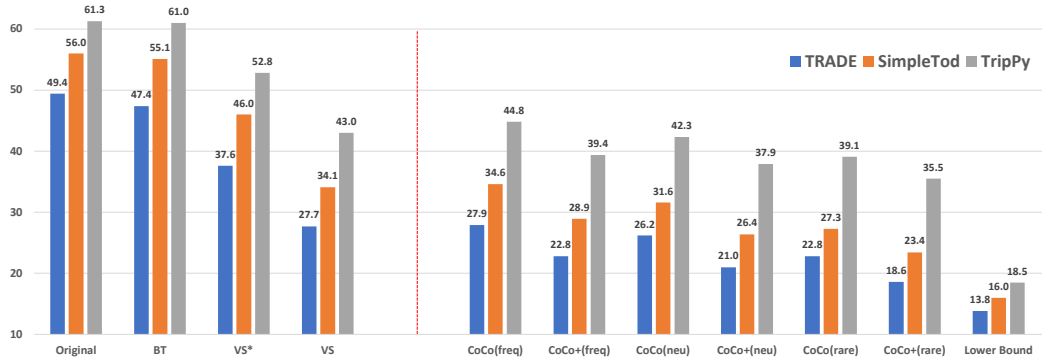

Figure 3: Joint goal accuracy (%) across different methods. "Original" refers to the results on the original held-out test set. * denotes results obtained from in-domain unseen slot-value dictionary (*I*). VS, CoCo and CoCo+ results use out-of-domain slot-value dictionary (*O*). For brevity, we omit CoCo and CoCo+ results using in-domain slot-value dictionary. See Appendix C for the full results. *freq*, *neu*, and *rare* indicate which slot-combination dictionary is used. Lower bound refers to the percentage of correct predictions on turns with empty turn-level belief state over original held-out test set.

ator network and contains a slot classification gate and a state generator module to generate states. TripPy (Heck et al., 2020) introduces a classification gate and a triple copy module, allowing the model to copy values either from the conversation context or previous turns' predictions or system informs. SimpleTod (Hosseini-Asl et al., 2020) models DST as a conditional generation problem with conversation history as its condition and belief state as its target and finetunes on GPT2.

Evaluation. We train each of these three models following their publicly released implementations on the standard train/dev/test split of MultiWOZ 2.1 (Eric et al., 2019) from scratch. We use the joint goal accuracy to evaluate the performance of DST models. It measures the model accuracy by comparing the predicted belief state with the ground-truth, where the prediction is marked correct if and only if the set of (*domain-slot*, value) pairs in the model output exactly matches the oracle one.

Slot-Value Dictionary. We carefully design two sets of slot-value dictionaries to capture the effect of unseen slot values from two perspectives, namely *in-domain* (*I*) and *out-of-domain* (*O*). *I* is a dictionary that maps each slot to a set of values that appear in MultiWOZ test set, but not in the training set.[5] On the other hand, we construct *O* using external values (e.g., hotel names from Wikipedia) that fall completely outside of the MultiWOZ data for the slots (e.g., *hotel-name*, *restaurant-name*, etc.). Otherwise, we follow a similar fall-back strategy for slots (e.g., *hotel-internet*) with no possible external values beyond the ones (e.g., *yes* and *no*) in the original data.

Slot-Combination Dictionary. As illustrated in Table 2, held-out evaluation set follows almost the same slot co-occurrence distribution with training data. This makes it difficult to estimate how well DST models would generalize on the valid conversation scenarios that just do not obey the same distribution. CoCo's flexibility at generating a conversation for an arbitrary turn-level belief state naturally allows us to seek an answer to this question. To this end, we design three slot combination dictionaries, namely *freq*, *neu* and *rare*. A slot combination dictionary directly controls how different slots can be combined while generating counterfactual goals. As suggested by their names, *freq* contains frequently co-occurring slot combinations (e.g., *book people* is combined only with *book day* and *book time* slots), while *rare* is the opposite of *freq* grouping rarely co-occurring slots together, and *neu* is more neutral allowing any meaningful combination within the same domain.[6]

5.2 Main Results

Before we begin reporting our results, it is important to note that several different post-processing strategies (e.g., output cleaning, employing semantic dictionary mapping) are used by different DST

[5]When this set is empty for a slot (e.g., *hotel-area*), we use the set of all possible values (e.g., *center*, *east*, *west*, *south*, *north*) for this slot from training data. Please see Appendix H for further details.

[6]Please see Appendix G for further details.

models. To make a fair comparison across different models, we follow the same post-processing strategy employed by SIMPLETOD evaluation script for TRADE and TRIPPY as well. We summarize our main results in Figure 3. While all three DST models are quite robust to back-translation (BT), their performance significantly drop on counterfactual conversations generated by each of VS, COCO and COCO+ compared to MultiWOZ held-out set accuracy (original).

Unseen Slot-Value Generalization. We analyze the effect of unseen slot values for the two dictionaries (*I* and *O*) introduced in the previous section compared to the original set of slot values that have large overlap with the training data. Results presented on the left part of Figure 3 show that the performance of DST models significantly drops up to 11.8% compared to original accuracy even on the simple counterfactuals generated by VS strategy using in-domain unseen slot-value dictionary (I). Furthermore, using out-of-domain slot-value dictionary (O) results in about 10% additional drop in accuracy consistently across the three models. Consistent and similar drop in accuracy suggests that TRADE, SIMPLETOD, and TRIPPY are almost equally susceptible to unseen slot values.

Generalization to Novel Scenarios. The right section of Figure 3 presents the main results in our effort to answer the central question we posed at the beginning of this paper. Based on these results, we see that state-of-the-art DST models are having a serious difficulty generalizing to novel scenarios generated by both COCO and COCO+ using three different slot combination strategies. The generalization difficulty become even more serious on counterfactuals generated by COCO+. As expected, the performance drop consistently increases as we start combining less and less frequently co-occurring slots (ranging from *freq* to *rare*) while generating our counterfactual goals. In particular, COCO+(rare) counterfactuals drops the accuracy of TRADE from 49.4% to 18.6%, pushing its performance very close to its lower bound of 13.8%. Even the performance of the most robust model (TRIPPY) among the three drops by up to 25.8%, concluding that held-out accuracy for state-of-the-art DST models may not sufficiently reflect their generalization capabilities.

Transferability Across Models. As highlighted before, a significant difference and advantage of our proposed approach lies in its model-agnostic nature, making it immediately applicable for evaluation of any DST model. As can be inferred from Figure 3, the effect of COCO-generated counterfactuals on the joint goal accuracy is quite consistent across all three DST models. This result empirically proves the transferability of COCO, strengthening its reliability and applicability to be generally employed as a robustness evaluation of DST models by the future research.

5.3 HUMAN EVALUATION

We next examine the quality of our generated data from two perspectives: "human likeliness" and "turn-level belief state correctness". The human likeliness evaluates whether a user utterance is fluent and consistent with its dialog context. The turn-level belief state correctness evaluates whether (U_t^{sys}, $\hat{U}_t^{\text{usr}}$) exactly expresses goals in $\hat{L}_t$. Both metrics are based on binary evaluation. We randomly sample 100 turns in the original test data and their corresponding CoCo-generated ones. For the COCO-generated data, we have two different settings to examine its quality. The first is to use its original turn-level belief state to generate user utterance. We ask three individuals with proficient English and advanced NLP background to conduct the evaluation for original human response both in MultiWoZ and CoCo-generated counterpart. We use majority voting to determine the final human likeness and turn-level belief state correctness.

	Human likeliness	Correctness
Human	87%	85%
COCO(ori)	90%	91%
COCO(freq)	90%	99%
COCO(neu)	79%	98%
COCO(rare)	82%	96%

Table 3: Human evaluation.

Table 3 shows the results. We can see that the human evaluators could not distinguish between the MultiWOZ's human utterance and our generated utterance. Furthermore, COCO(ori) generated slightly more "correct" responses than the original utterances in MultiWoZ 2.1. A presumable reason is that annotation errors exist in MultiWoZ 2.1, while our COCO are trained on recently released cleaner MultiWoZ 2.2, making generated data have higher quality. The second setting is to evaluate COCO(freq)-, COCO(neu)- and COCO(rare)-generated data, as they hurt the DST models' accuracy significantly as show in Figure 3, and we need to verify the quality of the generated utterances. All three variants of the COCO-generated conversations consistently outperform human response in terms of the turn-level belief state correctness. Although COCO(neu) and COCO(rare) are slightly less human-like than the original human response, COCO(freq)-generated utterances are found to

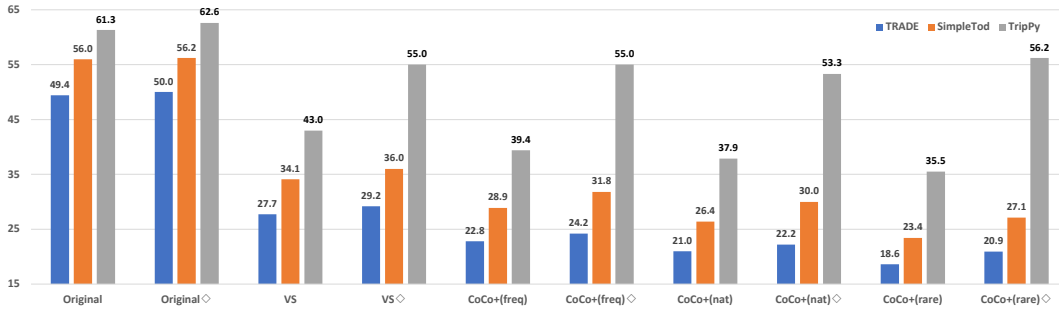

Figure 4: Comparison of retrained DST models (indicated by ◇) on CoCo+(rare)-augmented training data with their counterparts trained on original MultiWOZ train split. Original, VS and various CoCo+ versions share the same meaning as in Figure 3.

be difficult to distinguish from the original human utterances. These results demonstrate the effectiveness of our proposed approach in generating not only high-fidelity but also human-like user utterances, proving its potential to be adopted as part of robustness evaluation of DST models.

5.4 Analysis of CoCo+ as Data Augmentation Defense

So far, we have focused on the generalization capability of DST models on CoCo-generated conversations using different slot-value and slot-combination dictionaries. We have observed that all three DST models are consistently most susceptible to conversations generated by CoCo+(rare) strategy. Instead, we now seek to answer the following question: *Would using conversations generated by* CoCo+*(rare) to augment the training data help these DST models in better generalizing to unseen slot values and/or novel scenarios?* Towards exploring this direction in a principled way, we design a new slot value dictionary (*train-O*) similar to out-of-domain unseen slot-value dictionary (*O*). For a fair comparison, we make sure that the slot values in *train-O* (please refer to Appendix H for the complete dictionary) do not overlap with the one (*O*) used for generating test conversations.

We first retrain each DST model on the MultiWOZ training split augmented with CoCo+(rare)-generated conversations using *train-O* slot-value dictionary. Retrained DST models are then evaluated on original test set as well as on the counterfactual test sets generated by VS and various versions of CoCo+. Results presented in Figure 4 shows that retraining on the CoCo+(rare)-augmented training data improves the robustness of all three DST models across the board. Most notably, this data augmentation strategy rebounds the performance of TripPy on CoCo+(rare)-generated test set from 35.5% to 56.2%, significantly closing the gap with its performance (61.3%) on the original held-out test set. We also observe that retrained DST models obtains an improved joint goal accuracy on the original MultiWOZ test set compared to their counterparts trained only on the original MultiWOZ train split, further validating the quality of CoCo-generated conversations. Finally, we would like to highlight that retrained TripPy model achieves 62.6% joint goal accuracy, improving the previous state-of-the-art by 1.3%. We leave the exploration of how to fully harness CoCo as a data augmentation approach as future work.

6 Conclusion

We propose a principled, model-agnostic approach (CoCo) to evaluate dialogue state trackers beyond the held-out evaluation set. We show that state-of-the-art DST models' performances significantly drop when evaluated on the CoCo-generated conversations. Human evaluations validate that they have high-fidelity and are human-like. Hence, we conclude that these strong DST models have difficulty in generalizing to novel scenarios with unseen slot values and rare slot combinations, confirming the limitations of relying only on the held-out accuracy. When explored as a data augmentation method, CoCo consistently improves state-of-the-art DST models not only on the CoCo-generated evaluation set but also on the original test set. This further proves the benefit and potential of our approach to be adopted as part of a more comprehensive evaluation of DST models.

REFERENCES

M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1316. URL https://www.aclweb.org/anthology/D18-1316.

Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3(null):1137–1155, Mar. 2003. ISSN 1532-4435.

P. Budzianowski, T.-H. Wen, B.-H. Tseng, I. Casanueva, S. Ultes, O. Ramadan, and M. Gašić. 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*, 2018.

B. Byrne, K. Krishnamoorthi, C. Sankar, A. Neelakantan, B. Goodrich, D. Duckworth, S. Yavuz, A. Dubey, K.-Y. Kim, and A. Cedilnik. Taskmaster-1: Toward a realistic and diverse dialog dataset. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

G. Chao and I. Lane. Bert-dst: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. In *INTERSPEECH*, 2019.

M. Cheng, W. Wei, and C.-J. Hsieh. Evaluating and enhancing the robustness of dialogue systems: A case study on a negotiation agent. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3325–3335, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1336. URL https://www.aclweb.org/anthology/N19-1336.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

A. Einolghozati, S. Gupta, M. Mohit, and R. Shah. Improving robustness of task oriented dialog systems. *ArXiv*, abs/1911.05153, 2019.

M. Eric, R. Goel, S. Paul, A. Kumar, A. Sethi, A. K. Goyal, P. Ku, S. Agarwal, S. Gao, and D. Z. Hakkani-Tür. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *ArXiv*, abs/1907.01669, 2019.

S. Gao, A. Sethi, S. Agarwal, T. Chung, and D. Z. Hakkani-Tür. Dialog state tracking: A neural reading comprehension approach. *ArXiv*, abs/1908.01946, 2019.

I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6572.

M. Heck, C. van Niekerk, N. Lubis, C. Geishauser, H.-C. Lin, M. Moresi, and M. Gasic. 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*, pages 35–44, 1st virtual meeting, July 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.sigdial-1.4.

M. Henderson, B. Thomson, and S. Young. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299, Philadelphia, PA, U.S.A., June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4340. URL https://www.aclweb.org/anthology/W14-4340.

E. Hosseini-Asl, B. McCann, C.-S. Wu, S. Yavuz, and R. Socher. A simple language model for task-oriented dialogue, 2020.

M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1170. URL https://www.aclweb.org/anthology/N18-1170.

R. Jia and P. Liang. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1215. URL https://www.aclweb.org/anthology/D17-1215.

D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. Is bert really robust? natural language attack on text classification and entailment. *ArXiv*, abs/1907.11932, 2019.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

M. Lewis, D. Yarats, Y. Dauphin, D. Parikh, and D. Batra. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1259. URL https://www.aclweb.org/anthology/D17-1259.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ArXiv*, abs/1910.13461, 2020.

J. Li, M. Galley, C. Brockett, J. Gao, and W. Dolan. A diversity-promoting objective function for neural conversation models. *ArXiv*, abs/1510.03055, 2016.

B. Liu and I. Lane. End-to-end learning of task-oriented dialogs. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

A. Neelakantan, S. Yavuz, S. Narang, V. Prasad, B. Goodrich, D. Duckworth, C. Sankar, and X. Yan. Neural assistant: Joint action prediction, response generation, and latent knowledge reasoning. In *NeurIPS 2019 Converstional AI Workshop*, 2019.

N. Papernot, P. McDaniel, A. Swami, and R. E. Harang. Crafting adversarial input sequences for recurrent neural networks. *MILCOM 2016 - 2016 IEEE Military Communications Conference*, pages 49–54, 2016.

K. Patel, J. Fogarty, J. A. Landay, and B. Harrison. Investigating statistical machine learning as a tool for software development. In *CHI*, 2008.

B. Peng, C. Li, J. Li, S. Shayandeh, L. Liden, and J. Gao. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model, 2020.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

A. Radford, J. Wu, R. Child, L. David, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100, 000+ questions for machine comprehension of text. *ArXiv*, abs/1606.05250, 2016.

A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *arXiv preprint arXiv:1909.05855*, 2019.

M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020.

A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL https://www.aclweb.org/anthology/P17-1099.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL http://arxiv.org/abs/1312.6199.

J. Wang, Y. Zhang, T.-K. Kim, and Y. Gu. Modelling hierarchical structure between dialogue policy and natural language generator with option framework for task-oriented dialogue system. *ArXiv*, abs/2006.06814, 2020a.

K. Wang, J.-F. Tian, R. Wang, X. Quan, and J. Yu. Multi-domain dialogue acts and response co-generation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020b.

T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gašić, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain, Apr. 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/E17-1042.

C.-S. Wu, A. Madotto, E. Hosseini-Asl, C. Xiong, R. Socher, and P. Fung. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2019.

A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *6th International Conference on Learning Representations (ICLR)*, 2018.

J. Zhang, K. Hashimoto, C.-S. Wu, Y. Wan, P. S. Yu, R. Socher, and C. Xiong. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *ArXiv*, abs/1910.03544, 2019a.

Y. Zhang, Z. Ou, and Z. Yu. Task-oriented dialog systems that consider multiple appropriate responses under the same context, 2019b.

APPENDIX

A SLOT-LEVEL ANALYSIS

Closer Look at the Effect of CoCo+(rare) on TripPy. In Figure 5, we take a closer look at the robustness of TripPy through slot-level analysis across three major scenarios. Comparison of *blue* and *orange* lines reveals that counterfactuals generated by CoCo+(rare) consistently drops the performance of TripPy model (trained on the original MultiWOZ train split) across all the slots, significantly hurting the accuracy of all the slots in *train* domain along with *book day* slot for *hotel* domains. On the other hand, comparing *green* and *orange* lines clearly demonstrates the effectiveness of CoCo+(rare) as a data augmentation defense (see Section 5.4 for further details), assisting TripPy in recovering from most of the errors it made on CoCo+(rare) evaluation set. In fact, it rebounds the joint goal accuracy of TripPy from 35.5% to 56.2% as presented more quantitatively in Figure 4.

Figure 5: Slot-level accuracy analysis of TripPy. "Ori-TripPy-Clean" (blue) and "Ori-TripPy-CoCo+(rare)" (orange) denote TripPy (trained on original MultiWOZ training data) when evaluated against original test set and CoCo+(rare) generated test set, respectively. "Aug-TripPy-CoCo+(rare)" (green) indicates slot-level accuracy of TripPy when evaluated against test set generated by CoCo+(rare)

B ABLATION STUDY ON OPERATIONS

In Table 4, we present ablation results on three meta operations (i.e., *drop*, *change*, *add*) that are used to generate counterfactual goals. The result in the first row corresponds to the performance of three DST models on evaluation set generated by CoCo including all three meta operations along with the classifier filter. Each row analyzes the effects of the corresponding meta operation or classifier by removing it from full models. From Table 4, we observe that *add* operation hurts the performance of the three models the most. This may indicate that the investigated DST models are more vulnerable against user utterances including more rare slot combinations.

CoCo	TRADE	SIMPLETOD	TRIPPY
Full	26.2	31.6	42.3
-Drop	25.7	31.1	42.1
-Change	30.4	36.0	50.4
-Add	34.1	41.0	48.3
-Classifier	25.3	30.5	41.3

Table 4: Ablation study on the meta operations and classifier based filtering.

C FULL FIGURE FOR MAIN RESULT

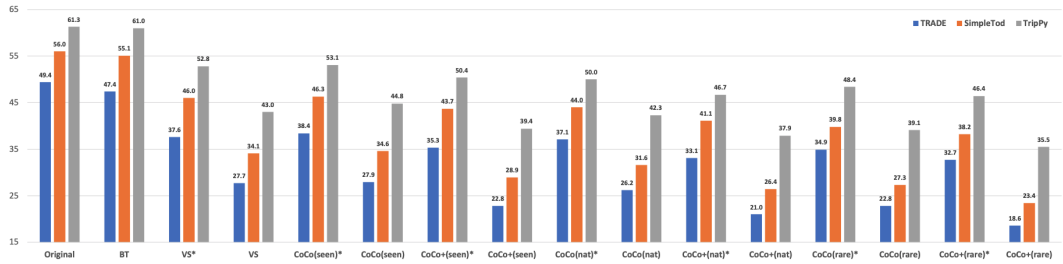

Figure 6: Joint goal accuracy (%) across different methods. "Original" refers to the results on the original held-out test set. * denotes results obtained from in-domain unseen slot-value dictionary (*I*) while other results use out-of-domain slot-value dictionary (*O*). *freq*, *neu*, and *rare* indicate which slot-combination dictionary is used.

D TRAINING DETAILS

D.1 TRAINING DETAILS OF CONTROLLABLE GENERATION MODEL

We instantiate $p_\theta(U_t^{usr}|U_t^{sys}, L_t)$ with `T5-small` (Raffel et al., 2020) and build it upon Huggingface `transformers` library. [7] During training, we use Adam optimizer (Kingma and Ba, 2015) with an initial learning rate $5e-5$ and set linear warmup to be 200 steps. The batch size is set to 36 and training epoch is set to be 10. The maximum sequence length of both encoder and decoder is set to be 100. We select the best checkpoint according to lowest perplexity on development set.

D.2 TRAINING DETAILS OF CLASSIFIER FILTER

We employ `BERT-base-uncased` as the backbone of our classifier filter. During training, we use Adam optimizer (Kingma and Ba, 2015) with an initial learning rate $2e-5$ and set linear warmup proportion to 0.1. We set maximum input length as 512, batch size as 108 and training epoch as 5. We select the best checkpoint based on the highest recall on development set.

[7] `https://huggingface.co/transformers`

E DIVERSITY EVALUATION

slot name	data	area	book day	book time	food	name	price range	entropy
book people	Ori-test	2.7	36.9	37.7	1.6	18.7	2.4	0.57
	CoCo-test	3.6	38.5	25.2	15.6	14.8	2.2	0.65

Table 5: Original test set (*Ori-test*) and CoCo generated test set (*CoCo-test*) co-occurrence distribution(%) comparisons of *book people* slot with other slots in *restaurant* domain within the same user utterance. The distribution entropy of *CoCo-test* is higher than its counterpart of *Ori-test* with an upper bound 0.78 corresponding to uniform distribution, meaning that *CoCo-test* is more diverse compared to *Ori-test* in terms of slot combinations.

Data	Distinct-1 ↑	Distinct-2 ↑	Distinct-3 ↑	Distinct-4 ↑
Ori-test	0.39	0.82	0.91	0.92
CoCo-test	0.38	0.82	0.92	0.92

Table 6: Language diversity comparisons between *Ori-test* and *CoCo-test*. We use unique n-gram ratio (Li et al., 2016) as our diversity metric. ↑ represents a higher number means more diversity. Overall, *CoCo-test* has similar diversity scores compared to *Ori-test*.

F GENERATED EXAMPLES BY COCO

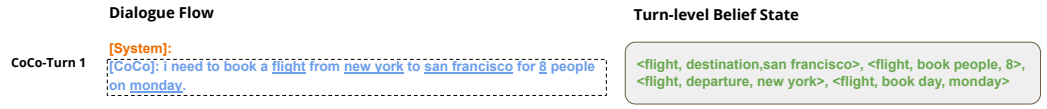

Figure 7: Zero-shot generation ability of CoCo on *flight* domain, which is never seen during training.

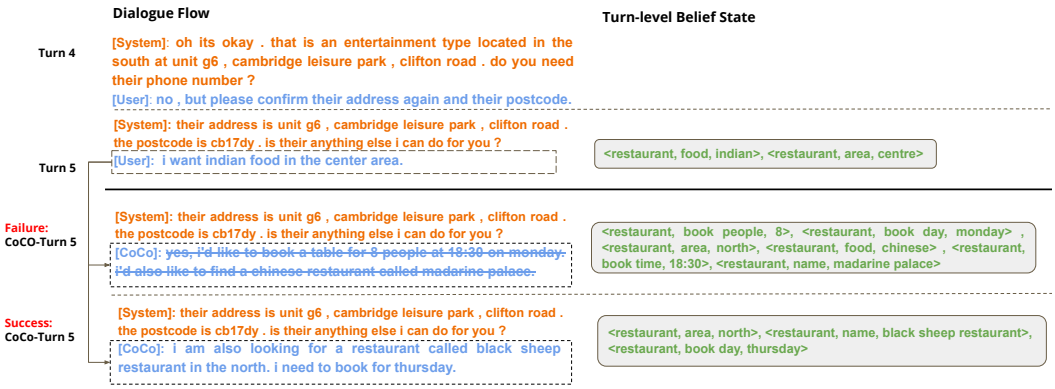

Figure 8: A success and failure example generated by CoCo with different slot-value combinations.

Figure 9: An example generated by CoCo with correct predictions by TRADE, SIMPLETOD and TRIPPY without retraining.

Figure 10: An example generated by CoCo with incorrect predictions by TRADE, SIMPLETOD and TRIPPY without retraining.

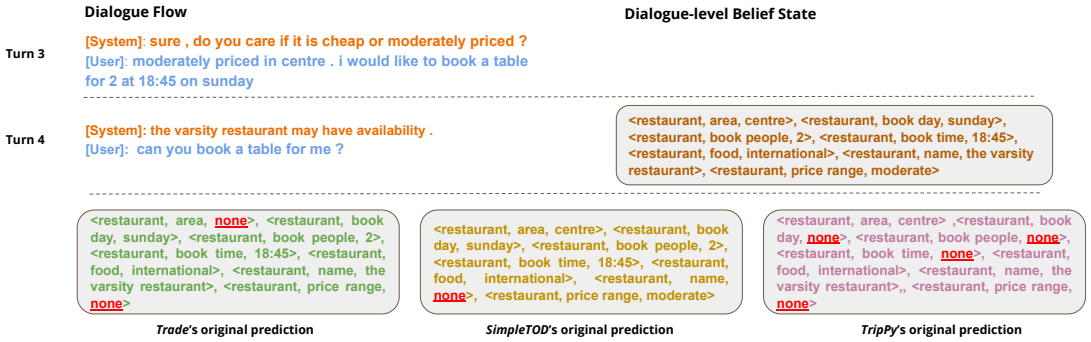

Figure 11: An example from original MultiWOZ test set, which is predicted incorrectly by original TRADE, SIMPLETOD and TRIPPY, is corrected by their retraining counterparts.

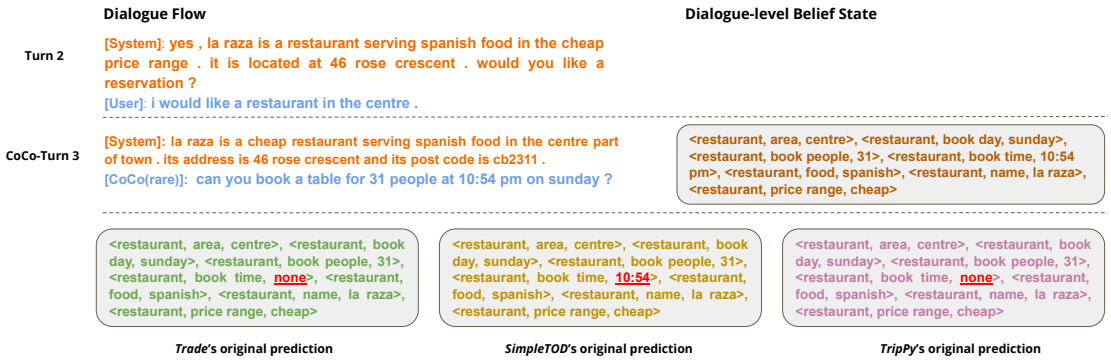

Figure 12: An example generated by CoCo(rare) evaluation set, which is predicted incorrectly by original TRADE, SIMPLETOD and TRIPPY, is corrected by their retraining counterparts.

G SLOT-COMBINATION DICTIONARY

Please find the different slot-combination dictionaries introduced in the main paper below.

domain-slot	*freq*
”hotel-internet”	[”hotel-area”,”hotel-parking”,”hotel-pricerange”,”hotel-stars”,”hotel-type”]
”hotel-type”	[”hotel-area”,”hotel-internet”,”hotel-parking”,”hotel-pricerange”,”hotel-stars”]
”hotel-parking”	[”hotel-area”,”hotel-internet”,”hotel-pricerange”,”hotel-stars”,”hotel-type”]
”hotel-pricerange”	[”hotel-area”,”hotel-internet”,”hotel-parking”,”hotel-stars”,”hotel-type”]
”hotel-book day”	[”hotel-book people”,”hotel-book stay”]
”hotel-book people”:	[”hotel-book day”,”hotel-book stay”]
”hotel-book stay”	[”hotel-book day”,”hotel-book people”]
”hotel-stars”	[”hotel-area”,”hotel-internet”,”hotel-parking”,”hotel-pricerange”,”hotel-type”]
”hotel-area”	[”hotel-internet”,”hotel-parking”,”hotel-pricerange”,”hotel-stars”,”hotel-type”]
”hotel-name”	[”hotel-book day”,”hotel-book people”,”hotel-book stay”]
”restaurant-area”	[”restaurant-food”,”restaurant-pricerange”]
”restaurant-food”	[”restaurant-area”,”restaurant-pricerange”]
”restaurant-pricerange”	[”restaurant-area”,”restaurant-food”]
”restaurant-name”	[”restaurant-book day”,”restaurant-book people”,”restaurant-book time”]
”restaurant-book day”	[”restaurant-book people”,”restaurant-book time”]
”restaurant-book people”	[”restaurant-book day”,”restaurant-book time”]
”restaurant-book time”	[”restaurant-book day”,”restaurant-book people”]
”taxi-arriveby”	[”taxi-leaveat”,”train-book people”]
”taxi-leaveat”	[”taxi-arriveby”,”train-book people”]
”taxi-departure”	[”taxi-destination”,”taxi-leaveat”,”taxi-arriveby”]
”taxi-destination”	[”taxi-departure”,”taxi-arriveby”,”taxi-leaveat”]
”train-arriveby”	[”train-day”,”train-leaveat”,”train-book people”]
”train-departure”	[”train-arriveby”,”train-leaveat”,”train-destination”,”train-day”,”train-book people”]
”train-destination”	[”train-arriveby”,”train-leaveat”,”train-departure”,”train-day”,”train-book people”]
”train-day”	[”train-arriveby”,”train-leaveat”,”train-book people”]
”train-leaveat”	[”train-day”]
”train-book people”	[]
”attraction-name”	[]
”attraction-area”	[”attraction-type”]
”attraction-type”	[”attraction-area”]

Table 7: Slot-combination dictionary for *freq* case.

slot-name	*neu*
'hotel-internet'	['hotel-book day','hotel-name','hotel-book stay','hotel-pricerange', 'hotel-stars','hotel-area','hotel-book people','hotel-type','hotel-parking']
'hotel-area'	['hotel-book day','hotel-name','hotel-book stay','hotel-pricerange', 'hotel-stars','hotel-book people','hotel-internet','hotel-type','hotel-parking']
'hotel-parking'	['hotel-book day','hotel-name','hotel-book stay','hotel-pricerange','hotel-stars', 'hotel-area','hotel-book people','hotel-internet','hotel-type']
'hotel-pricerange'	['hotel-book day','hotel-name','hotel-book stay','hotel-stars','hotel-area', 'hotel-book people','hotel-internet','hotel-type','hotel-parking']
'hotel-stars'	['hotel-book day','hotel-name','hotel-book stay','hotel-pricerange','hotel-area', 'hotel-book people','hotel-internet','hotel-type','hotel-parking']
'hotel-type'	['hotel-book day','hotel-book stay','hotel-pricerange','hotel-stars','hotel-area', 'hotel-book people','hotel-internet','hotel-parking']
'hotel-name'	['hotel-book day','hotel-book stay','hotel-pricerange','hotel-stars','hotel-area', 'hotel-book people','hotel-internet','hotel-parking']
'hotel-book day'	['hotel-name','hotel-book stay','hotel-pricerange','hotel-stars','hotel-area', 'hotel-book people','hotel-internet','hotel-type','hotel-parking']
'hotel-book people'	['hotel-book day','hotel-name','hotel-book stay','hotel-pricerange','hotel-stars', 'hotel-area','hotel-internet','hotel-type','hotel-parking']
'hotel-book stay'	['hotel-book day','hotel-name','hotel-pricerange','hotel-stars','hotel-area', 'hotel-book people','hotel-internet','hotel-type','hotel-parking']
'restaurant-area'	['restaurant-book day','restaurant-name','restaurant-food','restaurant-book people', 'restaurant-book time','restaurant-pricerange']
'restaurant-food'	['restaurant-book day','restaurant-book people','restaurant-book time', 'restaurant-area','restaurant-pricerange']
'restaurant-pricerange'	['restaurant-book day','restaurant-name','restaurant-food','restaurant-book people', 'restaurant-book time','restaurant-area']
'restaurant-name'	['restaurant-book day','restaurant-book people','restaurant-book time', 'restaurant-area','restaurant-pricerange']
'restaurant-book day'	['restaurant-name','restaurant-food','restaurant-book people','restaurant-book time', 'restaurant-area','restaurant-pricerange']
'restaurant-book people'	['restaurant-book day','restaurant-name','restaurant-food','restaurant-book time', 'restaurant-area','restaurant-pricerange']
'restaurant-book time'	['restaurant-book day','restaurant-name','restaurant-food','restaurant-book people', 'restaurant-area','restaurant-pricerange']
'taxi-departure'	['taxi-destination', 'taxi-leaveat', 'taxi-arriveby']
'taxi-destination'	['taxi-departure', 'taxi-leaveat', 'taxi-arriveby']
'taxi-leaveat'	['taxi-departure', 'taxi-destination', 'taxi-arriveby']
'taxi-arriveby'	['taxi-departure', 'taxi-destination', 'taxi-leaveat']
'train-arriveby'	['train-book people','train-day','train-leaveat','train-departure','train-destination']
'train-leaveat'	['train-book people','train-arriveby','train-day','train-departure','train-destination']
'train-departure'	['train-book people','train-arriveby','train-day','train-leaveat','train-destination']
'train-destination'	['train-book people','train-arriveby','train-day','train-leaveat','train-departure']
'train-day'	['train-book people','train-arriveby','train-leaveat','train-departure','train-destination']
'train-book people'	['train-arriveby','train-day','train-leaveat','train-departure','train-destination']
'attraction-name'	['attraction-area']
'attraction-area'	['attraction-type']
'attraction-type'	['attraction-area']

Table 8: Slot-combination dictionary for *neu* case.

slot-name	*rare*
'hotel-internet'	['hotel-book people','hotel-book day','hotel-name','hotel-book stay']
'hotel-area':	['hotel-book people','hotel-book day','hotel-name','hotel-book stay']
'hotel-parking'	['hotel-book people','hotel-book day','hotel-name','hotel-book stay']
'hotel-pricerange'	['hotel-book people','hotel-book day','hotel-name','hotel-book stay']
'hotel-stars'	['hotel-book people','hotel-book day','hotel-name','hotel-book stay']
'hotel-type'	['hotel-book people','hotel-book day','hotel-book stay']
'hotel-name'	['hotel-pricerange','hotel-stars','hotel-area','hotel-internet','hotel-parking']
'hotel-book day'	['hotel-name','hotel-pricerange','hotel-stars','hotel-area','hotel-internet', 'hotel-type','hotel-parking']
'hotel-book people'	['hotel-name','hotel-pricerange','hotel-stars','hotel-area','hotel-internet', 'hotel-type','hotel-parking']
'hotel-book stay'	['hotel-name','hotel-pricerange','hotel-stars','hotel-area','hotel-internet', 'hotel-type','hotel-parking']
'restaurant-area'	['restaurant-book day','restaurant-name','restaurant-book time', 'restaurant-book people']
'restaurant-food'	['restaurant-book day','restaurant-book time','restaurant-book people']
'restaurant-pricerange'	['restaurant-book day','restaurant-name','restaurant-book time', 'restaurant-book people']
'restaurant-name'	['restaurant-area','restaurant-pricerange']
'restaurant-book day'	['restaurant-name','restaurant-area','restaurant-food','restaurant-pricerange']
'restaurant-book people'	['restaurant-name','restaurant-area','restaurant-food','restaurant-pricerange']
'restaurant-book time'	['restaurant-name','restaurant-area','restaurant-food','restaurant-pricerange']
'taxi-departure'	[]
'taxi-destination'	[]
'taxi-leaveat'	['taxi-departure', 'taxi-destination']
'taxi-arriveby'	['taxi-departure', 'taxi-destination']
'train-arriveby'	['train-destination', 'train-departure']
'train-leaveat'	['train-destination','train-book people','train-arriveby','train-departure']
'train-departure'	[]
'train-destination'	[]
'train-day'	['train-destination', 'train-departure']
'train-book people'	['train-arriveby','train-departure','train-destination','train-day','train-leaveat']
'attraction-name'	['attraction-area']
'attraction-area'	['attraction-name']
'attraction-type'	[]

Table 9: Slot-combination dictionary for *rare* case.

H SLOT-VALUE DICTIONARY

Please find the different slot-value dictionaries introduced in the main paper below.

slot-name	train-O
”hotel-internet”	[’yes’]
”hotel-type”	[’hotel’, ’guesthouse’]
”hotel-parking”	[’yes’]
”hotel-pricerange”:	[’moderate’, ’cheap’, ’expensive’]
”hotel-book day”	[”march 11th”, ”march 12th”, ”march 13th”, ”march 14th”, ”march 15th”, ”march 16th”, ”march 17th”,”march 18th”, ”march 19th”, ”march 20th”]
”hotel-book people”	[”20”,”21”,”22”,”23”,”24”,”25”,”26”,”27”,”28”,”29”]
”hotel-book stay”	[”20”,”21”,”22”,”23”,”24”,”25”,”26”,”27”,”28”,”29”]
”hotel-area”	[’south’, ’north’, ’west’, ’east’, ’centre’]
”hotel-stars”	[’0’, ’1’, ’2’, ’3’, ’4’, ’5’]
”hotel-name”	[”moody moon”, ”four seasons hotel”, ”knights inn”, ”travelodge”, ”jack summer inn”, ”paradise point resort”]
”restaurant-area”	[’south’, ’north’, ’west’, ’east’, ’centre’]
”restaurant-food”	[’asian fusion’, ’burger’, ’pasta’, ’ramen’, ’taiwanese’]
”restaurant-pricerange”:	[’moderate’, ’cheap’, ’expensive’]
”restaurant-name”	[”buddha bowls”,”pizza my heart”,”pho bistro”, ”sushiya express”,”rockfire grill”,”itsuki restaurant”]
”restaurant-book day”	[”monday”,”tuesday”,”wednesday”,”thursday”,”friday”, ”saturday”,”sunday”]
”restaurant-book people”	[”20”,”21”,”22”,”23”,”24”,”25”,”26”,”27”,”28”,”29”]
”restaurant-book time”:	[”19:01”,”18:06”,”17:11”,”19:16”,”18:21”,”17:26”,”19:31”, ”18:36”,”17:41”,”19:46”,”18:51”,”17:56”, ”7:00 pm”, ”6:07 pm”,”5:12 pm”,”7:17 pm”,”6:17 pm”,”5:27 pm”, ”7:32 pm”,”6:37 pm”,”5:42 pm”,”7:47 pm”,”6:52 pm”, ”5:57 pm”, ”11:00 am”,”11:05 am”,”11:10 am”,”11:15 am”, ”11:20 am”,”11:25 am”,”11:30 am”,”11:35 am”,”11:40 am”, ”11:45 am”,”11:50 am”, ”11:55 am”]
”taxi-arriveby”	[”17:26”,”19:31”,”18:36”,”17:41”,”19:46”,”18:51”,”17:56”, ”7:00 pm”,”6:07 pm”,”5:12 pm”,”7:17 pm”,”6:17 pm”, ”5:27 pm”,”11:30 am”,”11:35 am”,”11:40 am”,”11:45 am”, ”11:50 am”,”11:55 am”]
”taxi-leaveat”:	[”19:01”,”18:06”,”17:11”,”19:16”,”18:21”,”7:32 pm”, ”6:37 pm”,”5:42 pm”,”7:47 pm”,”6:52 pm”, ”5:57 pm”,”11:00 am”,”11:05 am”,”11:10 am”, ”11:15 am”,”11:20 am”,”11:25 am”]
”taxi-departure”:	[”moody moon”, ”four seasons hotel”, ”knights inn”, ”travelodge”, ”jack summer inn”, ”paradise point resort”],
”taxi-destination”:	[”buddha bowls”,”pizza my heart”,”pho bistro”, ”sushiya express”,”rockfire grill”,”itsuki restaurant”]
”train-arriveby”:	[”17:26”,”19:31”,”18:36”,”17:41”,”19:46”,”18:51”, ”17:56”,”7:00 pm”,”6:07 pm”,”5:12 pm”,”7:17 pm”, ”6:17 pm”,”5:27 pm”,”11:30 am”,”11:35 am”,”11:40 am”, ”11:45 am”,”11:50 am”,”11:55 am”]
”train-leaveat”:	[”19:01”,”18:06”,”17:11”,”19:16”,”18:21”, ”7:32 pm”, ”6:37 pm”,”5:42 pm”,”7:47 pm”,”6:52 pm”,”5:57 pm”, ”11:00 am”,”11:05 am”,”11:10 am”,”11:15 am”,”11:20 am”,”11:25 am”]
”train-departure”	[”gilroy”,”san martin”,”morgan hill”,”blossom hill”, ”college park”,”santa clara”,”lawrence”,”sunnyvale”]
”train-destination”	[”mountain view”,”san antonio”,”palo alto”,”menlo park”, ”hayward park”,”san mateo”,”broadway”,”san bruno”]
”train-day”:	[”march 11th”, ”march 12th”, ”march 13th”, ”march 14th”, ”march 15th”, ”march 16th”, ”march 17th”,”march 18th”, ”march 19th”, ”march 20th”]
”train-book people”	[”20”,”21”,”22”,”23”,”24”,”25”,”26”,”27”,”28”,”29”]
”attraction-area”	[’south’, ’north’, ’west’, ’east’, ’centre’]
”attraction-name”	[”grand canyon”,”golden gate bridge”,”niagara falls”, ”kennedy space center”,”pike place market”,”las vegas strip”]
”attraction-type”	[’historical landmark’, ’aquaria’, ’beach’, ’castle’,’art gallery’]

Table 10: Slot value dictionary of train-O.

slot-name	I
”hotel-internet”	[’yes’]
”hotel-type”	[’hotel’, ’guesthouse’]
”hotel-parking”	[’yes’]
”hotel-pricerange”	[’moderate’, ’cheap’, ’expensive’]
”hotel-book day”	[’friday’, ’tuesday’, ’thursday’, ’saturday’, ’monday’, ’sunday’, ’wednesday’]
”hotel-book people”	[’1’, ’2’, ’3’, ’4’,’5’, ’6’, ’7’,’8’]
”hotel-book stay”	[’1’, ’2’, ’3’, ’4’,’5’, ’6’, ’7’,’8’]
”hotel-name”	[’alpha milton’, ’flinches bed and breakfast’, ’express holiday inn by cambridge’, ’wankworth house’, ’alexander b and b’, ’the gonville hotel’]
”hotel-stars”	[’0’, ’1’, ’3’, ’2’, ’4’, ’5’]
”hotel-area”	[’south’, ’east’, ’west’, ’north’, ’centre’]
”restaurant-area”	[’south’, ’east’, ’west’, ’north’, ’centre’]
”restaurant-food”	[’europeon’, ’brazliian’, ’weish’]
”restaurant-pricerange”	[’moderate’, ’cheap’, ’expensive’]
”restaurant-name”:	[’pizza hut in cherry’, ’the nirala’, ’barbakan’, ’the golden house’, ’michaelhouse’, ’bridge’, ’varsity restaurant’,’loch’, ’the peking’, ’charlie’, ’cambridge lodge’, ’maharajah tandoori’]
”restaurant-book day”	[’friday’, ’tuesday’, ’thursday’, ’saturday’, ’monday’, ’sunday’, ’wednesday’]
”restaurant-book people”	[’8’, ’6’, ’7’, ’1’, ’3’, ’2’, ’4’, ’5’]
”restaurant-book time”	[’14:40’, ’19:00’, ’15:15’, ’9:30’, ’7 pm’, ’11 am’, ’8:45’]
”taxi-arriveby”	[’08:30’, ’9:45’]
”taxi-leaveat”	[’7 pm’, ’3:00’]
”taxi-departure”	[’aylesbray lodge’, ’fitzbillies’, ’uno’, ’zizzi cambridge’, ’express by holiday inn’, ’great saint marys church’, ’county folk museum’,’riverboat’, ’bishops stortford’, ’caffee uno’, ’hong house’, ’gandhi’, ’cambridge arts’, ’the hotpot’, ’regency gallery’, ’saint johns chop shop house’] ,
”taxi-destination”	[’ashley’, ’all saints’, ”de luca cucina and bar’s”, ’the lensfield hotel’, ’oak bistro’, ’broxbourne’, ’sleeperz hotel’, ”saint catherine’s college”]
”train-arriveby”	[’4:45 pm’, ’18:35’, ’21:08’, ’19:54’, ’10:08’, ’13:06’, ’15:24’, ’07:08’, ’16:23’, ’8:56’, ’09:01’, ’10:23’, ’10:00 am’, ’16:44’, ’6:15’, ’06:01’, ’8:54’,’21:51’, ’16:07’, ’12:43’, ’20:08’, ’08:23’, ’12:56’, ’17:23’, ’11:32’, ’20:54’, ’20:06’, ’14:24’, ’18:10’, ’20:38’, ’16:06’, ’3:00’, ’22:06’, ’20:20’, ’17:51’,’19:52’, ’7:52’, ’07:44’, ’16:08’],
”train-leaveat”	[’13:36’, ’15:17’, ’14:21’, ’3:15 pm’, ’6:10 am’, ’14:40’, ’5:40’, ’13:40’, ’17:11’, ’13:50’, ’5:11’, ’11:17’, ’5:01’, ’13:24’, ’5:35’, ’07:00’, ’8:08’, ’7:40’, ’11:54’, ’12:06’, ’07:01’, ’18:09’, ’13:17’, ’21:45’, ’06:40’, ’01:44’, ’9:17’, ’20:21’, ’20:40’, ’08:11’, ’07:35’, ’14:19’, ’1 pm’, ’19:17’, ’19:48’, ’19:50’, ’10:36’, ’09:19’, ’19:35’, ’8:06’, ’05:29’, ’17:50’, ’15:16’, ’09:17’, ’7:35’, ’5:29’, ’17:16’, ’14:01’, ’10:21’, ’05:01’, ’15:39’, ’15:01’, ’10:11’, ’08:01’],
”train-departure”:	[’london liverpool street’, ’kings lynn’, ’norwich’, ’birmingham new street’, ’london kings cross’,’broxbourne’]
”train-destination”	[’bishops stortford’, ’cambridge’, ’ely’, ’stansted airport’, ’peterborough’, ’leicester’, ’stevenage’]
”train-day”	[’friday’, ’tuesday’, ’thursday’, ’monday’, ’saturday’, ’sunday’, ’wednesday’]
”train-book people”	[’9’]
”attraction-name”:	[’the cambridge arts theatre’, ’the churchill college’, ’the castle galleries’, ’cambridge’, ”saint catherine’s college”, ’street’, ’corn cambridge exchange’, ’fitzwilliam’, ’cafe jello museum’],
”attraction-area”:	[’south’, ’east’, ’west’, ’north’, ’centre’],
”attraction-type”	[’concerthall’, ’museum’, ’entertainment’, ’college’, ’multiple sports’, ’hiking’, ’architecture’, ’theatre’, ’cinema’, ’swimmingpool’, ’boat’, ’nightclub’, ’park’]

Table 11: Slot-value dictionary for I case.

slot-name	*O*
”hotel-internet”	[’yes’]
”hotel-type”	[’hotel’, ’guesthouse’]
”hotel-parking”	[’yes’]
”hotel-pricerange”	[’moderate’, ’cheap’, ’expensive’]
”hotel-book day”	[”april 11th”, ”april 12th”, ”april 13th”, ”april 14th”, ”april 15th”, ”april 16th”, ”april 17th”,”april 18th”, ”april 19th”, ”april 20th”]
”hotel-book people”	[”30”,”31”,”32”,”33”,”34”,”35”,”36”,”37”,”38”,”39”]
”hotel-book stay”	[”30”,”31”,”32”,”33”,”34”,”35”,”36”,”37”,”38”,”39”]
”hotel-area”	[’south’, ’east’, ’west’, ’north’, ’centre’]
”hotel-stars”	[’0’, ’1’, ’2’, ’3’, ’4’, ’5’]
”hotel-name”	[”white rock hotel”, ”jade bay resort”, ”grand hyatt”, ”hilton garden inn” ,”cottage motel”,”mandarin oriental”],
”restaurant-area”	[’south’, ’east’, ’west’, ’north’, ’centre’]
”restaurant-food”	[”sichuan”, ”fish”, ”noodle”, ”lobster”, ”burrito”, ”dumpling”, ”curry”,”taco”]
”restaurant-pricerange”	[’moderate’, ’cheap’, ’expensive’]
”restaurant-name”:	[”lure fish house”,”black sheep restaurant”,”palapa restaurant”, ”nikka ramen”, ”sun sushi”,”super cucas”]
”restaurant-book day”:	[”monday”,”tuesday”,”wednesday”,”thursday”,”friday”,”saturday”,”sunday”]
”restaurant-book people”	[”30”,”31”,”32”,”33”,”34”,”35”,”36”,”37”,”38”,”39”]
”restaurant-book time”	[”20:02”,”21:07”,”22:12”,”20:17”,”21:22”,”22:27”,”20:32”,”21:37”,”22:42”, ”20:47”,”21:52”,”22:57”,”8:00 pm”,”9:04 pm”,”10:09 pm”,”8:14 pm”, ”9:19 pm”,”10:24 pm”,”8:29 pm”,”9:34 pm”,”10:39 pm”,”8:44 pm”,”9:49 pm”, ”10:54 pm”,”10:00 am”,”10:06 am”,”10:11 am”,”10:16 am”,”10:21 am”,”10:26 am”, ”10:31 am”,”10:36 am”,”10:41 am”,”10:46 am”,”10:51 am”,”10:56 am”],
”taxi-arriveby”:	[”20:02”,”21:07”,”22:12”,”20:17”,”21:22”,”22:27”,”9:34 pm”,”10:39 pm”, ”8:44 pm”,”9:49 pm”,”10:54 pm”, ”10:00 am”,”10:06 am”,”10:11 am”, ”10:16 am”,”10:21 am”,”10:26 am”],
”taxi-leaveat”:	[”21:37”,”22:42”,”20:47”,”21:52”,”22:57”,”8:00 pm”,”9:04 pm”,”10:09 pm”, ”8:14 pm”,”9:19 pm”,”10:24 pm”,”8:29 pm”,”10:31 am”,”10:36 am”, ”10:41 am”, ”10:46 am”,”10:51 am”,”10:56 am”],
”taxi-departure”:	[”lure fish house”,”black sheep restaurant”,”palapa restaurant”, ”nikka ramen”, ”sun sushi”,”super cucas”],
”taxi-destination”:	[”white rock hotel”, ”jade bay resort”, ”grand hyatt”, ”hilton garden inn”, ”cottage motel”,”mandarin oriental”]
”train-departure”	[”northridge”,”camarillo”,”oxnard”,”morepark”,”simi valley”,”chatsworth”, ”van nuys”,”glendale”]
”train-destination”	[”norwalk”,”buena park”,”fullerton”,”santa ana”,”tustin”,”irvine”, ”san clemente”,”oceanside”],
”train-arriveby”:	[”20:02”,”21:07”,”22:12”,”20:17”,”21:22”,”22:27”, ”9:34 pm”,”10:39 pm”, ”8:44 pm”,”9:49 pm”,”10:54 pm”,”10:00 am”,”10:06 am”,”10:11 am”, ”10:16 am”,”10:21 am”,”10:26 am”],
”train-day”:	[”april 11th”, ”april 12th”, ”april 13th”, ”april 14th”, ”april 15th”, ”april 16th”, ”april 17th”,”april 18th”, ”april 19th”, ”april 20th”],
”train-leaveat”:	[”21:37”,”22:42”,”20:47”,”21:52”,”22:57”,”8:00 pm”,”9:04 pm”,”10:09 pm”, ”8:14 pm”,”9:19 pm”,”10:24 pm”,”8:29 pm”,”10:31 am”,”10:36 am”, ”10:41 am”,”10:46 am”,”10:51 am”,”10:56 am”],
”train-book people”:	[”30”,”31”,”32”,”33”,”34”,”35”,”36”,”37”,”38”,”39”]
”attraction-area”:	[’south’, ’east’, ’west’, ’north’, ’centre’]
”attraction-name”:	[”statue of liberty”,”empire state building”,”mount rushmore”, ”brooklyn bridge”,”lincoln memorial”,”times square”],
”attraction-type”:	[”temple”, ”zoo”, ”library”, ”skyscraper”,”monument”]

Table 12: Slot-value dictionary for *O* case.