Know Thy Strengths: Comprehensive Dialogue State Tracking Diagnostics

Hyundong Cho[1,2*] **Chinnadhurai Sankar**[2] **Christopher Lin**[2] **Kaushik Ram Sadagopan**[2]
Shahin Shayandeh[2] **Asli Celikyilmaz**[2] **Jonathan May**[1] **Ahmad Beirami**[3*]
[1]University of Southern California Information Sciences Institute [2]Meta AI [3]Google Research
hd.justincho@gmail.com

Abstract

Recent works that revealed the vulnerability of dialogue state tracking (DST) models to distributional shifts have made holistic comparisons on robustness and qualitative analyses increasingly important for understanding their relative performance. We present our findings from standardized and comprehensive DST diagnoses, which have previously been sparse and uncoordinated, using our toolkit, CheckDST, a collection of robustness tests and failure mode analytics. We discover that different classes of DST models have clear strengths and weaknesses, where generation models are more promising for handling language variety while span-based classification models are more robust to unseen entities. Prompted by this discovery, we also compare checkpoints from the same model and find that the standard practice of selecting checkpoints using validation loss/accuracy is prone to overfitting and each model class has distinct patterns of failure. Lastly, we demonstrate how our diagnoses motivate a pre-finetuning procedure with non-dialogue data that offers comprehensive improvements to generation models by alleviating the impact of distributional shifts through transfer learning.

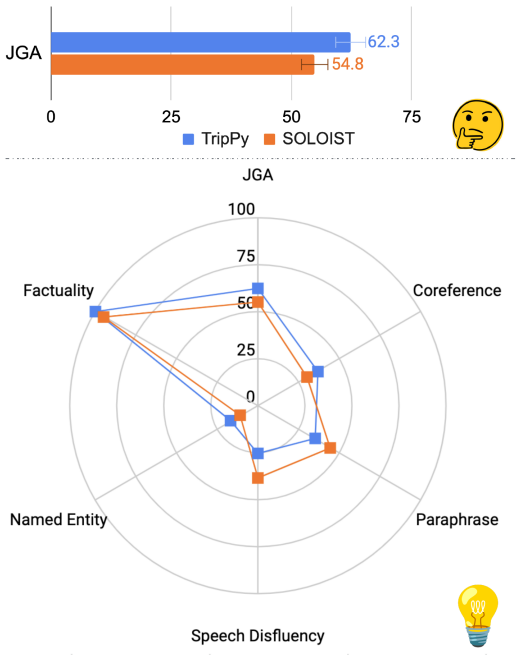

Figure 1: *Top:* performance differences on current benchmarks provide limited information on how DST models compare in robustness and various modes of failure. *Bottom:* CheckDST diagnoses paints a richer summary that highlights strengths and weaknesses through standardized and comprehensive comparisons of metrics that isolate various perturbations and failure modes. Higher is better for all metrics.

1 Introduction

A crucial skill for task-oriented dialogue models, which serve as backbones to modern digital assistants, is slot filling, formally known as dialogue state tracking (DST). It requires understanding the users' intents to populate slots in API queries that request information needed to fulfill their goals. To encourage the development of DST models, various large-scale datasets that double as benchmarks have been developed, such as MultiWOZ (Budzianowski et al., 2018), Taskmaster (Byrne et al., 2019), and the Schema-guided Dialogue (SGD) dataset (Rastogi et al., 2020).

As shown in the bar chart on the top side of Figure 1, recent state-of-the-art approaches in DST leaderboards such as MultiWOZ (Budzianowski et al., 2018) are compared to one another on a single metric that provides limited information and their modes of failure. Moreover, since these models demonstrate a sharp drop in joint goal accuracy (JGA), the average accuracy of correctly predicting all slots for a dialogue turn, when exposed with realistic distributional shifts (Li et al., 2020; Qian et al., 2021; Liu et al., 2021; Peng et al., 2021b), it is increasingly important to shift the focus from

[*] The work of HC and AB was done at Meta AI when HC was an intern. This work was done prior to JM joining Amazon.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 5374 - 5388

answering the question *"**Is** one DST model better than another?"* to *"**How** does one DST model compare to another?"* through robustness tests and qualitative analyses to understand relative performance. Yet these efforts have mostly been sparse and uncoordinated, precluding in-depth comparisons and opportunities to discover improvements that combine strengths from different models.

To address this gap, we first consolidate previous efforts in robustness testing and qualitative analyses to design our toolkit, CheckDST[1]. CheckDST facilitates comprehensive diagnosis by quantifying robustness with augmented test sets that represent various perturbations in isolation and measuring the frequency of common failure modes. For measuring robustness, we introduce *consistency* JGA (cJGA) to place more emphasis on prediction consistency and obviate the assumption that the perturbations should be more difficult than their corresponding original samples.

With a CheckDST diagnosis, we can quickly examine a rich comparison summary that highlights relative performance in multiple dimensions, as illustrated in the bottom of Figure 1. Evaluating a subset of models from two major classes of state-of-the-art models, span-based classification models and SimpleTOD-style generation models (henceforth denoted as classification models and generation models, respectively), on the MultiWOZ (Budzianowski et al., 2018) leaderboard, we demonstrate the value of our diagnosis through the revelation that robustness is not proportionate to higher JGA and that the model classes have clearly different strengths and weaknesses. In particular, while classification models attain higher JGA, generation models are significantly more robust to various perturbations except to swapped named entities. This disconnect between JGA and robustness found in inter-model comparisons naturally prompts comparisons between different checkpoints from the same model. By diagnosing multiple intermediate checkpoints with CheckDST, we show that the current standard practice of selecting a checkpoint according to performance on the validation set is prone to overfitting, regardless of model class.

Finally, we demonstrate how the findings from our diagnoses with CheckDST can help develop more robust DST models with strategies that alleviate the trade-off between JGA and robustness and mitigate the failure modes. We share our experiments with PrefineDST, building on the robustness of generation models via a pre-finetuning procedure with instruction prompts, similarly to the T0 model (Sanh et al., 2021), with non-dialogue tasks to acquire skills that should intuitively boost robustness. We show that comprehensive improvements that reduce the trade-off between JGA and robustness can be achieved without directly exposing the model to similar perturbations used in the test sets during training.

In summary, our contributions include:

- CheckDST, a toolkit for facilitating a standardized and comprehensive diagnosis of DST models;
- a rich empirical study enabled by CheckDST that discovers the disconnect between JGA and robustness, that major classes of DST models have clear strengths and weaknesses that is not portrayed in the performance difference on the original test set, and that current checkpoint selection criteria are prone to overfitting; and
- PrefineDST, a simple model motivated by CheckDST diagnosis that achieves a comprehensive improvement in robustness and JGA through a multitasking pre-finetuning step with instruction-prompts.

2 DST Diagnostics with CheckDST

To conduct standardized robustness tests and qualitative analyses, we first consolidate previous efforts to design *Checklist for Dialogue State Tracking* (CheckDST), a toolkit for quantifying robustness with a collection of augmented test sets and diagnostics that help identify commonly known failure modes. In this section, we introduce *consistent* JGA (cJGA) and summarize the perturbations and failure modes of interest that are shown in Table 1, as well as their corresponding metrics.

2.1 Measuring counterfactual robustness with Consistent JGA (cJGA)

By exposing DST models to perturbations, we want to quantify DST models' responses to valid perturbations that may be encountered at deployment in order to answer the question *"How do their robustness compare to other models?"* In this work, we define *robustness* as the "degree of performance consistency to realistic distribution shifts."

Based on this definition, robustness can be measured by comparing JGA to JGA on a perturbed test

[1]Our code and data are available at https://github.com/wise-east/CheckDST.

	Input Example	Example DST Prediction
Original	*I would like to leave from Cambridge.*	train departure cambridge
Paraphrase	*Please book me one ticket departing from Cambridge.*	train departure cambridge
Speech Disfluency	*I would like to uh leave from London no I meant Cambridge.*	train departure cambridge
Unseen Entity	*I would like to leave from mbadgceir.*	train departure mbadgceir
Coreference	*I need you to book the restaurant for Tuesday.* ... *I'm also looking for a train to London Kings Cross on the same day as the restaurant booking.*	... restaurant day tuesday train day tuesday ...
Hallucination	*I would like to leave from London.*	train departure cambridge train departure london

Table 1: An overview of perturbations and failure modes captured in CheckDST illustrated with simple examples. Blue segments indicate changes from the original input. Dialogue states in green are correct predictions, while those in red are incorrect.

set ($\widetilde{\text{JGA}}$), but this requires the perturbed test set to strictly contain corresponding samples that are more difficult for the model such that the performance drop represents a lack of robustness. There may be cases where certain perturbed samples are easier than the original, leading a model to achieve $\widetilde{\text{JGA}}$ similar to JGA, even though it makes many *inconsistent* predictions between the original and perturbed pairs. Therefore, we measure robustness through *consistent* JGA (cJGA) to obviate the difficulty requirement of the perturbations.

cJGA simply measures the frequency of the cases where the prediction is correct on both the original and perturbed samples. Given a DST model (with parameters θ), let function $f(z;\theta) \rightarrow \{0,1\}$ indicate whether the joint goal is satisfied on sample $z = (x, y)$, where x is the dialogue history and y is the reference belief state. Further, let $\widetilde{z} = (\widetilde{x}, \widetilde{y})$ denote a perturbed sample (e.g., with paraphrased dialog history). Then, we define cJGA for a sample set $[n] := \{1, \ldots, n\}$ as:

$$\text{cJGA} := \frac{1}{n} \sum_{i \in [n]} \mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1), \quad (1)$$

where $\mathbf{1}(\cdot)$ denotes the indicator function.

When labels are preserved, i.e. y and $\widetilde{y}$ are identical, cJGA is an adaptation of the CheckList *invariance* test, and if changes from y to $\widetilde{y}$ are mirrored in changes from x to $\widetilde{x}$, it is an adaptation of the Checklist *directional* test (Ribeiro et al., 2020). We also make the mathematical case for the usefulness of cJGA by proving that $\text{cJGA} \leq \min\{\text{JGA}, \widetilde{\text{JGA}}, 1 - |\text{JGA} - \widetilde{\text{JGA}}|\}$ in Lemma 1 (Appendix A), with equality only if the model performance is *consistent* on perturbed samples and original ones. This establishes that cJGA captures robustness beyond the JGA drop as it additionally captures the consistency of performance across the original and perturbed test set.

2.2 Perturbations

We select perturbations from those suggested by Liu et al. (2021); Peng et al. (2021b); Qian et al. (2021) and we elaborate them briefly here.[2]

Paraphrase. A robust DST model should make consistent predictions for utterances that have the same semantics, regardless of language variety. There is a wide spectrum for what is considered a paraphrase, including single word replacements with synonyms.[3] For CheckDST, we focus on more complex paraphrases with minimal word overlap.

In the context of DST, paraphrasing is defined as any change to the wording of utterances that preserves the dialogue belief states. Thus, Paraphrase Invariant cJGA (PI cJGA) measures whether a model can make correct slot predictions consistently for two semantically equivalent utterances.

Speech Disfluency. Many task-oriented dialogue applications are built around voice-based digital

[2]Note that the perturbations we select are relatively simple ones applied at the utterance level. There are previous works that create augmented test sets with entire new dialogues generated with counterfactual goals (Li et al., 2020) or insert additional dialogue turns (Peng et al., 2021b). While dialogue-level perturbation is interesting, its difficulty to decouple from other lower-level perturbations such as paraphrases and unseen named entities make it inadequate for isolated analysis and therefore we do not include it in CheckDST.

[3]According to Li et al. (2020), DST models only drop 2% in JGA for these kinds of simple paraphrases. However, when the paraphrases share only a few words with the original, the models demonstrate significant drops in JGA (Peng et al., 2021b; Liu et al., 2021), indicating that understanding paraphrases is still a challenge.

assistants. Therefore, a DST model's resilience to speech artifacts is a crucial criterion of a TOD model's success. Speech disfluencies are common speech artifacts that include the restart of requests mid-sentence, use of non-lexical vocables or filler words, and stammering and repetition that occur within the flow of otherwise fluent speech (Wang et al., 2020). As with PI cJGA, Speech Disfluency Invariant cJGA (SDI cJGA) measures how often a model maintains correct predictions with the presence of speech disfluencies.

Unseen Entities. DST models should be able to generalize performance to unseen entities, but it is a known problem that they can overfit to entities that appear frequently in the training set (Qian et al., 2021). To measure their robustness to entities not seen during training, we replace named entities in the dialogue belief states and conversations with scrambled entities. Named Entity Directional cJGA (NED cJGA) tracks how frequently a model correctly mirrors a change in the conversation to its prediction to obtain the right slot values.

2.3 Failure modes

Hallucination. Generation models, models that autoregressively generate a sequence of text from an open vocabulary, have become popular for task-oriented dialogue (Su et al., 2021; Peng et al., 2021a; Hosseini-Asl et al., 2020) following their success with various NLP tasks, but they are known to suffer from content *hallucination*, providing irrelevant entities memorized from training (Massarelli et al., 2020; Maynez et al., 2020). Therefore, we measure hallucination frequency as well in CheckDST with Factuality (F). F is equal to 1 if the predicted named entity is in the dialogue history and 0 otherwise. Since hallucination occurs even without any perturbations, CheckDST reports F on both the original test set and one used for NED cJGA ($F_{\texttt{orig}}$ and $F_{\texttt{swap}}$, respectively).

Coreference resolution. Long conversations with coreferences that span multiple turns are especially challenging, as shown by the performance improvement when coreference annotations are present (Quan et al., 2019; Han et al., 2020). As a proxy for measuring a model's ability to understand longer conversations and resolve coreferences to make correct predictions, we simply calculate the JGA for samples in the original test set that require coreference resolution and denote it as CorefJGA.

3 Experimental Setup

3.1 DST Benchmark

We use MultiWOZ (Budzianowski et al., 2018) as an example TOD dataset that we apply CheckDST to. It is an open-source dataset released with the Apache 2.0 license and we use it for research purposes only. We specifically use MultiWOZ 2.3 (Han et al., 2020), which includes corrections from MultiWOZ 2.1 (Eric et al., 2020) and coreference annotations, and use its original train/dev/test splits.

Perturbation Tools. For augmented test sets with paraphrases and speech disfluencies, we use the augmented test sets from LAUG (Liu et al., 2021)[4] and update the dialogue states with those from the official MultiWOZ 2.3 dataset to resolve inconsistencies we found. LAUG is an open-source augmentation toolkit that can be used for any task-oriented dialogue dataset that has dialogue acts and belief state annotations. To create a test set with unseen entities, we scramble the character order of named entity slot values (Huang et al., 2021) to create unseen entities, as shown in Table 1.[5]

3.2 Models

From the top-performing models reported on the MultiWOZ 2.0 repository and the MultiWOZ 2.3 repository, we implement a subset that has replicable code. We train all models for 10 epochs with the default hyperparameters reported in their corresponding reports, if applicable. We provide all other training details in Appendix B.3. For inter-model comparisons, we conduct CheckDST diagnosis on checkpoints with the best validation JGA.

Recent DST models that attain competitive results can largely be divided into two classes: classification models and generation models.

[4]Paraphrases are collected with a SC-GPT model (Peng et al., 2020) trained to generate the user response given its corresponding dialogue history and the dialogue act. The resulting paraphrases are heuristically filtered to ensure that slot values in the belief states are preserved. The degree of paraphrasing with LAUG is significant, replacing 74% of all words. Speech disfluencies are inserted according to their occurrence frequency in the Switchboard corpus (Godfrey et al., 1992). For both perturbations, more than 97% were considered appropriate by human evaluators. Refer to Liu et al. (2021) for further details.

[5]Instead, we can swap with more realistic entities not seen during training, such as those from Schema Guided Dialogue (SGD) (Rastogi et al., 2020; Qian et al., 2021). However, since some baseline models are pre-trained with SGD, we choose scrambled entities as the default for a fair comparison.

	Model	JGA	CorefJGA	PI cJGA	SDI cJGA	NED cJGA	F_{orig}	F_{swap}
SCLS	TripPy (2020)	$62.3_{0.2}$	$37.0_{0.7}$	$36.2_{0.1}$	$27.7_{0.5}$	$16.5_{0.4}$	100_{0}	100_{0}
	ConvBERT-DG (2020)	$62.0_{0.1}$	$36.6_{0.7}$	$35.9_{0.2}$	$29.5_{0.4}$	$16.2_{0.1}$	100_{0}	100_{0}
GEN	BART-DST (2020)	$52.5_{0.2}$	$25.5_{1.2}$	$43.0_{0.8}$	$36.5_{1.3}$	$9.8_{0.7}$	$94.8_{0.2}$	$75.4_{1.8}$
	SOLOIST (2021a)	$54.8_{0.4}$	$30.4_{1.1}$	$44.5_{0.6}$	$38.5_{0.5}$	$10.7_{0.4}$	$94.9_{0.1}$	$81.1_{1.6}$
	MUPPET-DST (2021)	$54.9_{0.4}$	$29.9_{1.9}$	$45.4_{0.4}$	$39.1_{0.7}$	$7.8_{1.4}$	$94.6_{0.1}$	$68.4_{3.8}$

Table 2: CheckDST diagnosis overview on the MultiWOZ 2.3 dataset clearly shows a significant divergence of robustness properties between the two model classes. SCLS is short for span-based classification models while GEN is short for generation models. All results are percentages aggregated over five runs with different seed values, presented as $\texttt{median}_{\texttt{se}}$, where se is short for standard error. x marks the best score for the column while x marks the worst. If there is an overlap between $\texttt{median} - \texttt{se}$ and $\texttt{median} + \texttt{se}$ with the best/worst score, the difference is considered statistically insignificant and all overlapping scores are highlighted.

Classification models. These models predict the required dialogue state operation and then specify the starting and ending index of slot values in the context to copy from or choose labels from a predefined ontology for those that are not directly in the context. The domains and their slot types are fixed, and predictions are made for every possible (domain, slot-type) pair using a classification layer.

(i) **TripPy** (Heck et al., 2020) is a model based on BERT (Devlin et al., 2019) that determines whether slot values can be copied from the current utterance, the previous system utterance, or the previous turns dialogue belief state.

(ii) **ConvBERT-DG** (Mehri et al., 2020) is TripPy with BERT replaced with ConvBERT-DG, a BERT model that is further pretrained on more than 70 million conversations of open-domain dialogue and then finetuned on the DialoGLUE benchmark.

Generation models. Generation models for DST predict belief states in the same way the underlying model generates text. They sequentially generate the domain, slot-type, and slot-value. Belief states are usually generated usually through greedy sampling on $P(x_t|x_{1:t-1}, C; \theta))$, where $X = \{x_1, x_2, ...x_t\}$ is the flattened text format of the belief state, e.g. domain slot-type slot-value, C is the dialogue context, and θ is the set of model parameters. Generation models are becoming increasingly popular as they can easily be expanded to perform end-to-end task-oriented dialogue by also generating the dialogue policy and responses after the belief states.

(i) **BART-DST** (Lewis et al., 2020) is a model that is trained in the SimpleTOD approach (Hosseini-Asl et al., 2020) to generate the dialogue belief states in domain slot-type slot-value format given a conversation.

(iii) **SOLOIST** (Peng et al., 2021a) is also similar to BART-DST, but it excludes dialogue act prediction during end-to-end training and adds a pre-training step with the SGD dataset. We use BART instead of GPT-2 in our experiments.

(iv) **MUPPET** (Aghajanyan et al., 2021) is a BART-DST model that is pre-finetuned on more than 50 natural language tasks. MUPPET adds auxiliary layers that take the representation of the final token in BART to perform classification tasks and does standard autoregressive language modeling for generation tasks.

4 CheckDST Diagnosis Results

4.1 Inter-model comparison

Each model class has distinct strengths and weaknesses. With our results in Table 2 we can immediately see a dramatic divergence of robustness properties between the classification and generation models.[6] Overall, generation models are much more robust to language variety, as captured by paraphrases and speech disfluencies, while classification models are significantly more robust to unseen entities. That the classification models' robustness is significantly lower against paraphrases and speech disfluencies, approximately 10% for both, is surprising, especially given that they attain a higher JGA and CorefJGA by a significant margin.

On the other hand, it is not surprising that classification models are more robust to unseen entities since their copying mechanism prevents hallucination altogether the distributional shift of correctly predicting spans is smaller than that of copying unseen entities by autoregressively generating from

[6]We assess the generation models the same way as we would for classification models by including "dontcare" slots. This makes an enormous difference. Otherwise, JGA and CheckDST metrics are boosted by more than 5%.

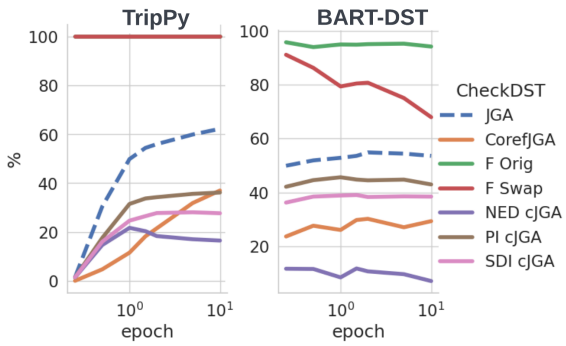

Figure 2: To examine how models overfit in different ways, we plot CheckDST for intermediate checkpoints. Most of the gains for all metrics except for JGA and CorefJGA are reached before the first few epochs and continue to steadily increase while others stagnate or deteriorate. Classification models show superior performance for perturbations to named entities due to their copy mechanism, but fail to reach similar levels of robustness as generation models for language variety. The x-axis uses a logarithmic scale to better visualize the progression in earlier stages of training.

an open vocabulary.[7] In addition, TripPy models have an advantage for coreference resolution as they observe shorter segments of text since older dialogue history is summarized in the previous dialogue belief state. SimpleTOD-style generation models, on the other hand, must examine the entire dialogue history at once. Among each class of models, generation models seem to more readily benefit from a pre-finetuning step as both SOLOIST and MUPPET-DST enjoy boosts on most metrics compared to SimpleTOD, which can be considered their baseline. However, the large amount of pre-training with open-domain dialogue for ConvBERT-DG seems to be only effective for becoming more robust to speech disfluencies, while other metrics have insignificant differences. These results encourage further exploring pre-finetuning generation models.

4.2 Intra-model comparison

The decoupling between JGA and robustness revealed by CheckDST between in our inter-model comparisons makes us wonder whether a similar problem exists between different checkpoints of the same model. We answer this curiosity by running CheckDST on checkpoints saved every 0.25 epochs until the second epoch and every epoch afterwards to observe how each model's performance on each metric in CheckDST fares across different checkpoints. Here, we share our findings from these intra-model comparisons.

Stopping training early is better for robustness. Similar to our findings in inter-model comparisons, we found that even among checkpoints of the same model, higher JGA can lower robustness. In Figure 2, with TripPy and BART-DST as representative examples, we observe similar trends for both models but with varying degrees. While JGA and CorefJGA continue to increase for both models even after the first two epochs, other metrics stagnate or deteriorate, an indication of overfitting.

To understand the nature of overfitting, we perform qualitative analyses to identify patterns of failure that become apparent over time. For each model, we inspect 100 predictions from the NED test set that were correctly predicted by an earlier checkpoint with the highest cJGA and incorrectly predicted by the final checkpoint selected as the best model.

Classification models give up on span prediction with more training. As training progresses, we observe that more than 80% of these cases for TripPy and ConvBERT-DG become none labels for slot values, indicating that the models are choosing to forgo span predictions over making incorrect span predictions. For example, the span for a scrambled entity for the restaurant name slot was correctly predicted to retrieve "*osdi jkal*" in the first epoch, reflecting the model's generalizability. However, the model with the best JGA predicted that a span for the same slot does not exist, indicating overfitting to entities that they have been repeatedly exposed to during training.

Generation models have difficulty correctly copying out-of-domain slot values. Generation models also struggle with unseen named entities, but their types of failure are more mixed. They either (*i*) fail to copy the slot values correctly and produce substrings of the correct slot value or (*ii*) determine that the slot value does not exist and generates nothing, which is equivalent to classification models predicting none. For the generation

[7] The advantages of classification models for unseen entities and hallucination needs to be taken with a grain of salt. The copy mechanism makes TripPy and ConvBERT vulnerable to even simple typos, and to overcome this issue the original implementation defines a mapping for typos that appear in the training set. Since the mapping makes not attempt to directly cover the test set, we keep the same practice, but this unfairly places these models at an advantage over generation models for a test set with similar entities as the training set since the generation models need to memorize typo fixes in order to predict the correct slot.

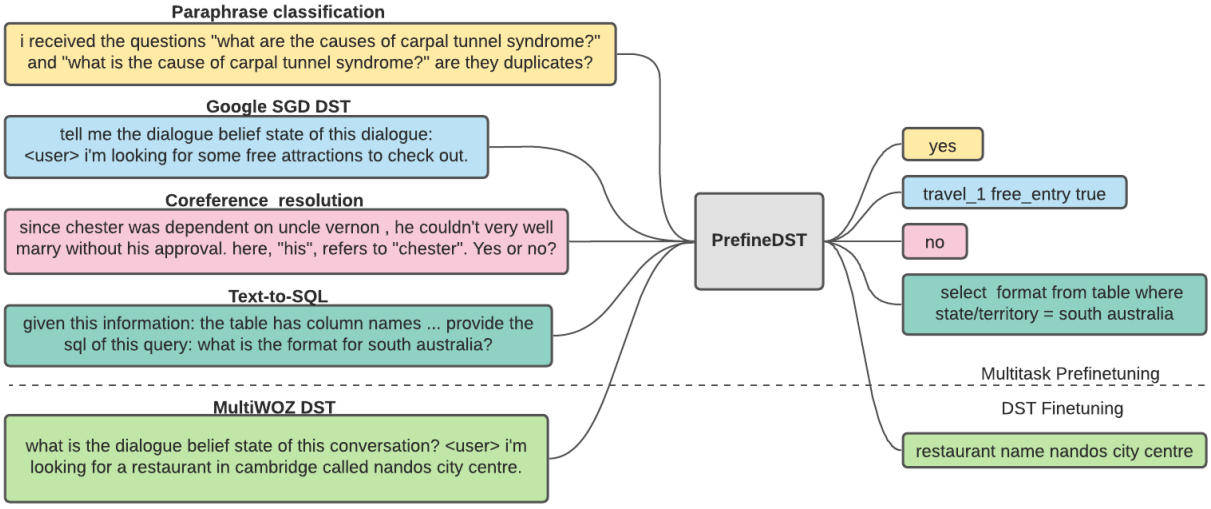

Figure 3: The pre-finetuning step (above dotted line) in PrefineDST is similar to T0 (Sanh et al., 2021), using randomly chosen instruction templates to format every task as a generation task. After pre-finetuning, we add a finetuning step for the downstream DST task (below dotted line).

models, we saw that about 40% of the drops are attributed to incorrect predictions while about 60% is attributed to empty predictions, which is consistent with the drop in F_{swap} and NED cJGA in Figure 2. For instance, in an earlier epoch, BART-DST correctly generates "`restaurant name osdi jkal`", but later instead produces "`restaurant name osjkal`". In fewer cases, the prediction becomes empty, similar to the behavior of classification models. From this observation, we hypothesize that making the copying process more robust for generation models will significantly boost robustness of generation-based DST models.

5 PrefineDST

The weaknesses exposed by CheckDST guide us toward approaches that can boost robustness without compromising JGA. Inspired by the strong results of massive multi-task learning on many NLP tasks in recent work, such as MUPPET (Aghajanyan et al., 2021), T0 (Sanh et al., 2021) and FLAN (Wei et al., 2021), and the ease of expanding a generation model to end-to-end task-oriented dialogue (Gunasekara et al., 2020), we explore a simple multi-tasking approach for a generation model that we name PrefineDST, short for *Pre-finetuned DST*, to train a more well-rounded DST model.

As shown in Figure 3, PrefineDST first pre-finetunes a pre-trained BART-large model with the same method as T0, but with a narrower set of tasks that can address the robustness issues captured by CheckDST. We select tasks that require understanding paraphrases, generating exact spans of text from the context, and resolving coreferences, with the expectation that skills required to perform well on them will be transferred to the fine-tuning step on a downstream DST task and eventually be reflected in better scores on CheckDST. Details on the chosen tasks and implementation details can be found in Appendix C.

Pre-finetuning with non-target data is a promising avenue for a robust DST model. Overall, results in Table 3 show that the simple and intuitive approach behind PrefineDST is successful in maintaining the robustness advantage that generation models have over classification models and performs on-par or better on all CheckDST metrics among competitive generation model baselines except for on F_{swap}. This well-rounded performance is maintained even when including classification models, as reflected by the lowest average slack, the difference from the highest scoring model, for all metrics Table 4.

PrefineDST is most directly comparable to SOLOIST and MUPPET in that they all incorporate a pre-finetuning step. Thus, it is notable that PrefineDST achieves a higher JGA than both while simultaneously achieving comparable or better results in all dimensions measured by CheckDST, except for F_{swap}. This indicates some degree of successful knowledge transfer from the pre-finetuning tasks, but the relatively lower F_{swap} signals that generalizing to correctly copying unseen entities through pre-finetuning is ineffective.

PrefineDST's superior results to MUPPET-DST,

Model	JGA	CorefJGA	PI cJGA	SDI cJGA	NED cJGA	F_{orig}	F_{swap}
BART-DST	$52.5_{0.2}$	$25.5_{1.2}$	$43.0_{0.8}$	$36.5_{1.3}$	$9.8_{0.7}$	$94.8_{0.2}$	$75.4_{1.8}$
SOLOIST	$54.8_{0.4}$	$30.4_{1.1}$	$44.5_{0.6}$	$38.5_{0.5}$	$10.7_{0.4}$	$94.9_{0.1}$	$81.1_{1.6}$
MUPPET-DST	$54.9_{0.4}$	$29.9_{1.9}$	$45.4_{0.4}$	$39.1_{0.7}$	$7.8_{1.4}$	$94.6_{0.1}$	$68.4_{3.8}$
PrefineDST	$55.7_{0.3}$	$30.5_{0.5}$	$46.1_{1.0}$	$41.8_{1.0}$	$11.0_{0.5}$	$95.2_{0.2}$	$76.7_{1.4}$

Table 3: CheckDST diagnosis results including PrefineDST and other generation models. The annotations are the same as those for Table 2. Compared to competitive baselines, PrefineDST improves on the previous highest score for all dimensions except F_{swap}.

	Average Δ from Best $\downarrow$
TripPy (Heck et al., 2020)	<u>4.55</u>
ConvBERT-DG (Mehri et al., 2020)	4.57
SimpleTOD (Hosseini-Asl et al., 2020)	6.93
SOLOIST (Peng et al., 2021a)	4.98
MUPPET-DST (Aghajanyan et al., 2021)	5.40
PrefineDST	**3.92**

Table 4: Average slack of each model from the best performing model on every dimension diagnosed with CheckDST and JGA based on results in Table 2 and Table 3. **Bold** indicates the best performing model and <u>underline</u> denotes the second best model. Compared to strong baselines, PrefineDST is the most well-rounded model.

which has been pre-finetuned with more than 40 tasks compared to 8 for PrefineDST, show that choosing NLP tasks that require skill related to the downstream task is more useful than having more tasks. Also, the results indicate that multitasking with all tasks as generation tasks is more effective than additional auxiliary layers when DST is also formulated as a generation task. In fact, MUPPET's poor performance compared to SimpleTOD on NED cJGA and F_{swap} shows that pre-finetuning can actually be harmful to robustness to unseen entities.

In conclusion, using a simple and intuitive approach, PrefineDST shows that pre-finetuning with non-target datasets is a promising direction for boosting robustness. We leave it to future work to leverage CheckDST as a guide to explore more sophisticated pre-finetuning strategies and non-target tasks to improve on PrefineDST, especially for handling unseen entities.

6 Related Work

Pretrained language models continue to make impressive strides on NLP benchmarks, surpassing human baseline scores on many of them (Lee et al., 2020; Reddy et al., 2019; Rajpurkar et al., 2016; Wang et al., 2019, 2018). These results lead to questions of whether these models are acquiring the intelligence required for their performance to be robust or instead are taking advantage of spurious correlations (Bender and Koller, 2020; Clark et al., 2019). Many works show that the latter is the case and seek adversarial techniques to test these models to new limits (Gardner et al., 2021; Wallace et al., 2019; Hosseini et al., 2017) and train them to be more robust (Oren et al., 2019; Jia et al., 2019; Jones et al., 2020; Zhang et al., 2022).

Robustness in dialogue models has also been similarly questioned. Perturbations to the dialogue history have exposed that dialogue models do not effectively use dialogue structure information (Sankar et al., 2019) and commonsense probes show that they struggle with commonsense reasoning(Zhou et al., 2021). Specifically for the dialogue state tracking task, several works report drops in performance for conversations with entities unseen during training (Qian et al., 2021; Huang et al., 2021; Heck et al., 2020) or with adversarially created dialogue flows (Li et al., 2020). Liu et al. (2021) and Peng et al. (2021b) recently initiated a rigorous study into the robustness of TOD models to realistic natural language perturbations.

We extend their work to establish a framework that further facilitates robustness analysis with additional metrics that capture coreference resolution performance and frequency of well-known problems to generation models. Moreover, we propose cJGA, a simple yet rigorous metric that enables measuring robustness in DST without making assumptions about the difficulty of perturbations.

PrefineDST is motivated by the recent line of work that uses generation models for DST. SimpleTOD (Hosseini-Asl et al., 2020) first reported viability of formulating TOD tasks in a completely end-to-end manner with a generation model and SOLOIST (Peng et al., 2021a) added a pretraining step to improve on data efficiency. PrefineDST, inspired by recent work on impressive results from massive multi-tasking prefinetuning (Aghajanyan et al., 2021; Sanh et al., 2021; Wei et al., 2021), extends SimpleTOD and SOLOIST by adding more prefinetuning tasks. Concurrent work from Gupta et al. (2022) explore a similar procedure for task-oriented dialogue, but their analysis is centered around zero- and few-shot generalization.

7 Conclusion

We shared our findings enabled by CheckDST, a toolkit that standardizes a comprehensive diagnosis of DST models. We confirm that benchmark results are insufficient for understanding relative performance as relative robustness is not correlated to benchmark performance. We find generation models to be a more promising direction as they are significantly more robust to language variety without making any task-specific design choices. We also observed that the standard practice of choosing a checkpoint with validation loss or accuracy is prone to overfitting, exacerbating the trade-off between JGA and robustness for both classification and generation models. Finally, we use the robustness issues exposed by CheckDST to guide the development of PrefineDST, a model that attains well-rounded improvements through a prefinetuning step that multi-tasks on reasoning skills. Moving forward, we encourage future work on task-oriented dialogue to conduct CheckDST diagnosis for a comprehensive comparison and analysis of DST performance, which we believe will make the search for ideas to build robust task-oriented dialogue models significantly more efficient.

Limitations and Broader Impacts

In this paper, we show that CheckDST can be used to reveal insights about the robustness of DST models and we hope that the task-oriented dialogue research community will build on and improve CheckDST as a means for performing holistic comparisons with other work to accelerate our understanding of effective DST approaches. We acknowledge that CheckDST cannot capture generalization to arbitrary distribution shifts in practice as the perturbations against which we measure robustness have to be known ahead of time; and mechanisms to simulate such perturbations need to be built and incorporated, which can be considered a limitation of our work. We also recognize that our analysis has been conducted only in English and therefore our empirical findings may not necessarily be true for DST models built for other languages.

CheckDST has broad implications as the accurate comparison of robustness and comparative strengths of various DST approaches will help the task-oriented dialogue community take a more informed step towards holistically improving the robustness of best-performing models. Consequently, this will improve the quality and accessibility of the numerous services that are powered by task-oriented dialogue models.

Acknowledgments

The work of JM is based in part upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00112290025.

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. *arXiv preprint arXiv:2101.11038*.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 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 (EMNLP)*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Ben Goodrich, Daniel Duckworth, Semih Yavuz, Amit Dubey, Kyu-Young Kim, and Andy Cedilnik. 2019. 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)*, pages 4516–4525, Hong Kong, China. Association for Computational Linguistics.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs. *URL https://www.kaggle. com/c/quora-question-pairs*.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. 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)*, pages 4069–4082, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 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*, pages 422–428.

Matt Gardner, William Merrill, Jesse Dodge, Matthew Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. 2021. Competency problems: On finding and removing artifacts in language data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1801–1813, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, volume 1, pages 517–520. IEEE Computer Society.

Chulaka Gunasekara, Seokhwan Kim, Luis Fernando D'Haro, Abhinav Rastogi, Yun-Nung Chen, Mihail Eric, Behnam Hedayatnia, Karthik Gopalakrishnan, Yang Liu, Chao-Wei Huang, et al. 2020. Overview of the ninth dialog system technology challenge: Dstc9. *arXiv preprint arXiv:2011.06486*.

Prakhar Gupta, Cathy Jiao, Yi-Ting Yeh, Shikib Mehri, Maxine Eskenazi, and Jeffrey P Bigham. 2022. Improving zero and few-shot generalization in dialogue through instruction tuning. *arXiv preprint arXiv:2205.12673*.

Ting Han, Ximing Liu, Ryuichi Takanobu, Yixin Lian, Chongxuan Huang, Dazhen Wan, Wei Peng, and Minlie Huang. 2020. Multiwoz 2.3: A multi-domain task-oriented dialogue dataset enhanced with annotation corrections and co-reference annotation. *arXiv preprint arXiv:2010.05594*.

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*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. Deceiving google's perspective api built for detecting toxic comments. *arXiv preprint arXiv:1702.08138*.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Tianjian Huang, Shaunak Halbe, Chinnadhurai Sankar, Pooyan Amini, Satwik Kottur, Alborz Geramifard, Meisam Razaviyayn, and Ahmad Beirami. 2021. Dair: Data augmented invariant regularization.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. 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)*, pages 4129–4142, Hong Kong, China. Association for Computational Linguistics.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. 2020. Robust encodings: A framework for combating adversarial typos. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2752–2765, Online. Association for Computational Linguistics.

Gyeongbok Lee, Seung-won Hwang, and Hyunsouk Cho. 2020. SQuAD2-CR: Semi-supervised annotation for cause and rationales for unanswerability in SQuAD 2.0. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5425–5432, Marseille, France. European Language Resources Association.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

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*, pages 7871–7880, Online. Association for Computational Linguistics.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2020. Coco: Controllable counterfactuals for evaluating dialogue state trackers. In *International Conference on Learning Representations*.

Jiexi Liu, Ryuichi Takanobu, Jiaxin Wen, Dazhen Wan, Hongguang Li, Weiran Nie, Cheng Li, Wei Peng, and Minlie Huang. 2021. Robustness testing of language understanding in task-oriented dialog. In *Proceedings of the 59th Annual Meeting of the Association for*

Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 2467–2480, Online. Association for Computational Linguistics.

Luca Massarelli, Fabio Petroni, Aleksandra Piktus, Myle Ott, Tim Rocktäschel, Vassilis Plachouras, Fabrizio Silvestri, and Sebastian Riedel. 2020. How decoding strategies affect the verifiability of generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 223–235, Online. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

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

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84, Copenhagen, Denmark. Association for Computational Linguistics.

Yonatan Oren, Shiori Sagawa, Tatsunori B. Hashimoto, and Percy Liang. 2019. Distributionally robust language modeling. 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)*, pages 4227–4237, Hong Kong, China. Association for Computational Linguistics.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021a. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Baolin Peng, Chunyuan Li, Zhu Zhang, Chenguang Zhu, Jinchao Li, and Jianfeng Gao. 2021b. RADDLE: An evaluation benchmark and analysis platform for robust task-oriented dialog systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4418–4429, Online. Association for Computational Linguistics.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 172–182, Online. Association for Computational Linguistics.

Kun Qian, Ahmad Beirami, Zhouhan Lin, Ankita De, Alborz Geramifard, Zhou Yu, and Chinnadhurai Sankar. 2021. Annotation inconsistency and entity bias in MultiWOZ. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 326–337, Singapore and Online. Association for Computational Linguistics.

Jun Quan, Deyi Xiong, Bonnie Webber, and Changjian Hu. 2019. GECOR: An end-to-end generative ellipsis and co-reference resolution model for task-oriented dialogue. 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)*, pages 4547–4557, Hong Kong, China. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Chinnadhurai Sankar, Sandeep Subramanian, Chris Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. In *Proceedings of*

the 57th Annual Meeting of the Association for Computational Linguistics, pages 32–37, Florence, Italy. Association for Computational Linguistics.

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.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. 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)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Shaolei Wang, Wangxiang Che, Qi Liu, Pengda Qin, Ting Liu, and William Yang Wang. 2020. Multi-task self-supervised learning for disfluency detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9193–9200.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Yunxiang Zhang, Liangming Pan, Samson Tan, and Min-Yen Kan. 2022. Interpreting the robustness of neural NLP models to textual perturbations. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3993–4007, Dublin, Ireland. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Pei Zhou, Pegah Jandaghi, Hyundong Cho, Bill Yuchen Lin, Jay Pujara, and Xiang Ren. 2021. Probing commonsense explanation in dialogue response generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4132–4146, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Appendix

A Further Justification for cJGA

Lemma 1. *Let*

$$\texttt{JGA} := \frac{1}{n}\sum_{i\in[n]} f(z_i;\theta), \tag{2}$$

$$\widetilde{\texttt{JGA}} := \frac{1}{n}\sum_{i\in[n]} f(\widetilde{z}_i;\theta), \tag{3}$$

$$\texttt{cJGA} := \frac{1}{n}\sum_{i\in[n]} \mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1), \tag{4}$$

where $\mathbf{1}(\cdot)$ *denotes the indicator function. Then,*

$$\texttt{cJGA} \leq \min\{\texttt{JGA}, \widetilde{\texttt{JGA}}, 1 - |\texttt{JGA} - \widetilde{\texttt{JGA}}|\}. \tag{5}$$

Proof. It is clear that $\texttt{cJGA} \leq \min\{\texttt{JGA}, \widetilde{\texttt{JGA}}\}$. The proof to show that $\texttt{cJGA} \leq 1 - |\texttt{JGA} - \widetilde{\texttt{JGA}}|$ follows from the following set of inequalities:

$$\begin{aligned}
\texttt{cJGA} &= \frac{1}{n}\sum_{i\in[n]} \mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1) \\
&\leq 1 - \frac{1}{n}\sum_{i\in[n]} |f(z_i;\theta) - f(\widetilde{z}_i;\theta)| \qquad (6) \\
&\leq 1 - \left|\frac{1}{n}\sum_{i\in[n]} (f(z_i;\theta) - f(\widetilde{z}_i;\theta))\right| \qquad (7) \\
&= 1 - |\texttt{JGA} - \widetilde{\texttt{JGA}}|,
\end{aligned}$$

where (6) follows from the fact that for any $i \in [n]$,

$$\mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1) \leq 1 - |f(z_i;\theta) - f(\widetilde{z}_i;\theta)|$$

and (7) follows from Jensen's inequality. □

Lemma 1 shows that cJGA not only captures the discrepancy between JGA and $\widetilde{\texttt{JGA}}$, but it can actually capture counterfactual robustness beyond that. As an example, consider a case where $\texttt{JGA} = \widetilde{\texttt{JGA}} = 0.6$, hence no drop is observed. In this case if $\texttt{cJGA} \approx 0.6$, it means that the performance is robust but the model is struggling with learning some particular flows. On the other hand, if cJGA is low, e.g., 0.2, it means that the performance is statistically fragile and the JGA is mostly affected by model robustness. This would not have been revealed by solely quantifying the JGA drop. As a second example, consider a case where the $\texttt{JGA} = 0.8$ whereas $\widetilde{\texttt{JGA}} = 0.6$. It is straightforward to show that cJGA cannot be larger than 0.75 (see Lemma 1), hence capturing the JGA drop. On the other hand, cJGA may be (much) smaller than 0.75 if there are further statistical model variations due to lack of robustness (inconsistency of performance across original and perturbed samples), which would not be revealed by the JGA drop.

B Further notes on CheckDST

B.1 Results on MultiWOZ2.1

Our preliminary experiments with MultiWOZ2.1 (Eric et al., 2020) showed similar results as MultiWOZ2.3. We shifted to using the latter because it has cleaner annotations that enable more accurate follow-up diagnosis. However, acknowledging that many results are still provided using 2.1, we report PrefineDST's JGA on MultiWOZ2.1, which is 53.8 ± 0.3. BART-DST's JGA is 51.5 ± 0.2. There is a relatively smaller gap than that for MultiWOZ2.3, which is in line with results for other models shown in the official MultiWOZ2.3 benchmark.[8]

B.2 Generalizability of CheckDST

For CheckDST to be applied to a TOD dataset, the dataset must have dialogue act and belief state annotations at the minimum. If these annotations are available, the LAUG toolkit can be used to insert speech disfluencies and generate paraphrases with a SC-GPT model (Peng et al., 2020; Liu et al., 2021). To replace named entities, named entity slot types must be pre-defined such that these values can be automatically scrambled or replaced, both in the annotations and dialogue. In the same vein, the named entity slot types are used to determine hallucination frequency by measuring how often their slot values are not values from the given text. CorefJGA is the least portable metric in CheckDST as it requires coreference annotations. However, using simple regular expressions for pronouns and frequently used terms such as *"same X as"* can discover many coreference cases with high precision. These subsets can then be used for measuring CorefJGA.

B.3 Baseline Training Details

Most models are trained on MultiWOZ 2.1 (Eric et al., 2020) and therefore we retrain them on MultiWOZ 2.3 (Han et al., 2020) before assessing them on CheckDST. Unless otherwise specified, we use

[8] https://github.com/lexmen318/MultiWOZ-coref

the set of hyperparameters mentioned by the original work and run five iterations with different seed values for results to have more statistical significance. If not provided, we do a hyperparameter search for the best learning rate and choose the configuration that leads to the best median JGA on the validation set. For each baseline, we train with five different seeds and report the median and standard error of these runs.

For finetuning MUPPET (Aghajanyan et al., 2021) with MultiWOZ, we follow the same setup used in the original work for finetuning on downstream tasks. We drop the additional layers and use only the parameters that are part of the original BART architecture to finetune MUPPET on MultiWOZ in the same way as BART-DST.

For intra-model comparisons, we conduct CheckDST diagnosis on checkpoints at the following epochs: [0.25, 0.5, 0.75, 1, 1.5, 2, 5, 10].

C PrefineDST Details

C.1 Implementation details

Task formulation. We take the same approach as T0 in uniformly formatting all datasets, reusing prompts for tasks that are already used for T0 and designing new ones for those that are not. For each example from a dataset, we randomly sample from a corresponding set of instruction templates and modify each sample according to the chosen template.

Prompts. For tasks that are not used in T0 such as WikiSQL (Zhong et al., 2017) and SGD (Rastogi et al., 2020), we modify applicable prompts from different tasks to create at least five different prompt templates for each task. One of these templates are randomly chosen for training time and inference time. The random seed is changed during training time but kept the same at test time to ensure replicability.

Training details. Following Sanh et al. (2021), we do not adjust the sampling rate based on the sample size of each task that we multitask with during prefinetuning. Since all tasks are formatted as a sequence-to-sequence generation task, we do not need any additional layers as was needed for MUPPET nor form heterogeneous batches that contain samples from multiple tasks. For the prefinetuning step, we do a hyperparameter search with only five different learning rates and keep the batch size at 64 per GPU to find the model with the lowest loss value on the test set. We use 8 A100 GPUs and train for 10 epochs, early stopping on the loss value of the validation set with a patience of 3. This process amounts to a total of approximately 400 GPU hours. We get best results with a learning rate of $1e^{-5}$.

Then, we finetune the prefinetuned model. We vary both the learning rate and the batch size and train for 10 epochs on a single A100 GPU, running five iterations with different seed values, after which we choose the checkpoint with the best JGA on the validation set. The best performing model uses a batch size of 4 and learning rate of $5e^{-5}$. This amounts to about 170 GPU hours in total. We use ParlAI (Miller et al., 2017) for all of our experiments.

C.2 Prefinetuning Tasks

We choose prefinetuning tasks based on their intuitive potential for improving on qualities measured by CheckDST. They can largely be categorized into copying, paraphrase classification, and coreference resolution tasks.

Copying. One of the key skills required for DST that seemed difficult to apply for out-of-domain samples is copying the correct entities mentioned in the conversation to the slot values. This skill is relevant to many other natural language understanding tasks that provide multiple candidates that can be chosen for copying, e.g., question answering and structured text generation such as text-to-SQL. To teach better copying skills, we include SQuAD v2.0 (Rajpurkar et al., 2018), CoQA (Reddy et al., 2019), WikiSQL (Zhong et al., 2017), and Schema Guided Dialogue (SGD) (Rastogi et al., 2020).

Paraphrase Classification. To internalize an understanding of semantic similarities such that the downstream model become robust to paraphrases, we leverage two paraphrase classification tasks: The Microsoft Research Paraphrase corpus (Dolan and Brockett, 2005) and the Quora Question Pairs corpus (Chen et al., 2018).

Coreference Resolution. With the expectation that seeing examples that require coreference resolution from other tasks will also help solve cases that need the same skill in DST, we include coreference resolution tasks to our prefinetuning step. We use the Winograd Schema Challenge (WSC) dataset (Levesque et al., 2012) from the SuperGLUE benchmark (Wang et al., 2019) and Wino-

Dataset	Type	Train / Valid / Test Size	Targeted CheckDST metrics
MSR (Dolan and Brockett, 2005)	Paraphrase	4,076 / 862 / 863	PI cJGA
QQP (Chen et al., 2018)	Paraphrase	305,408 / 38,176 / 38,176	PI cJGA
WSC* (Levesque et al., 2012)	Coref	554 / 104	CorefJGA
WNLI* (Wang et al., 2018)	Coref	635 / 71	CorefJGA
SQuAD v2* (Rajpurkar et al., 2018)	Q&A	130,319 / 11,873	NEI cJGA, NoHF
CoQA* (Reddy et al., 2019)	Q&A	108,647 / 7,983	NEI cJGA, NoHF, CorefJGA
WikiSQL (Zhong et al., 2017)	Text-SQL	56,355 / 8,421 / 15,878	NEI cJGA, NoHF
SGD (Rastogi et al., 2020)	TOD	164,982 / 24,363 / 42,297	NEI cJGA, NoHF, CorefJGA

Table 5: A summary of prefinetuning datasets that we use for PrefineDST. *These datasets do not have a separate test set. We reuse the validation set for these datasets.

grad NLI (WNLI) (Wang et al., 2018). /The difference changes the entity that the pronouns in the sentence must resolve to.

C.3 Prefinetuning Task Details

The full list of tasks that we use for the prefinetuning step is summarized in Table 5.