CheckDST: Measuring Real-World Generalization of Dialogue State Tracking Performance

Hyundong Cho[1]*, **Chinnadhurai Sankar**[2], **Christopher Lin**[2], **Kaushik Ram Sadagopan**[2], **Shahin Shayandeh**[2], **Asli Celikyilmaz**[2], **Jonathan May**[1], **Ahmad Beirami**[2]
[1] University of Southern California, Information Sciences Institute [2] Meta AI
jcho@isi.edu

Abstract

Neural dialogue state tracking (DST) models employ various strategies to further push state-of-the-art results in DST benchmarks. However, it is difficult to gauge whether these results, which are on test sets distributionally similar to those seen during training, translate to better real-world performance as these models tend to fail when faced with realistic natural language variations. Qualitative analysis that go beyond comparing benchmark metrics have been sparse, limiting our understanding of comparative strengths of different approaches to DST and subsequently impeding opportunities to combine them. Therefore, a comprehensive and consolidated comparison of DST performance is crucial. To this end, we design CheckDST, a framework with a collection of metrics that formally quantifies robustness to perturbations and facilitates qualitative comparisons by measuring commonly known challenges and problematic behaviors. Using CheckDST, we discover that the robustness and reaction to perturbations of top-performing DST models vary significantly despite comparable joint goal accuracy results (`JGA`). Although span-based classification models achieve slightly better `JGA` than generation models, they are significantly less robust to distribution shift and give up entirely on their predictions while generation models continue to attempt at a prediction for the correct slots. Secondly, we observe that while stopping training early, e.g., at the first epoch, hurts `JGA`, the resulting models are significantly more robust to distribution shift. Lastly, guided by weaknesses that CheckDST expose, we propose PrefineDST, a simple generation model pretrained with non-target datasets to internalize reasoning skills relevant to DST, to simultaneously boost `JGA` and robustness.

1 Introduction

The growing desire and feasibility to interact with intelligent systems through conversations have driven recent efforts in task-oriented dialogue (TOD) models, which form the backbone of digital assistants such as Siri, Google Assistant, and Amazon Alexa. A crucial skill for these models is dialogue state tracking (DST), which requires understanding the users' intents and extracting slot values to populate API queries in order to fulfill their goals. So far, state-of-the-art for DST performance has been determined by performance on an in-domain test set (Dai et al., 2021a; Su et al., 2021; Heck et al., 2020).

However, performance on a test set that is distributionally similar to the training set can be misguiding: it does not capture how well a model performs on out-of-distribution examples, which are inevitable in real-world deployment. For example, Qian et al. (2021) showed that recent DST models face a significant performance drop when test-set named entities are replaced with ones unseen during training. Liu et al. (2021) developed LAUG, an automatic augmentation tool for TOD datasets, and found a similar lack of robustness in DST models when they are exposed to various realistic perturbations. Similarly, Peng et al. (2021b) introduced a robustness benchmark for TOD models

[*] Work was done during an internship at Meta AI.

Metric	Examples		Correct DST Predictions
`PI JGA`	Original	*I would like to leave from cambridge*	`train departure cambridge`
	Perturbed	*Please book me one departing from cambridge*	
`SDI JGA`	Original	*I would like to leave from cambridge*	`train departure cambridge`
	Perturbed	*I would like to uh leave from london no i meant cambridge*	

`NED JGA,`
`NoHF`
	Original	*I would like to leave from cambridge*	`train departure cambridge`

	Perturbed	*I would like to leave from mbadgceir*	`train departure mbadgceir`

`Coref JGA`	*<user> I need you to book the restaurant for me if that's okay.*
	For 2 people at 19:45 on Tuesday...
	<user> Actually, I'm also looking for a train. I need to go
	to London Kings Cross on the same day as the restaurant booking.
			`...`
			`restaurant day tuesday`
			`train day tuesday`
			`...`

Table 1: An overview of metrics in CheckDST. For `cJGA` metrics, we are interested in tracking how often both original and perturbed samples are correctly predicted when either one of them is correct.

in the low-resource setting. Chen et al. (2021); Dai et al. (2021b) also highlight the importance of making comparisons that go beyond accuracy on a given test set.

In this paper, we first introduce and motivate CheckDST[1] – a framework for quantifying DST robustness for both full-shot and few-shot settings to facilitate comprehensive assessments and comparisons of DST performance. It provides a general framework for test set augmentation that can incorporate existing augmentation schemes. We define new metrics that measure prediction consistency via *conditional* joint goal accuracy (`JGA`) as opposed to separately measuring `JGA` on the perturbed test sets. We argue both theoretically and empirically that the conditional metrics additionally quantify the consistency of performance on original and perturbed test sets, which is crucial for model robustness against statistical variations, not captured by previous work. CheckDST also separately highlights performance on cases that are known to be more challenging, such as those that requiring coreference resolution (Han et al., 2020). It also tracks the frequency of hallucination, a problem occurring frequently in popular generation models, such as GPT-2 (Radford et al., 2019) and BART (Lewis et al., 2020).

Our second contribution is the analytical insights derived from using CheckDST. We show that models with higher `JGA` on the original test set may be significantly less robust. In particular, we evaluate two popular classes of state-of-the-art models, span-based classification models and models based on autoregressive language models (henceforth *classification models* and *generation models*, respectively). Results show that while classification models attain modestly higher `JGA` and do not hallucinate, generation models are significantly more robust to various perturbations. We also find that robustness *degrades* as training progresses by examining each model's intermediate checkpoints and elucidate *how* the degradation is manifested from qualitative analyses. These results verify that comparing `JGA` and using it as a stopping criterion during training is a poor practice as it misses useful information for quantifying real-world performance when a model faces the inevitable distribution shift at deployment.

Finally, we demonstrate how the weaknesses exposed by CheckDST can guide the development of more robust DST models that reduce the trade-off between `JGA` and robustness. We share our experiments with PrefineDST, a DST model pretrained with tasks in a similar fashion to the T0 model (Sanh et al., 2021) to acquire skills that should intuitively boost robustness as quantified by CheckDST. Our results demonstrate preliminary success in transferring such skills from non-target datasets for bridging the gaps in robustness as quantified by CheckDST.

2 CHECKDST

CheckDST stands for *Checklist for Dialogue State Tracking* and is an implementation of principles advocated by CheckList (Ribeiro et al., 2020).

CheckDST is motivated by realistic *perturbations* that robust DST models are expected to be resilient to. In this section, we motivate and formally define the metrics and perturbations that form the basis of CheckDST. An overview of the metrics is shown in Table 1.

[1] Our code and data are available at `https://anonymous.4open.science/r/CheckDST-22A4`.

2.1 Measuring robustness with Conditional JGA (cJGA)

With CheckDST, we want to answer the questions: (*i*) *"To what degree is the performance of DST models invariant to or reflective of valid perturbations that may be encountered at deployment, such as paraphrases and unseen named entities?"* and (*ii*) *"How does their robustness compare to other models?"*

To this end, one can capture robustness by comparing JGA to JGA on the perturbed test set ($\widetilde{\texttt{JGA}}$), but this assumes that the perturbed test set is more difficult to the model, and hence 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 $\widetilde{\texttt{JGA}}$ similar to JGA, even though it makes lots of *inconsistent* predictions between the original and perturbed pairs. Therefore, to capture the consistency of performance between original and perturbed samples in addition to the performance drop due to difficulty of the perturbed test set, we choose to make our comparisons using *conditional* JGA (cJGA).

cJGA measures the frequency of the cases where the prediction is correct on both the original and perturbed samples when either one of them is correct. Given a DST model (with parameters θ), let function $f(z;\theta) \to \{0,1\}$ 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:

$$\texttt{cJGA} := \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1),$$

where $\mathbf{1}(\cdot)$ denotes the indicator function and $\mathcal{I}$ is the index set of all examples in N with at least one of $f(z_i;\theta)$ or $f(\widetilde{z}_i;\theta)$ equal to one.

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 $\texttt{cJGA} \leq 1 - |\texttt{JGA} - \widetilde{\texttt{JGA}}|$ 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.

We now discuss the types of perturbations that we apply to z, their importance for robust DST, and how we measure resilience to them with cJGA.

Paraphrase Invariance cJGA (PI cJGA). Users may employ a wide variety of styles and nuances to express the same intent. Hence, the predictions of a robust DST model should be consistent for utterances that have the same semantics. There is a wide spectrum for what is considered a paraphrase, including single word replacements with a synonym. According to Li et al. (2020), DST models only drop 2% in JGA for these kinds of simple paraphrases that were generated via back-translation. However, when the paraphrases become more complex and share only a few words, as they would be in a real world situation, the models demonstrate significant drops in JGA (Peng et al., 2021b; Liu et al., 2021), indicating that understanding paraphrases is still a challenge.

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

Speech Disfluency Invariance cJGA (SDI cJGA). Many task-oriented dialogue applications are built around voice-based digital assistants. Therefore, a DST model's resilience to speech artifacts is a crucial criterion 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, SDI cJGA measures how often a model maintains a correct predictions even with the presence of speech disfluencies.

Named Entity Directional cJGA (NED cJGA). As highlighted by the motivation for DST models that explicitly employ a copy mechanism (Gu et al., 2016; Heck et al., 2020; Mehri et al., 2020; Li et al., 2020), DST models should not memorize named entities from training data so that their performance is generalizable to unseen entities.

However, generation DST models often overfit and incorrectly predict named entity slot values with entities that appear frequently in the training set (Qian et al., 2021). In order to determine the extent of overfitting to named entities seen during training, we replace named entities in the dialogue belief states and conversations with scrambled entities. NED cJGA tracks how frequently a model correctly mirrors a change in the conversation to its prediction to obtain the right slot values.

2.2 COREFERENCE JGA (COREF JGA)

In addition to cJGA metrics, we track performance for cases that require coreferences. Long conversations with coreferences that span multiple turns are relatively more difficult, as shown by the performance improvement when their 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 for making correct predictions, we simply calculate the JGA for samples in the original test set that require coreference resolution.

2.3 NO HALLUCINATION FREQUENCY (NOHF)

Generation models have become popular following recent success with various NLP tasks (Radford et al., 2019; Lewis et al., 2020; Sanh et al., 2021; Wei et al., 2021; Aghajanyan et al., 2021), including task-oriented dialogue (Su et al., 2021; Peng et al., 2021a; Hosseini-Asl et al., 2020). However, content hallucination, providing irrelevant entities memorized from training, is a well-known issue for generation models (Massarelli et al., 2020; Maynez et al., 2020). Despite being a common phenomenon, it is only indirectly measured by NED cJGA, so we measure hallucination frequency as well in CheckDST.

When a model makes a prediction for a named entity slot, we verify whether the predicted value is contained in the dialogue history, i.e., NoHF is equal to 1 if the predicted named entity is in the dialogue history and 0 otherwise. CheckDST reports NoHF on both the original test set and one used for NED cJGA (NoHF Orig and NoHF Swap).

2.4 CHECKDST IS EXTENDABLE AND DATASET-AGNOSTIC

Just as CheckList allows bring-your-own tools, CheckDST is easily extendable since any augmentation tool can be used to introduce new dimensions of robustness and measure it with cJGA as long as the belief state labels are aligned with the perturbation. Also, CheckDST is dataset-agnostic as long as the same inputs for the augmentation tools are available in other task-oriented datasets. In fact, CheckDST can be applied to other NLP tasks that can benefit from similar perturbations and robustness quantification through cJGA, e.g., sentiment analysis. However, this paper is focused on DST. We explain CheckDST's generalizability further in Appendix B.1.

3 EXPERIMENTS

We now describe the dataset and the competitive models we evaluate with CheckDST to make fine-grained comparisons on their robustness.

3.1 DATASET

Here, we use MultiWOZ (Budzianowski et al., 2018), a corpus with more than 10,000 multi-domain and single-domain task-oriented dialogues, as an example TOD dataset that we apply CheckDST to. 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 LAUG (Liu et al., 2021) for augmenting its test sets. We use MultiWOZ 2.3 in its original train/test/dev splits. 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.

Dataset Perturbations. To compute `PI cJGA` and `SDI cJGA`, we use the test sets augmented with paraphrases and speech disfluencies using LAUG. The degree of paraphrasing with LAUG is significant, replacing 74% of all words. For `SDI cJGA`, LAUG inserts speech disfluencies according to their occurrence frequency in the Switchboard corpus (Godfrey et al., 1992). More than 97% of the perturbations were considered appropriate by human evaluators.

For `NED cJGA`, we scramble the character order of named entity slots, such as *restaurant name*, to create unseen entities as done by Huang et al. (2021). Instead, we could have swapped with real 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 pretrained with SGD, we choose scrambled entities as the default for a fair comparison.

Since CheckDST can be calculated for each sample and its augmented counterpart, we can use it on any subset of a given dataset. Therefore, we construct the same few-shot setting in Peng et al. (2021a) and compare model robustness in a low-resource single-domain environment. The few-shot dataset contains 50 single-domain conversations from each of the `attraction`, `train`, `taxi`, `hotel`, and `restaurant` domain for the training set and validation set and 200 for the test set.

3.2 Models

From those models reported on the MultiWOZ 2.0 repository and the MultiWOZ 2.3 repository , we implement a subset that has replicable code. All models are trained for 10 epochs in the full-shot setting and 20 epochs in the few-shot setting. We provide more training details in Appendix B.2.

Recent DST models that attain competitive results can be largely divided into two groups: span-based classification models and generation models.

Span-based classification models. These models predict the starting and ending index of slot values that must be extracted from the context 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 uses a three-level copying strategy to predict dialogue belief states. For every `domain-slot type` pair, it 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) has the same architecture as TripPy but it replaces BERT with ConvBERT-DG, which itself is a BERT model that has been pretrained on more than 70 million conversations of open-domain dialogue and then finetuned on the DialoGLUE benchmark. Another difference with TripPy is that ConvBERT-DG multitasks with the masked language modeling objective before and during the finetuning process.

Generation models. Generation models for DST predict belief states in the same way the underlying model generates text. It sequentially generates the `domain`, `slot-type`, and `slot-value`. Belief states are generated usually via 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 model parameters. Generation models are becoming increasingly popular as they can be expanded to perform end-to-end task-oriented dialogue by also generating the dialogue policy and responses after the belief states.

(***i***) **SimpleTOD** (Hosseini-Asl et al., 2020) is a GPT-2 model that is trained to generate the dialogue belief states in `domain slot-type slot-value` format given a conversation.

(***ii***) **BART-DST** (Lewis et al., 2020; Qian et al., 2021) is the same as SimpleTOD except it uses BART instead of GPT-2.

(***iii***) **SOLOIST** (Peng et al., 2021a) is also similar to SimpleTOD but it excludes dialogue act prediction during end-to-end training and adds a pretraining step with SGD.

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

		JGA	Coref JGA	PI cJGA	SDI cJGA	NED cJGA	NoHF Orig	NoHF Swap
CLS	TripPy (2020)	62.4 ± 0.1	36.8 ± 0.5	55.2 ± 0.4	44.5 ± 0.5	3.3 ± 1.0	100 ± 0	100 ± 0
	ConvBERT-DG (2020)	62.0 ± 0.2	36.0 ± 0.6	54.9 ± 0.2	46.9 ± 0.7	2.5 ± 0.5	100 ± 0	100 ± 0
GEN	SimpleTOD (2020)	55.5 ± 0.8	29.4 ± 0.2	84.6 ± 0.8	70.8 ± 0.7	21.6 ± 0.2	93.6 ± 0.2	78.2 ± 1.0
	BART-DST (2020)	61.1 ± 0.3	38.1 ± 0.3	79.8 ± 0.6	71.6 ± 1.0	19.8 ± 0.7	95.9 ± 0.1	71.6 ± 0.8
	SOLOIST (2021a)	60.7 ± 0.2	35.6 ± 0.3	82.8 ± 0.8	70.1 ± 0.6	15.4 ± 0.7	95.8 ± 0.0	66.2 ± 1.5
	MUPPET-DST (2021)	59.4 ± 0.7	31.6 ± 2.3	87.1 ± 0.8	74.1 ± 0.7	7.0 ± 1.2	95.8 ± 0.2	60.6 ± 1.3
	PrefineDST (Ours)	61.8 ± 0.4	37.1 ± 1.1	84.5 ± 0.5	75.7 ± 0.6	19.8 ± 0.9	95.7 ± 0.1	73.4 ± 1.0

Table 2: CheckDST results on MultiWOZ 2.3 full-shot training. CLS: Classification, GEN: Generation. All results are percentages, presented as the median ± standard error over five runs. x marks the best score for the column while x marks the worst. If there is an overlap between median - standard error and median + standard error with the best/worst score, the difference is considered statistically insignificant and all overlapping scores are highlighted.

for generation tasks. MUPPET reports improved performance on downstream tasks and better data efficiency.

(v) Lastly, **PrefineDST** is our contribution, which is based on a multi-tasking prefinetuning step similar to (Sanh et al., 2021), specifically targeted at acquiring skills that intuitively should improve robustness as quantified by CheckDST. We describe it in more detail later in Section 4.4.

4 CheckDST Results

4.1 Better JGA does not correlate with more robustness

First, we evaluate classification models and generation models in the full-shot setting to examine their robustness. For all models, we select the model with the best validation set JGA in 10 epochs of training and report the results in Table 2, which demonstrate a dramatic divergence of robustness properties between the classification and generation models. Although the classification models attain slightly higher JGA than the best performing generation model and never hallucinate by design, they are much less robust than generation models against all perturbations.

The classification models' relative lack of robustness to replaced named entity slots is somewhat surprising given that identifying spans of text for slot prediction intuitively feels like an easier task than trying to generate the unseen slot values. We will study these in more detail next.

4.2 Training less is better for more robustness

The divergence in robustness revealed by CheckDST despite close JGA between classification models and generation models in the full-shot setting led to the question of *"how do robustness metrics evolve throughout training?"* We answer this question by running CheckDST on all the training checkpoints to observe how each model's performance on each metric in CheckDST fares across different training checkpoints.

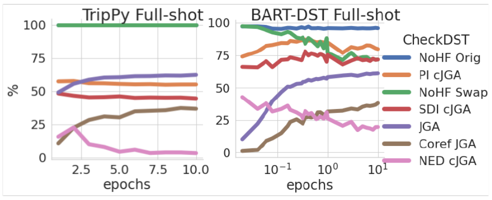

Figure 1: Most of the gains for TripPy and BART-DST on JGA are reached before the first few epochs and continues to steadily increase, but CheckDST metrics continue to deteriorate except for Coref JGA. The x-axis for BART-DST uses a logarithmic scale to better visualize the progression in the first epoch.

In Figure 1, we use TripPy and BART-DST as representative examples, as trends among the same type of models are similar, to compare classification and generation models and plot how scores on JGA and CheckDST metrics evolve throughout training. Overall, we can see the trends for each metric are similar across model types, where PI cJGA and SDI cJGA are quite flat while NED cJGA continues to deteriorate. The starting points of these metrics differ and that the relative strength of generation models on these metrics are maintained throughout the full length of training. In particular, NoHF Swap rapidly exacerbates as training proceeds for generation models.

We also observe a trade-off between most of CheckDST metrics and JGA (except for Coref JGA which increases proportionately with JGA) for all models. This trade-off for extra training is summarized in Figure 2, where we compare the model's performance at the first and the tenth epochs.

The full CheckDST results on the first epoch used for Figure 2 and the CheckDST trend charts for other models can be found in Appendix B.3.

To understand how robustness degrades, we also perform a qualitative analysis to identify patterns of failure that become apparent over time. For each model, we inspect 100 examples from each perturbed 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. Here we discuss our findings.

Classification models give up on span prediction with more training. As training progresses, we observe that TripPy and ConvBERT-DG start to produce more `none` labels for slot values and tend to not make any span predictions (rather than 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 second epoch, but the final checkpoint decides a span for the slot does not exist and does not produce a span.

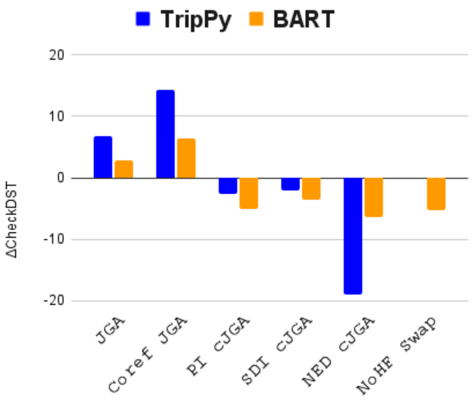

Figure 2: Relative gains and losses on CheckDST for TripPy and BART-DST when subtracting scores of the checkpoint with the best validation `JGA` (epoch 10) from those of epoch 1.

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 or (*ii*) determine that the slot value does not exist and generate nothing. In an earlier epoch, BART-DST correctly generates "`restaurant name osdi jkal`", but later instead produces "`restaurant name osjkal`". In other cases, the prediction becomes empty, similar to the behavior of classification models.

4.3 Few-shot results show a smaller divergence in robustness performance.

The robustness properties of DST models in the few-shot setting follow a similar pattern as the full-shot setting, albeit with a much smaller divergence between span-based classification and generation models. As illustrated in Figure 3 which shows the median performance of each group of models in full-shot and few-shot settings. Overall, we see a much smaller difference between the two group of models. The full results of the few-shot setting can be found in Appendix B.3.

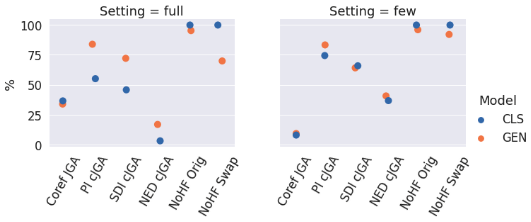

Figure 3: The difference on CheckDST between the median values for classification and generation models are much less pronounced in the few-shot setting (right).

The number of gradient update steps taken during training in the few-shot setting for 20 epochs is equivalent to that of only 0.2 epochs in the full-shot setting. Therefore, it seems that larger number of updates is more accountable to degrading robustness and the wider disparity in CheckDST for the full-shot setting than how often the same data samples are observed during training. These results from the few-shot setting also reinforce our finding that more training can deteriorate robustness for most models.

4.4 PrefineDST results

The weaknesses exposed by CheckDST guide us towards approaches that can boost robustness without compromising JGA. Motivated 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), we explore PrefineDST, short for *Prefinetuned DST*, to train a more robust DST model. PrefineDST is a BART model that is first prefinetuned with the same method as T0 on tasks that require understanding paraphrases, generating exact spans of text from the context, and resolving coreferences, with the expectation that similar skills will be transferred when finetuned on a downstream DST task and eventually be reflected in better scores on CheckDST. Details on the chosen tasks and our implementation can be found in Appendix C.

PrefineDST is a promising avenue for a robust DST model. Overall, results in Table 2 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 NoHF and NED cJGA, even for which PrefineDST ranks second best. This well-rounded performance is summarized in Table 3 and also reflected in the few-shot setting.

	Full-shot ↓	Few-shot ↓
TripPy (2020)	11.45	4.38
ConvBERT-DG (2020)	11.44	6.71
SimpleTOD (2020)	7.51	14.11
BART-DST(2020)	6.36	4.13
SOLOIST (2021a)	8.30	**0.85**
MUPPET-DST (2021)	10.72	6.38
PrefineDST (Ours)	**4.97**	1.83

Table 3: Average slack of each model from the best performing model on every metric in CheckDST and JGA based on results in Table 2 and Table 4. **Bold** indicates the best performing model and underline denotes the second best model.

PrefineDST is most directly comparable to BART-DST and thus it is notable that it achieves a higher JGA in both full-shot and few-shot setting while simultaneously achieving comparable or better results in all CheckDST metrics. This is reflective of robustness being enhanced through knowledge transfer from the prefinetuning tasks.

In addition, PrefineDST's superior results to MUPPET-DST, which has been prefinetuned with more than 40 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 BART-DST on NED cJGA and NoHF Swap shows that prefinetuning can actually be harmful to robustness.

In conclusion, using a simple and intuitive approach, PrefineDST shows that prefinetuning 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 prefinetuning strategies and non-target tasks to improve on PrefineDST.

5 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 led to questions of whether these models were acquiring the intelligence required for their performance to be robust or instead taking advantage of spurious correlations (Bender & Koller, 2020; Clark et al., 2019). Many work showed that the latter was the case and sought 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).

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 showed that they struggle with commonsense reasoning(Zhou et al., 2021). Specifically for the dialogue state tracking task, several work reported 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. They are most related to CheckDST.

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.

6 Conclusion

We introduced CheckDST, a DST robustness framework, and used it to reveal the gap in robustness between span-based classification models and generation models with similar `JGA`, verifying that performance on a test set distributionally similar to a training set does not capture a model's robustness to the inevitable deployment-time distribution shifts. We also observed a trade-off between `JGA` and robustness as CheckDST deteriorate as training proceeds while `JGA` increases, calling for more robust finetuning approaches. Finally, we use the robustness issues exposed by CheckDST to guide the development of PrefineDST, a model that better maintains both `JGA` and CheckDST metrics through a prefinetuning step to multi-task on reasoning skills that should intuitively boost robustness as quantified by CheckDST.

We encourage future work on task-oriented dialogue to adopt CheckDST and conduct a comprehensive analysis of DST robustness. We believe that the analysis CheckDST enables will pave clearer paths for future research that make task-oriented dialogue models more reliable when deployed to the real world.

Broader Impact

In this paper, we showed that CheckDST could be used to reveal insights about the robustness of DST models and we hope that the task-oriented dialogue research community would build on and improve CheckDST as a means for reliable deployment of models in the real world. 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.

MultiWOZ Budzianowski et al. (2018) is an open-source dataset released with the Apache 2.0 license and we use it for research purposes only.

References

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

Emily M. Bender and Alexander Koller. 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*, pp. 5185–5198, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.463. URL `https://aclanthology.org/2020.acl-main.463`.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica 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 (EMNLP)*, 2018.

Moya Chen, Paul A. Crook, and Stephen Roller. Teaching models new apis: Domain-agnostic simulators for task oriented dialogue, 2021.

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

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 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)*, pp. 4069–4082, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1418. URL `https://aclanthology.org/D19-1418`.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialogue state tracking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 879–885, Online, August 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-short.111. URL `https://aclanthology.org/2021.acl-short.111`.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialog state tracking, 2021b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

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

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 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*, pp. 422–428, 2020.

Matt Gardner, William Merrill, Jesse Dodge, Matthew Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. Competency problems: On finding and removing artifacts in language data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1801–1813, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.emnlp-main.135.

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

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1631–1640, 2016.

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

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica 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*, pp. 35–44, 1st virtual meeting, July 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.sigdial-1.4.

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

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

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

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 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)*, pp. 4129–4142, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1423. URL https://aclanthology.org/D19-1423.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. Robust encodings: A framework for combating adversarial typos. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2752–2765, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.245. URL https://aclanthology.org/2020.acl-main.245.

Gyeongbok Lee, Seung-won Hwang, and Hyunsouk Cho. 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*, pp. 5425–5432, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.667.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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*, pp. 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703.

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

Jiexi Liu, Ryuichi Takanobu, Jiaxin Wen, Dazhen Wan, Hongguang Li, Weiran Nie, Cheng Li, Wei Peng, and Minlie Huang. 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)*, pp. 2467–2480, 2021.

Luca Massarelli, Fabio Petroni, Aleksandra Piktus, Myle Ott, Tim Rocktäschel, Vassilis Plachouras, Fabrizio Silvestri, and Sebastian Riedel. How decoding strategies affect the verifiability of generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 223–235, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.22. URL https://aclanthology.org/2020.findings-emnlp.22.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1906–1919, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173.

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

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. ParlAI: A dialog research software platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 79–84, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-2014. URL https://aclanthology.org/D17-2014.

Yonatan Oren, Shiori Sagawa, Tatsunori B. Hashimoto, and Percy Liang. 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)*, pp. 4227–4237, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1432. URL https://aclanthology.org/D19-1432.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. Few-shot natural language generation for task-oriented dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 172–182, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.17. URL https://aclanthology.org/2020.findings-emnlp.17.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824, 2021a. doi: 10.1162/tacl_a_00399. URL https://aclanthology.org/2021.tacl-1.49.

Baolin Peng, Chunyuan Li, Zhu Zhang, Chenguang Zhu, Jinchao Li, and Jianfeng Gao. Raddle: An evaluation benchmark and analysis platform for robust task-oriented dialog systems. In *ACL-IJCNLP 2021*, August 2021b. URL https://www.microsoft.com/en-us/research/publication/raddle-an-evaluation-benchmark-and-analysis-platform-for-robust-task-oriented-dial

Kun Qian, Ahmad Beirami, Zhouhan Lin, Ankita De, Alborz Geramifard, Zhou Yu, and Chinnadhurai Sankar. Annotation inconsistency and entity bias in MultiWOZ. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 326–337, Singapore and Online, July 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.sigdial-1.35.

Jun Quan, Deyi Xiong, Bonnie Webber, and Changjian Hu. 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)*, pp. 4547–4557, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1462. URL https://aclanthology.org/D19-1462.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 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)*, pp. 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL https://aclanthology.org/P18-2124.

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

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

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4902–4912, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.442. URL https://aclanthology.org/2020.acl-main.442.

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

Chinnadhurai Sankar, Sandeep Subramanian, Chris Pal, Sarath Chandar, and Yoshua Bengio. 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*, pp. 32–37, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1004. URL https://aclanthology.org/P19-1004.

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

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 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)*, pp. 2153–2162, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1221. URL https://aclanthology.org/D19-1221.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 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*, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446.

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

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

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

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

Pei Zhou, Pegah Jandaghi, Hyundong Cho, Bill Yuchen Lin, Jay Pujara, and Xiang Ren. Probing commonsense explanation in dialogue response generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 4132–4146, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.findings-emnlp.349.

Appendix

A Further Justification for cJGA

Lemma 1. *Let*

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

$$\widetilde{\mathit{JGA}} := \frac{1}{n}\sum_{i\in[n]} f(\widetilde{z}_i;\theta), \tag{2}$$

$$c\mathit{JGA} := \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1), \tag{3}$$

where $\mathbf{1}(\cdot)$ *denotes the indicator function and* $\mathcal{I}$ *is given by*

$$\mathcal{I} := \{i \quad | \quad \max\{f(z_i;\theta), f(\widetilde{z}_i;\theta)\} = 1\}. \tag{4}$$

Then,

$$c\mathit{JGA} \leq 1 - \frac{|\mathit{JGA} - \widetilde{\mathit{JGA}}|}{\max\{\mathit{JGA}, \widetilde{\mathit{JGA}}\}} \leq 1 - |\mathit{JGA} - \widetilde{\mathit{JGA}}|. \tag{5}$$

Proof. First notice that for any $i \in \mathcal{I}$,

$$\mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1) = 1 - |f(z_i;\theta) - f(\widetilde{z}_i;\theta)|. \tag{6}$$

Hence,

$$\begin{aligned}
\sum_{i\in\mathcal{I}} \mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1) &= |\mathcal{I}| - \sum_{i\in\mathcal{I}} |f(z_i;\theta) - f(\widetilde{z}_i;\theta)| && (7)\\
&\leq |\mathcal{I}| - \left|\sum_{i\in\mathcal{I}} (f(z_i;\theta) - f(\widetilde{z}_i;\theta))\right| && (8)\\
&= |\mathcal{I}| - \left|\sum_{i\in[n]} (f(z_i;\theta) - f(\widetilde{z}_i;\theta))\right| && (9)\\
&= |\mathcal{I}| - n|\mathtt{JGA} - \widetilde{\mathtt{JGA}}|, && (10)
\end{aligned}$$

where equation 8 follows from Jensen's inequality, equation 9 follows from the fact that $f(z_i;\theta) - f(\widetilde{z}_i;\theta) = 0$ for $i \notin \mathcal{I}$ and hence we can increase the domain of summation from $\mathcal{I}$ to $[n]$, and equation 10 follows from the definition. Notice that equation 10 is achieved with equality if and only if $f(z_i;\theta) = f(\widetilde{z}_i;\theta)$ or $f(z_i;\theta) = 1 - f(\widetilde{z}_i;\theta)$, for all $i \in \mathcal{I}$. Hence,

$$\begin{aligned}
\mathtt{cJGA} &= \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbf{1}(f(z_i;\theta) = f(\widetilde{z}_i;\theta) = 1) && (11)\\
&\leq 1 - \frac{n|\mathtt{JGA} - \widetilde{\mathtt{JGA}}|}{|\mathcal{I}|} && (12)\\
&\leq 1 - \frac{|\mathtt{JGA} - \widetilde{\mathtt{JGA}}|}{\max\{\mathtt{JGA}, \widetilde{\mathtt{JGA}}\}} && (13)\\
&\leq 1 - |\mathtt{JGA} - \widetilde{\mathtt{JGA}}|, && (14)
\end{aligned}$$

where equation 12 follows from equation 10 and equation 13 follows from the fact that $|\mathcal{I}| \geq n \times \max\{\mathtt{JGA}, \widetilde{\mathtt{JGA}}\}$, and equation 14 follows from the fact that $\max\{\mathtt{JGA}, \widetilde{\mathtt{JGA}}\} \leq 1$. Notice that equation 14 is achieved with equality if and only if $\max\{\mathtt{JGA}, \widetilde{\mathtt{JGA}}\} = 1$. This completes the proof. □

Lemma 1 shows that $\texttt{cJGA}$ not only captures the discrepancy between $\texttt{JGA}$ and $\widetilde{\texttt{JGA}}$, but it can actually capture 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 $\texttt{cJGA} \approx 1$, it means that the performance is robust but the model is struggling with learning some particular flows. On the other hand, if $\texttt{cJGA}$ is low, e.g., 0.2, it means that the performance is statistically fragile and the $\texttt{JGA}$ is mostly affected by model robustness. This would not have been revealed by solely quantifying the $\texttt{JGA}$ 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 $\texttt{cJGA}$ cannot be larger than 0.75 (see Lemma 1), hence capturing the $\texttt{JGA}$ drop. On the other hand, $\texttt{cJGA}$ 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 $\texttt{JGA}$ drop.

B Further notes on CheckDST

B.1 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, we can use the LAUG toolkit to insert speech disfluencies and generate paraphrases with a SC-GPT model Liu et al. (2021); Peng et al. (2020). 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. $\texttt{Coref JGA}$ 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 $\texttt{Coref JGA}$.

B.2 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 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 $\texttt{JGA}$ 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 the few-shot setting, we make some adjustments to the hyperparamters from the full-shot setting to allow for at least 5,000 gradient updates before training ends. Every model is trained for 20 epochs with a batch size of 4 in order to provide each model with the same amount of training. The total GPU hours for baseline models is about 600 hours including all full-shot and few-shot experiments.

B.3 CheckDST Result Details

CheckDST results for the first epoch and the few-shot setting are shown in **??** and Table 4, respectively. Plots for CheckDST over time for classification models are in Figure 5 and for generation models are in Figure 4.

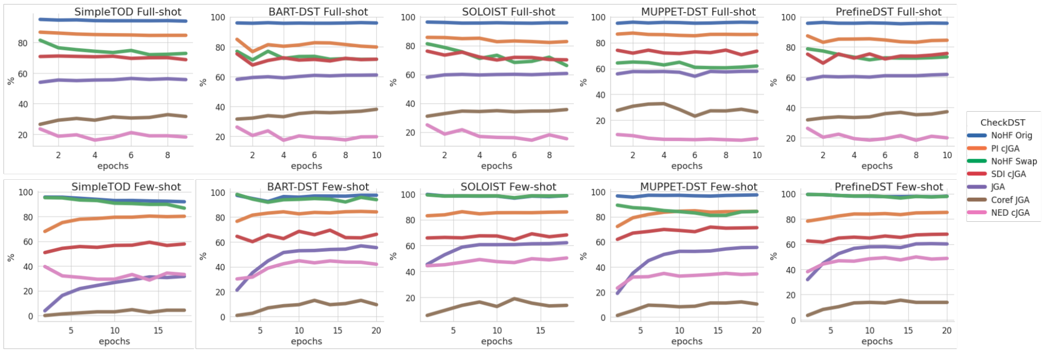

Figure 4: CheckDST over different epochs for generation models.

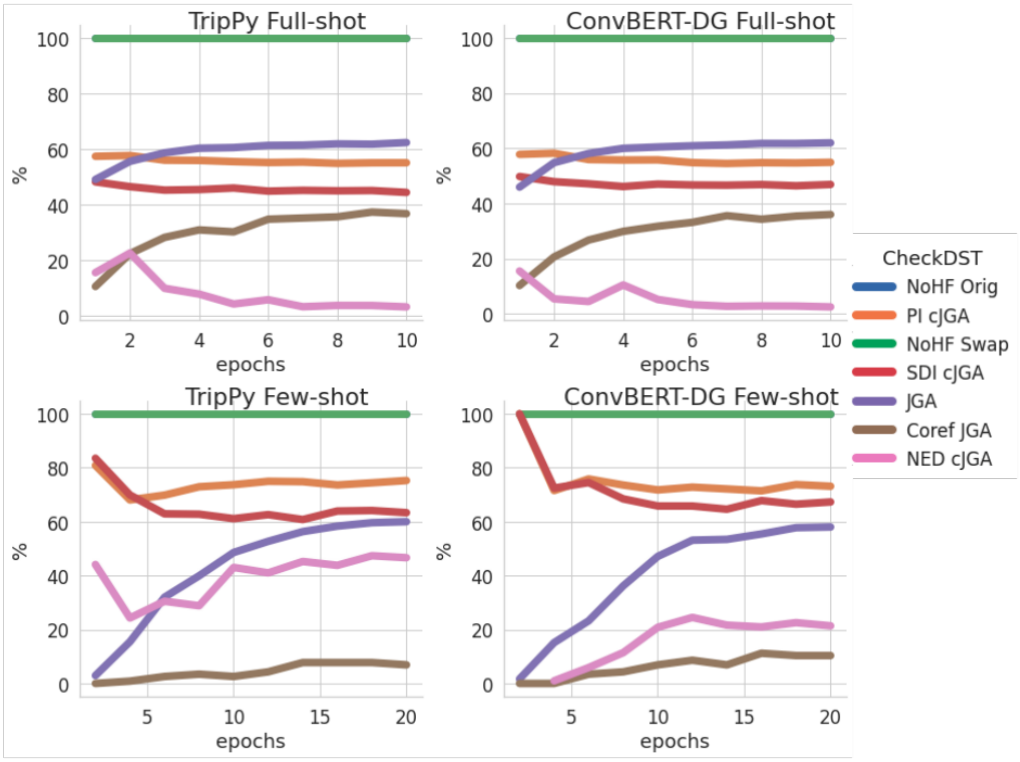

Figure 5: CheckDST over different epochs for classification models. By design, span-based classification models do not hallucinate, so both `NoHF Orig` and `NHF Swap` are always 100%.

		JGA	Coref JGA	PI cJGA	SDI cJGA	NED cJGA	NoHF Orig	NoHF Swap
CLS	TripPy (2020)	60.0 ± 0.4	6.9 ± 0.8	75.3 ± 0.5	63.3 ± 1.0	46.6 ± 1.4	100 ± 0	100 ± 0
	ConvBERT-DG (2020)	58.6 ± 1.4	8.6 ± 1.3	73.6 ± 0.6	68.3 ± 0.8	26.6 ± 3.3	100 ± 0	100 ± 0
GEN	SimpleTOD (2020)	31.6 ± 0.4	4.3 ± 0.3	79.2 ± 0.9	56.1 ± 0.8	33.5 ± 1.2	92.6 ± 0.4	91.0 ± 1.1
	BART-DST (2020)	56.7 ± 1.8	12.9 ± 0.9	84.4 ± 1.2	63.2 ± 1.7	43.5 ± 2.0	97.4 ± 0.5	95.7 ± 0.5
	SOLOIST (2021a)	62.2 ± 0.5	12.9 ± 0.2	86.0 ± 0.3	68.2 ± 0.3	50.4 ± 0.5	98.6 ± 0.1	98.5 ± 0.4
	MUPPET-DST (2021)	55.5 ± 0.2	11.2 ± 0.6	84.2 ± 0.4	71.2 ± 0.9	34.4 ± 1.2	97.4 ± 0.2	84.2 ± 1.0
	PrefineDST (Ours)	60.2 ± 0.2	12.1 ± 0.4	85.2 ± 0.2	67.9 ± 0.5	48.7 ± 0.3	98.0 ± 0.2	97.9 ± 0.4

Table 4: CheckDST results on MultiWOZ 2.3 few-shot training as described in Section 3.1. The few-shot dataset only contains single-domain conversations and therefore these results are not meant to be directly compared with results in Table 2. We annotate the table the same way as the full-shot table.

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 $5e^{-5}$ for the full-shot setting and $1e^{-5}$ for the few-shot setting. 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:

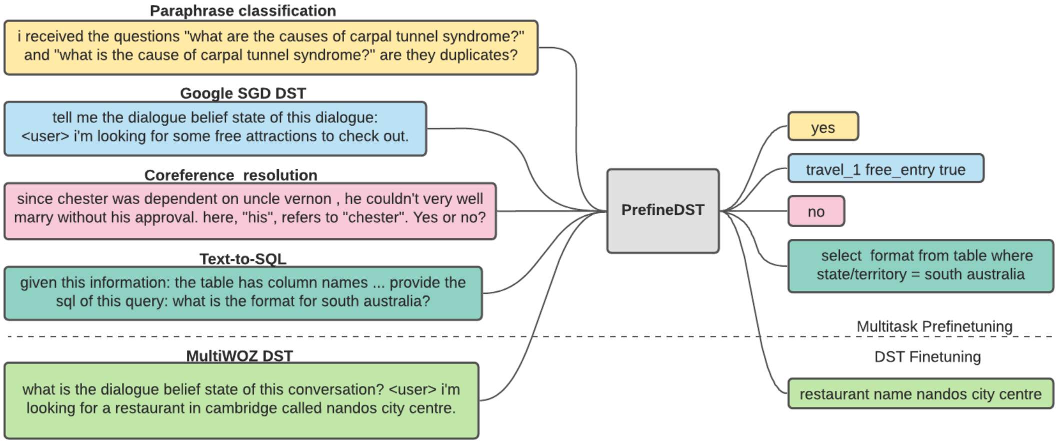
Paraphrase classification
i received the questions "what are the causes of carpal tunnel syndrome?" and "what is the cause of carpal tunnel syndrome?" are they duplicates?
Google SGD DST
tell me the dialogue belief state of this dialogue: <user> i'm looking for some free attractions to check out.
Coreference resolution
since chester was dependent on uncle vernon , he couldn't very well marry without his approval. here, "his", refers to "chester". Yes or no?
Text-to-SQL
given this information: the table has column names ... provide the sql of this query: what is the format for south australia?
MultiWOZ DST
what is the dialogue belief state of this conversation? <user> i'm looking for a restaurant in cambridge called nandos city centre.
PrefineDST
yes
travel_1 free_entry true
no
select format from table where state/territory = south australia
Multitask Prefinetuning
DST Finetuning
restaurant name nandos city centre

Figure 6: PrefineDST takes the same approach as T0 Sanh et al. (2021) for prefinetuning (above dotted line) and then adds a finetuning step for a downstream task (below dotted line).

The Microsoft Research Paraphrase corpus Dolan & 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 Winograd 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.

Dataset	Type	Train / Valid / Test Size	Targeted CheckDST metrics
MSR Dolan & 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	Coref JGA
WNLI* Wang et al. (2018)	Coref	635 / 71	Coref JGA
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, Coref JGA
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, Coref JGA

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.