Robust Dialogue State Tracking with Weak Supervision and Sparse Data

Michael Heck, Nurul Lubis, Carel van Niekerk,
Shutong Feng, Christian Geishauser, Hsien-Chin Lin, Milica Gašić
Heinrich Heine University Düsseldorf, Germany
{heckmi,lubis,niekerk,fengs,geishaus,linh,gasic}@hhu.de

Abstract

Generalizing dialogue state tracking (DST) to new data is especially challenging due to the strong reliance on abundant and fine-grained supervision during training. Sample sparsity, distributional shift, and the occurrence of new concepts and topics frequently lead to severe performance degradation during inference. In this paper we propose a training strategy to build extractive DST models without the need for fine-grained manual span labels. Two novel input-level dropout methods mitigate the negative impact of sample sparsity. We propose a new model architecture with a unified encoder that supports value as well as slot independence by leveraging the attention mechanism. We combine the strengths of triple copy strategy DST and value matching to benefit from complementary predictions without violating the principle of ontology independence. Our experiments demonstrate that an extractive DST model can be trained without manual span labels. Our architecture and training strategies improve robustness towards sample sparsity, new concepts, and topics, leading to state-of-the-art performance on a range of benchmarks. We further highlight our model's ability to effectively learn from non-dialogue data.

1 Introduction

Generalization and robustness are among the key requirements for naturalistic conversational abilities of task-oriented dialogue systems (Edlund et al., 2008). In a dialogue system, dialogue state tracking (DST) solves the task of extracting meaning and intent from the user input, and keeps track of the user's goal over the continuation of a conversation as part of a dialogue state (DS) (Young et al., 2010). A recommendation and booking system for places, for instance, needs to gather user preferences in terms of budget, location, and so forth. Concepts like these are assembled in an ontology on levels of domain (e.g., *restaurant* or *hotel*), slot (e.g., *price* or *location*), and value (e.g., ''expensive'' or ''south''). Accurate DST is vital to a robust dialogue system, as the system's future actions depend on the conversation's current estimated state. However, generalizing DST to new data and domains is especially challenging. The reason is the strong reliance on supervised training.

Virtually all top-performing DST methods either entirely or partially extract values directly from context (Ni et al., 2021). However, training these models robustly is a demanding task. Extractive methods usually rely on fine-grained labels on word level indicating the precise locations of value mentions. Given the richness of human language and the ability to express the same canonical value in many different ways, producing such labels is challenging and very costly, and it is no surprise that datasets of such kind are rare (Zhang et al., 2020b; Deriu et al., 2021). Reliance on detailed labels has another downside; datasets are usually severely limited in size. This in turn leads to the problem of sample sparsity, which increases the risk for models to over-fit to the training data, for instance, by memorizing values in their respective contexts. Overfitting prevents a state tracker to generalize to new contexts and values, which is likely to break a dialogue system entirely (Qian et al., 2021). Recently, domain-independent architectures have been encouraged to develop systems that may be built once and then applied to new scenarios with no or little additional training (Rastogi et al., 2020a,b). However, training such flexible models robustly remains a challenge, and the ever-growing need for more training samples spurs creativity to leverage non-dialogue data (Heck et al., 2020a; Namazifar et al., 2021).

We propose novel strategies for extractive DST that address the following four issues of robustness and generalization. (1) We solve *the*

Transactions of the Association for Computational Linguistics, vol. 10, pp. 1175–1192, 2022. https://doi.org/10.1162/tacl_a_00513
Action Editor: Claire Gardent. Submission batch: 3/2022; Revision batch: 6/2022; Published 11/2022.

problem of requiring fine-grained span labels with a self-supervised training scheme. Specifically, we learn from random self-labeled samples how to locate occurrences of arbitrary values. All that is needed for training a full DST model is the dialogue state ground truth, which is undoubtedly much easier to obtain than fine-grained span labels. (2) We handle *the sample sparsity problem* by introducing two new forms of input-level dropout into training. Our proposed dropout methods are easy to apply and provide a more economical alternative to data augmentation to prevent memorization and over-fitting to certain conversation styles or dialogue patterns. (3) We add a value matching mechanism on top of extraction to enhance *robustness towards previously unseen concepts*. Our value matching is entirely optional and may be utilized if a set of candidate values is known during inference, for instance, from a schema or API. (4) We propose a new architecture that is entirely domain-agnostic to facilitate *transfer to unseen slots and domains*. For that, our model relies on the attention mechanism and conditioning on natural language slot descriptions. The established slot-independence enables zero-shot transfer. We will demonstrate that we can actively teach to track new domains by learning from non-dialogue data. This is non-trivial as the model must learn to interpret dialogue data from exposure to unstructured data.

2 Related Work

Traditional DS trackers perform prediction over a fixed ontology (Mrkšić et al., 2017; Liu and Lane, 2017; Zhong et al., 2018) and therefore have various limitations in more complex scenarios (Ren et al., 2018; Nouri and Hosseini-Asl, 2018). The idea of fixed ontologies is not sustainable for real world applications, as new concepts become impossible to capture during test time. Moreover, the demand for finely labeled data quickly grows with the ontology size, causing scalability issues.

Recent approaches to DST extract values directly from the dialogue context via span prediction (Xu and Hu, 2018; Gao et al., 2019; Chao and Lane, 2019), removing the need for fixed value candidate lists. An alternative to this mechanism is value generation via soft-gated pointer-generator copying (Wu et al., 2019; Kumar et al., 2020; Kim et al., 2020). Extractive methods have limitations as well, since many values may be expressed variably or implicitly. Contextual models such as BERT (Devlin et al., 2019) support generalization over value variations to some extent (Lee et al., 2019; Chao and Lane, 2019; Gao et al., 2019), and hybrid approaches try to mitigate the issue by resorting to picklists (Zhang et al., 2020a).

TripPy (Heck et al., 2020b) jointly addresses the issues of coreference, implicit choice, and value independence with a triple copy strategy. Here, a Transformer-based (Vaswani et al., 2017) encoder projects each dialogue turn into a semantic embedding space. Domain-slot specific slot gates then decide whether or not a slot-value is present in the current turn in order to update the dialogue state. In case of presence, the slot gates also decide which of the following three copy mechanisms to use for extraction. (1) Span prediction extracts a value directly from input. For that, domain-slot specific span prediction heads predict per token whether it is the beginning or end of a slot-value. (2) Informed value prediction copies a value from the list of values that the system informed about. This solves the implicit choice issue, where the user might positively but implicitly refer to information that the system provided. (3) Coreference prediction identifies cases where the user refers to a value that has already been assigned to a slot earlier and should now also be assigned to another slot in question. TripPy shows good robustness towards new data from known domains since it does not rely on a priori knowledge of value candidates. However, it does not support transfer to new topics, since the architecture is ontology specific. Transfer to new domains or slots is therefore impossible without re-building the model. TripPy also ignores potentially available knowledge about value candidates, since its copy mechanisms operate solely on the input. Lastly, training requires fine-grained span labels, complicating the transfer to new datasets.

While contemporary approaches to DST leverage parameter sharing and transfer learning (Rastogi et al., 2020a; Lin et al., 2021), the need for finely labeled training data is still high. Sample sparsity often causes model biases in the form of memorization or other types of over-fitting. Strategies to appease the hunger of larger models are the exploitation of out-of-domain dialogue data for transfer effects (Wu et al., 2020) and data augmentation (Campagna et al., 2020; Yu et al.,

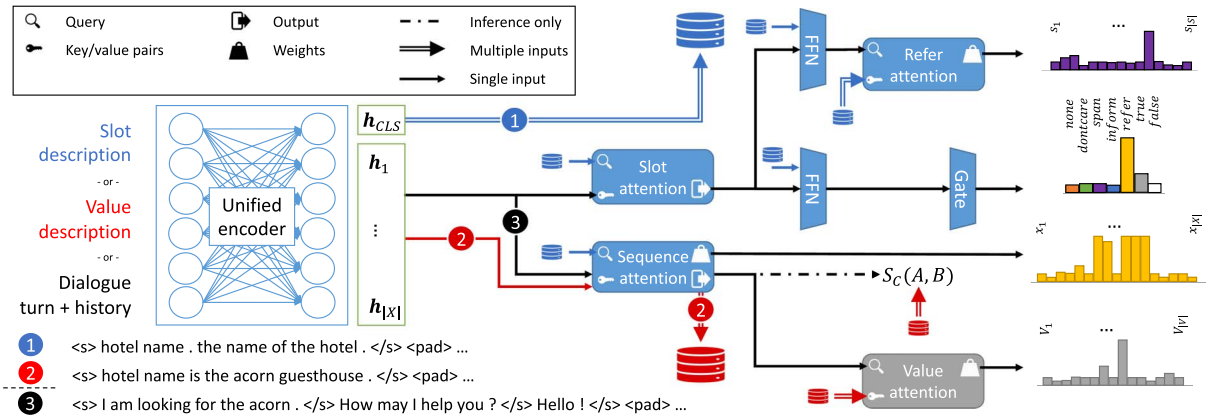

Figure 1: Proposed model architecture. TripPy-R takes the turn and dialogue history as input and outputs a DS. All inputs are encoded separately with the same fine-tuned encoder. For inference, slot and value representations are encoded once and then stored in databases for retrieval.

2020; Li et al., 2020; Dai et al., 2021). However, out-of-domain dialogue data is limited in quantity as well. Data augmentation still requires high level knowledge about dialogue structures and an adequate data generation strategy. Ultimately, more data also means longer training. We are aware of only one recent work that attempts DST with weak supervision. Liang et al. (2021) take a few-shot learning approach using only a subset of fully labeled training samples—typically from the end of conversations—to train a soft-gated pointer-generator network. In contrast, with our approach to spanless training, we reduce the level of granularity needed for labels to train extractive models. Note that these strategies are orthogonal.

3 TripPy-R: Robust Triple Copy DST

Let $\{(U_1, M_1), \ldots, (U_T, M_T)\}$ be the sequence of turns that form a dialogue. U_t and M_t are the token sequences of the user utterance and preceding system utterance at turn t. The task of DST is (1) to determine for every turn whether any of the domain-slot pairs in $S = \{S_1, \ldots, S_N\}$ is present, (2) to predict the values for each S_n, and (3) to track the dialogue state DS_t. Our starting point is triple copy strategy DST (Heck et al., 2020b), because it has already been designed for robustness towards unseen values. However, we propose a new architecture with considerable differences to the baseline regarding its design, training, and inference to overcome the drawbacks of previous approaches as laid out in Section 2. We call our proposed framework *TripPy-R* (pronounced ''trippier''), **R**obust **tri**ple co**py** strategy DST[1]. Figure 1 is a depiction of our proposed model.

3.1 Model Layout

Joint Components We design our model to be entirely domain-agnostic, adopting the idea of conditioning the model with natural language descriptions of concepts (Bapna et al., 2017; Rastogi et al., 2020b). For that, we use data-independent prediction heads that can be conditioned with slot descriptions to solve the tasks required for DST. This is different to related work such as in Heck et al. (2020b), which uses data-dependent prediction heads whose number depends on the ontology size. In contrast, prediction heads in TripPy-R are realized via the attention mechanism (Bahdanau et al., 2015). Specifically, we use scaled dot-product attention, implemented as multi-head attention according to and defined by Vaswani et al. (2017). We utilize this mechanism to query the input for the presence of information. Among other things, we deploy attention to predict whether or not a slot-value is present in the input, or to conduct sequence tagging—rather than span prediction—by assigning importance weights to input tokens.

Unified Context/Concept Encoder Different from other domain-agnostic architectures (Lee et al., 2019; Ma et al., 2019), we rely on a single encoder that is shared among encoding tasks. This unified encoder is used to produce representations for dialogue turns and natural language

[1] https://gitlab.cs.uni-duesseldorf.de/general/dsml/trippy-r-public.

slot and value descriptions. The encoder function is $\text{Enc}(X) = [\boldsymbol{h}_{\text{CLS}}, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{|\text{X}|}]$, where X is a sequence of input tokens. $\boldsymbol{h}_{\text{CLS}}$ can be interpreted as a representation of the entire input sequence. The vectors $\boldsymbol{h}_1$ to $\boldsymbol{h}_{|\text{X}|}$ are contextual representations for the sequence of input tokens. We define $\text{Enc}_{\text{P}}(X) = [\boldsymbol{h}_{\text{CLS}}]$ and $\text{Enc}_{\text{S}}(X) = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{|\text{X}|}]$ as the pooled encoding and sequence encoding of X, respectively.

Dialogue turns and natural language slot and value descriptions are encoded as

$$\begin{aligned}
\boldsymbol{R}_t &= \text{Enc}_{\text{S}}(x_{\text{CLS}} \oplus U_t \oplus x_{\text{SEP}} \oplus M_t \oplus \\
&\qquad x_{\text{SEP}} \oplus H_t \oplus x_{\text{SEP}}), \\
\boldsymbol{r}_{S_i} &= \text{Enc}_{\text{P}}(x_{\text{CLS}} \oplus S_i \oplus \text{"."} \oplus S_i^{\text{desc}} \oplus x_{\text{SEP}}), \\
\boldsymbol{R}_{V_{S_i,j}} &= \text{Enc}_{\text{S}}(x_{\text{CLS}} \oplus S_i \oplus \text{"is"} \oplus V_{S_i,j} \oplus x_{\text{SEP}}),
\end{aligned}$$

where $H_t = \{(U_{t-1}, M_{t-1}), \ldots, (U_1, M_1)\}$ is the history of the dialogue up to turn t. The special token x_{CLS} initiates every input sequence, and x_{SEP} is a separator token to provide structure to multi-sequence inputs. S_i^{desc} is the slot description of slot S_i and $V_{S_i,j}$ is a candidate value j for slot S_i.

Conditioned Slot Gate The slot gate outputs a probability distribution over the output classes $C = \{none, dontcare, span, inform, refer, true, false\}$. Our slot gate can be conditioned to perform a prediction for one particular slot, allowing our architecture to be ontology independent. The *slot attention* layer attends to token representations of a dialogue turn given the representation of a particular slot S_i as query, that is,

$$[\boldsymbol{g}_{\text{o}}, \boldsymbol{g}_{\text{w}}] = \text{MHA}_{\text{g}}(\boldsymbol{r}_{S_i}, \boldsymbol{R}_t, \boldsymbol{R}_t), \qquad (1)$$

where $\text{MHA}_{(\cdot)}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \hat{\boldsymbol{k}})$ is a multi-head attention layer that expects a query matrix $\boldsymbol{Q}$, a key matrix $\boldsymbol{K}$, a value matrix $\boldsymbol{V}$ and an optional masking parameter $\hat{\boldsymbol{k}}$. $\boldsymbol{g}_{\text{o}}$ is the layer-normalized (Ba et al., 2016) attention output and $\boldsymbol{g}_{\text{w}}$ are the attention weights. For classification, the attention output is piped into a feed-forward network (FFN) conditioned with S_i,

$$\boldsymbol{g}_{\text{s}} = \text{softmax}(\text{L}_3(\text{G}_2(\boldsymbol{r}_{S_i} \oplus \text{G}_1(\boldsymbol{g}_{\text{o}})))) \in \mathbb{R}^7,$$

where $\text{L}_{(\cdot)}(\boldsymbol{x}) = \boldsymbol{W}_{(\cdot)} \cdot \boldsymbol{x} + \boldsymbol{b}_{(\cdot)}$ is a linear layer, and $\text{G}_{(\cdot)}(\boldsymbol{x}) = \text{GeLU}(\text{L}_{(\cdot)}(\boldsymbol{x}))$ (Hendrycks and Gimpel, 2016).

Sequence Tagging In order to keep the value extraction directly from the input ontology-independent as well, our model re-purposes attention to perform sequence tagging. If the slot gate predicts *span*, the *sequence attention* layer attends to token representations of the current dialogue turn given $\boldsymbol{r}_{S_i}$ as query, analogous to Eq. (1):

$$[\boldsymbol{q}_{\text{o}}, \boldsymbol{q}_{\text{w}}] = \text{MHA}_{\text{q}}(\boldsymbol{r}_{S_i}, \boldsymbol{R}_t, \boldsymbol{R}_t, \hat{\boldsymbol{r}}_t). \qquad (2)$$

Here, $\hat{\boldsymbol{r}}_t$ is an input mask that only allows attending to representations of user utterances.

In contrast to other work that leverages attention for DST (Lee et al., 2019; Wu et al., 2019), we explicitly teach the model *where* to put the attention. This way, the predicted attention weights $\boldsymbol{q}_{\text{w}}$ become the sequence tagging predictions. Tokens that belong to a value are assigned a weight of 1, all other tokens are weighted 0. Since $\|\boldsymbol{q}_{\text{w}}\|_1 = 1$, we scale the target label sequences during training. During inference, we normalize $\boldsymbol{q}_{\text{w}}$, namely,

$$\hat{\boldsymbol{q}}_{\text{w}} = [\hat{q}_1, \ldots, \hat{q}_{|X|}], \text{with } \hat{q}_j = \frac{q_{\text{w},j} - \frac{1}{|X|}}{\max_{\forall q \in \boldsymbol{q}_{\text{w}}} q}, \qquad (3)$$

so that we can infer sequence tags according to an ''*IO*'' tagging scheme (Ramshaw and Marcus, 1995). All $\hat{q}_j > 0$ are assigned the ''*I*'' tag, all others the ''*O*'' tag. The advantage of sequence tagging over span prediction is that training can be performed using labels for multiple occurrences of the same slot-value in the input (for instance in the current turn and the dialogue history), and that multiple regions of interest can be predicted. To extract a value from the context, we pick the sequence with the highest average token weight according to $\hat{\boldsymbol{q}}_{\text{w}}$ among all sequences of tokens that were assigned the ''*I*'' tag and denote this value prediction as $\text{Val}(\hat{\boldsymbol{q}}_{\text{w}})$.

Informed Value Prediction We adopt informed value prediction from TripPy. Ontology independence is established via our conditioned slot gate. The inform memory $I_t = \{I_t^1, \ldots, I_t^{|S|}\}$ tracks slot-values that were informed by the system in the current dialogue turn t. If the user positively refers to an informed value, and if the user does not express the value such that sequence tagging can be used (i.e., the slot gate predicts $inform$), then the value ought to be copied from I_t to DS_t.

We know from works on cognition that ''all collective actions are built on common ground and its accumulation'' (Clark and Brennan, 1991). In other words, it must be established in a conversation what has been understood by all participants. The process of forming mutual understanding is known as *grounding*. Informed value prediction in TripPy-R serves as a grounding component. As long as the information shared by the system has not yet been grounded (i.e., confirmed by the user), it is not added to the DS. This is in line with information state and dialogue management theories such as devised by Larsson and Traum (2000), which view grounding as essential to the theory of information states and therefore DST.

Coreference Prediction Although TripPy supports coreference resolution, this mechanism is limited to an a priori known set of slots. We use attention to establish slot independence for coreference resolution to overcome this limitation. If the slot gate predicts $refer$ for a slot S_i, namely, that it refers to a value that has previously been assigned to another slot, then the *refer attention* needs to predict the identity of said slot S_j, that is,

$$[\boldsymbol{f}_{\mathrm{o}}, \boldsymbol{f}_{\mathrm{w}}] = \mathrm{MHA}_{\mathrm{f}}(\mathrm{G}_5(\boldsymbol{r}_{S_i} \oplus \mathrm{G}_4(\boldsymbol{g}_{\mathrm{o}})), \boldsymbol{R}_S, \boldsymbol{R}_S),$$

where the slot attention output $\boldsymbol{g}_{\mathrm{o}}$ is first piped through an FFN. $\boldsymbol{R}_S = [\boldsymbol{r}_{S_1}, \ldots, \boldsymbol{r}_{S_{|S|}}] \in \mathbb{R}^{d \times |S|}$ is the matrix of stacked slot representations and $\boldsymbol{f}_{\mathrm{w}}$ is the set of weights assigned to all candidate slots for S_j. The slot with the highest assigned weight is then our referred slot S_j. To resolve a coreference, S_i is updated with the value of S_j. During inference, $\boldsymbol{R}_S$ can be modified as desired to accommodate new slots.

Value Matching In contrast to picklist based methods such as that of Zhang et al. (2020a), TripPy-R performs value matching as an optional step. We first create slot-value representations for all value candidates $V_{S_i,j}$ of slot S_i, and learn matching of dialogue context $\boldsymbol{q}_{\mathrm{o}}$ to the list of candidate values via *value attention*:

$$\begin{aligned}[\boldsymbol{r}_{V_{S_i,j}}, \boldsymbol{v}_{\mathrm{w}}] &= \mathrm{MHA}_{\mathrm{q}}(\boldsymbol{r}_{S_i}, \boldsymbol{R}_{V_{S_i,j}}, \boldsymbol{R}_{V_{S_i,j}}), \quad (4)\\ [\boldsymbol{m}_{\mathrm{o}}, \boldsymbol{m}_{\mathrm{w}}] &= \mathrm{MHA}_{\mathrm{m}}(\boldsymbol{q}_{\mathrm{o}}, \boldsymbol{R}_{V_{S_i}}, \boldsymbol{R}_{V_{S_i}}).\end{aligned}$$

where $\boldsymbol{R}_{V_{S_i}} = [\boldsymbol{r}_{V_{S_i,1}}, \ldots, \boldsymbol{r}_{V_{S_i,|V_{S_i}|}}] \in \mathbb{R}^{d \times |V_{S_i}|}$. $\boldsymbol{m}_{\mathrm{w}}$ should place a weight close to 1 on the correct value and weights close to 0 on all the others. Dot-product attention as used in our model is defined as $\mathrm{softmax}(\boldsymbol{Q} \cdot \boldsymbol{K}^\top) \cdot \boldsymbol{V}$. Computing the dot product between input and candidate value representations is proportional to computing their cosine similarities, which is $\cos(\theta) = \frac{\boldsymbol{q} \cdot \boldsymbol{k}}{\|\boldsymbol{q}\| \cdot \|\boldsymbol{k}\|}$ $\forall \boldsymbol{q} \in \boldsymbol{Q}, \boldsymbol{k} \in \boldsymbol{K}$. Therefore, optimizing the model to put maximum weight on the correct value and to minimize the weights on all other candidates forces representations of the input and of values occurring in that input to be closer in their common space, and vice versa.

3.2 Training and Inference

Each training step requires the dialogue turn and all slot and value descriptions to be encoded. Our unified encoder re-encodes all slot descriptions at each step. Because the number of values might be in the range of thousands, we encode them once for each epoch. The encoder is fine-tuned towards encoding all three input types. We optimize our model given the joint loss for each turn,

$$\mathcal{L} = \lambda_{\mathrm{g}} \cdot \mathcal{L}_{\mathrm{g}} + \lambda_{\mathrm{q}} \cdot \mathcal{L}_{\mathrm{q}} + \lambda_{\mathrm{f}} \cdot \mathcal{L}_{\mathrm{f}} + \lambda_{\mathrm{m}} \cdot \mathcal{L}_{\mathrm{m}}, \quad (5)$$

$$\begin{aligned}
\mathcal{L}_{\mathrm{g}} &= \Sigma_i \ell(\boldsymbol{g}_{\mathrm{s}}, L^{\mathrm{g}}_{S_i}), & L^{\mathrm{g}}_{S_i} &\in C,\\
\mathcal{L}_{\mathrm{q}} &= \Sigma_i \ell(\boldsymbol{q}_{\mathrm{w}}, \boldsymbol{l}^{\mathrm{q}}_{S_i} / \|\boldsymbol{l}^{\mathrm{q}}_{S_i}\|_1), & \boldsymbol{l}^{\mathrm{q}}_{S_i} &\in \{0,1\}^{|X|},\\
\mathcal{L}_{\mathrm{f}} &= \Sigma_i \ell(\boldsymbol{f}_{\mathrm{w}}, \boldsymbol{l}^{\mathrm{f}}_{S_i}), & \boldsymbol{l}^{\mathrm{f}}_{S_i} &\in \{0,1\}^{|S|},\\
\mathcal{L}_{\mathrm{m}} &= \Sigma_i \ell(\boldsymbol{m}_{\mathrm{w}}, \boldsymbol{l}^{\mathrm{m}}_{S_i}), & \boldsymbol{l}^{\mathrm{m}}_{S_i} &\in \{0,1\}^{|V_{S_i}|}.
\end{aligned}$$

Here, $\ell(\cdot,\cdot)$ is the loss between a prediction and a ground truth. $\mathcal{L}_{\mathrm{g}}$, $\mathcal{L}_{\mathrm{q}}$, $\mathcal{L}_{\mathrm{f}}$ and $\mathcal{L}_{\mathrm{m}}$ are the joint losses of the slot gate, sequence tagger, coreference prediction and value matching. It is $\|\cdot\|_1 = 1$ for $\boldsymbol{l}^{\mathrm{f}}_{S_i}$ and $\boldsymbol{l}^{\mathrm{m}}_{S_i}$, that is, labels for coreference prediction and value matching are 1-hot vectors. Back-propagating $\mathcal{L}_{\mathrm{m}}$ also affects the sequence tagger. We scale $\boldsymbol{l}^{\mathrm{q}}_{S_i}$ since sequence tagging may have to label more than one token as being part of a value.

During inference, the model can draw from the rich output of the model, namely, slot gate predictions, coreference prediction, sequence tagging and value matching to adequately update the dialogue state. Slot and value descriptions are encoded only once with the fine-tuned encoder, then stored in databases, as illustrated in Figure 1 in steps ① and ②. Pre-encoded slots condition the attention and FFN layers, and pre-encoded values are used for value matching. Note that it is straightforward to update these databases on-the-fly for a

running system, thus easily expanding its capacities. Step ③ is the processing of dialogue turns to perform dialogue state update prediction.

3.3 Dialogue State Update

At turn t, the slot gate predicts for each slot S_i how it should be updated. $none$ means that no update is needed. $dontcare$ denotes that any value is acceptable to the user. $span$ indicates that a value is extractable from any of the the user utterances $\{U_t, \ldots, U_1\}$. $inform$ denotes that the user refers to a value uttered by the system in M_t. $refer$ indicates that the user refers to a value that is already present in DS_t in a different slot. Classes $true$ and $false$ are used by slots that take binary values.

If candidate values are known at inference, TripPy-R can utilize value matching to benefit from supporting predictions for the $span$ case. Because sequence tagging and value matching predictions would compete over the slot update, we use confidence scores to make an informed decision. Given the current input, and candidate values for a slot, we can use the attention weights $\boldsymbol{m}_{\text{w}}$ of the value attention as individual scores for each value. We can also use the L2-norm between input and values, namely, $e_{S_i,j} = \|\boldsymbol{q}_{\text{o}} - \boldsymbol{r}_{V_{S_i},j}\|_2$, and $\boldsymbol{e}_{S_i} = [e_{S_i,1}, \ldots, e_{S_i,|V_{S_i}|}]$ is the score set. Then

$$\text{Conf}(C) = 1 - \frac{\min_{\forall c \in C} c}{\left(\left(\sum_{c \in C} c\right) - \min_{\forall c \in C} c\right)/|C|},$$

is applied to $\boldsymbol{m}_{\text{w}}$ and $\boldsymbol{e}_{S_i}$ (interpreting them as multisets rather than vectors) to compute two confidence scores $\text{Conf}(\boldsymbol{m}_{\text{w}})$ and $\text{Conf}(\boldsymbol{e}_{S_i})$ for the most likely value candidate. This type of confidence captures the notion of difference between the best score and the mean of all other scores, intuitively expressing model certainty. $\text{Val}(\boldsymbol{m}_{\text{w}}) = \text{argmax}(\boldsymbol{m}_{\text{w}})$ and $\text{Val}(\boldsymbol{e}_{S_i}) = \text{argmax}(\boldsymbol{e}_{S_i})$ are the most likely candidates according to value attention and L2-norm. For any slot that was predicted as $span$, the final prediction is

$$S_i^* = \begin{cases} \text{Val}(\boldsymbol{m}_{\text{w}}), \text{if} & S_i \text{ is categorical}^2 \wedge \\ & \text{Conf}(\boldsymbol{m}_{\text{w}}) > \tau \\ \text{Val}(\boldsymbol{e}_{S_i}), \text{else if} & \text{Conf}(\boldsymbol{e}_{S_i}) > \tau \\ \text{Val}(\hat{\boldsymbol{q}}_{\text{w}}), \text{else}, & \end{cases}$$

[2]For the distinction of categorical and non-categorical slots, see Rastogi et al. (2020b) and Zang et al. (2020).

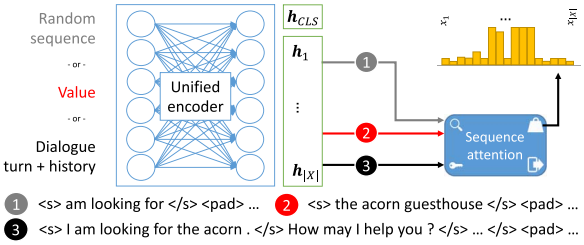

Figure 2: The proto-DST model for value tagging.

where $\tau \in [0, 1]$ is a threshold parameter that controls the level of the model's confidence needed to still consider its value matching predictions.

4 Levels of Robustness in DST

We propose the following methods to improve robustness in DST on multiple levels.

4.1 Robustness to Spanless Labels

Our framework introduces a novel training scheme to learn from data without span labels, therefore lowering the demand for fine-grained labels. We teach a proto-DST model that uses parts of TripPy-R's architecture to tag random token sub-sequences that occur in the textual input. We use this model to locate value occurrences in each turn t of a dialogue as listed in the labels for DS_t.

The proto-DST model consists of the unified encoder and the sequence attention of TripPy-R, as depicted in Figure 2. Let $D_t = (U_t, M_t)$ be the input to the model, which is encoded as $\boldsymbol{R}'_t = \text{Enc}_{\text{S}}(x_{\text{CLS}} \oplus x_{\text{NONE}} \oplus x_{\text{SEP}} \oplus U_t \oplus x_{\text{SEP}} \oplus M_t \oplus x_{\text{SEP}})$. Let $Y \in D_t$ be a sub-sequence of tokens that was randomly picked from the input, encoded as $\boldsymbol{r}_Y = \text{Enc}_{\text{P}}(x_{\text{CLS}} \oplus Y \oplus x_{\text{SEP}})$. In Figure 2, this corresponds to input types ① and ③. The sequence tagger is then described as

$$[\boldsymbol{q}'_{\text{o}}, \boldsymbol{q}'_{\text{w}}] = \text{MHA}_{\text{q}}(\boldsymbol{r}_Y, \boldsymbol{R}'_t, \boldsymbol{R}'_t),$$

analogous to Eq. (2). For training, we minimize

$$\mathcal{L}_{\text{q}} = \ell(\boldsymbol{q}'_{\text{w}}, \boldsymbol{l}^{\text{q}}_Y / \|\boldsymbol{l}^{\text{q}}_Y\|_1), \quad \boldsymbol{l}^{\text{q}}_Y \in \{0, 1\}^{|X|},$$

analogous to Eq. (5). At each training step, a random negative sample $\bar{Y} \notin D_t$ rather than a positive sample is picked for training with probability p_{neg}. For the $Y \in D_t$, the label $\boldsymbol{l}^{\text{q}}_Y$ marks the positions of all tokens of Y in D_t. For the $\bar{Y} \notin D_t$, the label $\boldsymbol{l}^{\text{q}}_{\bar{Y}}$ puts a weight of 1 onto special token x_{NONE} and 0 everywhere else. The desired

Training	PMUL1188	x_{CLS}	x_{NONE}	x_{SEP}	I	need	a	train	to	leave	from	Cambridge	after	15:30	x_{SEP}	
$Y \in D_t$	labels	0	0	0	0	1	1	1	0	0	0	0	0	0	0	
$\bar{Y} \notin D_t$	labels	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Tagging	PMUL2340	x_{CLS}	x_{NONE}	x_{SEP}	Hi,	I	am	looking	for	an	upscale	restaurant	in	the	centre	x_{SEP}
rest.-price=expensive	prediction	0	.25	0	0	0	0	0	0	0	.87	0	0	0	0	0
rest.-area=centre	prediction	0	0	0	0	0	0	0	0	0	0	0	0	0	.99	0

Table 1: Top: Training samples for the proto-DST. $Y = \{\text{``need''}, \text{``a''}, \text{``train''}\}$ is a randomly picked sub-sequence in D_t. The model needs to tag all tokens belonging to Y. For any random sequence $\bar{Y} \notin D_t$, all probability mass should be assigned to x_{NONE}. Bottom: Example of tagging the training data with a proto-DST given only spanless labels. The model needs to tag all tokens belonging to the respective values. Note how the proto-DST successfully tagged the word 'upscale' as an occurrence of the canonical value *restaurant-price=expensive*.

behavior of this model is therefore to distribute the maximum amount of the probability mass uniformly among all tokens that belong to the randomly picked sequence. In case a queried sequence is absent from the input, all probability mass should be assigned to x_{NONE}. Table 1 lists positive and negative training examples.

In order to tag value occurrences in dialogue turns for training with spanless labels, we predict for each value in DS_t its position in D_t, given the proto-DST. Let s_t^i be the value label for slot S_i in turn t, which is encoded as $\boldsymbol{r}_{s_t^i} = \text{Enc}_{\text{P}}(x_{\text{CLS}} \oplus s_t^i \oplus x_{\text{SEP}})$. Value tagging is performed as

$$[\boldsymbol{q}'_{\text{o}}, \boldsymbol{q}'_{\text{w}}] = \text{MHA}_{\text{q}}(\boldsymbol{r}_{s_t^i}, \boldsymbol{R}'_t, \boldsymbol{R}'_t, \hat{\boldsymbol{r}}_t), \forall s_t^i \in DS_t,$$

which corresponds to input types ② and ③ in Figure 2. $\boldsymbol{q}'_{\text{w}}$ is normalized according to Eq. (3). Table 1 shows examples of value tagging with the proto-DST. A set of tag weights $\hat{\boldsymbol{q}}'_{\text{w}}$ is accepted if more than half the probability mass is assigned to word tokens rather than x_{NONE}. We use a morphological closing operation (Serra, 1982) to smooth the tags, that is,

$$\hat{\boldsymbol{q}}'_{\text{w}} \bullet \omega = \delta_{>\nu}(\hat{\boldsymbol{q}}'_{\text{w}} \oplus \omega) \ominus \omega, \tag{6}$$

where $\oplus$ and $\ominus$ are the dilation and erosion operators, δ is an indicator function, $\hat{\boldsymbol{q}}'_{\text{w}}$ is interpreted as an array, $\omega = [1, 1, 1]$ is a kernel, and ν is a threshold parameter that allows filtering of tags based on their predicted weights.

Contextual representations enable our value tagger to also identify positions of value variants, i.e., different expressions of the same value (see Table 1 for an example). We tag turns without their history. To generate labels for the history portion, we simply concatenate the tags of the preceding turns with the tags of the current turn.

4.2 Robustness to Sample Sparsity

We propose new forms of input-level dropout to increase variance in training samples while preventing an increase in data and training time.

Token Noising Targeted feature dropout (Xu and Sarikaya, 2014) has already been used successfully in the form of slot value dropout (SVD) to stabilize DST model training (Chao and Lane, 2019; Heck et al., 2020b). During training, SVD replaces tokens of extractable values in their context by a special token x_{UNK} with a certain probability. The representation of x_{UNK} amalgamates the contextual representations of all tokens that are not in the encoder's vocabulary V_{enc} and therefore carries little semantic meaning.

Instead of randomly replacing target tokens with x_{UNK}, we use random tokens from a frequency-sorted V_{enc}. Specifically, a target token is replaced with probability p_{tn} by a token $x_k \in V_{\text{enc}}$, where k is drawn from a uniform distribution $\mathcal{U}(1, K)$. Since the least frequent tokens in V_{enc} tend to be nonsensical, we use a cut-off $K \ll |V_{\text{enc}}|$ for k. The idea behind this *token noising* is to avoid a train-test discrepancy. With SVD, x_{UNK} is occasionally presented as target during training, but the model will always encounter valid tokens during inference. With token noising, this mismatch does not occur. Further, token noising increases the variety of observed training samples, while SVD potentially produces duplicate inputs by masking with a placeholder.

History Dropout We propose history dropout as another measure to prevent over-fitting due to

sample sparsity. With probability p_{hd}, we discard parts of the turn history H_t during training. The cut-off is sampled from $\mathcal{U}(1, t-1)$. Utilizing dialogue history is essential for competitive DST (Heck et al., 2020b). However, models might learn correlations from sparse samples that do not hold true on new data. The idea of history dropout is to prevent the model from over-relying on the history so as to not be thrown off by previously unencountered conversational styles or contents.

4.3 Robustness to Unseen Values

Robustness to unseen values is the result of multiple design choices. The applied triple copy strategy as proposed by Heck et al. (2020b) facilitates value independence. Our proposed token noising and history dropout prevent memorization of reoccurring patterns. TripPy-R's value matching provides an alternative prediction for the DS update, in case candidate values are available during inference. Our model is equipped with the partial masking functionality (Heck et al., 2020b). Masking may be applied to informed values in the system utterances $M_t, \ldots, M_1$ using x_{UNK}, which forces the model to focus on the system utterances' context information rather than specific mentions of values.

4.4 Robustness to Unseen Slots and Domains

Domain transfer has the highest demand for generalizability and robustness. A transfer of the strong triple copy strategy DST baseline to new topics post facto is not possible due to ontology dependence of slot gates, span prediction heads, inform memory, and classification heads for coreference resolution. The latter two mechanisms in particular contribute to robustness of DST towards unseen values within known domains (Heck et al., 2020b). However, the proposed TripPy-R architecture is absolutely vital to establish robustness of triple copy strategy DST to unseen slots across new domains. TripPy-R is designed to be entirely domain-agnostic by using a model architecture whose parts can be conditioned on natural language descriptions of concepts.

5 Experimental Setup

5.1 Datasets

We use MultiWOZ 2.1 (Eric et al., 2020), WOZ 2.0 (Wen et al., 2017), sim-M, and sim-R (Shah et al., 2018) for robustness tests. MultiWOZ 2.1 is a standard benchmark for multi-domain dialogue modeling that contains 10000+ dialogues covering 5 domains (train, restaurant, hotel, taxi, attraction) and 30 unique domain-slot pairs. The other datasets are significantly smaller, making sample sparsity an issue. We test TripPy-R's value independence on two specialized MultiWOZ test sets, OOO_{Heck} (Heck et al., 2020b) and OOO_{Qian} (Qian et al., 2021), which replace many values with out-of-ontology (OOO) values. In addition to MultiWOZ version 2.1, we test TripPy-R on 2.0, 2.2, 2.3, and 2.4 (Budzianowski et al., 2018; Zang et al., 2020; Han et al., 2021; Ye et al., 2021a).

5.2 Evaluation

We use joint goal accuracy (JGA) as the primary metric to compare between models. The JGA given a test set is the ratio of dialogue turns for which all slots were filled with the correct value (including $none$). For domain-transfer tests, we report per-domain JGA, and for OOO prediction experiments, we also report per-slot accuracy. We repeat each experiment 10 times for small datasets, and three times for MultiWOZ and report averaged numbers and maximum performance. For evaluation, we follow Heck et al. (2020b).

5.3 Training

We initialize our unified encoder with RoBERTa-base (Liu et al., 2019). The input sequence length is 180 after WordPiece tokenization (Wu et al., 2016). The loss weights are $(\lambda_{\text{g}}, \lambda_{\text{q}}, \lambda_{\text{f}}, \lambda_{\text{m}}) = (0.8, \frac{1-\lambda_{\text{g}}}{2}, \frac{1-\lambda_{\text{g}}}{2}, 0.1)$. $\ell_{\text{g}}, \ell_{\text{f}}$ are cross entropy loss, and $\ell_{\text{q}}, \ell_{\text{m}}$ are mean squared error loss. We use the Adam optimizer (Kingma and Ba, 2015) and back-propagate through the entire network including the encoder. We also back-propagate the error for slot encodings, since we re-encode them at every step. The learning rate is 5e-5 after a warmup portion of 10% (5% for MultiWOZ), then decays linearly. The maximum number of epochs is 20 for MultiWOZ, 50 for WOZ 2.0, and 100 for sim-M/R. We use early stopping with patience (20% of max. epoch), based on the development set JGA. The batch size is 16 (32 for MultiWOZ). During training, the encoder output dropout rate is 30%, and $p_{\text{tn}} = p_{\text{hd}} = 30\%$ (10% for MultiWOZ). The weight decay rate is 0.01. For token noising, we set $K = 0.2 \cdot |V_{\text{enc}}|$. We weight ℓ_{g} for $none$ cases with 0.1. For value matching we tune τ in decrements of 0.1 on the development sets.

Models	sim-M		sim-R		WOZ 2.0		MultiWOZ 2.1	
	average	best	average	best	average	best	average	best
TripPy (baseline)	88.7±2.7	94.0	90.4±1.0	91.5	**92.3**±0.6	**93.1**	55.3±0.9	56.3
TripPy-R w/o value matching	95.1±0.9	96.1	92.0±0.9	93.8	91.3±1.2	92.2	54.2±0.2	54.3
TripPy-R	**95.6**±1.0	**96.8**	92.3±2.7	**96.2**	91.5±0.6	92.6	**56.0**±0.3	**56.4**
w/o History dropout	95.4±0.5	96.1	**93.2**±0.9	94.7	91.6±1.0	93.0	55.5±0.6	56.2
w/o Token noising	88.6±3.6	94.4	92.7±1.2	94.9	91.3±0.7	92.5	54.8±0.4	55.3
w/o Joint components	87.2±3.9	92.6	90.8±0.9	91.9	91.7±0.6	92.8	54.9±0.3	55.3
TripPy-R w/ spanless training	95.2±0.8	96.0	92.0±1.5	94.0	91.1±0.8	92.4	55.1±0.5	55.7
w/o value matching	92.0±1.4	93.6	91.6±1.3	94.5	89.0±0.7	90.0	51.4±0.4	51.9
w/ variants	/	/	/	/	/	/	55.2±0.1	55.3

Table 2: DST results in JGA (± denotes standard deviation). *w/o value matching* refers to training and inference.

For spanless training, the maximum length of random token sequences for the proto-DST model training is 4. The maximum number of epochs is 50 for the WOZ datasets and 100 for sim-M/R. The negative sampling probability is $p_{\text{neg}} = 10\%$.

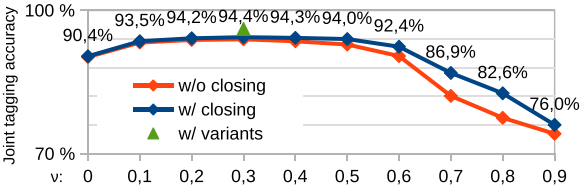

Figure 3: Tagging performance of the proto-DST model depending on the weight threshold ν.

6 Experimental Results

6.1 Learning from Spanless Labels

The quality of the proto-DST for value tagging determines whether or not training without explicit span labels leads to useful DST models. We evaluate the tagging performance on the example of MultiWOZ 2.1 by calculating the ratio of turns for which all tokens are assigned the correct ''*IO*'' tag. Figure 3 plots the joint tagging accuracy across slots, dependent on the weight threshold in Eq. (6). It can be seen that an optimal threshold is $\nu = 0.3$. We found this to be true across all datasets. We also found that the morphological closing operation generally improves tagging accuracy. Typical errors that are corrected by this post-processing are gaps caused by occasionally failing to tag special characters within values, for example, ''`:`'' in times with `hh:mm` format, and imprecisions caused by insecurities of the model when tagging long and complex values such as movie names. Average tagging accuracy across slots is 99.8%. This is particularly noteworthy since values in MultiWOZ can be expressed with a wide variety (e.g., ''expensive'' might be expressed as ''upscale'', ''fancy'', and so on). We attribute the high tagging accuracy to the expressiveness of the encoder-generated semantic contextual representations.

Table 2 lists the JGA of TripPy-R when trained without manual span labels. For the small datasets we did not use x_{NONE} and negative sampling, as it did not make a significant difference. We see that performance is on par with models that were trained with full supervision. If value matching on top of sequence tagging is not used, performance is slightly below its supervised counterparts. We observed that value matching compensates for minor errors caused by the sequence tagger that was trained on automatic labels.[3]

Impact of Tagging Variants While our proto-DST model already achieves very high accuracy on all slots including the ones that expect values with many variants, we tested whether explicit tagging of variants may further improve performance. For instance, if a turn contains the (canonical) value ''expensive'' for slot *hotel-pricerange*, but expressed as ''upscale'', we would explicitly tag such variants. While this strategy further improved the joint tagging accuracy from 94.4% to 96.1% (Figure 3), we did not see a rise in DST performance (Table 2). In other words, the

[3]Note that training with value matching also affects the training of the sequence tagger, be it with or without using span labels.

contextual encoder is powerful enough to endow the proto-DST model with the ability to tag variants of values, based on semantic similarity, which renders any extra supervision for this task unnecessary.

6.2 Handling Sample Sparsity

Impact of Token Noising We experienced that traditional SVD leads to performance gains on sim-M, but not on any of the other tested datasets, confirming Heck et al. (2020b). In contrast, token noising improved the JGA for sim-M/R considerably. Note that in Table 2, the TripPy baseline for sim-M already uses SVD. On MultiWOZ 2.1, we observed minor improvements. As with SVD, WOZ 2.0 remained unaffected. The ontology for WOZ 2.0 is rather limited and remains the same for training and testing. This is not the case for the other datasets, where values occur during testing that were never seen during training. By all appearances, presenting the model with a more diverse set of dropped-out training examples helps generalization more than using a single placeholder token. This seems especially true when there are only few value candidates per slot, and few training samples to learn from. A particularly exemplaric case is found in the sim-M dataset. Without token noising, trained models regularly end up interpreting the value ''last christmas'' as *movie-date* rather than *movie-name*, based on its semantic similarity to dates. Token noising, on the other hand, forces the model to put more emphasis on context rather than token identities, which effectively removes the occurrence of this error.

Impact of History Dropout Table 2 shows that history dropout does not adversely affect DST performance. This is noteworthy because utilizing the full dialogue history is the standard in contemporary works due to its importance for adequate tracking. History dropout effectively reduces the amount of training data by omitting parts of the model input. At the same time, training samples are diversified, preventing the model from memorizing patterns in the dialogue history and promoting generalization. Figure 4 shows the severe effects of over-fitting to the dialogue history on small datasets, when not using history dropout. Here, models were only provided the current turn as input, without historical context. Models with history dropout fare considerably better, showing that they do not over-rely on the historical information. Models without history dropout do not only perform much worse, their performance is also extremely unstable. On sim-R, the span from least to highest relative performance drop is 0% to 39.4%. The results on MultiWOZ point to the importance of the historical information for proper tracking on more challenging scenarios. Here, performance drops equally in the absence of dialogue history, whether or not history dropout was used.

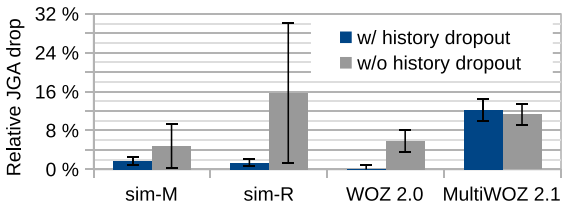

Figure 4: Performance loss due to mismatched training and testing conditions. Here, history is provided during training, but not during testing. sim-M/R and WOZ 2.0 show clear signs of over-fitting without history dropout.

Models	OOO_{Heck}	OOO_{Qian}
TripPy	40.1±1.9	29.2±1.9
Qian et al. (2021)	/	27.0±2.0
TripPy-R	42.2±0.8	29.7±0.7
TripPy-R + masking	**43.0**±1.5	**36.0**±1.6

Table 3: Performance in JGA on artificial out-of-ontology test sets (± denotes standard deviation).

6.3 Handling Unseen Values

We probed value independence on two OOO test sets for MultiWOZ. OOO_{Heck} replaces most values by fictional but still meaningful values that are not in the original ontology. Replacements are consistent, that is, the same value is always replaced by the same fictional stand-in. The overall OOO rate is 84%. OOO_{Qian} replaces only values of slots that expect names (i.e., *name*, *departure*, and *destination*) with values from a different ontology. Replacements are not consistent across dialogues, and such that names are shared across all slots, for example, street names may become hotel names, restaurants may become train stops and so on—that is, the distinction between concepts is lost.

Table 3 lists the results. The performance loss is more graceful on OOO_{Heck}, and we see that

Models	Domains					
	hotel	rest.	attr.	train	taxi	avg.
TRADE (2019; 2020)	19.5	16.4	22.8	22.9	59.2	28.2
MA-DST (2020)	16.3	13.6	22.5	22.8	59.3	26.9
SUMBT (2019; 2020)	**19.8**	16.5	22.6	22.5	59.5	28.2
Li et al. (2021)*	18.5	**21.1**	23.7	**24.3**	59.1	**29.3**
TripPy-R	18.3	15.3	**27.1**	23.7	**61.5**	29.2
Li et al. (2021)**	24.4	26.2	31.3	29.1	59.6	34.1

Table 4: Best zero-shot DST results for various models on MultiWOZ 2.1 in JGA. * Li et al. (2021) presents considerably higher numbers for models with data augmentation. We compare against a model without data augmentation. ** is a model with three times as many parameters as ours.

TripPy-R has an advantage over TripPy. The performance drop is more severe on OOO_{Qian}, with comparable JGA to the baseline of Qian et al. (2021), which is a generative model. The authors of that work attribute the performance degradation to hallucinations caused by memorization effects. For our extractive model the main reason is found in the slot gate. The relative slot gate performance drop for the *train* domain-slots is 23.3%, while for other domain-slots it is 6.4%. We believe the reason is that most of the arbitrary substitutes carry no characteristics of train stops, but of other domains instead. This is less of a problem for the *taxi* domain for instance, since taxi stops are of a variety of location types. The issue of value-to-domain mismatch can be mitigated somewhat with informed value masking in system utterances (Section 4.3). While this does not particularly affect our model on the regular test set or on the more domain-consistent OOO_{Heck}, we can see much better generalization on OOO_{Qian}.

6.4 Handling Unseen Slots and Domains

Table 2 shows that moving from slot specific to slot independent components only marginally affects DST performance, while enabling tracking of dialogues with unseen domains and slots.

Zero-shot Performance We conducted zero-shot experiments on MultiWOZ 2.1 by excluding all dialogues of a domain d from training and then evaluating the model on dialogues of d. In Table 4, we compare TripPy-R to recent models that support slot independence. Even though we did not specifically optimize TripPy-R for zero-shot abilities, our model shows a level of robustness that is competitive with other contemporary methods.

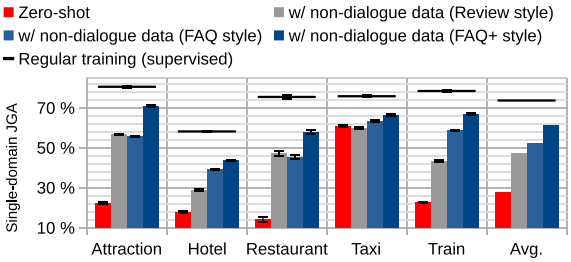

Figure 5: Performance of TripPy-R after training with non-dialogue style data from a held-out domain.

Impact of Non-dialogue Data Besides zero-shot abilities, we were curious, is it feasible to improve dialogue state tracking by learning the required mechanics purely from non-dialogue data? This is a non-trivial task, as the model needs to generalize knowledge learned from unstructured data to dialogue, that is, sequences of alternating system and user utterances. We conducted this experiment by converting MultiWOZ dialogues of a held-out domain d into non-dialogue format for training. For d, the model only sees isolated sentences or sentence pairs without any structure of a dialogue. Consequently, there is no ‘‘turn’’ history from which the model could learn. The assumption is that one would have some way to label sequences of interest in non-dialogue sentences, for instance with a semantic parser. As this is a feasibility study, we resort to the slot labels in DS_t, which simulates having labels of very high accuracy. We tested three different data formats, (1) *Review style*: Only system utterances with statements are used to learn from; (2) *FAQ style*: A training example is formed by a user question and the following system answer. Note that this is contrary to what TripPy-R naturally expects, which is a querying system and a responding user; and (3) *FAQ+ style*: Combines review and FAQ style examples and adds user questions again as separate examples.

Figure 5 shows that we observed considerable improvements across all held-out domains when using non-dialogue data to learn from. Learning from additional data, even if unstructured, is particularly beneficial for unique slots, such as the *restaurant-food* slot which the model can not learn about from any other domain in MultiWOZ (as

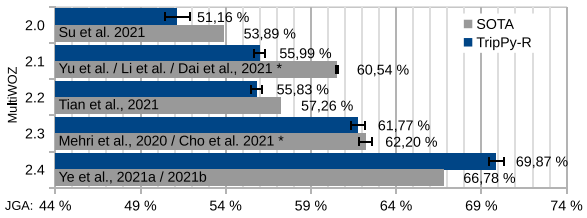

Figure 6: Comparison of TripPy-R and SOTA open vocabulary DST models. * denotes TripPy-style models.

is reflected in a poor zero-shot performance as well). We also found that learning benefits from the combination of different formats. The heightened performance given the FAQ+ style data is not an effect of more data, but of its presentation, since we mainly *re-use* inputs with different formats. This observation is reminiscent of findings in psychology. Horst et al. (2011) showed that children benefited from being read the same story repeatedly. Furthermore, Johns et al. (2016) showed that contextual diversity positively affects word learning in adults. Note that this kind of learning is in contrast to few-shot learning and leveraging artificial dialogue data, which either require fine-grained manual labels or high-level knowledge of how dialogues are structured. Even though the data we used is far-removed from what a dialogue state tracker expects, TripPy-R still manages to learn how to appropriately track these new domains.

6.5 Performance Comparison

We evaluated on five versions of MultiWOZ to place TripPy-R among contemporary work. Versions 2.1 and 2.2 mainly propose general corrections to the labels of MultiWOZ 2.0. Version 2.3 unifies annotations between dialogue acts and dialogue states. In contrast, version 2.4 removes all values that were mentioned by the system from the dialogue state, unless they are proper names. Figure 6 plots the results. The performance of TripPy-R is considerably better on versions 2.3 and 2.4. This can be attributed to a more accurate prediction of the $inform$ cases due to better test ground truths.

For fairness, we restricted our comparison to models that have the same general abilities, that is, they ought to be open-vocabulary and without data-specific architectures. The SOTA on 2.0 (Su et al., 2022) proposes a unified generative dialogue model to solve multiple tasks including DST and benefits from pre-training on various dialogue corpora. While profiting from more data in general, its heterogeneity in particular did not affect DST performance. Yu et al. (2020), Li et al. (2020), and Dai et al. (2021) currently share the top of the leaderboard for 2.1, all of which propose TripPy-style models that leverage data augmentation. The main reason for performance improvements lies in the larger amount of data and in diversifying samples. TripPy-R does not rely on more data, but diversifies training samples with token noising and history dropout. On version 2.2, the method of Tian et al. (2021) performs best with a two-pass generative approach that utilizes an error recovery mechanism. This mechanism can correct generation errors such as caused by hallucination, which is a phenomenon that does not occur with TripPy-R. However, their error recovery also has the potential to avoid propagation of errors made early in the dialogue, which is demonstrated by a heightened performance. Cho et al. (2021) report numbers for the method of Mehri et al. (2020) on version 2.3, which is another TripPy-style model using an encoder that was pre-trained on millions of conversations, thus greatly benefiting from specialized knowledge. For version 2.4, the current SOTA with the properties as stated above is presented by Ye et al. (2021b) and reported in Ye et al. (2021a), which is now surpassed by TripPy-R. The major difference to our model is the use of slot self-attention, which allows their model to learn correlations between slot occurrences. While TripPy-R does not model slot correlations directly, it does however explicitly learn to resolve coreferences.

6.6 Implications of the Results

The zero-shot capabilities of our proposed TripPy-R model open the door to many new applications. However, it should be noted that its performance on an unseen arbitrary domain and on unseen arbitrary slots will likely degrade. In such cases it would be more appropriate to perform adaptation, which the TripPy-R framework facilitates. This means that one would transfer the model as presented in Sections 4.3 and 4.4 and continue fine-tuning with limited—and potentially unstructured (see Section 6.4)—data from the new domain. Nonetheless, in applications such as e-commerce (Zhang et al., 2018) or customer support (García-Sardiña et al., 2018), whenever new slots or even domains are introduced, they

are to a great extent related to ones that a deployed system is familiar with. We believe that the zero-shot performance presented in Table 4 is highly indicative of this set-up, as MultiWOZ domains are different, yet to some extent related.

Further, the TripPy-R model facilitates future applications in complex domains such as healthcare. One of the biggest obstacles to harnessing large amounts of natural language data in healthcare is the required labeling effort. This is particularly the case for applications in psychology, as can be seen from the recent work of Rojas-Barahona et al. (2018), where only 5K out of 1M interactions where labeled with spans for so called *thinking-errors* by physiologists. A framework like TripPy-R can completely bypass this step by utilizing its proto-DST, as presented in Section 4.1, eliminating the overbearing labeling effort.

7 Conclusion

In this work we present methods to facilitate robust extractive dialogue state tracking with weak supervision and sparse data. Our proposed architecture—TripPy-R—utilizes a unified encoder, the attention mechanism, and conditioning on natural language descriptions of concepts to facilitate parameter sharing and zero-shot transfer. We leverage similarity based value matching as an optional step after value extraction, without violating the principle of ontology independence.

We demonstrated the feasibility of training without manual span labels using a self-trained proto-DST model. Learning from spanless labels enables us to leverage data with weaker supervision. We showed that token noising and history dropout mitigate issues of pattern memorization and train-test discrepancies. We achieved competitive zero-shot performance and demonstrated in a feasibility study that TripPy-R can learn to track new domains by learning from non-dialogue data. We achieve either competitive or state-of-the-art performance on all tested benchmarks. For future work we continue to investigate learning from non-dialogue data, potentially in a continuous fashion over the lifetime of a dialogue system.

Acknowledgments

We thank the anonymous reviewers and the action editors for their valuable feedback. M. Heck, N. Lubis, C. van Niekerk, and S. Feng are supported by funding provided by the Alexander von Humboldt Foundation in the framework of the Sofja Kovalevskaja Award endowed by the Federal Ministry of Education and Research, while C. Geishauser and H.-C. Lin are supported by funds from the European Research Council (ERC) provided under the Horizon 2020 research and innovation programme (grant agreement no. STG2018804636). Computing resources were provided by Google Cloud and HHU ZIM.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450v1.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA.

Ankur Bapna, Gokhan Tür, Dilek Hakkani-Tür, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. In *Proceedings of Interspeech 2017*, pages 2476–2480. https://doi.org/10.21437/Interspeech.2017-518

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, 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*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1547

Giovanni Campagna, Agata Foryciarz, Mehrad Moradshahi, and Monica Lam. 2020. Zero-shot transfer learning with synthesized data for multi-domain dialogue state tracking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 122–132, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.12

Guan-Lin Chao and Ian Lane. 2019. BERT-DST: Scalable end-to-end dialogue state tracking

with bidirectional encoder representations from transformer. In *Proceedings of Interspeech 2019*, pages 1468–1472. https://doi.org/10.21437/Interspeech.2019-1355

Hyundong Cho, Chinnadhurai Sankar, Christopher Lin, Kaushik Ram Sadagopan, Shahin Shayandeh, Asli Celikyilmaz, Jonathan May, and Ahmad Beirami. 2021. CheckDST: Measuring real-world generalization of dialogue state tracking performance. *CoRR*, abs/2112.08321v1.

Herbert H. Clark and Susan E. Brennan. 1991. Grounding in communication. In Lauren B. Resnick, John M. Levine, and Stephanie D. Teasley, editors, *Perspectives on Socially Shared Cognition*, pages 127–149. American Psychological Association, Washington, USA. https://doi.org/10.1037/10096-006

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

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2021. Survey on evaluation methods for dialogue systems. *Artificial Intelligence Review*, 54(1):755–810. https://doi.org/10.1007/s10462-020-09866-x, PubMed: 33505103

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.

Jens Edlund, Joakim Gustafson, Mattias Heldner, and Anna Hjalmarsson. 2008. Towards human-like spoken dialogue systems. *Speech Communication*, 50(8–9):630–645. https://doi.org/10.1016/j.specom.2008.04.002

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

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tür. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 264–273, Stockholm, Sweden. Association for Computational Linguistics.

Laura García-Sardiña, Manex Serras, and Arantza del Pozo. 2018. ES-Port: A spontaneous spoken human-human technical support corpus for dialogue research in Spanish. In *Proceedings of the 11th International Conference on Language Resources and Evaluation*, pages 781–785, Miyazaki, Japan. European Language Resources Association.

Ting Han, Ximing Liu, Ryuichi Takanabu, Yixin Lian, Chongxuan Huang, Dazhen Wan, Wei Peng, and Minlie Huang. 2021. MultiWOZ 2.3: A multi-domain task-oriented dialogue dataset enhanced with annotation corrections and co-reference annotation. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 206–218. Springer. https://doi.org/10.1007/978-3-030-88483-3_16

Michael Heck, Christian Geishauser, Hsien-chin Lin, Nurul Lubis, Marco Moresi, Carel van Niekerk, and Milica Gašić. 2020a. Out-of-task training for dialog state tracking models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6767–6774, Barcelona, Spain (Online). International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.596

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco

Moresi, and Milica Gašić. 2020b. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21st Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (GELUs). *CoRR*, abs/1606.08415v4.

Jessica S. Horst, Kelly L. Parsons, and Natasha M. Bryan. 2011. Get the story straight: Contextual repetition promotes word learning from storybooks. *Frontiers in Psychology*, 2:17. https://doi.org/10.3389/fpsyg.2011.00017, PubMed: 21713179

Brendan T. Johns, Melody Dye, and Michael N. Jones. 2016. The influence of contextual diversity on word learning. *Psychonomic Bulletin & Review*, 23(4):1214–1220. https://doi.org/10.3758/s13423-015-0980-7, PubMed: 26597891

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*. San Diego, CA, USA.

Adarsh Kumar, Peter Ku, Anuj Kumar Goyal, Angeliki Metallinou, and Dilek Hakkani-Tür. 2020. MA-DST: Multi-attention based scalable dialog state tracking. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, volume 34, pages 8107–8114. https://doi.org/10.1609/aaai.v34i05.6322

Staffan Larsson and David R. Traum. 2000. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering*, 6(3–4):323–340. https://doi.org/10.1017/S1351324900002539

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5478–5483, Florence, Italy. 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*.

Shuyang Li, Jin Cao, Mukund Sridhar, Henghui Zhu, Shang-Wen Li, Wael Hamza, and Julian McAuley. 2021. Zero-shot generalization in dialog state tracking through generative question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1063–1074, Online. Association for Computational Linguistics.

Shuailong Liang, Lahari Poddar, and Gyuri Szarvas. 2021. Attention guided dialogue state tracking with sparse supervision. *CoRR*, abs/2101.11958v1.

Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Zhenpeng Zhou, Paul Crook, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, and Pascale Fung. 2021. Zero-shot dialogue state tracking via cross-task transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7890–7900, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bing Liu and Ian Lane. 2017. An end-to-end trainable neural network model with belief tracking for task-oriented dialog. In *Proceedings of Interspeech 2017*, pages 2506–2510. https://doi.org/10.21437/Interspeech.2017-1326

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692v1.

Yue Ma, Zengfeng Zeng, Dawei Zhu, Xuan Li, Yiying Yang, Xiaoyuan Yao, Kaijie Zhou, and Jianping Shen. 2019. An end-to-end dialogue

state tracking system with machine reading comprehension and wide & deep classification. *CoRR*, abs/1912.09297v2.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tür. 2020. DialoGLUE: A natural language understanding benchmark for task-oriented dialogue. *CoRR*, abs/2009.13570v1.

Nikola Mrkšić, Diarmuid Ó. Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/P17-1163

Mahdi Namazifar, Alexandros Papangelis, Gokhan Tür, and Dilek Hakkani-Tür. 2021. Language model is all you need: Natural language understanding as question answering. In *Proceedings of the 2021 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7803–7807. IEEE. https://doi.org/10.1109/ICASSP39728.2021.9413810

Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, Vinay Adiga, and Erik Cambria. 2021. Recent advances in deep learning based dialogue systems: A systematic survey. *CoRR*, abs/2105.04387v4.

Elnaz Nouri and Ehsan Hosseini-Asl. 2018. Toward scalable neural dialogue state tracking model. In *Proceedings of the 2nd Conversational AI workshop at the 32nd Conference on Neural Information Processing Systems*. Montréal, Canada.

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.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the 3rd Workshop on Very Large Corpora*, pages 82–94.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020a. Schema-guided dialogue state tracking task at DSTC8. *CoRR*, abs/2002.01359v1.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020b. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696. https://doi.org/10.1609/aaai.v34i05.6394

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2780–2786, Brussels, Belgium. Association for Computational Linguistics.

Lina M. Rojas-Barahona, Bo-Hsiang Tseng, Yinpei Dai, Clare Mansfield, Osman Ramadan, Stefan Ultes, Michael Crawford, and Milica Gašić. 2018. Deep learning for language understanding of mental health concepts derived from cognitive behavioural therapy. In *Proceedings of the 9th International Workshop on Health Text Mining and Information Analysis*, pages 44–54, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-5606

Jean Serra. 1982. *Image Analysis and Mathematical Morphology*. Academic Press, Inc.

Pararth Shah, Dilek Hakkani-Tür, Gökhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry P. Heck. 2018. Building a conversational agent overnight with dialogue self-play. *CoRR*, abs/1801.04871v1.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676, Dublin, Ireland. Association for Computational Linguistics.

Xin Tian, Liankai Huang, Yingzhan Lin, Siqi Bao, Huang He, Yunyi Yang, Hua Wu,

Fan Wang, and Shuqi Sun. 2021. Amendable generation for dialogue state tracking. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 80–92, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.nlp4convai-1.8

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

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

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144v2.

Puyang Xu and Qi Hu. 2018. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1448–1457, Melbourne, Australia. Association for Computational Linguistics.

Puyang Xu and Ruhi Sarikaya. 2014. Targeted feature dropout for robust slot filling in natural language understanding. In *Fifteenth Annual Conference of the International Speech Communication Association*, pages 258–262.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2021a. MultiWOZ 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. *CoRR*, abs/2104.00773v1.

Fanghua Ye, Jarana Manotumruksa, Qiang Zhang, Shenghui Li, and Emine Yilmaz. 2021b. Slot self-attentive dialogue state tracking. In *Proceedings of the Web Conference 2021*, pages 1598–1608.

Steve Young, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The hidden information state model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174. https://doi.org/10.1016/j.csl.2009.04.001

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2020. SCoRe: Pre-training for context representation in conversational semantic parsing. In *International Conference on Learning Representations*.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2: A dialogue dataset with additional annotation

corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.nlp4convai-1.13

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wang, Philip Yu, Richard Socher, and Caiming Xiong. 2020a. Find or classify? Dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the 9th Joint Conference on Lexical and Computational Semantics*, pages 154–167, Barcelona, Spain (Online). Association for Computational Linguistics.

Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. 2020b. Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences*, 63:2011–2027. https://doi.org/10.1007/s11431-020-1692-3

Zhuosheng Zhang, Jiangtong Li, Pengfei Zhu, Hai Zhao, and Gongshen Liu. 2018. Modeling multi-turn conversation with deep utterance aggregation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3740–3752, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467, Melbourne, Australia. Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-1135