# Improving Dialogue State Tracking with Turn-based Loss Function and Sequential Data Augmentation

**Jarana Manotumruksa**
University College London
j.manotumruksa@ucl.ac.uk

**Jeffrey Dalton**
University of Glasgow
jeff.dalton@glasgow.ac.uk

**Edgar Meij**
Bloomberg
emeij@bloomberg.net

**Emine Yilmaz**
University College London, Amazon
emine.yilmaz@ucl.ac.uk

## Abstract

While state-of-the-art Dialogue State Tracking (DST) models show promising results, all of them rely on a traditional cross-entropy loss function during the training process, which may not be optimal for improving the joint goal accuracy. Although several approaches recently proposed augmenting the training set by copying user utterances and replacing the real slot values with other possible or even similar values, they are not effective at improving the performance of existing DST models. To address these challenges, we propose a Turn-based Loss Function (TLF) that penalises the model if it inaccurately predicts a slot value at the early turns more so than in later turns in order to improve joint goal accuracy. We also propose a simple but effective Sequential Data Augmentation (SDA) algorithm to generate more complex user utterances and system responses to effectively train existing DST models. Experimental results on two standard DST benchmark collections demonstrate that our proposed TLF and SDA techniques significantly improve the effectiveness of the state-of-the-art DST model by approximately 7-8% relative reduction in error and achieves a new state-of-the-art joint goal accuracy with 59.50 and 54.90 on MultiWOZ2.1 and MultiWOZ2.2, respectively.

## 1 Introduction

Task-based Virtual Personal Assistants (VPAs) interact with users in natural language to help complete tasks such as making hotel bookings and restaurant reservations. Dialogue State Tracking (DST) is an essential component for VPAs that aims to track the dialogue state from the user's utterances at each turn (Rastogi et al., 2019). Based on the current dialogue state, VPAs decide the next action to perform. In general, existing DST models rely on an ontology that defines slots for a particular domain/task (e.g. hotel-name and taxi-destination). To accomplish the tracking task, given the user's current utterance, a slot to track and dialogue history, the DST models need to 1) predict if the user has mentioned the given slot and 2) if so, predict/extract its value from the current utterance.

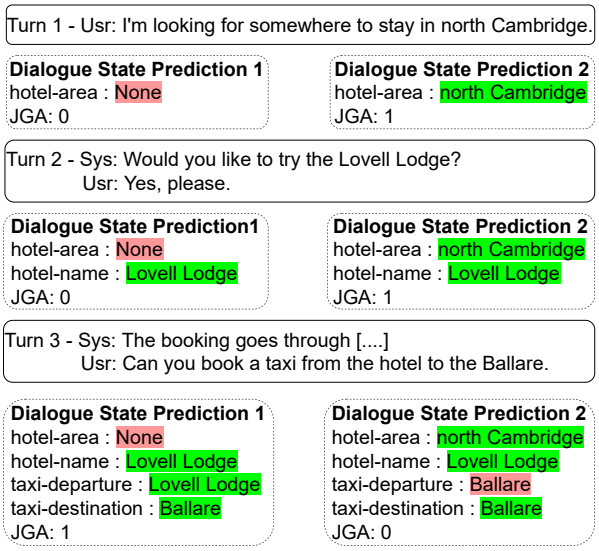



Figure 1: An example of multi-domain dialogue. The terms highlighted in green and red represent correctly and incorrectly predicted dialogue states respectively.

Joint Goal Accuracy (JGA) is a widely used metric to evaluate the effectiveness of DST models (Zang et al., 2020; Eric et al., 2019; Shah et al., 2018; Wen et al., 2017). At each turn, the joint goal accuracy is 1.0 if and only if all domain-slot and value pairs are predicted correctly, otherwise 0. The existing DST models rely on the traditional cross-entropy loss function during the training process. We argue that this is not effective for optimizing joint goal accuracy. We illustrate this issue in Figure 1 in Dialogue State Prediction 1 & 2, with the traditional cross-entropy loss function, if two models only make one mistake during the training process they will be penalised equally. However, the consequence that the first model incorrectly predicts the value at the first turn is worse than when the second model fails to predict the value at the



*Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1674–1683
November 7–11, 2021. 

forth turn (i.e. average JGA across 3 turns is 0 and 0.66 for model 1 and model 2).

Training current DST models currently requires annotated dialogue datasets that cover a wide variety of diverse conversation flows. However, existing dialogue datasets (e.g. MultiWOZ2.1 (Eric et al., 2019)) are relatively small and do not provide coverage of all slot values for open-vocabulary slots (e.g. restaurant name and destination). To alleviate this problem, several recent data augmentation techniques propose (Giovanni et al., 2020; Summerville et al., 2020; Song et al., 2020) replacing the ground-truth values of particular slots using additional information (e.g. restaurant and movie name corpus). Although these augmented dialogues increase coverage of all slot values, the complexity of the dialogues remains the same[1].

To address the aforementioned challenges, we propose a novel Turn-based Loss Function and Sequential Data Augmentation algorithm that improves the effectiveness of DST models. Our contributions are:

- We modify the traditional cross-entropy loss function to take into account the turn information during the training process. Our proposed Turn-based Loss Function (TLF) penalises the DST models more heavily if they fail to predict dependent slot values in subsequent turns. To the best of our knowledge, this work is the first to incorporate turn dependence into the loss function of the DST models.

- We propose a simple but effective Sequential Data Augmentation algorithm (SDA) to generate complex dialogues that can be used to train DST models to generalize more effectively.

- We conduct comprehensive experiments on two DST benchmark datasets. Experimental results demonstrate that our TLF and SDA approaches consistently and significantly improve the effectiveness of the state-of-the-art DST model in terms of joint goal accuracy. In particular, we study the state-of-the-art DST model behaviour based on turn depth, domains, slot complexity, and robustness using perturbed dialogue history. We find the model does not perform well on later turn depths, dialogues with more active slots, and does not depend heavily on aspects of the dialog history, while the model with our proposed TLF and SDA approach can effectively address these challenges.

[1]We will further explain the complexity of dialogues in Section 4.2

## 2 Related Work

DST models can be categorised into two types: the ontology-based (Zhang et al., 2019; Chen et al., 2020) and span-based models (Heck et al., 2020; Kim et al., 2020; Wu et al., 2019). Zhang et al. (2019) propose an ontology-based DST model that leverages a pre-defined ontology to predict dialogue state based on the similarity between the encoded candidate values and encoded user utterance and slot description. Recent work in DST focuses on the span-based approach to address the scalability and generalisation issues of previous ontology-based models. Wu et al. (2019) proposed a scalable span-based DST model that encodes the whole dialogue context and decodes the value for every slot using a copy-augmented decoder. Recently, several DST models (Kim et al., 2020; Heck et al., 2020) incorporate the predicted dialogue state from previous turns when tracking the dialogue state at the current turn using a copy mechanism.

Data augmentation is widely used to improve the effectiveness of the existing DST models (Hou et al., 2018; Giovanni et al., 2020; Song et al., 2020). Hou et al. (2018) use a sequence-to-sequence model and delexicalisation to generate a variety of diverse utterances based on the original utterances. These generated augmented utterances help improve the language understanding of the DST models. In addition, the span-based DST models often encounter out-of-vocabulary words (e.g. unseen restaurant name) at inference time. As a result, these DST models are likely to fail to extract unseen words from the utterances. To address this problem, Summerville et al. (2020) augment the training dataset by randomly replacing original slot values with other possible values obtained from external corpora (e.g. restaurant name corpus). Similar to (Summerville et al., 2020), Song et al. (2020) augment the training data by copying user utterances and replace the ground-truth slot values with randomly generated strings. Recently, Li et al. (2021) proposed to use the pre-trained utterance generator and counterfactual goal generator to create novel user utterances that are correlated with the original system response. Their approach showed significant improvement on the DST performance.

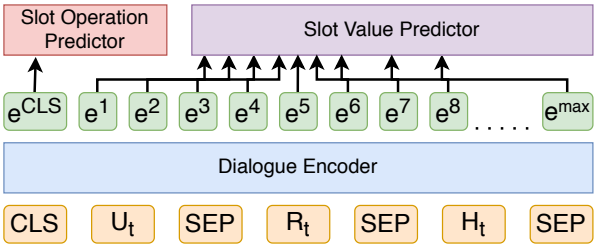



Figure 2: Generic architecture of existing DST models.

## 3 Neural Models for DST

We first formalise the task of Dialogue State Tracking and define key notations. Then, we briefly describe the general architecture of existing neural network DST models (see Figure 2 that consist of three main components: the dialogue encoder, the slot operation predictor, and the slot value predictor.

### 3.1 Problem Statement and Notation

DST tracks the state of the user at a particular turn given the user's utterance and system response. Let $\mathcal{X} = \{(U_1, R_1), (U_2, R_2), ..., (U_T, R_T)\}$ be the sequence of user utterance $U$ and system response $R$ pairs, given a dialogue context with $T$ turns. Each $(U_t, R_t)$ pair can involve a single or multiple domains (e.g. restaurant and taxi) and a certain number of slots (e.g. restaurant-name and taxi-destination) associated with the domains. Let $\mathcal{B} = \{B_1, B_2, ..., B_t\}$ be the dialogue state of the user for each turn. We denote all the $N$ possible domain-slot pairs as $\mathcal{S} = \{s_1, s_2, ..., s_N\}$. Each dialogue state $B_t$ is a set of tuples $(s, v)$, where $s \in \mathcal{S}$ is a domain-slot pair and $v$ is a value associated with the domain-slot $s$.

### 3.2 The Dialogue Encoder

The dialogue encoder is the core component of DST models that aims to capture the user's intent from the dialogue context (see the blue box in Figure 2). The input of the dialogue encoder is the dialogue context at turn $t$ that consists of the current utterance $U_t$, system response $R_t$ and dialogue history $H_t = (U_{t-1}, R_{t-1}), ..., (U_1, R_1)$. Existing DST models exploit pre-trained language models (e.g. BERT(Devlin et al., 2019)) to encode the input as follows:

$$E_t = BERT([C] \oplus U_t \oplus [S] \oplus R_t \oplus [S] \oplus H_t \oplus [S]),$$

where $\oplus$ is the concatenation operation, $[C]$ and $[S]$ are BERT's special CLS and SEP tokens. $E_t = [e_t^{CLS}, e_t^1, ..., e_t^{seq_{max}}]$ is the output of the dialogue encoder that represent each token in the dialogue context. In particular, $e_t^{CLS} \in \mathcal{R}^d$, where $d$ is BERT's contextual embedding dimension, is the aggregated representation of the total input tokens that captures the user's intent from the whole dialogue context, while $[e_t^1, e_t^2, ..., e_t^{seq_max}]$ is the token-level representation.

### 3.3 The Slot Operation Predictor

The slot operation predictor aims to predict an operation for each slot as one of the slot operations $O_{slot} = \{none, dontcare, update\}$ (see the red box in Figure 2). *none* and *dontcare* operations denotes that the slot does not take a value or could be any value, respectively. The *update* operation denotes that a value of the given slot could be predicted or extracted from the current utterance $U_t$ (see Turn 1 & 2 in Figure 1). If the slot operation predictor predicts that a value of the given domain-slot pair then the DST models will obtain the value from the slot value predictor described in Section 3.4. The input to the slot operation predictor is the aggregated representation $e_t^{CLS}$ and the probability distribution over the slot operations $O_{slot}$ for domain-slot pair $s$ at turn $t$ is defined as follows:

$$\hat{y}_{t,s}^{slot} = softmax(W_s^{slot} \cdot (e_t^{CLS} \oplus i_t \oplus ds_t)^\top + b_s^{slot}) \quad (1)$$

where $W_s^{slot} \in \mathcal{R}^{|O_{slot}| \times d}$ and $b_s^{slot}$ are learnable parameters and bias. Then, the cross-entropy loss function for the slot operation prediction is defined as follows:

$$\mathcal{L}_{slot} = \sum_{t=1}^{T} \sum_{s=1}^{N} -log(y_{t,s}^{slot} \cdot (\hat{y}_{t,s}^{slot})^\top) \quad (2)$$

where $y_{t,s}^{slot}$ is the one-hot slot operation label for domain-slot pair $s$ at turn $t$.

### 3.4 The Slot Value Predictor

The final component of DST models is the slot value predictor that aims to extract a value for each domain-slot pair from the dialogue context (the violet box in Figure 2). The slot value predictor takes the token-level representations $[e_t^1, e_t^2, ..., e_t^{seq_{max}}]$ of the entire dialogue context for turn $t$ as input and applies a two-way linear mapping to compute the probability of the terms being the start and the end position of the span for slot $s$, $\hat{y}_{t,s}^{start}$ and $\hat{y}_{t,s}^{end}$,

respectively, as follows:

$$[\alpha_t^s, \beta_t^s] = W_s^{value} \cdot ([[e_t^1, e_t^2, ..., e_t^{seq_{max}}]]) + b_s^{value}$$
$$\hat{y}_{t,s}^{start} = softmax(\alpha_t^s)$$
$$\hat{y}_{t,s}^{end} = softmax(\beta_t^s)$$

Similar to the slot operation predictor's loss function (Equation (2)), the loss function for the slot value prediction, $\mathcal{L}_{value}$, is defined as follows:

$$\sum_{t=1}^{T}\sum_{s=1}^{N} \frac{-log(y_{t,s}^{start} \cdot (\hat{y}_{t,s}^{start})^\top) - log(y_{t,s}^{end} \cdot (\hat{y}_{t,s}^{end})^\top}{2} \quad (3)$$

where $y_{t,s}^{start}$ and $y_{t,s}^{end}$ are the one-hot start and end position label for domain-slot pair $s$ at turn $t$. Finally, the DST models are trained using the following joint loss function:

$$\mathcal{L} = \mu_{slot} \cdot \mathcal{L}_{slot} + \mu_{value} \cdot \mathcal{L}_{value}, \quad (4)$$

where $\mu_{slot}$ and $\mu_{value}$ are hyperparameters that control the weights of the slot operation prediction and the slot value prediction, respectively. Note that the joint loss function in Equation (4) is widely used by the existing DST models (e.g. (Heck et al., 2020; Kim et al., 2020; Zhang et al., 2019)).

# 4 Proposed Methods

We now describe the proposed methods that improve the effectiveness of DST model.

## 4.1 Turn-based Loss Function

We start by describing our Turn-based Loss Function which improves the effectiveness of the core DST model. As discussed in Section 1, most existing DST models (e.g. (Heck et al., 2020; Kim et al., 2020; Zhang et al., 2019)) still rely on the traditional cross-entropy loss function during the training process, which may not be optimal to improve the joint goal accuracy. To address this, we incorporate the turn information during the training process. Our proposed TLF penalises the DST model more heavily if it inaccurately predicts a slot value at the early turns than the later turns. This is important to avoid the error cascade in early turns that results in highly degraded JGA in later turns. To model this dependency explicitly during the training process we modify the joint loss function as shown in Equation (4) as follows:

$$\mathcal{L} = [\mu_{slot} \cdot \mathcal{L}_{slot} + \mu_{value} \cdot \mathcal{L}_{value}] \cdot [1 + (T - t) * \lambda] \quad (5)$$

where $T$ is the total number of turns for a given dialogue, $t$ is the current turn number and $\lambda$ is a turn weight parameter that controls the influence of the turn-based penalty. For example, if a given dialogue consists of 5 turns ($T = 5$), the model will be penalised more heavily if it makes a mistake at the first turn ($t = 1$) than the last ($t = 5$).

## 4.2 Sequential Data Augmentation

Our proposed Sequential Data Augmentation algorithm improves the generalizability of DST models. The overall training process of DST algorithms with SDA is summarised in Algorithm 1. In particular, for each turn $t$, given the current utterance $U_t$, system response $R_t$ and dialogue history $H_t$, we generate augmented training data by concatenating $U_t$ and $R_t$ with $U_{t+1}, U_{t+2}, ..., U_{t+\eta}$ and $R_{t+1}, R_{t+2}, ..., R_{t+\eta}$, respectively. The hyperparameter $\eta$ controls the complexity of the augmented dialogues. For example, we can generate augmented data for the dialogue in turn 1 in Figure 1 with $\eta = 2$, as follows:

> $U_1^{augment}$ = I'm looking for somewhere to stay in north Cambridge. Yes, Please. Can you book a taxi from the hotel to the Ballare
> $R_1^{augment}$ = Would you like to try the lovell lodge? The booking goes through [...].

The larger $\eta$ is, the more complex the augmented dialogue becomes. We hypothesize that the complex augmented dialogues help the DST model to learn to more effectively track dialogue state for several reasons. First, the longer augmented dialogues help the model to understand the intent deeper in the conversation than the original dialogues, which are often relatively short. Second, the augmented dialogues contain more ground truth labels than the original dialogues[2], which helps to train the DST models more effectively to extract the domain-slot values from the utterance and system response. For example, in Figure 1, the utterance on turn 1, $U_1$, consists only of one ground truth label, whereas the augmented utterance $U_1^{augment}$ and the augmented system response $R_1^{augment}$ contains 3 ground truth labels, highlighted in green. Our proposed SDA algorithm differs from the previous data augmentation algorithms (e.g. (Summerville et al., 2020;

[2]Indeed, on average, the user's utterance and system's response in the original dialogues contain only a few ground truth labels(Zang et al., 2020; Eric et al., 2019)

Song et al., 2020)) in two key ways: 1) SDA takes into account the sequential property of dialogues when generating the augmented training data, while (Summerville et al., 2020; Song et al., 2020) do not and 2) (Summerville et al., 2020; Song et al., 2020) require external information (e.g. a restaurant name corpus), while SDA does not.

**Algorithm 1** Training process for existing DST models with our TLF and SDA approaches

**Input:** training data $\mathcal{X}$, hyperparameter $\eta$
**Output:** DST model's parameters $\Theta$
Initialise $\Theta$ with pre-trained BERT
**repeat**
  **for** $x \in \mathcal{X}$ **do**
    **for** $t \leftarrow 1$ to $T_x$ **do**
      $U_t, R_t, H_t, i_t, ds_t = x_t$
      Compute $\mathcal{L}$ (Eq.(5))
      Update model's params $\Theta$
    **end for**
    **for** $j \leftarrow 1$ to $\eta$ **do** // Seq. Data Augmentation
      **for** $t \leftarrow 1$ to $T_x$ **do**
        Init augment dialogue $U_a, R_a = U_t, R_t$
        **for** $k \leftarrow t+1$ to $t+j$ **do**
          $U_k, R_k, H_k, i_k, ds_k = x_k$
          $U_a = U_a \oplus U_k$
          $R_a = R_a \oplus R_k$
          Compute $\mathcal{L}$ (Eq.(5))
          Update model's params $\Theta$
        **end for**
      **end for**
    **end for**
  **end for**
**until** convergence

## 5 Experimental Setup

We conduct experiments on the two most widely used multi-domain task-based dialogue state tracking datasets (MultiWOZ2.1 (Eric et al., 2019) and MultiWOZ2.2 (Zang et al., 2020)). These two are the largest datasets which contain over 10,000 dialogues across seven domains: restaurant, taxi, attraction, hotel, train, hospital and police. Following (Wu et al., 2019; Zhang et al., 2019), we remove hospital and police domains in MultiWOZ2.1 and MultiWOZ2.2 because they only appear in the training dataset. This results in five domains with 30 domain-slot pairs. We use the standard training/validation/test splits provided in the original datasets. Following previous literature (Heck et al., 2020; Zhang et al., 2019), we evaluate all the DST models using the Joint Goal Accuracy (JGA) metric (Henderson et al., 2014). At each turn JGA is 1.0 if and only if all domain-slot and values pairs are correctly predicted, otherwise 0. The score is averaged across all turns in the test set.

### 5.1 Baseline Models

We compare our proposed approaches with a variety of recent DST baselines. **TRADE** (Wu et al., 2019) encodes the whole dialogue context using bidirectional Gated Recurrent Units (GRU) and generates the value for every slot using the GRU-based copy mechanism. **Picklist-DST** (Zhang et al., 2019) is the ontology-based DST model that requires a pre-defined ontology with all possible values for each domain-slot pair. **DS-DST** (Zhang et al., 2019) is a hybrid DST model that jointly trains both the ontology- and span-based models. **SOM-DST** (Kim et al., 2020) is the span-based DST model that uses the copy-mechanism for the slot operation prediction and uses GRU for the slot value prediction. **TripPy** (Heck et al., 2020) is the state-of-the-art span-based DST model that uses the triple copy mechanism to track the dialogue state. **DialoGLUE** (Mehri et al., 2020) is the TripPy model that uses ConvBERT, a fine-tuned BERT trained on an open-domain dialogue corpus consisting of 700M conversations, as dialogue encoder. **TripPy-V** is the TripPy model that uses the existing Value-based Data Augmentation (VDA) (Summerville et al., 2020), that randomly replaces original slot values with other possible values. **TripPy-CoCo** (Li et al., 2021) is the TripPy model trained on the augmented data generated by the Controllable Counterfactual (CoCo) data augmentation algorithm that consists of three main components: value substitution, controllable counterfactual generation and classifier filter.

### 5.2 Implementation Details

We implement our proposed TLF and SDA approaches using PyTorch[3]. The hyperparameters of TLF and SDA (i.e. the turn weight parameter $\lambda$ and the sequence number parameter $\eta$) are tuned on the validation set. We use the pre-trained BERT-base-uncased model(Devlin et al., 2019) with 12 hidden layers and embedding dimension $d = 768$ as the dialogue encoder[4]. For all baselines, we optimise them similarly using cross-entropy loss and the Adam optimiser (Kingma and Ba, 2014) with a learning rate of $2e^{-5}$. For the hyperparameters, we use the optimized parameters reported in the original papers.

[3] https://github.com/feay1234/TLF-SDA
[4] https://huggingface.co/transformers/pretrained_models.html

Table 1: Effectiveness of various DST models on MultiWOZ2.1 and MultiWOZ2.2 with the best result is highlighted in bold; $*$ denotes a significant difference to the best performing result according to a paired t-test at $p < 0.01$. † indicates a previously reported result; we are unable to test these for statistical significance.

| Models | Encoder | MultiWOZ2.1 | MultiWOZ2.2 |
|---|---|---|---|
| TRADE† | GRU | 45.60 | 45.40 |
| DS-DST† | BERT | 51.21 | 51.70 |
| SOM-DST | BERT | 52.37* | - |
| DST-picklist† | BERT | 53.30 | - |
| TripPy | BERT | 55.52* | 50.71* |
| DialoGLUE† | ConvBERT | 58.70 | - |
| TripPy-TS | BERT | **59.50** | **54.90** |
| TripPy-T | BERT | 56.44* | 52.85* |
| TripPy-S | BERT | 58.03* | 53.89* |
| TripPy-V | BERT | 56.14* | 52.02* |
| TripPy-CoCo (1x)† | BERT | 56.00 | - |
| TripPy-CoCo (2x)† | BERT | 56.94 | - |
| TripPy-CoCo (4x)† | BERT | 59.73 | - |
| TripPy-CoCo (8x)† | BERT | **60.53** | - |

## 6 Experimental Results and Discussion

Tables 1 reports the effectiveness of DST models in terms of joint goal accuracy on the two datasets. The table contains two groups of rows: The first group reports the effectiveness of the TripPy model that uses our proposed Turn-based Loss Function (TLF) and Sequential Data Augmentation (SDA) approaches compared to the baselines. The second group reports the effectiveness of TLF, SDA and the existing data augmentation algorithms (i.e. VDA and CoCo). The encoder column indicates the pre-trained language model used by the baselines as the dialogue encoder, described in Section 3.2. Due to their recency or a lack of details, we were not able to re-implement all baselines. For those baselines, we include the as-reported results and are unable to test for statistical significance.

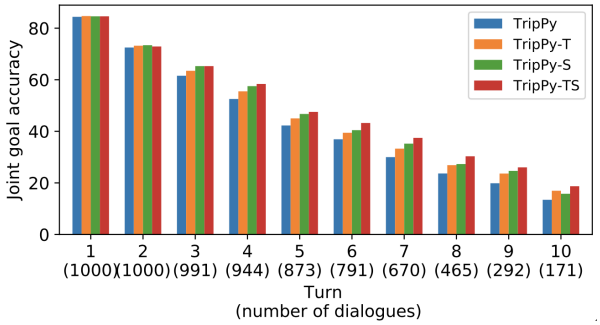
We first reproduce results with TripPy and SOM-DST models in Table 1. We find that the relative dialogue state tracking effectiveness of these two models on MultiWOZ2.1 is consistent with the results reported in the original papers (Heck et al., 2020; Kim et al., 2020). For instance, SOM-DST outperforms both TRADE and DS-DST and is as effective as the state-of-the-art ontology-based DST model (DST-picklist). Similarly, we observe that TripPy outperforms all the ontology-based and span-based baselines on the MultiWOZ2.1 and MultiWOZ2.2 datasets. Note that MultiWOZ2.2 is the most recent DST dataset and has not widely used in the previous literature, hence some baselines results are not available yet on MultiWOZ2.2[5]. The results of TRADE and DS-DST on MultiWOZ2.2 are those reported in (Zang et al., 2020).

Comparing the baseline model that uses our TLF and SDA approaches (TripPy-TS) with baselines across the two datasets in Table 1, we observe that TripPy-TS consistently and significantly outperforms all the ontology and span-based DST baselines in terms of JGA across all datasets. TripPy-TS improves joint goal accuracy by 7.17%, 11.63% and 14.12% relative reduction in error over the base TripPy, DST-picklist and SOM-DST models that use BERT as the dialogue encoder on MultiWOZ2.1. Comparing TripPy-TS with DialoGLUE, the TripPy model that uses the fine-tuned BERT on 700 million open-domain dialogues, we observe that TripPy-TS still outperforms DialoGLUE by 1.36% relative reduction in error on MultiWOZ2.1, although TripPy-TS only uses the pre-trained BERT-base-uncased model as the dialogue encoder. Similar results are observed on MultiWOZ2.2 where TripPy-TS outperforms TripPy, DS-DST and TRADE by 8.26%, 6.19% and 20.93% relative reduction in error, respectively. These results imply that our TLF and SDA approaches significantly and consistently improve the effectiveness of the state-of-the-art DST model, TripPy.

Next, we further analyse the effectiveness of TLF and SDA using an ablation study. We note that TripPy-T and TripPy-S are the baseline models using TLF and SDA. Comparing TripPy-TS to TripPy-T, we observe a significant decrease of effectiveness in terms of JGA across both datasets. The relative reduction in error decreases around 1.55-5% in TripPy-TS's performance compared to TripPy-T. These results indicate the importance of SDA in enhancing the effectiveness. In addition, comparing TripPy-S and TripPy-V that uses the existing Value-based Data Augmentation, we find that SDA is more effective than the VDA in improving the effectiveness of TripPy. Comparing TripPy-TS and TripPy-S, we observe approximately 2.53% and 1.87% relative reduction in error decreases of the TripPy-TS's performance on MultiWOZ2.1 and MultiWOZ2.2 datasets, respectively. These results are intuitive because TLF improves the effectiveness of DST models by penalising the DST models heavily if they fail to accurately predict the dialogue state at the early to mid turn depths.

[5]The SOM-DST implementation does not currently support MultiWOZ2.2

Figure 3: The performance of TripPy with/without TLF and SDA across different turns on MultiWOZ2.2.




Overall, we find that our proposed TLF and SDA approaches together consistently and significantly improve the effectiveness of state-of-the-art DST model (TripPy) across the two used datasets.

We further compare the performance of SDA (TripPy-S) and the state-of-the-art CoCo data augmentation algorithm (TripPy-CoCo). Note that TripPy-CoCo (2x) denotes the TripPy model that is trained on the augmented data two times larger than the original training data. First, we observe that TripPy-S outperforms TripPy-CoCo (1x) and (2x) by 3.63% and 1.91% relative reduction in error on MultiWOZ2.1. Although TripPy-CoCo (4x) and (8x) are more effective than TripPy-S, such comparison is not fair due to several reasons. First, the value substitution component of CoCo relies on a pre-defined value set for each domain-slot which is manually created. Second, CoCo initialises the parameters of the controllable counterfactual generation model and the classifier filter using the pre-trained T5 (Raffel et al., 2020) and BERT models. Third, CoCo uses MultiWOZ2.2 to train the controllable counterfactual generation model and the classifier filter, yet evaluate the performance of TripPy-CoCo on MultiWOZ2.1. In contrast, our SDA approach does not use any pre-defined value set for each domain-slot, the advanced pre-trained models (i.e. T5) as well as the MultiWOZ2.2 dataset during the training process.

### 6.1 Turn Depth and Domain-specific JGA

Dialogues vary in length and longer dialogues are likely to be more challenging. In this section, we study the relationship between the depth of dialogue and the effectiveness of different models[6]. The trend in Figure 3 clearly shows the effectiveness of all models decreases steadily and dramatically as the turn depth increases. Comparing the effectiveness of TripPy that uses the traditional cross-entropy loss function and TripPy-T that uses our proposed Turn-based Loss Function, we observe that TripPy-T outperforms the baseline starting from the third turn consistently through to turn ten. The improvement of TripPy-T compared to TripPy from the early to mid turn depths has a large impact in the later turns. These results imply that we should penalise the model more heavily if it fails to predict the slot value early-ish in the conversation as the error from the first turn cascades in later turns, degrading JGA.

Next, we investigate the utility of our proposed SDA algorithm to improve the quality of DST. We see that the performance of both TripPy and TripPy-S are similar on the first and second turns. Then, from the third to the tenth turn, TripPy-S consistently outperforms TripPy on MultiWOZ2.2. When we compare TripPy with TripPy-T, SDA does not improve the performance of TripPy at the early turns but increases the effectiveness of the model in the later turns. Finally, the performance of TripPy-TS with the base model is comparable at the first and second turn. Interestingly, we observe that TripPy-TS consistently outperforms TripPy from the third turn to the tenth turn. This demonstrates that TLF and SDA are complementary and together play an important role in improving the quality of state tracking across increasing turn depths.

We also analyse the effectiveness of the algorithms examining joint goal accuracy for each domain over turn depths. First, in Figure 4, the results show that TripPy-TS consistently outperforms TripPy across the first five turns on the restaurant, hotel, attraction and train domains on MultiWOZ2.2[7]. In particular, TripPy-TS outperforms TripPy by approximately 15%, 7%, 14% and 20% relative reduction in error for the restaurant, hotel, attraction and train domains. We also observe that TripPy-TS outperforms TripPy from the third turn on the taxi domain. On the fourth turn TripPy completely fails to track the state from fourteen dialogues while TripPy-TS accurately predicts three. The performance of TripPy-TS averaged across all turns on the taxi domain is better than the performance of TripPy by approximate 22% relative reduction in error (i.e. 31.70 JGA compared to 25.79 JGA).

[6]We conduct this experiment on both MultiWOZ2.1 and MultiWOZ2.2 datasets and we obtain similar results. We omit the results on MultiWOZ.21 due to space limitations.

[7]Note that the results after the fifth turn are relatively similar to the first five turns' results across the two used datasets and we omit them due to the space limitation.

Figure 4: The effectiveness of TripPy-TS on different turn depths across domains on MultiWOZ2.2. The results shown on top of each figure report the average joint goal accuracy of each model across all turns.

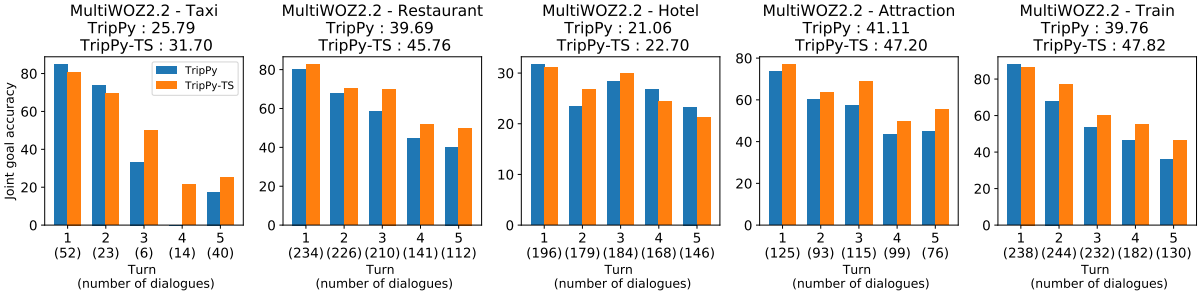



Figure 5: The performance of TripPy with/without TLF and SDA on different turns with a particular number of active slots. There are 2,414 (56%), 1,525 (35%), 314 (7%) and 75 (2%) turns with 1, 2 ,3 and 4 active slots.

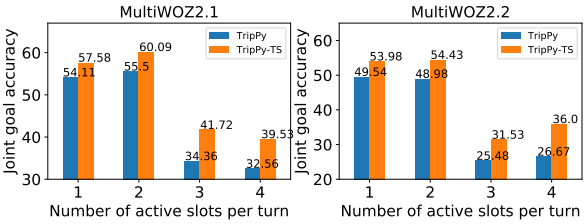
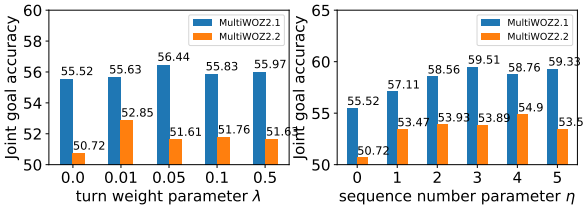



Figure 6: The sensitivity of hyperparameters for turn weight $\lambda$ and sequence number $\eta$.




## 6.2 Complex Turn-specific JGA

We compare TripPy and TripPy-TS on different turns with a particular number of active slots, i.e. a simple turn contains either one or two active slots. We define a complex turn to be one containing either three or four active slots. In Figure 5, we observe that TripPy-TS consistently outperforms TripPy on the simple turns by approximately 6-10% relative reduction in error in terms of JGA across MultiWOZ2.1 and MultiWOZ2.2. It is clear on both datasets that TripPy-TS is more effective in predicting state for the complex turns. TripPy-TS outperforms TripPy by 17-21% and 23-34% relative reduction in error in terms of JGA over the complex turns on MultiWOZ2.1 and MultiWOZ2.2, respectively. These results imply that both TLF and SDA consistently improves the effectiveness of TripPy in accurately predicting the state over the turns with more active slots. As illustrated in Section 4.2, the augmented dialogues generated by our algorithm are contain more active slots than the original dialogues in the training set, that usually have a small number of active slots per turn. These augmented dialogues help the model to learn to effectively extract the slot values from complex turns.

## 6.3 Impact of hyperparameters

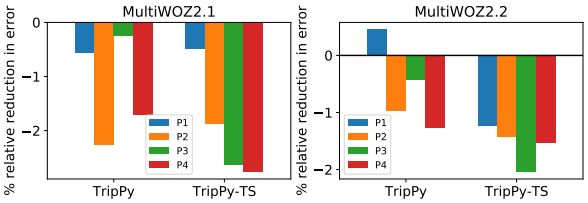
We evaluate the sensitivity of the hyperparameters for our proposed methods. First, we study the effect of the turn weight parameter $\lambda$ of TLF in Equation 5 by varying its value in the range of $\{0, 0.01, 0.05, 0.1, 0.5\}$[8]. Note that $\lambda = 0$ corresponds to the baseline model without TLF. From the left figure in Figure 6, we observe that setting $\lambda = 0.01, 0.05, 0.1$ or $0.5$ is more effective than the baseline with $\lambda = 0$ across both datasets. Next, we study the impact of the sequence number parameter $\eta$ of the Sequential Data Augmentation algorithm. From the right figure in Figure 6, we observe that $\eta = 1, 2, 3, 4$ or $5$ is more effective than $\eta = 0$. The most effective is obtained when $\eta = 3$ or $\eta = 4$ respectively.

## 6.4 Impact of perturbed dialogue history

Recent work by Sankar et al. (2019) shows existing task-based models seldom understand or use the dialogue history effectively. We first study the behavior of the state-of-the-art TripPy model to use it (i.e. $H_t$ in Section 3.2). Then, we compare behavior of the proposed TLF and SDA methods. We apply four types of perturbation operations to the dialogue history in the test set. P1 and P2 are utterance-level perturbation operations that shuffle and reverse the sequence of utterances in the dia-

[8]Other values up to 1 were tried in preliminary experiments and were not as effective as the intervals we report.

Figure 7: The effect of varying dialogue history perturbations on TripPy and TripPy-TS.




logue history. P3 and P4 are word-level perturbation operations that randomly shuffle words within an utterance and reverse the ordering of words.

In Figure 7, we observe that the effectiveness of TripPy and TripPy-TS decreases by 0.5-2.5% relative reduction in error across different perturbation operations on MultiWOZ2.1. These results are intuitive because the DST models are less likely to capture intent from the perturbed history, hence being less effective. For the P3 operation that randomly shuffles the words within the utterance, the results show that this operation negatively impacts the performance of TripPy-TS, while it only slightly affects TripPy. Similar to the results observed on MultiWOZ2.1, on MultiWOZ2.2 we also observe that both TripPy and TripPy-TS suffer from the four perturbation operations, except on P1 with TripPy. As desired, it shows that TripPy-TS is more sensitive than TripPy to the four perturbation operations. This implies that our proposed TLF and SDA approach rely more heavily on turn and word order. This is behavior that we expect and desire in a tracking model. We hypothesize that the augmented dialogues (which are complex and long) generated by SDA force the model to incorporate the dialogue order during the training process.

## 7 Conclusion

We propose two novel algorithms, TLF (Turn-based Loss Function) and SDA (Sequential Data Augmentation), that improve the effectiveness of state-of-the-art dialogue state tracking models. TLF penalizes such models more heavily if they fail to predict slot values in the middle of the conversation. On the other hand, SDA generates dialogues used to train existing DST models to generalize more effectively. Our comprehensive experiments on multiple benchmark datasets demonstrate the combined utility of both TLF and SDA to improve the effectiveness of the leading model in the literature. Indeed, TLF and SDA significantly improve the effectiveness of TripPy by approximately 8.26% relative reduction in error on MultiWOZ2.2, which constitutes the new state-of-the-art result on this most recent benchmark dataset. For future work, we plan to extend both TLF and SDA to incorporate additional information such as dialogue length and the number of active slots during the training process to be even more effective for long and complex dialogues. We also plan to investigate the effectiveness of TLF and SDA on other existing DST models.

## Acknowledgements

This project was funded by the EPSRC Fellowship titled "Task Based Information Retrieval", grant reference number EP/P024289/1, and supported by the Engineering and Physical Sciences Research Council grant EP/V025708/1 as well as the Bloomberg Data Science Research Grant.

## References

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *Proc. of AAAI*.

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

M. Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, A. Sethi, Anuj Kumar Goyal, Peter Ku, Sanchit Agarwal, Shuyang Gao, and Dilek Z. Hakkani-Tür. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *ArXiv*, abs/1907.01669.

Campagna Giovanni, Foryciarz Agata, Moradshahi Mehrad, and Lam Monica S. 2020. Zero-shot transfer learning with synthesized data for multi-domain dialogue state tracking. In *Proc. of ACL*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gašić. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Proc. of SIGdial*.

Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014. The second dialog state tracking challenge. In *Proc. of SIGDIAL*, pages 263–272.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-sequence data augmentation for dialogue language understanding. In *Proc. of ACL*.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proc. of ACL*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2021. Coco: Controllable counterfactuals for evaluating dialogue state trackers. In *Proc. of ICLR*.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *arXiv preprint arXiv:2009.13570*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *arXiv preprint arXiv:1909.05855*.

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

Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018. Building a conversational agent overnight with dialogue self-play. *arXiv preprint arXiv:1801.04871*.

Xiaohui Song, Liangjun Zang, Yipeng Su, Xing Wu, Jizhong Han, and Songlin Hu. 2020. Data augmentation for copy-mechanism in dialogue state tracking. *arXiv preprint arXiv:2002.09634*.

Adam Summerville, Jordan Hashemi, James Ryan, et al. 2020. How to tame your data: Data augmentation for dialog state tracking. In *Proc. of ConvAI*, pages 32–37.

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 *Proc. of EACL*.

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 *Proc. of ACL*.

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 *Proc. of the 2nd Workshop on Natural Language Processing for Conversational AI, ACL*, pages 109–117.

Jian-Guo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wan, Philip S Yu, Richard Socher, and Caiming Xiong. 2019. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *arXiv preprint arXiv:1910.03544*.