How to Stop an Avalanche? JoDeM: Joint Decision Making through Compare and Contrast for Dialog State Tracking

Haoming Wang **Xin Wang** **Yuantao Gu** [*]
Department of Electronic Engineering, Tsinghua University, Beijing, China
{wanghm20, wangxin20}@mails.tsinghua.edu.cn, gyt@tsinghua.edu.cn

Abstract

Dialog state tracking (DST) is a core component in task-oriented dialog systems. Existing state-of-the-art DST model incorporates insight and intuition from the human experience into design of supplementary labels, which greatly assisted the training process of turn-by-turn DST model. Though the turn-by-turn scheme and supplementary labels enabled satisfactory performance on the task, most of the DST models of this fashion label or process the raw dialogue data on the premise that the last turn dialogue state is always correct, which is usually not the case. In this paper, we address the negative impact resulted from the premise above as the avalanche phenomenon. After that, we propose JoDeM, a state-of-the-art DST model which can tackle the Avalanche phenomenon with two mechanisms. First mechanism is a jointly decision making method to extract key information from the dialogue. Second mechanism is a compare and contrast dialogue update technique to prevent error accumulation. Example study and graph analysis are presented to support our claim about the harmfulness of avalanche phenomenon. We also conduct quantitative and qualitative experiments on the high quality MultiWOZ2.3 corpus dataset to demonstrate that the proposed model not only outperforms the existing state-of-the-art methods, but also proves the validity of solving avalanche degradation problem.

1 Introduction

Goal-oriented dialog (GOD) systems, or Task-oriented dialogue (TOD) systems have recently attracted growing attention and significant progress has been made (Zhang et al., 2020; Neelakantan et al., 2019; Peng et al., 2020). Well-known commercial dialogue systems include the Apple Siri, Amazon Alexa, or Microsoft Cortana. In a complete GOD system, Dialog State Tracking (DST) serves as a cognitive and comprehending component, where it understands and extracts the user's goal in a well-constructed manner. The user's goal is then provided to downstream for recommendation, booking, or other subsequent dialogue policy components to determine the system action and response. Hence, as the backbone of a dialogue system, it is crucial to have a DST module with exceptional performance to guarantee the base for the performance of subsequent components (Takanobu et al., 2020).

Since the blossom of the application of pre-trained language model, the accuracy of DST models has increased tremendously. Especially, turn-by-turn schematic DST models (Liao et al., 2021) with insightful design of auxiliary labels and data structure have dominated the field, where most of the among-the-best works are of this genre (Heck et al., 2020; Liao et al., 2020). However, this type of models all suffer from a major flaw, the avalanche phenomenon. The avalanche phenomenon is the result of wrong premise during the labeling process which will only occur in the DST models with turn-by-turn scheme.

As oppose to the trending turn-by-turn scheme, early multi-domain DST methods follow a dialog history scheme. Model of this scheme takes the whole or window-sized dialogue history as input. It predicts slot value without explicitly discriminating over turns of utterances. Despite the benefits of making prediction based on a more comprehensive and complete data at once, dialog history scheme has several drawbacks. The length of dialogues is often too long for pre-trained language model to process. More essentially, processing an entire dialogue at once violates the instant update nature of DST. Aligning with the need of instant update, turn-by-turn scheme was proposed (Kim et al., 2019). Models of this scheme input the dialogue state generated from the previous turn and the most recent turn utterance and output the up-

[*] Corresponding author.

dated dialogue state. The advantages of turn-by-turn scheme resulted in great performance boost, in which most of the among-the-best works are of this scheme. On top of the choice of better scheme, to achieve a superior performance, these state-of-the-art DST models made the best use of auxiliary labels.

Basic input of turn-by-turn schematic DST models are the current turn utterance and last turn dialogue state, where the basic output is the updated current turn dialogue state. Using only basic output as golden training label inevitably leads to a sub-optimal result due to complexity of DST task. Mainstream DST systems typically incorporate supplemental labels to guide the model towards better performance. For example, Zhang et al. 2019; Heck et al. 2020 obtain key information from the dialogue span directly labels the starting and ending index of the key phrase for DST models to learn span detection.

However, the high utilization of supplementary labels in turn-by-turn schematic models have induced a new obstacle in developing a more robust and high quality DST system. In practice, a training instance, which is a turn of dialog in an entire dialogue for turn-by-turn systems, are randomly shuffled along with instance from other dialogues. For convenience and effective training, supplementary labels are made under the assumption that the input previous turn dialogue state is correct. While in a considerable amount of cases, models have to make prediction under incorrect last turn dialog state. In those cases, the supplementary labels will also be incorrect themselves because they are also made under the false assumption. These facts add up to a poor robustness against noisy input, making the final performance way lower than expectation. To reflect this kind of characteristic where errors induce more errors, we name this phenomenon the **avalanche phenomenon**. Although solutions and strategies on similar error accumulation phenomenon are widely explored in auto-regressive characteristic tasks Ranzato et al. 2015; Bengio et al. 2015 , there are significant differences, resulting in different approaches to tackle the issue. Detailed comparison and analysis can be found in the *discussion* part of the appendix.

In this paper, we propose **JoDeM**: **J**oint **De**cision **M**aking DST system with a compare and contrast mechanism. As mentioned, there are two major issues that directly contribute to the existence of the avalanche phenomenon, incorrect last turn dialogue state and inflexible training labels. To address the former issue where DST models often perform worse when the input last turn dialogue state is incorrect, we simply exempt dialogue state from the data flow of DST model, and strictly update it in a compare and contrast fashion. In other words, the extraction of key information is accomplished by a series of fluent back propagatable operations while the update process is not. To tackle the later issue, JoDeM deploys a joint decision making structure to successfully update dialogue state in a more robust and flexible manner despite the fact that training labels are fixed.

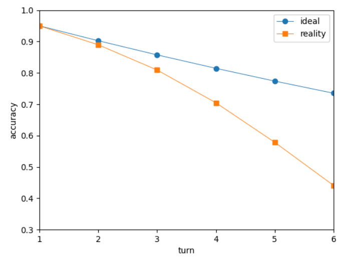


Figure 1: An example of the avalanche phenomenon creating a deficit between the reality and the ideal on the joint accuracy of a DST systems.

The JoDeM model contains eight modules that divides the whole DST process into three stages. The first stage contains a utterance encoder. The second stage contains four parallel modules, namely, a domain update, a slot gate, a slot type, and a span detection module. The third stage contains a dialogue state update module. As shown in the figure, first, we use BERT as the pre-trained language module to embed turn utterance. Then, a parallel decision making procedure is adopted by the four modules to extract key information from the embedded utterance. At last, the dialogue state update module designed to address the avalanche phenomenon is applied to output the updated dialogue state.

After introducing related work and the details of JoDeM, we conduct multiple standard and customized evaluation and analysis in this paper to show that not only JoDeM achieved a state-of-the-art performance, but also the reason why it achieved such robustness against the avalanche phenomenon. In short, our contribution is twofold:

1. We bring up the attention to the avalanche phenomenon, a previous uncharted territory

in dialogue state tracking task, and present quantitative evidence to show its existence and severity to the performance of DST systems.

2. We proposed a DST model to verify the feasibility of a solution to address the avalanche phenomenon, targeting straight to the roots of the phenomenon. After that, we performed quantitative and qualitative experiments to show the validity of our work and that our model has achieved a state-of-the-art performance on the qualified MultiWOZ2.3 dataset.

2 Related Work

Depending on the inputs, existing DST models are categorized to history-based and turn-by-turn based (Liao et al., 2021). The former scheme takes the whole or window-sized dialogue history as input to recurrent neural networks or networks (Goel et al., 2019; Gao et al., 2019). For example, HJST considers the full dialogue history using a hierarchical RNN (Gao et al., 2019; Serban et al., 2015). Works such as Wu et al. 2019 treats the entire dialogue as a concatenated sequence while using BiLSTM or RNN as an encoder. There are also works inputting the whole history or window-sized dialogue history into BERT such as Lee et al. 2019a. In order to overcome the limitations of history-based scheme mentioned in the introduction, turn-by-turn DST systems was developed. Typically, model of this scheme takes the previous turn dialogue state and the current turn utterance as input to generate new dialogue state (Chao and Lane, 2019; Ren et al., 2019; Heck et al., 2020).

Basic label of the DST task is the correct dialogue state at each turn, which is often insufficient for the model to learn from effectively. The most common example of supplementary label is the starting index and ending index of the value phrase utilized in the span-based models (Zhang et al., 2019; Heck et al., 2020; Chen et al., 2020b). Kim et al. 2019, a turn-by-turn model designed a set of operation-based labels to guide the updating process of dialogue state. Heck et al. 2020 defined three copy strategy and labeled the original dialogue state tracking process with more refined information. These attempts have made significant result on the performance by incorporating human knowledge to the training process by applying supplementary labels. However, these labels are created under the assumption that the last dialogue state at every turn is flawless, while in reality it is usually not the case. The gap between ideal and reality creates a major drawback on the performance and robustness. In our JoDeM model, we not only design our supplementary label base on fine intuition, but also address the drawback resulted from the avalanche phenomenon.

3 JoDeM: Joint Decision Making through Compare and Contrast

The proposed JoDeM model in Figure 2 consists of eight components that are located in three different stages of the DST process. The first stage is the *Utterance Encoder* that encodes the basic inputs, i.e., system and user utterance into vector embedding. After that, the utterance embedding is sent to the second stage, which is the *Joint Decision Making* stage. In this stage, key information is extracted from the utterance embedding by the following component, *Domain Update*, *Slot Gate*, *Type Prediction*, *Span Detection* and *Co-ref Classification*. At the last stage, compare and contrast mechanism is applied by the *Dialogue State Update* component to update the dialogue state according to the key information from the second stage and the previous turn dialog state.

Before formally getting into the detail of the JoDeM model, we first layout the necessary mathematical notations and proper definition for the DST problem. We define a complete dialogue as $X = \{(S_1, U_1), ..., (S_T, U_T)\}$, which has T sets, or turns of system and user utterance that are in a sequential order. The dialogue states of an entire dialogue which is a set of dialogue state from all T turns is defined as $\mathrm{DS} = \{\mathrm{DS}_1, ..., \mathrm{DS}_T\}$, where DS_i is the dialogue state of the ith turn. Each turn's dialogue state is a set which takes multiple triplets of format $(domain, slot, value)$ as its elements. To complete a DST task is equivalent to the following statement: for any turn t, given the turn utterance (S_t, U_t) and the last turn dialogue state DS_{t-1} as input, we should output DS_t, which contains the correct set of triplets $(domain, slot, value)$.

3.1 Utterance Encoder

Utterance encoder is the cornerstone of all NLP task including the DST task. At each turn t, we use the pre-trained BERT (Devlin et al., 2018) as the front-end encoder to encode the dialog utterance (S_t, U_t) as

$$\mathbf{R}_t = \mathrm{BERT}([\mathrm{CLS}] \oplus S_{\mathrm{t}} \oplus [\mathrm{SEP}] \oplus U_t), \quad (1)$$

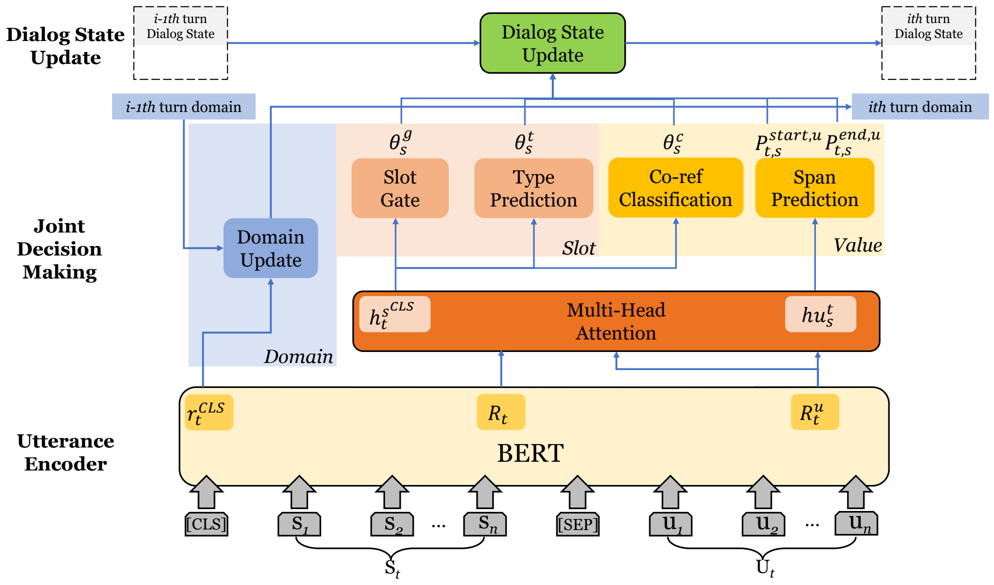


Figure 2: The architecture of the proposed JoDeM model comprised of three stages of eight components.

where $\mathbf{R}_t$ is the embedding of utterance from turn t. $\oplus$ is the concatenation operator. Special token CLS is the starting token for BERT and SEP is the separation token separating system utterance S_t and user utterance U_t. The embedding of utterance can also be denoted as $\mathbf{R}_t = [\mathbf{r}_t^{CLS}, \mathbf{r}_t^{S_1}, \mathbf{r}_t^{S_2}, ..., \mathbf{r}_t^{SEP}, \mathbf{r}_t^{U_1}, \mathbf{r}_t^{U_2}, ...]$, where $\mathbf{r}_t^{CLS}$ is the vector representation of the entire turn dialogue. The vector $\mathbf{r}_t^i$ is the contextual representations for the ith token in the utterance. The dimension of the embedding is h, which is a hyper-parameter of BERT. Above sentence embedding is then utilized for joint decision making.

3.2 Joint Decision Making

The intuition behind the *Joint Decision Making* stage is to break down and imitate the human reasoning process. Human beings complete the DST task by solving the triplets of $(domain, slot, value)$ in a joint fashion, rather than solving the elements in a triplet in an order or individually. For example, one would not first determine the state of a $(domain, slot)$ pair, then search for its value. Instead, the context regarding different $(domain, slot)$ pairs and their possible values within the utterance are considered jointly so that comprehensive judgement on the state of different $(domain, slot, value)$ triplets can be made. Bearing this intuition in mind, we propose the *Joint Decision Making* stage consisting of five parallel components that jointly solve all the $(domain, slot, value)$ triplets in a dialogue state, covering every possible scenario.

3.2.1 Domain Update

We obtain the domain of turn t by updating it from the last turn $t-1$ domain. As shown in the dialogue example in Figure 3, the domain element of the dialog state is highly correlated to its last turn domain. Generally, if the turn utterance doesn't contain any trace of or sufficient domain information, the domain from the last turn will still be in use by the continuity of the context. Therefore, we design the *Domain Update* component to obtain the turn domain by taking the utterance representation $\mathbf{r}_t^{CLS}$ as an input to detect new domain and the last turn domain as a bias. The probability distribution of the turn domain $\mathbf{D}_t$ over all possible domains $\mathrm{D} = \{train, taxi, restaurant, hotel, attraction\}$ is obtained by

$$\mathbf{D}_t = \mathrm{softmax}(\boldsymbol{\gamma} \cdot (\mathbf{W}^{DU} \cdot \mathbf{r}_t^{CLS} + \mathbf{b}^{DU})) \in \mathbb{R}^5, \tag{2}$$

where $\mathbf{W}^{DU}$ and $\mathbf{b}^{DU}$ are the trainable parameters of a standard linear transformation, respectively. Diagonal coefficient matrix $\boldsymbol{\gamma} = (\mathrm{diag}(\mathbf{d}_{t-1}) + \mathbf{E})$ where $\mathbf{E}$ is the identity matrix , $\mathbf{d}_{t-1}$ is the normalized resulted from the last turn domain $\mathbf{D}_{t-1}$, and $\mathrm{diag}(\cdot)$ transforms vectors into diagonal matrices. Due to the uniqueness of domain in each turn, the class with the highest probability from $\mathbf{D}_t$ is the turn domain. The design of the impact of last turn domain is oriented to the following purpose: we require the impact from the last turn play a dominate role when there's no new domain predicted. At the same time, if there is new domain involved, the influence of the last turn should be ignored. If

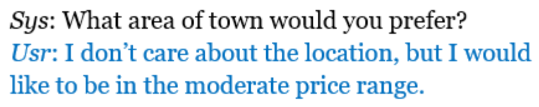

Sys: What area of town would you prefer?
Usr: I don't care about the location, but I would like to be in the moderate price range.

Figure 3: Example for the case when the turn domain is entirely dependent on the last turn context

$\gamma = \mathrm{diag}(\mathbf{d}_{t-1})$, any newly discovered domain would be covered up by the scaling effect from the last turn domain. Also, in order to diminish the impact from the last turn when new domain is predicted, we make the bias itself relevant to the outcome of the linear transformation. Only when there is no domain discovered, i.e., the outcome of the linear part is equally distributed , will the bias of $\mathrm{diag}(\mathbf{d}_{t-1})$ dominate the result.

3.2.2 Slot Gate & Type Prediction

Our model is equipped with a *Slot Gate* and a *Type Prediction* components for each slots. The *Slot Gate* aims to determine whether a slot should be updated, i.e., the output of a slot gate $\mathbf{G}_s$ is a binary probability distribution. Inspired by Heck et al. 2020 and Kim et al. 2019, we summarize the possible updates into the following four types $\{U, S, C, N\}$. U and S indicates that the value of the slot should be found in the span of user utterance $\mathbf{U}_t$ and system utterance $\mathbf{S}_t$ respectively. C indicates that the value of the slot has a co-reference relationship with a certain $(domain, slot)$ pair in the last turn dialogue state. N means that the user intend to delete the existing value of the corresponding slot in the dialogue state without providing any alternative value.

To make the above prediction for each slot, we first employ the multi-head attention mechanism (Vaswani et al., 2017) to calculate the attended context vector $\mathbf{h}_t^s$ between $\mathbf{R}_t$ and the user utterance embedding $\mathbf{R}_t^u$ at t as

$$\mathbf{h}_t^s = \mathrm{MultiHeadAtte}(\mathbf{Q}, \mathbf{K}, \mathbf{V}), \qquad (3)$$

where $\mathbf{Q}$ is the embedding of the entire utterance embedding, $\mathbf{R}_t$. $\mathbf{K}$ and $\mathbf{V}$ are the embedding of the user utterance embedding $\mathbf{R}_t^u = [\mathbf{r}_t^{U_1}, \mathbf{r}_t^{U_2}, ...]$. The reason to apply the multi-head attention mechanism is that the confirmation from a user is the essence of dialog state update, no matter the type of update. Therefore, the relationship between the entire utterance and the user utterance is needed.

After obtaining the attended embedding of the entire utterance, for each slot s, slot gates and type predictions are made by two parallel trainable linear layer classification,

$$\boldsymbol{\theta}_s^g = \mathrm{softmax}((\mathbf{W}_s^g \cdot \mathbf{h}_t^{s^{CLS}} + \mathbf{b}_s^g)) \in \mathbb{R}^2, \quad (4)$$

$$\boldsymbol{\theta}_s^v = \mathrm{softmax}((\mathbf{W}_s^v \cdot \mathbf{h}_t^{s^{CLS}} + \mathbf{b}_s^v)) \in \mathbb{R}^4. \quad (5)$$

3.2.3 Span Detection & Co-Ref Classification

Span detection and *co-ref classification* are equipped to solve the possible value for each slots.

Span detection is utilized for the slots whose values are found in the utterance. The attended utterance embedding is separated into two parts, the attended vector for user $\mathbf{hu}_t^s$ and the attended vector for system $\mathbf{hs}_t^s$. A slot specific span detection layer performs a user/system specific span detection on the attended context vector $\mathbf{hu}_t^s$ and system context vector $\mathbf{hs}_t^s$ separately to obtain the span of potential values in the utterance to update. The expression of the process, using *span detection* on the user utterance as an example, is

$$[\boldsymbol{\alpha}_{t,i}^{s,u}, \boldsymbol{\beta}_{t,i}^{s,u}] = (\mathbf{W}_s^{span} \cdot \mathbf{hu}_{t,i}^s + \mathbf{b}_s^{span}) \in \mathbb{R}^2$$

$$P_{t,s}^{start,u} = \mathrm{argmax}(\boldsymbol{\alpha}_t^s)$$

$$P_{t,s}^{end,u} = \mathrm{argmax}(\boldsymbol{\beta}_t^s)$$

i is the index of a token in the attended context of user utterance, $P_{t,s}^{start,u}$ is the starting position of span in the user utterance $\mathbf{U}_t$ for slot s in turn t and $P_{t,s}^{end,u}$ is the corresponding ending position.

Co-ref classification is utilized for the slots whose value should be filled via co-referencing with a known value in the last turn dialog state. We simply take $\mathbf{h}_t^{s^{CLS}}$ which is the attended context embedding of the representation token for the entire utterance and perform a linear layer classification,

$$\boldsymbol{\theta}_s^c = (\mathbf{W}_s^c \cdot \mathbf{h}_t^{s^{CLS}} + \mathbf{b}_s^c) \in \mathbb{R}^{31}, \qquad (6)$$

where the output $\boldsymbol{\theta}_s^c$ is a probability distribution on all possible thirty $(domain, slot)$ pairs and one none class.

3.3 Dialogue State Update

Dialogue State Update is the key part of any turn-by-turn schematic DST systems, which is the procedure where the avalanche phenomenon originated from. We mentioned that the conflict between incorrect last turn dialogue state and the supplementary labels which are based on the correct last dialog state is the main contributor to the avalanche phenomenon. Therefore, we exclude the dialogue state updating process from the forward and backward propagation of data processing flow, by updating the dialogue state by carefully comparing and contrasting through the information that we obtained from the previous *Joint Decision Making* stage. At last, to achieve better robustness of the model, we apply a trick in the training process. The overall dialogue state update procedure is shown in Algorithm 1.

Algorithm 1: DS Update

```
Input: $\boldsymbol{\theta}_s^g$,$\boldsymbol{\theta}_s^v$,$\boldsymbol{\theta}_s^c$,
       $P_s^{start,u}$,$P_s^{end,u}$,$P_s^{start,s}$,$P_s^{end,s}$,
       $\mathbf{D}_t$,DS$_{t-1}$
Output: DS$_t$
Specify the turn Domain via $(\mathbf{D}_t)$
for each slots $s$ in the turn Domain do
    if $\boldsymbol{\theta}_s^g$ then
        if $\boldsymbol{\theta}_s^v = U$ then
            $v \leftarrow$ U$_t[P_s^{start,u}$:$P_s^{end,u}]$
        else if $\boldsymbol{\theta}_s^v = S$ then
            $v \leftarrow$ S$_t[P_s^{start,s}$:$P_s^{end,s}]$
        end
        else if $\boldsymbol{\theta}_s^v = C$ then
            $v \leftarrow$ DS$_{t-1}[\boldsymbol{\theta}_s^c]$
        end
        else if $\boldsymbol{\theta}_s^v = N$ then
            $v \leftarrow$ none
        end
    end
    if DS$_t\{\mathbf{D}_t, s\} \neq v$ then
        DS$_t\{\mathbf{D}_t, s\} \leftarrow v$
    else
        if Training then
            $\boldsymbol{\theta}_s^g$,$\boldsymbol{\theta}_s^v$,$\boldsymbol{\theta}_s^c$,
             $P_s^{start,u}$,$P_s^{end,u}$,$P_s^{start,s}$,$P_s^{end,s}$
             $\leftarrow$ GoldenLabel
        end
    end
end
```

First, we specify the domain by the result of the *Domain Update* component $\mathbf{D}_t$. Second, we determine whether to update a slot within the domain through the *Slot Gate* result $\boldsymbol{\theta}_s^g$. If it equals to 1, that is, $\boldsymbol{\theta}_s^g = 1$, we move on to the next step. In the third step, we go through the slots with $\boldsymbol{\theta}_s^g = 1$ and determine their corresponding values according to their *Type Prediction* $\boldsymbol{\theta}_s^v$. For the slots whose $\boldsymbol{\theta}_s^v = U$ or $\boldsymbol{\theta}_s^v = S$, we obtain their values by getting the corresponding span from the user or system utterance. The span is determined by the corresponding starting and ending index $P_s^{start,u}$,$P_s^{end,u}$,$P_s^{start,s}$,$P_s^{end,s}$. If $\boldsymbol{\theta}_s^v = C$, the values of the slots will be determined by the co-refered $(domain, slot)$ pairs $\boldsymbol{\theta}_s^c$ from the last turn dialogue state. At last, for slots with $\boldsymbol{\theta}_s^v = N$, we simply delete the values that were stored previously. Finally, in the last step, we perform the update by comparing and contrasting new triplets $(domain, slot, value)$ and the ones in the last dialogue state.

As mentioned above, we perform a special operation at this stage during the training process. During training, if the potential value is equal to the last turn dialogue state, we set all the output from the forward propagation to the golden label. Thus, preventing the back propagation process to alter the trainable parameters in the model. This operation can enable the model to develop the ability to self-correct, resulting in a better performance. More details can be found in the *example study* in the appendix.

4 Experiment

4.1 Dataset

We evaluate our model on the public dataset: MultiWOZ2.3, which is a fully-labeled task-oriented corpora comprised of human-human written conversation. It contains 8439 multi-turn dialogues with dialogue having 6.84 turns on average. The difference between the MultiWOZ2.3 dataset and the previous versions of MultiWOZ dataset is that MultiWOZ2.3 has a cleaner and more accurate annotation as opposed to the noisier annotation of the previous MultiWOZ versions (Zhou and Small, 2019a; Han et al., 2020; Zang et al., 2020). Following previous work, only five domains, $(restaurant, hotel, attraction, taxi, train)$ are employed in out experiments.

4.2 Training Configuration

We use the pre-trained BERT-based-uncased model as the utterance encoder in our model, which has 12 hidden layers with 768 units. The limitation of the maximum sequence length isn't problematic, therefore setting length $l = 256$ would suffice.

In our experiments, Adam optimizer is utilized, whose learning rate linearly decreases from $5e-5$. We have trained the model with 25 epochs.

4.3 DST result

Both standard metrics and customized evaluation are carried out to compare our model and the state-of-the-art models. Standard metrics include Joint accuracy and Domain-Slot accuracy. Joint accuracy is the accuracy of the prediction of dialogue states. It requires that all of the thirty $(domain, slot, value)$ triplets in the dialogue state to be predicted and updated correctly. Only when the turn output Dialogue State is completely correct will $JA = 1$. In other cases, $JA = 0$, which is likely to happen when the input last turn dialogue state is wrong in the first place because models of turn-by-turn scheme typically can't self-correct. Domain-Slot accuracy is the accuracy of all the labels for each *Domain-Slot* pair in a turn. In the case of the JoDeM model, labels of a *Domain-Slot* pair includes the turn Domain, $\mathbf{D}_t$, the slot gate for the slot $\boldsymbol{\theta}_s^g$, the type prediction of the slot $\boldsymbol{\theta}_s^v$, the coref classification of the slot $\boldsymbol{\theta}_s^c$, and all the index of span detection of the slot $P_s^{start,u}$,$P_s^{end,u}$,$P_s^{start,s}$, and $P_s^{end,s}$. There are thirty *Domain-Slot* pairs in total. It's apparent that JA is a much demanding criterion to achieve and is also the most crucial metric to evaluate a dialogue state tracking system.

We make a thorough comparison over our model with the following state-of-the-art models from both schemes including TRADE (Wu et al., 2019), DS-DST (Zhang et al., 2019), IL-DST (Zhang et al., 2021), SUMBT (Lee et al., 2019a), PIN (Chen et al., 2020b), SOM-DST (Kim et al., 2019), COMER (Ren et al., 2019),DSTQA (Zhou and Small, 2019b), NA-DST (Le et al., 2020), TEN (Chen et al., 2020a), ReDST (Liao et al., 2020), ReInf (Liao et al., 2021), CSFN-DST (Zhu et al., 2020a), SAVN (Wang et al., 2020b), TripPy (Heck et al., 2020), SimpleTod (Hosseini-Asl et al., 2020), and STAR (Ye et al., 2021). The first two columns of Table 1 are the results of standard metrics. The turn-by-turn schematic DST models have shown significant performance improvement over the dialog-history scheme in both Joint accuracy and Domain-Slot accuracy. By enhancing the accuracy at the turn level, turn-by-turn schematic DST models are able to gain a much higher joint accuracy at the end. Our model, the JoDeM DST model, despite having a Domain-Slot accuracy among the best, has achieve a state-of-the-art performance boost on the joint accuracy metric. This indicates that our model has a high robustness against the avalanche phenomenon, which resulted in a better overall performance.

Model	**J Acc**	**D-S Acc**	**A Coeff**
TRADE	49.2	96.94	0.932
DS-DST	55.2	97.67	0.941
IL-DST	58.3	98.50	0.940
SUMBT	52.6	91.02	**1.0023**
PIN	54.8	97.13	0.945
SOM-DST	55.0	97.93	0.938
COMER	50.5	95.48	0.950
DSTQA	52.1	97.15	0.938
NA-DST	51.7	95.42	0.954
TEN	47.3	94.93	0.947
ReDST	64.0	98.36	0.954
ReInf	59.5	98.21	0.945
CSFN-DST	54.8	97.39	0.942
SAVN	57.6	97.86	0.944
TripPy	63.2	98.63	0.949
SimpleTod	52.0	97.60	0.933
STAR	58.4	97.95	0.945
JoDeM	**74.9**	98.07	**0.979**

Table 1: Joint accuracy, slot accuracy and avalanche coefficient on the test sets of MultiWOZ2.3.

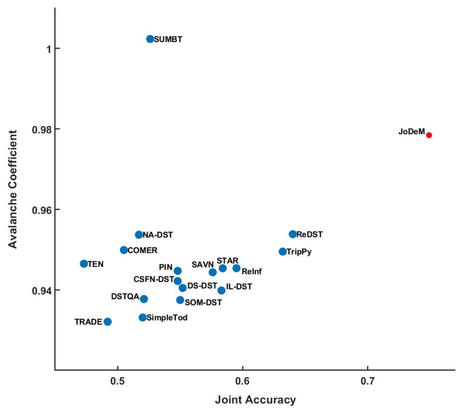


Figure 4: The correlation of joint accuracy and avalanche coefficient of various DST systems

Customized evaluation is designed to better evaluate and compare the robustness of different DST systems against the avalanche phenomenon. For quantification, we introduce a novel avalanche coefficient to describe the performance deficit ratio caused by the avalanche, α, which is calculated as $\alpha = \frac{\sqrt[\bar{l}]{\bar{p_j}}}{\bar{p_{ds}}}$, where $\bar{l}$, $\bar{p_j}$ and $\bar{p_{ds}}$ are the mean length of dialogues, Joint accuracy and Domain-Slot accuracy respectively. With fixed dialogues, the avalanche coefficient is model relevant only, which means it is an intrinsic parameter to DST systems.

From the definition, we deduce that higher the avalanche coefficient, the less a model suffers from the avalanche phenomenon. The avalanche coefficient of a DST model equals 1 when the model doesn't suffer from the avalanche phenomenon. As shown in Figure 4, despite the poor joint accuracy performance, dialogue history scheme based models has an avalanche coefficient higher than 1. Our model, among with other turn-by-turn schematic DST models, has an avalanche coefficient lower than 1, but way closer to 1 than the current state-of-the-art models, resulting in a much better overall Joint accuracy performance. This proves that addressing the avalanche is crucial for obtaining higher Joint accuracy in DST models.

4.4 Component Analysis

In order to dig deeper into the black box of the JoDeM model, we carry out detailed analysis to show the sufficiency and necessity of different components in the JoDeM model and how our design is aligned with our intuition.

To examine the *Domain Update* component, we conduct two sets of control experiments with unique variation on the original *Domain Update* component.

Variation one: We set the diagonal coefficient matrix in 2 to $\gamma = \mathbf{E}$ during the training process. This setting means that the component learns to obtain the turn domain without utilizing any last turn information.

Variation two: Similarly but different from the variation one, we set the diagonal coefficient matrix in 2 to $\boldsymbol{\gamma} = \mathbf{E}$ only during the testing process. This setting means that the model is trained given the last turn domain but being denied that information while performing on the test set.

The results from the original JoDeM model and the two variation are presented in Table 2. The metric we investigate is domain accuracy, which is the accuracy of the prediction of the turn domain. As you can see, the first column, which is the original JoDeM model, has the highest domain accuracy. The second column corresponds to variation one, which is the one with $\boldsymbol{\gamma} = \mathbf{E}$ during training process. We can see that although a model can predict the turn domain solely using the turn utterance information, but the performance is sub-par compared to the one with last turn domain. The third column is the one with $\boldsymbol{\gamma} = \mathbf{E}$ during testing only, whose decline of the domain accuracy is massive.

Original JoDeM	**Training with $\gamma = E$**	**Testing with $\gamma = E$**
97.75	91.93	73.94

Table 2: Domain Accuracy Analysis with Different Settings of the JoDeM Model.

Original JoDeM	**Variation One**	**Variation Two**
74.9	67.3	41.3

Table 3: Joint accuracy comparison on the JoDeM model with different usage setting of the multi-head attention mechanism

The significance of this set of control experiment is to demonstrate that the last turn domain plays a key role or is relied heavily in the prediction of the turn domain.

Next, we focus on the question which is the purpose of the extra multi-head attention layer before applying the *slot gate*, *type prediction*, *span detection* and *co-ref classification* components. The intuition behind utilizing multi-head attention layer between user utterance embedding and the entire dialogue embedding is that any update from the dialogue state is based on the consent of user. For example, the system may recommend a piece of information about a restaurant, but whether that information should be inserted into the dialogue state is up to whether the user takes the advice. To fairly evaluate, we train two control JoDeM models under two variation respectively. The metric we investigate is the joint accuracy.

Variation one: Instead of attending the user utterance embedding to the entire turn utterance embedding, we apply two multi-head self-attention layers on user and system utterance separately. The purpose of this variation is to examine and explore exactly what kind of attended relationship is the crux to dialogue state tracking.

Variation two: We discard the multi-head attention layer entirely, the input sequence for the *slot gate*, *type prediction*, *span detection* and *co-ref classification* components is the direct embedding of the pre-trained BERT. The goal of this variation is to examine the necessity of applying attention mechanism in the first place.

The results are shown in Table 3. Apparently, applying an additional attention layer is not only necessary but also crucial for the performance for dialogue state tracking. This observation is consistent with respect to other previous analytical work on dialogue state tracking. Furthermore, applying a multi-head cross-attention layer has the edge over a self-attention layer. This indicates that learning the relationship between the user utterance and the whole utterance is important in dialogue state tracking, which aligns with our intuition and the interactive nature of dialogue itself.

5 Conclusion

We proposed a novel, robust DST model JoDeM to address the rarely discussed problem, the Avalanche phenomenon. We showed that the trending topnotch DST systems all suffer from the Avalanche phenomenon with quantitative results and evidence. By multiple control experiments, we demonstrated how the overall structure and different techniques served the performance and robustness of the JoDeM model. We achieved a state-of-the-art performance on Joint accuracy and the criterion we design for measuring the impact of the Avalanche phenomenon. Finally, through the success of JoDeM, we show that the Avalanche phenomenon is worth solving and that there is more potential in this perspective for the DST task.

References

Vevake Balaraman and Bernardo Magnini. 2021. Domain-aware dialogue state tracker for multi-domain dialogue systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. *arXiv e-prints*, page arXiv:1506.03099.

Jie Cao and Yi Zhang. 2021. A comparative study on schema-guided dialogue state tracking.

Guan-Lin Chao and Ian R. Lane. 2019. BERT-DST: scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. *CoRR*, abs/1907.03040.

Junfan Chen, Richong Zhang, Yongyi Mao, and Jie Xu. 2020a. Neural dialogue state tracking with temporally expressive networks. *CoRR*, abs/2009.07615.

Junfan Chen, Richong Zhang, Yongyi Mao, and Jie Xu. 2020b. Parallel interactive networks for multi-domain dialogue state generation. *CoRR*, abs/2009.07616.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tür. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *CoRR*, abs/1907.01669.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tür. 2019. Dialog state tracking: A neural reading comprehension approach. *CoRR*, abs/1908.01946.

Rahul Goel, Shachi Paul, and Dilek Hakkani-Tür. 2019. Hyst: A hybrid approach for flexible and accurate dialogue state tracking. *CoRR*, abs/1907.00883.

T. Han, X Liu, R. Takanobu, Y. Lian, C. Huang, W. Peng, and M. Huang. 2020. Multiwoz 2.3: A multi-domain task-oriented dataset enhanced with annotation corrections and co-reference annotation.

M. Heck, C Van Niekerk, N. Lubis, C. Geishauser, H. C. Lin, M. Moresi, and M. Gai. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking.

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

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2019. Efficient dialogue state tracking by selectively overwriting memory. *CoRR*, abs/1911.03906.

Hung Le, Richard Socher, and Steven C. H. Hoi. 2020. Non-autoregressive dialog state tracking. *CoRR*, abs/2002.08024.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. *CoRR*, abs/2109.07506.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019a. 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.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Zheng Zhang, Yaoqin Zhang, Xiang Li, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. 2019b. ConvLab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 64–69, Florence, Italy. Association for Computational Linguistics.

S. Li, S. Yavuz, K. Hashimoto, J. Li, T. Niu, N. Rajani, X. Yan, Y. Zhou, and C. Xiong. 2020. Coco: Controllable counterfactuals for evaluating dialogue state trackers.

Lizi Liao, Yunshan Ma, Wenqiang Lei, and Tat-Seng Chua. 2020. Rethinking dialogue state tracking with reasoning. *CoRR*, abs/2005.13129.

Lizi Liao, Tongyao Zhu, Le Long, and Tat Chua. 2021. Multi-domain dialogue state tracking with recursive inference. pages 2568–2577.

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

Arvind Neelakantan, Semih Yavuz, Sharan Narang, Vishaal Prasad, Ben Goodrich, Daniel Duckworth, Chinnadhurai Sankar, and Xifeng Yan. 2019. Neural assistant: Joint action prediction, response generation, and latent knowledge reasoning. *CoRR*, abs/1910.14613.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. SOLOIST: few-shot task-oriented dialog with A single pre-trained auto-regressive model. *CoRR*, abs/2005.05298.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence Level Training with Recurrent Neural Networks. *arXiv e-prints*, page arXiv:1511.06732.

Liliang Ren, Jianmo Ni, and Julian J. McAuley. 2019. Scalable and accurate dialogue state tracking via hierarchical sequence generation. *CoRR*, abs/1909.00754.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2015. Hierarchical neural network generative models for movie dialogues. *CoRR*, abs/1507.04808.

Ryuichi Takanobu, Qi Zhu, Jinchao Li, Baolin Peng, Jianfeng Gao, and Minlie Huang. 2020. Is your goal-oriented dialog model performing really well? empirical analysis of system-wise evaluation. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 297–310, 1st virtual meeting. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Dingmin Wang, Chenghua Lin, Li Zhong, and Kam-Fai Wong. 2020a. Dialogue state tracking with pretrained encoder for multi-domain trask-oriented dialogue systems. *CoRR*, abs/2004.10663.

Yexiang Wang, Yi Guo, and Siqi Zhu. 2020b. Slot attention with value normalization for multi-domain dialogue state tracking. pages 3019–3028.

Chien-Sheng Wu, Steven Chu-Hong Hoi, and Caiming Xiong. 2020. Improving limited labeled dialogue state tracking with self-supervision. *CoRR*, abs/2010.13920.

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. *CoRR*, abs/1905.08743.

Fanghua Ye, Jarana Manotumruksa, Qiang Zhang, Shenghui Li, and Emine Yilmaz. 2021. Slot self-attentive dialogue state tracking. *CoRR*, abs/2101.09374.

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.

Jianguo 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. *CoRR*, abs/1910.03544.

Ye Zhang, Yuan Cao, Mahdis Mahdieh, Jeffrey Zhao, and Yonghui Wu. 2021. Improving longer-range dialogue state tracking. *CoRR*, abs/2103.00109.

Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Task-oriented dialog systems that consider multiple appropriate responses under the same context. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:9604–9611.

J. Zhao, M. Mahdieh, Y. Zhang, Y. Cao, and Y. Wu. 2021. Effective sequence-to-sequence dialogue state tracking.

L. Zhou and K. Small. 2019a. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering.

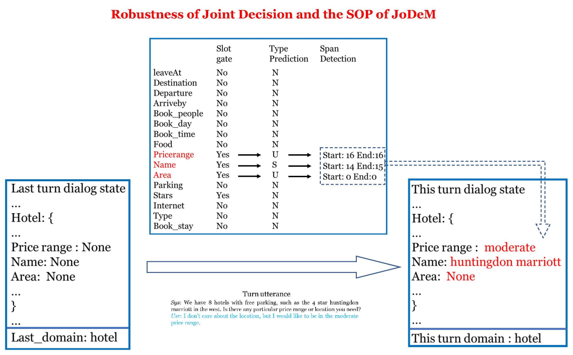
Li Zhou and Kevin Small. 2019b. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *CoRR*, abs/1911.06192.

Su Zhu, Jieyu Li, Lu Chen, and Kai Yu. 2020a. Efficient context and schema fusion networks for multi-domain dialogue state tracking. *CoRR*, abs/2004.03386.

Su Zhu, Jieyu Li, Lu Chen, and Kai Yu. 2020b. Efficient context and schema fusion networks for multi-domain dialogue state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 766–781, Online. Association for Computational Linguistics.

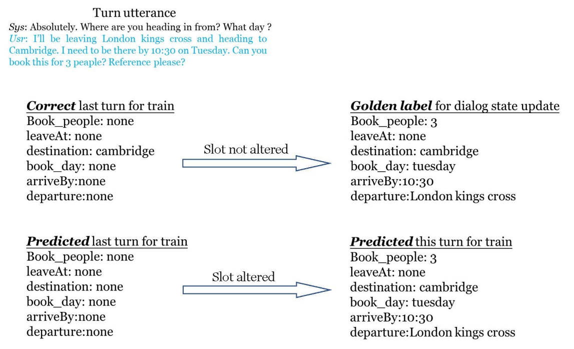
A Appendix: Example Study

To inspect the actual effect the JoDeM model have on the update prediction of dialogue states, we provide two examples to demonstrate the strength of the JoDeM model.



Sys: We have 8 hotels with free parking, such as the 4 star huntingdon marriott in the west. Is there any particular price range or location you need?
Usr: I don't care about the location, but I would like to be in the moderate price range.

Figure 5: Example on Robustness of Joint Decision Making



Sys: Absolutely. Where are you heading in from? What day ?
Usr: I'll be leaving London kings cross and heading to Cambridge. I need to be there by 10:30 on Tuesday. Can you book this for 3 peaple? Reference please?

Figure 6: Example on self-correcting of JoDeM

A.1 Example One

The first example is presented in Figure 5. It not only serves as a demonstration of the actual operation of JoDeM, but also can show the robustness of the joint decision making technique. First, the domain of the turn is obtained, which is *Hotel*. After domain is specified, the updating procedure will strictly be limited in the domain. As shown in the figure, after the domain is obtained, the focus shifts to *slot* information. According to the *Slot Gate*, slots *Price range*, *Name*, *Area* is altered from the context. After that, the value of the slot is extracted from the utterance according to the *Type Prediction* and *Span Detection*. As you can see, although *Slot gate* and *Type Prediction* made a false judgment on *Area*, it didn't lead to a wrongful update. The reason for that is that the corresponding *Span Detection* detected that the starting and ending index are appointed to the *[CLS]* token, which means no information is detected. Only when all the components have made wrongful decision will they result in a wrongful update, which is the reason why *Joint Decision Making* is a robust way to extract information in a DST system.

A.2 Example Two

The second example is presented in Figure 6. It shows that the JoDeM model can self-correct to a certain extend and why too many supplementary labels might be problematic. We focus on the *Destination* slot in the *Train* domain. As shown in the figure, the value of *Destination* is incorrect in the predicted last turn dialogue state. But it was rectified in this turn. If the predicted last turn dialogue state was correct, the correct operation at this turn is that *Slot gate* wouldn't have predicted the altering of the slot, which is aligned with the supplementary labels we have tagged. Therefore it would appear that the JoDeM model didn't get all the predictions right, but it enhanced the performance at the end. This ability of the JoDeM model takes credit from the trick we applied during the training process, which is setting the predicted values to the **GoldenLabel** when $DS_t\{D_\mathbf{t}, \boldsymbol{s}\} = v$. Had the system follow the operation of the correct labels, it wouldn't be able to right the wrongs from the past turns.

B Appendix: Discussion

Accumulation of error is a well discussed dilemma in the language generation field. Exposure bias caused by additional guiding during training resulted in subpar performance in testing. While the error accumulation in NLG and the avalanche phenomenon both originates from the auto-regressive characteristic and additional guiding during training, there are significant differences between them.

Even though the ground truth for NLG task is not unique, the guiding label during training is always one of the solutions. In DST tasks, supplementary label itself may be incorrect since they are made by comparing the ground truth dialogue state between consecutive turns.

Another significant difference is that in most auto-regressive tasks, the auto-regressive process happens within the model, which indicates that the model includes one or more auto-regressive output structured module. While in the DST systems, the auto-regressive characteristic is embedded in the pipeline of the task. The auto-regressive part is manually applied.

The above discussed differences are crucial because they render all the existing tactics and strategy during training ineffective for the DST systems. This work provides a model-based solution without altering the "pretrain + finetune" paradigm and training strategy.

C Appendix: Responsible NLP Research CheckList

C.1 Limitations and Risks

Although our work is evaluate on a public and high quality dataset, as we summarized in the abstract and introduction, the dialogue state tracking task in real world application is far more complicated. Therefore there is both limitation and risks on whether our model can perform well in application.

C.2 Use of scientific artifacts

The only scientific artifact our work applied is the dataset MultiWoz2.3 which is specifically designed for dialogue state tracking and publicly accessible. The content of the dataset doesn't contain any information that names or uniquely identifies individual people or offensive content.The dataset is about information regarding assorted places in Britain. The proportion of train/dev/test set is 8/1/1.

C.3 Computational Experiments

In our experiment, 8 GPU is used to train our model, which has 222M parameters. One training epoch takes 21 minutes. Any setting of hyperparameter, including the existing package of pretrained language bert model, is presented in the experiment section. Our result, as well as the compared result from other works, is the mean of multiple independent identically distributed tests.