A Span Extraction Approach for Dialog State Tracking: A Case Study in Hotel Booking Application

Hongjie Shi
Megagon Labs, Tokyo, Japan, Recruit Co., Ltd.
shi.hongjie@megagon.ai

1 Introduction

Traditional dialog state tracking is often formulated as a classification problem, where the dialog state is predicted as a distribution over a closed set of possible slot values within an ontology [1]. By doing so, the tracked dialog state which is a summary of current conversation, can be directly used for backend database querying or API calls in dialog system applications [2]. For example, a virtual assistant that helps users to book hotels, may fill a slot *price_range* with three possible values — high / low / intermediate defined by the query parameters of the backend hotel database. However, this classification approach of dialog state tracking may have two main disadvantages. First, the system will not be able to handle any unseen values beyond the predefined value set. While building a classifier that covers all possible values will result in impractical annotation and training cost. Second, we see that many modern designed databases [3] and search engine API can take direct natural language input as the search query. A predefined value set will limit the user query space and cause the matched results lack of variety.

To overcome these limitations, in this paper we explore the possibility of a span-extraction approach for the dialog state tracking, where the values are no longer constrained in a predefined set, but can be any string form as appeared in the dialog. The application of this approach is illustrated in Figure 1. We formulated our task as a multi-class multi-span extraction problem, with a goal to identify all contiguous spans along with its slot type from current user utterance. In this paper, we performed a particular case study on hotel booking domain, and set up 12 slot types that cover most common hotel requirement categories. In our system, the extracted spans (or the slot values) are then used as input search queries for different backend systems. The contribution of this paper is to initiate a new case study on this span-extraction dialog state tracking, and propose a simple BERT based approach for multi-slot multi-span extraction.

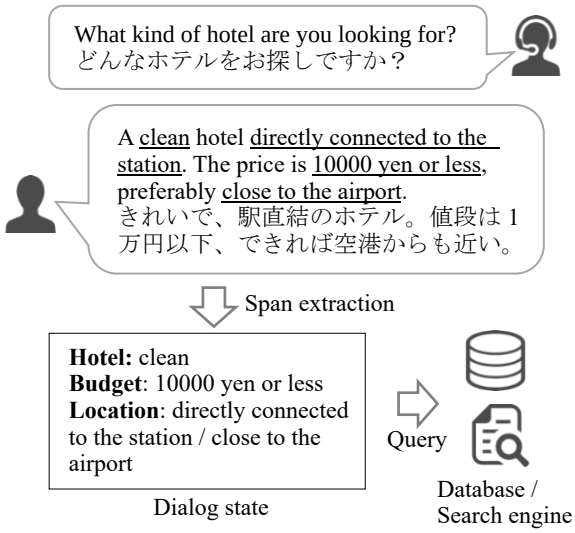

Figure 1: An example of span-extraction based dialog state tracking for hotel booking dialog system.

2 Related work

Free-form dialog state tracking Considering the limitation of traditional predefined slot values setup, free-form dialog state tracking that allow slots to be filled with any string value, have emerged in recent years. In one task track of DSTC 8 [2], the authors set up a problem to fill slots like *depart_date* with any string value derived from conversation. Others like [4] proposed to use dynamic vocabularies to track possible values beyond the predefined set. However these problems or approaches are still limited in narrow ranges of slot value form, for example the value is usually either a specific named entity

(movie name, restaurant name), or in a very specific format (date, time). In contrast, our problem considers a more wide range of possible value forms and span lengths, including adjectives, verbs, short phrases or even sentences.

Span extraction based approach The span extraction based approaches for dialog state tracking tasks have also been proposed recently. In the paper [1] the authors utilized a question answering (QA) model to predict slot value by directly pointing to the value span within the conversation. However the conventional QA model they used can not be directly applied to our multi-span extraction task, because the model output is restricted to a single answer span. On the other hand, multi-span QA models that capture multiple answer spans have also been proposed recently. In the paper [5] the author applied a sentence selection approach to identify all sentence-level spans, and in another paper [6] the authors casted the multi-span task as a sequence tagging problem. Our method resembles the latter approach, and we further extend the model with multi-label classification, as to extract spans for different slot types at the same time.

3 Dataset

The dialog corpus (Japanese) we used in this paper are the same corpus as our previous paper [7]. The data are collected between pairs of crowd workers who played travel agent and user roles respectively. The data contains 879 dialogs with 24963 agent and 11792 user utterances in total.

Annotation The span annotations are done for each user utterance in three steps:

1. Identify all span ranges for each user utterance.
2. Label each continuous span with one of the 12 slot types.
3. Label each span with *Negation* flag if the span represents a negation.

The length of span is determined so that each span has minimal string length that can be used as an independent hotel search query. The 12 slot types represent the common categories of the user requirement for the hotels, as shown in the y-axis label in Figure 2.

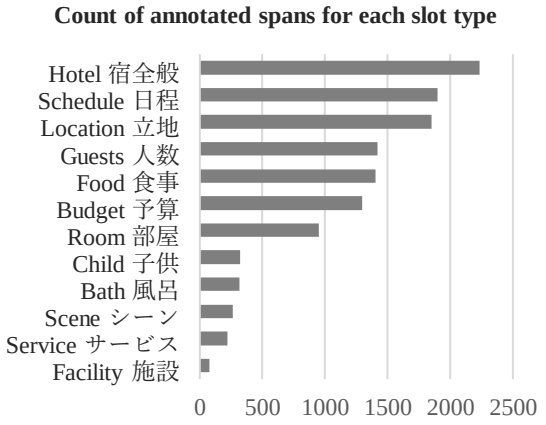

Figure 2: Distribution of annotated spans over 12 slot types.

Negation flag The *Negation* flag is used for a negative match in the database and it can be overlapped with any other of the 12 slot types. For example in the following utterance:

I don't like a room that is too cluttered.
あまりごみごみした部屋は好みではありません。

The span "too cluttered" is labeled with the slot type *Room* and the *Negation* flag simultaneously. In this paper we treat the *Negation* flag as one of the class along with 12 slot types, so that our problem can be framed as a multi-label classification.

Span statistics There are 12244 spans (including *Negation*) annotated among 6898 user utterances with average 1.8 spans for each utterance. The average span length is 5.3 characters with the shortest contains 1 and the longest contains 43 characters. The slot distribution of spans is shown in Figure 2.

4 Experiments

4.1 Models

The BERT based pre-training fine-tuning approach has achieved great success in almost all types of natural language processing tasks. In our experiment we also chose BERT as our start point. We incorporated BERT with a token classification head on top (a linear layer on top of the hidden states output), so that the model outputs a series of token-wise labels. This BERT architecture is

also used for other sequence labeling tasks such as Named-Entity-Recognition [8]. One modification we have done to the model is that we replace the output softmax to the binary sigmoid, so that each token can be associated with any number of labels. By doing so, we can train the *Negation* as one class along with the 12 slot types, and also do not need to add a dummy 'Outside/None' class for any non-span tokens.

Pre-training We compared two pre-trained BERT in our experiment. The first model is the one published in paper [9], which is pre-trained with Japanese wikipedia containing 18 million sentences. We call this model the general model. The second one is an in-domain model pre-trained with 20 million sentences of hotel reviews extracted from hotel booking site jalan.net. The pretraining details are described in paper [10]. Both models have 32k vocabulary built with Juman++[i] as the Japanese tokenizer and BPE as the subwork tokenizer. And both use 12 layers configuration as the original BERT-base configuration.

4.2 Experimental Setup

We split all user utterances into the training, validation and test dataset with portions of 70%, 15% and 15% respectively, where the training set contains 8254 user utterances and the valid / test set contains 1769 user utterances. The validation dataset is used for tuning the learning rate during training: the initial learning rate is set 0.00005 and if the loss for the validation dataset increased compared to the previous epoch, the learning rate is decayed by a rate of 0.7. The total training epochs is 7 and batch size is 32. For the model implementation, we use the transformers package from Hugging Face[ii]. We also apply the binary sigmoid instead of softmax in the output layer for multi-label output, and correspondingly modify the loss function to binary cross entropy.

Metrics The F-score used for evaluation is calculated token-wisely as following: If we represent the spans in a n-token sentence using a vector $\boldsymbol{y} = (y_1, y_2, \dots, y_n)$, where y_k is a binary digit with value of 1 if the k-th token is in the span, 0 otherwise. Then the F1 score is derived from harmonic mean of precision and recall as:

$$\text{Precision} = \frac{\boldsymbol{y} \cdot \hat{\boldsymbol{y}}}{\sum_{k=1}^{n} \hat{y}_k}, \ \text{Recall} = \frac{\boldsymbol{y} \cdot \hat{\boldsymbol{y}}}{\sum_{k=1}^{n} y_k}$$

where $\boldsymbol{y}$ is the true label and $\hat{\boldsymbol{y}}$ is the output of the model. To reduce the impact of the class imbalance, in evaluation we use the global average over all samples ignoring slot types (also known as micro-averaged F1).

5 Results

One clear conclusion from the result shown in Table 1 is that the in-domain BERT model outperforms the general model as large as almost 10%, and this improvement is most significant in Recall. If we look at the detail score for each slot type (Figure 3), a trend can be noticed is that the slot types with less training data, the more performance improvement is gained. This is not a surprising result because intuitively in-domain pre-training can help downstream fine tuning tasks with prior in-domain knowledge, and this effect is more dominant when the number of training instances is limited. This is also a general observation reported in many other tasks such as [11] and [12]. One complete example of dialog with its model output can be found in Appendix Table 3.

Table 1: Overview of performance comparison

	General model	In-domain model
Precision	0.75	0.81
Recall	0.65	0.77
Micro-F1	0.70	0.79

5.1 Error Analysis

In Table 2 we listed some typical error outputs from both models. From these examples, we can see that the in-domain model is relatively better at capturing unseen domain-specific named entity (example #1) and identifying indirect expression of negation (example #2). On the other hand the error output tends to be overlong (example #3) or with wrong slot types (example #4). One possible reason is that the likely overfitting to the domain may cause the model too sensitive to certain phrases. And also, the ambiguous interpretation of slot types may cause

[i] https://github.com/ku-nlp/jumanpp

[ii] https://github.com/huggingface/transformers

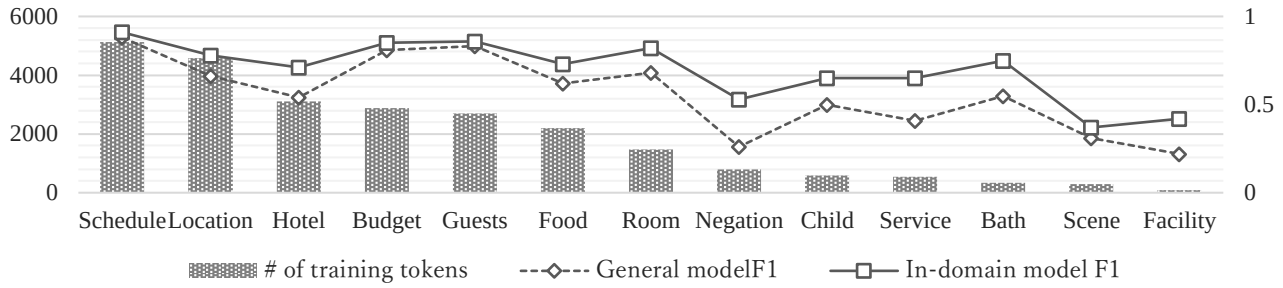

Figure 3: Comparison of accuracy for each slot type and Negation. Model details are described in Sect 4.1.

Table 2: Examples of error outputs (errors shown as red underlined).

#	User utterance	General model output	In-domain model output
1	では上州牛陶板焼つきのプランでお願いします。	**食事:** --	**食事:** 上州牛陶板焼
	I would like to book the plan with the Joshu beef roasted on a ceramic plate.	**Food:** --	**Food:** Joshu beef roasted on a ceramic plate.
2	山の眺めがすばらしいですが、少しこじんまりし過ぎの感じがします。	**宿:** 山の眺めがすばらしい **宿:** こじんまりし過ぎ	**宿:** 山の眺めがすばらしい **Neg+宿:** こじんまりし過ぎ
	The view of the mountains is great, but it feels a bit too small.	**Hotel:** The view of the mountains is great **Hotel:** too small	**Hotel:** The view of the mountains is great **Neg+Hotel:** too small
3	人数は妻と２人でお願いします。	**人数:** ２人	**人数:** 妻と２人
	There are 2 people including my wife.	**Guests:** 2 people	**Guests:** 2 people including my wife
4	静かなところがいいなあ。	**宿:** 静かな	**立地:** 静かな
	I like a quiet place.	**Hotel:** quiet	**Location:** quiet
5	都内です。	**立地:** 都	**立地:** 都内
	Within Tokyo.	**Location:** Tokyo	**Location:** Within Tokyo

a fluctuation in the annotation, such as the word “place” in example #4 can be both interpreted as *hotel* and *location*. However for using as a search query, these errors seem to be minor because the impact on the search result will be limited.

Example #5 exposes one limitation of our method, that is we do not include any dialog context in the model input. In this example, the user utterance is actually a response to the agent question “Where are you planning to come from?”, and therefore should not be interpreted as the *location* of the hotel. One way to take the dialog context into account is to concatenate previous utterances to the input with additional segment embedding [8]. We include this improvement in our future work.

6 Conclusion

In this paper we studied a new span-extraction based dialog state tracking task, and proposed a BERT-based approach for multi-slot multi-span extraction. The proposed method allows a slot to take any form of value as it is mentioned within the conversation, and therefore provides a more dynamic dialog state for querying the backend systems. While in this paper we only focused on the extraction from current user utterance, we plan to expand the model to multiple turns, including reference resolution and slot value update, with a better integration of dialog context.

Acknowledgements

We would like to thank Prof. Yuki Arase, Mai Aoki (IR-ALT) and other Megagon team members for their useful discussions and feedback.

References

[1] Shuyang Gao, et al., "Dialog State Tracking: A Neural Reading Comprehension Approach," *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue,* 2019.

[2] Abhinav Rastogi, et al., "Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset," *Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34. No. 05,* 2020.

[3] Yuliang Li, et al., "Subjective Databases," *Proceedings of the VLDB Endowment, 12(11),* 2019.

[4] Rahul Goel, et al., "Hyst: A hybrid approach for flexible and accurate dialogue state tracking," *Proc. Interspeech 2019,* pp. 1458-1462, 2019.

[5] Ming Zhu, et al., "Question Answering with Long Multiple-Span Answers," *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings,* 2020.

[6] Elad Segal, et al., "A Simple and Effective Model for Answering Multi-span Questions," *arXiv preprint arXiv:1909.13375,* 2019.

[7] Hongjie Shi, "A Sequence-to-sequence Approach for Numerical Slot-filling Dialog Systems," *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue,* 2020.

[8] Jacob Devlin, et al., "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805,* 2018.

[9] 柴田知秀, 河原大輔, and 黒橋禎夫, "BERT による日本語構文解析の精度向上," *言語処理学会 第25回年次大会発表論文集,* 2019.

[10] 林部祐太, "知識の整理のための根拠付き自然文間含意関係コーパスの構築," *言語処理学会第26回年次大会発表論文集,* 2020.

[11] Takeshi Sakaki, et al., "BERT Pre-trained model Trained on Large-scale Japanese Social Media Corpus," *https://github.com/hottolink/hottoSNS-bert,* 2019.

[12] Chi Sun, et al., "How to fine-tune bert for text classification?," *China National Conference on Chinese Computational Linguistics. Springer, Cham,* 2019.

[13] 林部祐太, "要約付き宿検索対話コーパス," *言語処理学会第27回年次大会発表論文集,* 2021.

Appendix

Table 3: An example of complete dialog and its corresponding model output. Here we are using the dialog collected in [13]. The slot values appeared in Agent utterances are not extracted by current model (future work).

Speaker	Utterance	In-domain model output
Agent	いらっしゃいませ。ご利用いただきまして、ありがとうございます。ご宿泊につきまして、お客様のご希望をお聞かせいただけますでしょうか。	
User	子供 2 人と一緒に、3 人で三重県に行きたいと思っています。11 月 13 日から 4 泊できる宿泊施設を探しています。	**日程：**11 月 13 日から 4 泊 **人数：**子供 2 人/3 人 **宿：**宿泊施設 **シーン：**三重県
Agent	かしこまりました。こちらのご宿泊は観光でのご利用でしょうか？具体的に泊まりたい場所はお決まりですか？	
User	観光です。何といいましたか、サミットが行われた辺りに行ってみたいと思うのですが。	**シーン：**観光 **立地：**サミットが行われた
Agent	伊勢・志摩地方でございますね。こちらですと、風光明媚な海の景色をお楽しみいただけます。オーシャンビューのお宿はいかがでしょうか？	
User	それは素敵ですね。ぜひお願いします。確か観光として遊覧船があったかと思うのですが。	**立地：**遊覧船
Agent	はい、伊勢湾の観光船でしょうか。もしよろしければ伊勢湾沿いのお宿はいかがでしょうか？	
User	遊覧船乗り場に近い宿がいいですね。子供がまだ小さいので。	**立地：**遊覧船乗り場に近い
Agent	さようでございますか。お子様が小さいとのことですが、近くに「志摩スペイン村」がございます。お子様をお連れになると、4 泊でも飽きることなく喜んでいただけるのではないかと思いますが、いかがでしょうか？	
User	楽しそうです。ぜひ行ってみたいのですが、車の運転ができないので交通手段がありません。バスなどは運航されているのでしょうか？	
Agent	かしこまりました。それでは、スペイン村にバスでアクセスできる場所でお宿をお探しいたします。お子様とスペイン村を観光されるようでしたら、お得なテーマパーク入場券付きプランをお探しいたしましょうか？	
User	はい、お願いします。宿泊する宿ですが、子供と一緒に入浴できる大浴場がある所がいいのですが。	**風呂：**子供と一緒に入浴できる大浴場
Agent	それでは、大浴場付きのお宿をお調べいたします。お食事は皆さま大人の料理をご用意してよろしいでしょうか？	
User	まだ小学生の子供が 1 人いますので、子供用のメニューがあると助かります。それとも、バイキングの方がいいかな。	**人数：**小学生の子供が 1 人 **子供：**子供用のメニュー **食事：**バイキング
Agent	さようでございますか。それでは、皆さまバイキングにして、お子様は子供料金があるお宿にしましょうか？	
User	はい、お願いします。それと最寄駅から送迎をしてほしいのですが。	**サービス：**最寄駅から送迎
Agent	かしこまりました。最寄り駅から送迎サービスのあるお宿でお調べいたします。	
Agent	それでは、希望条件に合うお宿をお探しいたします。	