A Sequence-to-sequence Approach for Numerical Slot-filling Dialog Systems

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

Abstract

Dialog systems capable of filling slots with numerical values have wide applicability to many task-oriented applications. In this paper, we perform a particular case study on the `number_of_guests` slot-filling in hotel reservation domain, and propose two methods to improve current dialog system model on 1. numerical reasoning performance by training the model to predict arithmetic expressions, and 2. multi-turn question generation by introducing additional context slots. Furthermore, because the proposed methods are all based on an end-to-end trainable sequence-to-sequence (seq2seq) neural model, it is possible to achieve further performance improvement on growing dialog logs in the future.

1 Introduction

Task-oriented dialog systems which assist users to complete tasks like hotel reservation, are drawing great attentions among both research and industry. Compared to conventional pipelined system, recently emerging end-to-end trainable dialog systems are showing many favorable characteristics — because of the neural models that directly learn from chatlogs of human-to-human conversation employed, such systems hold the promise of low data preparation cost, flexible response generation and the ability to evolve with new data.

In this work, we are going to explore the possibility to bring the end-to-end trainable dialog system model to the hotel reservation chatbot application, where we encounter two new problems: 1. numerical slots-filling and 2. multi-turn dialog management. To the best of our knowledge, both of them can not be fully solved using currently available end-to-end systems. In this paper, we will focus on these two problems and propose possible workarounds which can lead to satisfactory results.

2 Problem description

The hotel reservation application requires a dialog system to fill three slots with integer — number of adults (`slot:num_adult`), number of primary school children (aged 6-12) (`slot:num_c6_12`), and number of preschool children (aged 0-5) (`slot:num_c0_5`). These numerical slots are necessary because the applicable room plan (number of beds, quantity of amenities) and the pricing (food cost etc.) vary on the number of adults and children. This numerical slot-filling problem is also widely applicable to other domains such as restaurant reservation or flight booking, with slightly different slot configurations.

The challenges of building such dialog system mainly lie in two aspects. First challenge is the difficulty in the numerical slot value inference. Unlike most task-oriented dialog systems or datasets such as Wen et al. (2016); Henderson et al. (2013), where the slot filling can be either solved as a named entity extraction problem or a multi-label classification problem, the numerical slot-filling requires additional reasoning and calculation. For examples the simple expression *"My wife and me"* means 2 adults, and *"4 including 1 baby"* implies 3 adults. And moreover, the numerical inference sometimes involves with multi-turn dialog context, which brings to the second challenge.

Second challenge is the multi-turn dialog management. Many previous task-oriented dialog systems are designed in a turn-wise manner e.g. Lei et al. (2018) — the systems ask the question for particular slot in each turn and expect user to give explicit answer within that turn. If no exact slot value can be extracted from the response, the system will simply repeat the same question. This behavior is unfavorable for the numerical slot-filling, because of the likely ambiguity in the user responses. For example, no target slot value can be determined

, pages 127 - 132

from the user response *"4 people including 2 kids"*, while human agent may ask drill-down questions such as *"How old are the children"* to address this ambiguity. To achieve this human-level conversation, a dialog system capable of managing multi-turn strategy such as asking drill-down questions, is desirable.

3 Present methods

Several end-to-end model architectures have been proposed for task-oriented dialog system. Wen et al. (2016) proposed a modularly connected neural networks to enable end-to-end training. Later Lei et al. (2018) simplified this architecture to a single sequence-to-sequence model (SEQUICITY), which not only reduced the training cost but also improved the performance. More recently more advanced model like HaGAN has been applied to end-to-end learning (Fang et al., 2019). Wu et al. (2019) also explored the possibility of applying recent large pre-trained language model such as BERT and GPT-2 to the task-oriented dialog system.

After survey and review on different models, we consider the SQUICITY framework a particularly good point to start because of its simplicity and extendability. Its key idea is to encode the dialog states (slot values) into a text format which can be concatenated to the target utterances, so that any seq2seq models can handle both slot-filling and language generation at the same time. In this way, the model complexity and the training procedure are greatly simplified (refer to original paper for more details). Recently published T5 model (Raffel et al., 2019) also demonstrates the promising performance and the wide applicability of such text-to-text format training. Therefore we consider this SEQUICITY framework using seq2seq model has great potential and lower maintenance cost for commercial applications, and in this paper we chose this framework as our base model.

4 Proposed methods

4.1 Slot-filling with numerical reasoning

In order to enable the seq2seq model to perform numerical reasoning, we train the model to predict arithmetic expressions instead of numeric values. For example for the utterance "*three men and two women*", we modify the target output to be '3 + 2'[1] instead of the numeric value '5' during training. This encourages the seq2seq model to simply copy values from the input sentence[2], rather than manipulate the number directly. This method is also inspired by recent state-of-the-art models from the Discrete Reasoning Over Passages (DROP) dataset (Dua et al., 2019), where most models are trained to predict numerical spans and math operations respectively (Ran et al., 2019; Andor et al., 2019). Intuitively, by doing so, we can achieve better generalization performance because it can easily handle unseen combinations of different numbers and math operations.

4.2 Multi-turn dialog management for ambiguous user utterances

The original SQUICITY model only takes one single turn of previous utterance and slot values as model input for the response generation. This mechanism reduces the training cost, however, may hinder the model from learning multi-turn dialog strategy. For example in the following dialog:

Agent:	How many people is the reservation for?
User:	Four people including two kids. (1)
	`(total_num:4 num_child:2)`
Agent:	...
User:	...
Agent:	How old are the two children? (2)
User:	One is 5 and the other is 8.
	`(num_adult:2 num_c6_12:1 num_c0_5:1)`

It will not be possible for system to ask questions like (2) without being aware of earlier user utterance (1). To address this problem, we use additional slots to track down all necessary dialog context. We call them *context slots*. In this particular example, we use two context slots — `total_num` slot with value of 4 and `num_child` slot with value of 2 to track the information mentioned in user utterance (1). We treat these context slots just like other numerical slots — they will be carried on to the next turn's input until the goal is achieved, so that the model can refer to them at any position of the dialog. With the help of context slots, the dialog system can generate context-aware questions with less effort, and also is able to learn multi-turn dialog strategy from less data.

[1]This output consists of three tokens, which are '3', '+' and '1'

[2]Same as original SEQUICITY paper, we choose CopyNet (Gu et al., 2016) as our seq2seq architecture, so that some of the output tokens can be simply copied from the input sequence

5 Experiment and results

5.1 Slot-filling with numerical reasoning

To collect training and evaluation dataset with arithmetic expression, crowd sourcing service was used. We ask crowd workers to compose utterances using numbers given in the instruction, while avoiding directly including the answer in the sentence, so that each collected utterance requires numerical reasoning for inference. Samples of collected data are shown below[3] (with target slot values shown in the brackets):

Agent: How many people is the reservation for? **User:** One adult and one middle school child. (`num_adult:1+1 num_c6_12:0 num_c0_5:0`)
Agent: How many people is the reservation for? **User:** Four including one elementary school child. (`num_adult:4-1 num_c6_12:1 num_c0_5:0`)
Agent: Is the reservation for 4 adults? **User:** No, we have one 8-year-old child and one 3-year-old child. (`num_adult:4-1-1 num_c6_12:1 num_c0_5:1`)
Agent: Are there 3 adults and 1 preschool child? **User:** Oh, we have one more preschool child. (`num_adult:3 num_c6_12:0 num_c0_5:1+1`)

All utterances are trained with the arithmetic expressions as shown underlined above (more training details can be found in Appendix A). We also compared the proposed method to training the model with the numerical value directly. The result is summarized in the table below:

training / test data #[4]	Numerical values F1	Arithmetic expressions F1
899 / 1230	0.48	0.89
1829 / 1230	0.72	0.93
3666 / 1230	0.90	0.94

Our result shows that the proposed method (predicting algorithmic expression) outperforms predicting numeric value by a huge margin when the training data size is small. However, by increasing the training data size, the performance gap between two methods can be greatly reduced. These results can be interpreted as the following two reasons. 1. Algorithmic expression prediction have superior generalization performance for small size of training data, because it can easily handle unseen combination of numbers. On the other hand predicting with numeric value requires the model to also learn to manipulate numbers directly , therefore it may need more instances to train. 2. The algorithmic expressions appeared in the dataset is quite simple with limited range and variations. It is possible to train a seq2seq model with pretrained word embedding to be able to do simple calculation. This observation is consist with one recent paper, which also reported the good performance of neural models on addition calculation within the training range (Wallace et al., 2019).

In the real application, we can combine these two models to furthermore boost the performance in a ensemble learning way. And also, when two models give completely different answers, we can also tune the dialog system to confirm with user.

5.2 Multi-turn dialog management for ambiguous user utterances

To achieve dialog management resemble to real human-human conversation, we collected around 900 hotel reservation dialogs from pairs of workers who played agent or user roles. Each dialog covers all topics in hotel reservation, including location, price range, preference and so on. We then analyzed all sub dialog segments concerning `total_num/num_child` slot from each dialog, and extracted 7 representative drill-down questions as listed below:

- *Are all people above middle school students?*
- *Are there any children in the group?*
- *Are there any children who are primary school students or below?*
- *Are all people adults?*
- *How old is the child?*
- *If there is any child in the group, could you please tell me their ages?*
- *Is it for* `<total_num>` *adults?*

In order to collect more variations of possible user utterances which are applicable to these questions, again we used crowd sourcing service and asked workers to fill in the blank of the dialog below:

Agent:	How many people is the reservation for?
User:	(a)
Agent:	`<one of the questions shown in above list>`
User:	(b)
Agent:	Alright, so it is `<n1>` adults, `<n2>` child (6-12) and `<n3>` child (0-5).

Example of collected dialogs:

Agent:	How many people is the reservation for?
User:	For 6 people including kids.
Agent:	How old are the children?
User:	We have one primary school child, two preschool children.
Agent:	Alright, so it is 3 adults, 1 child (6-12) and 2 children (0-5).

[3] All data used in this paper are collected and trained in Japanese. The examples showing here are the English translations.

[4] All test data are identical. And F1 scores are the weighted average over all three slots.

Agent utterances	User utterances	Predicted slots	Generated response
How many people is the reservation for? お泊りの人数はお決まりでしょうか？	Four members of my family. 家族４名です。	total_num 4	Are there any children in the group? お子様はいらっしゃいますか？
Are there any children in the family? ご家族の中にお子様はいらっしゃいますか？	No there isn't. いません。	num_adult 4 num_c6_12 0 num_c0_5 0	<DIALOG_END>
How many people is the reservation for? お泊りの人数はお決まりでしょうか？	2 adults and 1 child please. 大人２人子供１人でお願いします。	num_adult 2 num_child 1	How old is the child? お子様のご年齢をお伺いできますか？
Could you tell me the age of your child? お子様のご年齢を教えていただけますか？	Four years old! ４歳です！	num_adult 2 num_c6_12 0 num_c0_5 1	<DIALOG_END>
How many people is the reservation for? お泊りの人数はお決まりでしょうか？	For 3 people. ３名です。	total_num 3	Are all people adults? 皆様大人の方でいらっしゃいますか？
Are all people adults? 皆様大人の方でいらっしゃいますか？	2 adults and 1 child. 大人２名と子供１名です。	num_adult 2 **num_c6_12 0** **num_c0_5 1**	**<DIALOG_END>**
How old is the child? お子様のご年齢をお伺いできますか？	10 years old. １０歳です。	N/A	N/A

Table 1: Samples of test dialogs with model outputs, including the slot values and next response. The model is trained and predicted in Japanese. Red bold texts are the results considered to be incorrect.

With these data, our model is ready to train for the multi-turn dialog strategy. During the training, turn (a) is trained with context slots (Sect. 4.2) and next-turn agent question. Turn (b) is trained with target slot values consistent with the last agent utterance, and a special token <DIALOG_END>.

To evaluate the model, we extracted 20 dialog segments that contain drill-down questions from actual human-human dialogs as hold-out test dataset, and only train the model with 2000 crowdsourcing collected dialogs. Samples of generated responses from the test data can be found in Table 1. Human evaluation shows that 80% of model generated responses are reasonable (more results including comparison with baseline can be found in Appendix B), however compared to actual human dialogs, the responses generated by model tend to be less diverse. In particular the question which quotes user context — "*Is it for* <total_num> *adults?*", rarely appears in the model output, even though it has same number of training data as the other questions. The lack of variation in the generated output is also a common issue that has been studied in previous general-purpose and task-oriented dialog models (Shao et al., 2017; Rajendran et al., 2018).

5.2.1 Agent utterance normalization

Another problem we encountered when training with large data is that, multiple correct next utterances corresponding to the same dialog state may exist in the training corpus. For example, followed by the same user response "*We have three people*", some of dialog contains next utterance "*Are there any children in the group?*", while others contain different utterance like "*Is it for 3 adults?*". This may cause the training difficult to converge, and therefore results in lower train and test accuracy.

To address this problem, Rajendran et al. (2018) proposed a method which uses a combination of supervised learning and reinforcement learning. However for the model we used, we found this reinforcement learning approach unstable and very sensitive to the heuristic determined rewards. After several unsuccessful trials, we decide to simply normalize all agent utterances based on the appearance frequency, so that for each unique dialog state (or user response), only one possible next utterance exists in the training corpus. By doing so, we are able to achieve almost 100% train accuracy and better test accuracy than previous results. This method works but is less sophisticated compared to the reinforcement learning. We will continuously explore alternatives to improve it in the future.

6 Conclusion and future work

In this paper, we proposed two methods for improving the original end-to-end dialog system on numerical slot-filling. By training the model to predict arithmetic expressions, the dialog system can perform numeric reasoning more robustly, and with newly included context slots, the dialog system is able to generate multi-turn questions for ambiguous user responses.

Future work may include extending the current seq2seq network to more recent large-scale pre-trained models such as RoBERTa, as suggested in Talmor et al. (2019), for a better performance in reasoning task. And also the proposed multi-turn dialog management approach should be extensively tested on other slots and domains.

Acknowledgments

I would like to thank Dr. Hidekazu Tamaki for the help of data collection, and Prof. Yuki Arase for helpful research advice.

References

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving bert a calculator: Finding operations and arguments with reading comprehension. *arXiv preprint arXiv:1909.00109*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

Ting Fang, Tingting Qiao, and Duanqing Xu. 2019. Hagan: Hierarchical attentive adversarial learning for task-oriented dialogue system. In *International Conference on Neural Information Processing*, pages 98–109. Springer.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Matthew Henderson, Blaise Thomson, and Jason Williams. 2013. Dialog state tracking challenge 2 & 3.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Janarthanan Rajendran, Jatin Ganhotra, Satinder Singh, and Lazaros Polymenakos. 2018. Learning end-to-end goal-oriented dialog with multiple answers. *arXiv preprint arXiv:1808.09996*.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. Numnet: Machine reading comprehension with numerical reasoning. *arXiv preprint arXiv:1910.06701*.

Louis Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating high-quality and informative conversation responses with sequence-to-sequence models. *arXiv preprint arXiv:1701.03185*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019. olmpics–on what language model pre-training captures. *arXiv preprint arXiv:1912.13283*.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do nlp models know numbers? probing numeracy in embeddings. *arXiv preprint arXiv:1909.07940*.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2016. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*.

Qingyang Wu, Yichi Zhang, Yu Li, and Zhou Yu. 2019. Alternating recurrent dialog model with large-scale pre-trained language models. *arXiv preprint arXiv:1910.03756*.

Agent utterances	User utterances	Predicted slots	Generated response
How many people is the reservation for? お泊りの人数はお決まりでしょうか？	There are four. ４人なんですが。		Are there any children in the group? お子様はいらっしゃいますか？
Are there any children in the group? お子様はいらっしゃいますか？	There is one child. 子供は一人です。	**num_c6_12 0** **num_c0_5 0**	**<DIALOG_END>**
Can you tell me how old the child is? お子様のご年齢も教えていただけますか？	5 years old. ５歳です。	N/A	N/A
How many people is the reservation for? お泊りの人数はお決まりでしょうか？	There are four. ４人なんですが。	num_adult 4	Are there any children in the group? お子様はいらっしゃいますか？
Are there any children in the group? お子様はいらっしゃいますか？	There is one child. 子供は一人です。	**num_adult 4** **num_c6_12 1** **num_c0_5 0**	**<DIALOG_END>**
Can you tell me how old the child is? お子様のご年齢も教えていただけますか？	5 years old. ５歳です。	N/A	N/A
How many people is the reservation for? お泊りの人数はお決まりでしょうか？	There are four. ４人なんですが。	total_num 4	Are there any children in the group? お子様はいらっしゃいますか？
Are there any children in the group? お子様はいらっしゃいますか？	There is one child. 子供は一人です。	total_num 4 num_child 1	How old is the child? お子様のご年齢をお伺いできますか？
Can you tell me how old the child is? お子様のご年齢も教えていただけますか？	5 years old. ５歳です。	num_adult 3 num_c6_12 0 num_c0_c5 1	<DIALOG_END>

Table 2: Samples of evaluation results of baseline #1, baseline #2 and proposed method in order from the top. Red bold texts are the results considered to be incorrect.

A Details of experiments for slot-filling with numerical reasoning

The SEQUICITY framework processes dialog output in two stages: in the first stage, it decodes a text form of slot values, which is called *belief span* (bspan) in the original paper; in the second stage, it decodes a machine response conditioning on the belief span decoded in the first stage. The processed input and output of each stage are summarized as below:

1st stage	
Inputs:	$bspan_{t-1} \oplus agent_{t-1} \oplus user_t$
Outputs:	$bpsan_t$
2nd stage	
Inputs:	$bspan_{t-1} \oplus agent_{t-1} \oplus user_t \oplus bspan_t$
Outputs:	$agent_t$

Table 3: Two-stage process used in SEQUICITY framework.

where $t-1$ represents the previous turn, t the current turn and $\oplus$ the concatenation operator. Since in the experiment 5.1 we only examine the model performance on slot-filling, only the first stage above is used. And also, because all data we collected in 5.1 only contain single turn (with no dialog history), $bspan_{t-1}$ is set empty during training and test. Sample of encoded $bpsan_t$ is shown below:

slots: num_adult:4-1 num_c6_12:1 num_c0_5:0
bspan: <slot1>␣4␣-␣1␣<slot2>␣1␣<slot3>␣0

where <slot1>, <slot2> and <slot3> are the special tokens for indicating num_adult, num_c6_12 and num_c0_5 slot respectively.

B Compare multi-turn dialog performance with baseline

To compare the proposed method (context slots) with original SEQUICITY framework, we performed additional comparison experiments. Here we modified collected multi-turn dialog's context slots in two ways so that it can be applied with original model: 1. simply delete total_num and num_child slots; 2. move total_num and num_child values to num_adult and num_c6_12 slots if unfilled.

Both of the baseline methods degrade in performance because: 1. deleting context slots causes missing out context information; 2. filling values with other slots causes indistinguishable value interpretation. Same as before, we evaluate 20 human-human dialog sections with both baseline methods. Compared to 80% success rate achieved by the proposed method, baseline #1 and #2 can only reach 20% and 70% respectively. Furthermore, we find that baseline #1 fails in almost all cases, while baseline #2 tends to wrongly generate <DIALOG_END> signal where drill-down question is necessary for more than one turn. As the example shown in Table 2, baseline #2 is able to generate correct drill-down question for the first turn, while fails on the second turn. This is partially because that the model can only access the dialog history by previous belief span (as explained in Table 3), which is inaccurate in this case due to lack of context slots.