Improving Sequence-to-Sequence Semantic Parser for Task Oriented Dialog

Chaoting Xuan
VMware
cxuan@vmware.com

Abstract

Task Oriented Parsing (TOP) attempts to map utterances to compositional requests, including multiple intents and their slots. Previous work focus on a tree-based hierarchical meaning representation, and applying constituency parsing techniques to address TOP. In this paper, we propose a new format of meaning representation that is more compact and amenable to sequence-to-sequence (seq-to-seq) models. A simple copy-augmented seq-to-seq parser is built and evaluated over a public TOP dataset, resulting in 3.44% improvement over prior best seq-to-seq parser (exact match accuracy), which is also comparable to constituency parsers' performance[1].

1 Introduction

Today, most virtual assistants like Alexa and Siri are task oriented dialog systems based on GUS architecture (Bobrow et al. 1977; Jurafsky and Martin. 2019). They parse users' utterances to semantic frames composed of intents and slots. An intent normally represents a web API call to some downstream domain application to fulfill certain task. Slots correspond to parameters required in web API calls. In this paper, the task of parsing utterances to semantic frames is called Task Oriented Parsing (TOP).

Many prior work (Liu and Lane, 2016; Goyal et al. 2018) concentrate on parsing single-intent requests in which one utterance contains only one intent and its slots. Shah et al. (2018) proposes a hierarchical TOP representation to model the nested requests: one utterance contains multiple recursive intents and their slots. Figure 1.a shows an example of the hierarchical TOP representation, which is called *base representation* in this paper. Other than expressiveness, base representation also enjoys the easy annotation, efficient parsing and low adoption barrier in practice. Two types of models have been employed to perform TOP tasks: seq-to-seq models, and constituency parsing models (Dyer et al., 2016; Gaddy et al. 2018). It has been reported that the latter consistently outperforms the former, probably because constituency parsing algorithms are dedicated to serving tree-based representation by design, while seq-to-seq architecture are purposed to serve more generalized form of representations such as graph and logical form (Dong and Lapata, 2016; Jia and Liang 2016).

In this paper we introduce a compact TOP representation, which has fewer tokens than base presentation. Further, we build a simple seq-to-seq model with attention-based copy mechanism to evaluate the effectiveness of the compact representation. Experimental results on a public TOP dataset show that this approach can significantly improve seq-to-seq parser's inference performance and close its gap to current constituency parsers, who cannot handle the new TOP representation.

2 Related Work

Shah et al. (2018) proposes the hierarchical TOP representation and uses RNNG (Dyer et al., 2016), a standard transition-based constituency parsing algorithm, to build a TOP parser, which outperforms the baseline seq-to-seq parsers by 2.64%. Einolghozati et al. (2018) further optimizes the RNNG parser using ensembling, contextual word embedding and language model re-ranking, leading to higher exact match accuracy. However, training a RNNG model is expensive and almost one-scale slower than training a seq-to-seq model. Later, Pasupat et al. (2019) presents a chart-based (constituency) TOP parser, and it can reach fast training and high inference accuracy simultaneously.

3 Representation

In base representation, words are terminals, and intents and slots are nonterminals. The root node is an intent, and an intent is allowed to be nested inside a slot. In addition, base representation

1. Source code is available at https://github.com/cxuan2019/Top

Proceedings of the First Workshop on Interactive and Executable Semantic Parsing, pages 18–22
Online, November 19, 2020. ©2020 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17

follows three constraints: 1. The top-level node must be an intent, 2. An intent can have words and/or slots as children, 3. A slot can have either words or an intent as children.

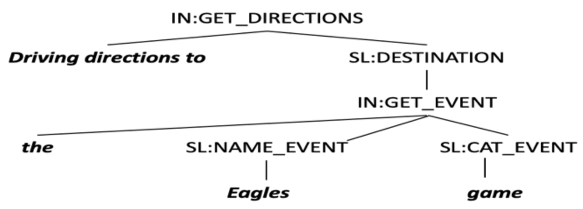


Fig 1.a: Base Representation. Intents are prefixed with IN: and slots with SL:.

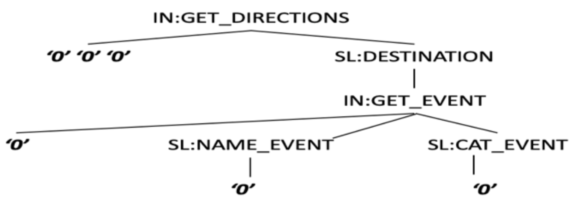


Fig 1.b: LOTV Representation. All words are replaced with token '0'.

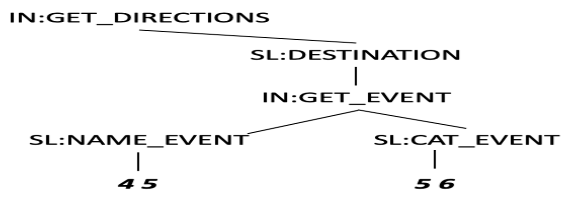


Fig 1.c: Compact Representation. Words are either gone or replaced with word indexes.

To simply seq-to-seq models, a single special token is used to replace multiple words in parses, which is called Limited Output Token Vocabulary (LOTV) representation (Shah et al., 2018). In the Figure 1.b, the special token used in LOTV representation is '0'. After using LOTV representation to substitute base representation, seq-to-seq model performs much better: almost 7% increase.

Compact representation is based on two observations: 1. Direct child tokens under an intent node are unnecessary to final execution of API calls; 2. A span of continuous words in the leaf of base representation can be encoded as a pair of positional indexes of starting word and ending word in source utterance. Specifically, compact representation is defined as a tree: root node is an intent; an intent node has either child slot nodes or no child node; a slot node has one child: either an intent node or a pair of word indexes that encode a continuous word span. Figure 1.c shows an example of compact representation.

Apparently, compact representation has fewer tokens than base representation and LOTV presentation. Its Vocabulary size is smaller than base representation, but bigger than LOTV representation.

4 Data

The TOP dataset[2] is introduced in the work of Shah et al. (2018), and it covers two domains: navigation and events. The utterances contain three types of queries: navigation, events and navigation to events. There are total 44783 annotated utterances with 25 intents and 36 slots. Each utterance is annotated with a hierarchical meaning representation. About 30% of records have nested requests. Among these data, the median depth of the trees is 2.54, and median length of the utterances is 8.93 tokens.

In this work, we remove the records that have IN:UNSUPPORTED intent from the dataset. After this, the dataset has 28414 training records, 4032 validation records and 8241 test records, identical to (Pasupat et al., 2019). Original dataset uses base representation, and we convert them to LOTV representation and compact representation. Average token lengths of LOTV and compact representations are 17 and 12; their vocabulary sizes are 60 and 93 respectively. Table 1 presents more statistics about the final dataset.

5 Model

We use a simple seq-to-seq with attention neural architecture to frame the TOP problem. Encoder is one-layer bi-directional recurrent neural network with LSTM (Hochreiter and Schmidhuber, 1997). The final output hidden states of both directions are concatenated and projected to the first input state of decoder through a linear layer. In decoder, attention and output token at time step t are computed as below:

$$z_t = [embed(y_{t-1});\ o_{t-1}] \quad (1)$$
$$h^{dec}, c^{dec} = LSTM(z_t, h^{dec}_{t-1}, c^{dec}_{t-1}) \quad (2)$$
$$e_t = (h^{dec}_t)^T W_{attProj} h^{enc} \quad (3)$$
$$\alpha_t = Softmax(e_t) \quad (4)$$
$$a_t = \sum_i^m \alpha_{t,i} h^{enc}_i \quad (5)$$
$$u_t = [h^{enc}_i;\ a_t] \quad (6)$$
$$o_t = Dropout(Tanh(W_u\, u_t)) \quad (7)$$
$$y_t = Softmax(W_{vocab} o_t) \quad (8)$$

Where y is output token, h, c are hidden state and context, α is attention score, a is attention, o is combined output. $W_{attProj}$ and W_u are trainable parameters.

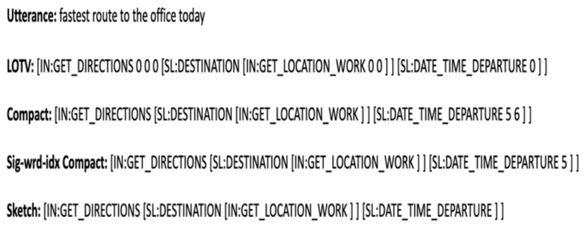
2. TOP dataset is available at http://fb.me/semanticparsingdialog

To better predict the word indexes in compact representation, we implement an attention-based copy mechanism, introduced by Eric and Manning (2017). First, we define the largest word index (utterance length) as system parameter and expand the decoder's vocabulary to include all word indexes from zero to the largest word index; then we modify the formula (6) to directly add the attention score α to compute the output tokens as below:

$$u_t = [h_i^{enc};\ a_t;\ \alpha_t]$$

Here, attention score is padded to the largest word index. The addition of attention score can provide useful signals to decoder to improve its prediction on word indexes.

We call the original model (without copy mechanism) as *vanilla seq-to-seq*, and the model with copy mechanism as *copy-augmented seq-to-seq*. In this paper, we make two hypotheses: 1. TOP parsers should benefit the shorten parses of compact representation and produce better inductive bias than LOTV representation despite the increase of token vocabulary size; 2. Copy mechanism should boost the prediction performance of seq-to-seq model.


Utterance: fastest route to the office today

LOTV: [IN:GET_DIRECTIONS 0 0 0 [SL:DESTINATION [IN:GET_LOCATION_WORK 0 0]] [SL:DATE_TIME_DEPARTURE 0]]

Compact: [IN:GET_DIRECTIONS [SL:DESTINATION [IN:GET_LOCATION_WORK]] [SL:DATE_TIME_DEPARTURE 5 6]]

Sig-wrd-idx Compact: [IN:GET_DIRECTIONS [SL:DESTINATION [IN:GET_LOCATION_WORK]] [SL:DATE_TIME_DEPARTURE 5]]

Sketch: [IN:GET_DIRECTIONS [SL:DESTINATION [IN:GET_LOCATION_WORK]] [SL:DATE_TIME_DEPARTURE]]

Fig 2: Examples of four representations in text format.

6 Evaluation

6.1 Representations

As mentioned before, with seq-to-seq model, LOTV representation can outperform base representation by large margin, so we exclude the base representation from the experiment. Besides LOTV and compact representations, we introduce two additional representations: *single-word-index compact representation* and *sketch*. In compact representation, a slot's content is denoted as a pair of word indexes, and it can be further reduced to a single word index for those slots that have exactly one word in its content. We would like to find out if this further token-size decrease by single-word-index compact representation can produce more inferencing benefits than compact representation.

As LOTV, compact and single-word-index compact representations share the same tree skeleton (nonterminal nodes) and only differ in leaves (terminal nodes), we extract the tree skeleton as a standalone representation, called *sketch*. We think studying sketch representation can help better understanding the nonterminal and terminal's contributions to prediction overheads among peer representations. Note that translating to a sketch parse cannot accomplish a TOP task by itself, as the parse has no slot contents (web API parameters). The sketch idea is inspired by Dong and Lapata (2018). Figure 2 shows an example of four representations in the experiment. Statistics of token lengths and vocabulary sizes of the representations are presented in Table 1.

Reps	***Non-terminal Len***	***Terminal Len***	***Total Len***	***Vocab Size***
LOTV	8	9	17	60
Compact	8	4	12	93
Sig-wrd-idx Compact	8	3	11	93
Sketch	8	0	8	59

Table 1: Average token lengths of four representations in test dataset (right bracket is counted as nonterminal)

6.2 Configurations

We use vanilla seq-to-seq model with LOTV representation as baseline and compare it with four other configurations: vanilla seq-to-seq model with compact representation; copy-augmented seq-to-seq model with compact representation; copy-augmented seq-to-seq model with single-word-index compact representation; and vanilla seq-to-seq with sketch representation. We choose exact match accuracy as metrics in this work, which is percentage of full trees that are correctly predicted.

6.3 Hyperparameters

Similar to previous TOP work, we use pre-trained 200b GloVe embeddings (Pennington at el. 2014). To make comparison fair, we ensure all four configurations share almost same set of hyper parameters: fixed random seed, batch size is 32; source input embedding size is 200; target input embedding size is 128; both encoder and decoder hidden size are 512; drop out value is 0.5; using Adam optimizer (Kingma and Ba, 2014) with learning rate 0.001 and decay rate 0.5; using cross entropy as loss function; running 50 epochs with early stops; top 2 beam search in inference.

6.4 Results

The main results are shown in Table 2. It can be observed that configuration 2 clearly outperforms configuration 1 by 2.61%, which confirms the first hypotheses: shorter token sequences are easier to learn and inference than longer token sequences, even with bigger-size vocabulary. One explanation is that compact representation has small vocabulary size (94), and seq-to-seq model is complex and powerful enough to accommodate the small increase of vocabulary size such that the performance of token prediction doesn't drop much. On the other hand, the longer token sequence makes the probability of exact match get worse quickly due to compounding conditional probabilities in a series of token predictions

Config ID	Model	Reps	Acc	Time (Sec)
1	Vanilla Seq2seq	LOTV	78.41	35
2	Vanilla Seq2seq	Compact	81.02	34
3	Copy-augmented Seq2seq	Compact	**81.68**	35
4	Copy-augmented Seq2seq	Sig-wrd-idx Compact	81.06	33
5	Vanilla Seq2seq	Sketch	84.03	28
6	RNNG Parser	Base	80.63	-
7	Span-based Parser	Base	**81.80**	-

Table 2. Exact match accuracies and training time per epoch of five configurations and two constituency parsers.

The configuration 3 performs better than the configuration 2 with edge of 0.66%, which confirms the second hypotheses: copy mechanism helps improving the word index prediction. Originally, learning word indexes requires model to have certain reasoning capability: connecting a 'word index' token to actual position in source utterance. In general, neural network is good at pattern recognition and but weak in reasoning. Copy mechanism can reduce the reasoning barrier and allows more leverage of neural network's strength in pattern recognition.

Comparing with compact representation, single-word-index compact representation has shorter token length, but its prediction performance gets worse, as observed in configuration 4's result. One possible reason is that compact representation has more predictable (word index) token occurrence pattern: its word index tokens always show up in pair right after a slot token, while single-word-index compact representation may have one or two word index tokens after a slot token, making tokens more unpredictable.

The configuration 5's result reveals the upper bound of other four configurations. The gap between configuration 3 and 5 is relatively small (2.35%), so we think the future research should pay more attention to improving the sketch's prediction, which is 84.03% at the point. Last, it can be seen that configuration 2, 3 and 4's accuracy results are comparable to two constituency parsers (Shah et al., 2018; Pasupat et al., 2019).

Config ID	Nonterminal Errors	Terminal Errors	Total Errors
1	1553	1188	1779
2	1300	971	1564
3	1243	945	1510
4	1293	987	1561
5	1316	0	1316

Table 3. Error counts of five configurations.

Error analysis. We count three types of inference errors in test dataset: nonterminal sequence (sketch) match errors; terminal sequence match errors; all token sequence match errors. When computing terminal sequence errors, consecutive terminals in a span are concatenated and treated as a single token. The result is listed in Table 3. Other than re-confirming the observations and arguments mentioned above, we have two new findings: 1. the copy mechanism seems able to boost both terminal and nonterminal inferences at same time (based on configuration 2 and 3's results). This is probably caused by the fact that decoder also gets some helpful clues from attention scores when predicting nonterminal tokens; 2. Compact representation (configuration 2 and 3) have less nonterminal errors than sketch representation (configuration 5). One possible explanation is that terminal (word index) token adds more contexts when predicting nonterminal tokens, e.g., if previous token is a word index, then current token cannot be intent, which narrows down the scope of token prediction.

7 Conclusions

In this paper, we propose a compact representation for TOP, which is more friendly to seq-to-seq parsers and demonstrates better performance than base representation and LOTV representation. It opens up another door to improve the semantic parsing for task oriented dialog.

References

D. G. Bobrow, R. M. Kaplan, M. Kay, D. A. Norman, H. S. Thompson, and T. Winograd. 1977. *GUS, A Frame-Driven Dialog System*. Artificial Intelligence, 8:155–173.

M. Eric and C. D. Manning. 2017. *A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue*. SIGDIAL .

L. Dong and M. Lapata. 2016. *Language to logi- cal form with neural attention.* In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 33–43, Berlin, Germany.

L. Dong and M. Lapata. *Coarse-to-fine decoding for neural semantic parsing*. 2018. arXiv preprint arXiv:1805.04793.

C. Dyer, A. Kuncoro, M. Ballesteros, and N. A. Smith. 2016. *Recurrent neural network grammars*. In Proc. of NAACL.

A. Einolghozati, P. Pasupat, S. Gupta, R. Shah, M. Mohit, M. Lewis, and L. Zettlemoyer. 2018. *Improving semantic parsing for task oriented dialog*. In Conversational AI Workshop at NeurIPS.

D. Gaddy, Mitchell Stern, and Dan Klein. 2018. *What's going on in neural constituency parsers? an analysis.* In North American Association for Computational Linguistics: Human Language Technologies (NAACL-HLT).

A. K. Goyal, A. Metallinou, and S. Matsoukas. *Fast and Scalable Expansion of Natural Language Understanding Functionality for Intelligent Agents.* 2018. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers). Association for Computational Linguistics.

S. Gupta, R. Shah, M. Mohit, A. Kumar, and M. Lewis. 2018. *Semantic parsing for task oriented dialog using hierarchical representations*. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. *Long short-term memory.* Neural computation, 9(8):1735–1780.

R. Jia and P. Liang. 2016. *Data recombination for neural semantic parsing*. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 12–22, Berlin, Germany.

D. Jurafsky, and J. H. Martin. 2019. *Speech and language processing: An introduction to natural language processing computational linguistics and speech recognition,* (Version 3).

D. P. Kingma and J. Ba. 2014. *Adam: A method for stochastic optimization*. arXiv preprint arXiv:1412.6980.

B. Liu and I. Lane. 2016. *Attention-based recur- rent neural network models for joint intent detection and slot filling*. In INTERSPEECH.

P. Pasupat, S. Gupta, R. Shah, M. Lewis, and L. Zettlemoyer. 2019. *Span-based Hierarchical Semantic Parsing for Task-Oriented Dialog*. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP).

J. Pennington, R. Socher, and C. Manning. 2014. *Glove: Global Vectors for Word Representation*. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, 1532–1543.