ConSmBop: Using Buttom Up Interaction History Representations for Context-Dependent Semantic Parsing

Itay Levy
itaylevy@cs.tau.ac.il

1 Introduction

1.1 Task

Context-Dependent semantic parsing is the task of mapping a sequence of context-dependent user utterances to structured queries. This task simulates a real-world setting, where users tend to ask a sequence of thematically related questions to learn about a particular topic or to achieve a complex goal. Context modeling is challenging because the users may explicitly refer to or omit previously mentioned entities and constraints, and may introduce refinements, additions or substitutions to what has already been said.

Dataset This work focuses on SParC (Yu et al., 2019) - a large-scale dataset of context-dependent questions over a number of databases in different domains annotated with the corresponding SQL representation. People involved in the question creation process of this dataset were encouraged to come up with context-dependent questions that have a thematic relation such as refinement, theme-entity, theme-property, and answer refinement/ theme as defined by (Bertomeu et al., 2006).

1.2 Hypothesis

Inspired by the characterization of thematic relations, we hypothesize that programs corresponding to utterances within a specific interaction, are composed of similar sub-programs. Therefore a parser capable of learning representations for meaningful semantic subprograms has a lot of potential for modeling this task. SmBoP (Rubin and Berant, 2021) possesses that special ability. SmBoP is a semi-autoregressive bottom-up semantic parser, which takes an utterance and a DB schema as input, and constructs at decoding step t the top-K sub-trees of height $\leq t$.

In this work we study several methods to enhance SmBoP's contextual modeling. We alter the encoder to address previous utterances, and enrich the decoder at each decoding step by reusing sub-trees of the same height predicted in previous turns within the same interaction.

Our best model achieves a 4.4% improvement over the baseline. Surprisingly, most of the contribution is due to the encoder modification, rather than reusing sub-trees at decoding.

2 Method

2.1 Transitioning from a single utterance to an interaction

SmBoP operates on a batch of independent instances, each corresponding to a single utterance. Modeling interaction context means that the model should be able to access input fields and computations over previous turns within an interaction. Hence we created nested instances where each instance corresponds to an interaction and each sub-instance corresponds to a single turn. Each model in this work encapsulates SmBoP in a loop, feeds it a sub-instance at each turn and sums the losses. This way, we are able to use standard data loading techniques and avoid forcing a specialized order for accessing predictions over previous turns.

2.2 Encoder

SmBoP's encoder is composed of RAT-SQL layers stacked on top of a pre-trained language model. The input to the encoder is the utterance concatenated to a linearized form of the database schema.

Concatenating all previous utterances (aka concat-encoder) The first attempt at modeling the context of the dialog is by encoding past utterances as well as the current one. We replace the single utterance used in SmBoP's original encoder, with the current utterance concatenated to all previous utterances separated by a special token $< s >$, as this simple strategy has been shown effective in

state-of-the-art context-dependent semantic parsing systems (Yu et al., 2021; Zhang et al., 2019). The maximum allowed sequence length in SmBoP is 512 due to the use of grappa-large as a pre-trained language model. Data analysis over the SParC train set reveals that when concatenating all utterances in an interaction to the linearized schema, only 86% of examples fit in 512 sequence length limit, and that 99% of examples fit in 1024 sequence length limit. Therefore we changed the limit to 1024 and replaced grappa-large with longoformer-base-4096. This encoder was used in all models presented in this work, except for when stated otherwise (e.g. baseline, efficient encoder experiments).

Baseline In order to perform a controlled experiment, the baseline method of this work parses each turn independently of other turns, and ignores the history and context of the dialog.

2.2.1 Efficient Encoder Attempt

A significant benefit of SmBoP is its speed-up compared to autoregressive parsers. Although it's not the main focus of this work, we were motivated to maintain this advantage in this follow-up work. Instead of repeating the encoding process for each turn, it is possible to perform a single forward pass of the encoder per the entire interaction and still get contextualized representations for each turn. However, we must make sure that each turn is not exposed to information from future turns. A possible solution is concatenating all utterances in the interaction and the linearized schema, and using autoregressive attention mechanism that enables each position to attend only to tokens of this turn and previous turns. Positions that correspond to schema tokens shouldn't attend to utterances to prevent indirect exposure. Due to generalization issues discussed in ablation 4.2.3, this encoder was not used in models of this work, but is still worth mentioning for completeness.

2.3 Decoder

Data Analysis We need to decide which previous sub trees to consider when parsing the current utterance. In order to make a data-driven decision we calculated the edit tree distance (Zhang and Shasha, 1989) between each pair of sub-trees of the same height belonging to different turns within the same interaction. The results after grouping by sub-tree height and distance between turns, and then averaging within each group are shown in figure 1. It is clear that the latest turn (turn distance = 1) is always closest in meaning. This result agrees with common intuition about interactions. Therefore all decoders in this work use information from the latest turn. Another expected insight is that short trees are closer to each other than tall trees (the distance is not normalized). The effect of tree height is further explored in the section 4.2.1.

Sub-trees height \ Distance between turns within the same interaction	1	2	3	4	5
0	0.90000	0.90000	0.90000	0.90000	0.90000
1	2.00000	2.10000	2.20000	2.20000	2.30000
2	3.60000	3.90000	4.10000	4.30000	4.20000
3	5.60000	6.40000	7.00000	8.00000	8.90000
4	7.50000	9.00000	10.00000	10.30000	nan
5	8.80000	10.70000	12.40000	nan	nan
6	10.20000	13.00000	16.50000	nan	nan
7	13.30000	17.30000	27.60000	nan	nan
8	15.30000	19.80000	nan	nan	nan

Figure 1: Average tree edit distance between sub-trees of same height belonging to different turns within the same interaction in SParC train set. nan values are due to the fact that program complexity increases as the interaction progresses.

Copying Previous Leaves (aka copy-leaf) This variation enriches SmBoP's vanilla decoder in each turn by allowing access to gold and predicted leaves (which can be either DB constants or spans from the utterance that represent DB values) from the latest turn. The term "copying" implies that no change is performed over leaves from the past. This method can be useful for example in theme-entity thematic relation where the current question asks for other properties about the same entity as a previous question (e.g. Prev_Q: What is the capacity of X?; Curr_Q: List all of the amenities which X has).
In detail, during training, gold leaves from the previous turn were teacher forced in the same manner as current gold leaves. During evaluation, the leaves of the best scoring tree predicted by the model in the previous turn, were given the best possible score.

Copying Previous Sub-trees (aka copy-tree) This variation resembles previous variation and extends it by copying entire gold or predicted sub-trees from the latest turn.

Contextualizing with Previous Sub-trees (aka contextualize-prev) The Copying mechanism has a coverage problem - it isn't able to properly reuse a previous sub-tree with a required change in a specific leaf. This is necessary when encountering refinement thematic relation where

the current question asks for the same type of entity as a previous question with a different constraint (e.g. Prev_Q: Which major has the fewest students? ; Curr_Q: What is the most popular one?). In those cases blindly copying previous sub-trees is not enough and a softer modeling can be more beneficial. This variation offers alternative softer modeling. Here, by contextualizing each beam candidate over all beam candidates of the same height from the latest turn, the model enriches its representation. This mechanism is inspired by SmBoP's contextualization technique. At the beginning of every decoding step t we use cross multi-head attention where the queries are the current tree representations (Z_t) and the keys and values are previous tree representations (Z_{t-1}). This enriched representation (Z_t^{prev}) is later further enriched with the current utterance ($\mathbf{x}$) to yield contextualized representations ($Z_t^{'}$) as part of SmBoP algorithm. The original beam representations (Z_t) remain unchanged and used later as part of SmBoP algorithm for scoring and representing the next beam (Z_{t+1}). So instead of SmBoP's contextualization:

1. $Z_t^{'} \leftarrow Attention(Z_t, \mathbf{x}, \mathbf{x})$

Now we have:

1. $Z_t^{prev} \leftarrow Attention(Z_t, Z_{t-1}, Z_{t-1})$
2. $Z_t^{'} \leftarrow Attention(Z_t^{prev}, \mathbf{x}, \mathbf{x})$

Contextualizing with Previous Sub-trees Augmented with the Current Utterance (aka contextualize-prev + augment) This variation introduces an improvement by first assessing the relevance of previous tree representations (Z_{t-1}) to the current turn by contextualizing them with the current utterance ($\mathbf{x}$). So now we have:

1. $Z_{t-1}^{next} \leftarrow Attention(Z_{t-1}, \mathbf{x}, \mathbf{x})$
2. $Z_t^{prev} \leftarrow Attention(Z_t, Z_{t-1}^{next}, Z_{t-1}^{next})$
3. $Z_t^{'} \leftarrow Attention(Z_t^{prev}, \mathbf{x}, \mathbf{x})$

3 Experiment Setup

Datasets The ablation presented in section 4.2.3 uses the Spider dataset (Yu et al., 2018). All other experiments are conducted using the SParC dataset (Yu et al., 2019).

Evaluation Metrics The main metric we used is dev set question match (QM), the exact set matching score over all questions as defined in Yu et al. (2019). The exact set matching score is 1 for each question only if all predicted SQL clauses are correct. In order to debug models in ablation 4.2.3, we also report SmBoP's specific metrics such as leaf accuracy, which is 1 for each question only if all gold leaves are in the predicted beam. Similarly, final beam accuracy is 1 for each question only if the gold tree is present in the final predicted beam.

Hyperparameters All models were trained using SmBoP's standard hyperparameters, except for changing the maximum sequence length as stated in 2.2.

Additional Models For reference we also report the results of the SCoRe model (Yu et al., 2021) which is the current state-of-the-art on SParC.

4 Results

4.1 Main Results

Table 1 presents the QM scores for main modeling approaches introduced in this work .The experiments in the upper two rows were conducted using the vanilla SmBoP decoder. As stated above, all specialized decoders were built upon the concat-encoder (sec 2.2). Therefore, the most significant contribution to performance is due the modification of the encoder. We notice that augmenting the previous sub-trees with the current utterance helps contextualization and achieves the best result among all methods investigated in this work (4.4% better than baseline). This can be somewhat surprising because while the copying mechanism addresses only gold previous sub-trees of limited height, the contextualization mechanism addresses also falsely-predicted previous sub-trees of all heights. Therefore, we infer that contextualization is more resilient to noise. The fact that copying leaves scores higher than copying trees is surprising and further investigated in ablation 4.2.1.

4.2 Ablations

4.2.1 Limiting Maximum Height for Tree Copying

Table 2 shows that copying shorter sub-tees is more beneficial. As the value for the maximum allowed height for copying is increased, performance deteriorates and there is a catastrophic decrease when approaching the maximum possible height. This can be explained by the rigid nature of tree copying which doesn't account for necessary alterations to previous sub-trees. In addition figure 1 shows that

Model	dev QM
baseline	40.2
concat-encoder	43.4
copy-leaf	44.3
copy-tree*	43.8
contextualize-prev	44.0
contextualize-prev + augment	**44.6**
SCoRe (SOTA)	**66.2**

Table 1: Dev set QM scores for main modeling approaches introduced in this work. **limited maximum height for tree copying to 2 according to ablation 4.2.1

the absolute tree edit distance increases dramatically at higher trees. As a result, optimization is subject to a great deal of noise when copying high sub-trees. Copying causes the model to give high scores to false options and therefore training loss can't reach 0.

Max Height for Tree Copying	dev QM
0 (copy-leaf)	**44.3**
2	43.5
5	43.1
8	25.9

Table 2: Dev set QM scores for different choices of maximum height for tree copying

4.2.2 Copying Only in Evaluation

Table 3 shows the QM scores of each copy model (copy-leaf and copy-tree at all heights) when performing copying both during training and evaluation versus only during evaluation. This comparison enables us to isolate the noisy optimization problem discussed in 4.2.1 because when training is performed without copying, optimization should not be affected (ignoring the effect of different learning rate scheduling). In the case of copying leaves it is better to perform copying also during training. It is probably because of lower train-test discrepancy and overall lower noise effect when dealing with leaves due to their low absolute edit distance. When copying all sub trees indeed we see the harming effect of noisy optimization.

4.2.3 Autoregressive Attention for Efficient Encoding

As stated in section 2.2.1, performing a single forward pass of the encoder per interaction requires preventing positions corresponding to schema tokens from attending to positions corresponding to

Model	Copy in training	dev QM
copy-leaf	X	43.2
copy-leaf	V	**44.3**
copy-tree	X	**27.4**
copy-tree	V	25.9

Table 3: Dev set QM scores for different choices whether to perform copying during training.

utterance tokens, across all encoder layers. In order to test the feasibility of this approach, we implemented the schema-to-utterance attention mask over the original SmBoP trained on Spider dataset. The implementation uses a block-wise triangular relative positions matrix. Table 4 shows that preventing the schema from attending to the utterance catastrophically hurts generalization. However, training performance is unaffected. A possible explanation comes from the way that the model initializes its beam. The model uses the representations of schema tokens to score and decide which schema constants are relevant to the current utterance and should be used as leaves. Therefore, masking the schema-to-utterance attention probably hurts leaves accuracy. The leaf accuracy column supports this explanation. Due to bottom up teacher forcing training accuracy shouldn't be affected by poor leaves scoring. However, in evaluation an error when choosing leaves is carried forward all the way up the parsing process, which greatly hurts generalization.

schema-utterance attention masking	dev QM	training final beam accuracy	validation leaf accuracy
X	**67.1**	97.2	94.6
V	3.9	97.1	35.5

Table 4: Metrics for different choices whether to mask the schema-to-utterance attention.

Acknowledgments

I thank Prof. Jonathan Berant and Ohad Rubin for their valuable guidance and feedback.

References

N. Bertomeu, H. Uszkoreit, A. Frank, Hans-Ulrich Krieger, and B. Jörg. 2006. Contextual phenomena and thematic relations in database qa dialogues:

results from a wizard-of-oz experiment. In *HLT-NAACL 2006*.

Ohad Rubin and Jonathan Berant. 2021. Smbop: Semi-autoregressive bottom-up semantic parsing. In *NAACL*.

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2021. Score: Pre-training for context representation in conversational semantic parsing. In *ICLR*.

Tao Yu, Rui Zhang, Kai-Chou Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Z Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *EMNLP*.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Y. Tan, Xi Victoria Lin, Suyi Li, H. Er, Irene Z Li, B. Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, V. Zhang, Caiming Xiong, R. Socher, and Dragomir R. Radev. 2019. Sparc: Cross-domain semantic parsing in context. In *ACL*.

K. Zhang and D. Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18:1245–1262.

Rui Zhang, Tao Yu, H. Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, R. Socher, and Dragomir R. Radev. 2019. Editing-based sql query generation for cross-domain context-dependent questions. *ArXiv*, abs/1909.00786.