A Pointer Network Architecture for Context-Dependent Semantic Parsing

Xuanli He[♣] **Quan Hung Tran**[♡] **Gholamreza Haffari**[♣]

[♣]Monash University, Australia
[♡]Adobe Research, San Jose, CA
{xuanli.he1, gholamreza.haffari}@monash.edu
qtran@adobe.com

Abstract

Semantic parsing targets at mapping human utterances into structured meaning representations, such as logical forms, programming snippets, SQL queries etc. In this work, we focus on logical form generation, which is extracted from an automated email assistant system. Since this task is dialogue-oriented, information across utterances must be well handled. Furthermore, certain inputs from users are used as arguments for the logical form, which requires a parser to distinguish the functional words and content words. Hence, an intelligent parser should be able to switch between generation mode and copy mode. In order to address the aforementioned issues, we equip the vanilla seq2seq model with a pointer network and a context-dependent architecture to generate more accurate logical forms. Our model achieves state-of-the-art performance on the email assistant task.

1 Introduction

Recently, due to the breakthrough of the deep learning, numerous and various tasks within the filed of natural language processing (NLP) have made impressive achievements (Vaswani et al., 2017; Devlin et al., 2018; Edunov et al., 2018). However, most these achievements are assessed by automatic metrics, which are relatively superficial and brittle, and can be easily tricked (Paulus et al., 2017; Jia and Liang, 2017; Läubli et al., 2018). Hence, understanding the underlying meaning of natural language sentences is crucial to NLP tasks.

As an appealing direction in natural language understanding, semantic parsing has been widely studied in the NLP community (Ling et al., 2016; Dong and Lapata, 2016; Jia and Liang, 2017). Semantic parsing aims at converting human utterances to machine executable representations. Most existing work focuses on parsing individual utterances independently, even they have an access to the contextual information. In spite of several pioneering efforts (Zettlemoyer and Collins, 2009; Srivastava et al., 2017), these pre-neural models suffer from complicated hand-crafted feature engineering, compared to their neural counterparts (Dong and Lapata, 2018; Rabinovich et al., 2017). One notable exception is the work of Suhr et al. (2018), who incorporate context into ATIS data with a neural approach.

In this work, we propose a neural semantic parser for email assistant task which incorporates the conversation context as well as a copy mechanism to fill-in the arguments of the logical forms from the input sentence. Our model achieves state-of-the-art (SOTA) performance. We further provide details analysis about where these improvements come from.

2 Models

To build our models, we follow a process of error-driven design. We first start with a simple seq2seq model, then we closely examine the errors, group them, and then propose a solution to each of these error groups. From our examination, we identify two main sources of errors of a seq2seq model: i) the overly strong influence of the language model component, and ii) the lack of contextual information. Thus we design our model to incorporate the *Pointer Mechanism* and *Context-dependent Mechanism* to solve these problems. From this point, we refer to the errors caused by the first source (language model) as *Copy-related errors*, and the ones caused by the second source (lack of context) as *Context-related errors*.

2.1 Word Copy using the Pointer Mechanism

With the basic seq2seq architecture, the model's generation is heavily influenced by the language

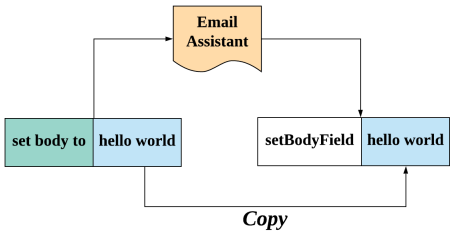

Figure 1: A example of semantic parsing on the email assistant system.

model aspect. Thus, it tends to use the strings it has seen in the training dataset (see Table 1).

current utterance:
set body to blue
logical form
reference: (setFieldFromString (getProbMutableFieldByFieldName body) (stringValue " blue "))
seq2seq: (setFieldFromString (getProbMutableFieldByFieldName body) (stringValue " charlie is on his way "))

Table 1: An error made by the base seq2seq model. Copy mechanism can fix it.

From this analysis, we realize that it would be crucial for the model to learn when to copy from the source sentence, and when to generate a new token. Thus, we incorporate the pointer mechanism into our base seq2seq approach.

As shown in Figure 1, for an email assistant system, users inputs are usually comprised of a functional part and a content part. A semantic parser should be able to distinguish and handle them in a different way. Specifically, the parser must generate a series of lambda-like functions for the functional part, while the content part should be copied to the argument slot.

Our pointer network is inspired by that of See et al. (2017) designed for the summarisation task. Given an utterance $\mathbf{x}$ and a logical form $\mathbf{y}$, at each time step t, we have a *soft* switch which determines the contributions of the token generator and the copier which uses a pointer over the words of the input utterance:

$$P(\mathbf{y}_t) = p_{gen} P_{vocab}(\mathbf{y}_t) + (1 - p_{gen}) \sum_{i:\mathbf{x}_i = \mathbf{y}_t} \alpha_i^t$$

where α_i^t is the attention score over the position i in the t-th generation step, and P_{vocab} is a probability distribution over the vocabulary. $p_{gen} \in [0, 1]$ is the *generation probability*, modelled as:

$$p_{gen} = \sigma(\mathbf{w}_c^T \mathbf{c}_t + \mathbf{w}_s^T \mathbf{s}_t + \mathbf{w}_x^T \mathbf{x}_t + b)$$

where $\mathbf{c}_t$ and $\mathbf{s}_t$ are the context vector and the decoder state respectively, while $\mathbf{w}_c^T$, $\mathbf{w}_s^T$, $\mathbf{w}_x^T$ and b are learnable parameters.

2.2 Conditioning on Conversation Context

Understanding conversations between a user and the system requires the comprehension of the flow of the discourse among sequence of utterances. Processing utterances independently within a conversation leads to misinterpreting users inputs, which will result in incorrect logical form generation (see Table 2). Therefore, we incorporate the context when processing the current utterance for a better generation.

dialog history
...
user: compose a new email. the recipient is mom. the subject is hello
user: cancel
...
current utterance:
cancel
logical form
reference: (undo)
seq2seq: (cancel)

Table 2: An error made by the base seq2seq model. It is clear that without the context information, the model cannot infer the correct logical form.

Basically, a conversation consists of a sequence of user utterances:$\{\mathbf{x}^1, ..., \mathbf{x}^T\}$ paired with a list of logical forms: $\{\mathbf{y}^1, ..., \mathbf{y}^T\}$. For a given utterance sequence $\mathbf{x}^i = \{\mathbf{x}_1^i, ..., \mathbf{x}_m^i\}$, a semantic parser should predict its associated logical form $\mathbf{y}^i = \{\mathbf{y}_1^i, ..., \mathbf{y}_n^i\}$. Inspired by Suhr et al. (2018), we introduce a hierarchical architecture to model both utterance-level and conversation-level information; see Figure 2. At the utterance level, we use an attentional seq2seq model to establish the mapping from an utterance $\mathbf{x}^i$ to its corresponding logical form $\mathbf{y}^i$:

$$\mathbf{h}_{1:m}^i = \text{Encoder}(\mathbf{x}_1^i, ..., \mathbf{x}_m^i), \quad (1)$$

$$\mathbf{c}_t^i = \text{Attention}(\mathbf{h}_{1:m}^i, \mathbf{s}_{t-1}^i), \quad (2)$$

$$\mathbf{y}_t^i, \mathbf{s}_t^i = \text{Decoder}(\mathbf{y}_{t-1}^i, \mathbf{s}_{t-1}^i, \mathbf{c}_t^i) \quad (3)$$

As the seq2seq model, we investigate the use of RNN-based and Transformer-based architectures. Furthermore, we make use of a conversation-level RNN to capture the wider conversational context:

$$\mathbf{g}_i = \text{RNN}(\mathbf{h}_m^i, \mathbf{g}_{i-1}) \quad (4)$$

where $\mathbf{h}_m^i$ is the last hidden state of the ith utterance, and $\mathbf{g}$ is the conversational hidden state.

In order to incorporate the conversational information into our model, we modify the Equ. 1 by injecting $\mathbf{g}_{i-1}$:

$$\mathbf{h}^i_{1:m} = \text{Encoder}([\mathbf{x}^i_1 : \mathbf{g}_{i-1}], ..., [\mathbf{x}^i_m : \mathbf{g}_{i-1}])$$

where $[:]$ denotes a concatenation operation.

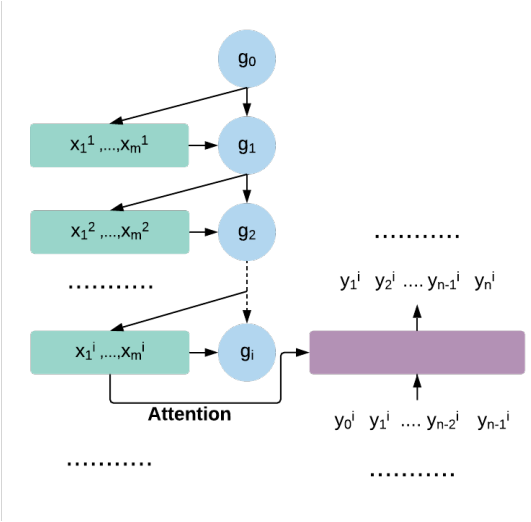

Figure 2: Overall architecture of our semantic parser. We omit the pointer network due to lack of space.

Similar to memory networks (Sukhbaatar et al., 2015), it is essential to give the decoder a direct access to the last k utterances, if we want to leverage the discourse information effectively. Hence, we concatenate the previous k utterance $\{\mathbf{x}_{i-k}, .., \mathbf{x}_{i-1}\}$ with the current utterance. Now Equ. 2 is rewritten as:

$$\mathbf{c}^i_t = \text{Attention}(\mathbf{h}^{i-k}_{1:m}, .., \mathbf{h}^{i-1}_{1:m}, \mathbf{h}^i_{1:m}, \mathbf{s}^i_{t-1})$$

In addition, since the importance of the concatenated utterances is different, it is significant to differentiate these utterances to reduce confusion. Therefore, as suggested by Suhr et al. (2018), we add relative position embeddings $\text{E}_{pos}[\cdot]$ to the utterances when we compute attention scores. Depending on their distances from the current utterance, we append $\text{E}_{pos}[0], .., \text{E}_{pos}[k]$ to the previous utterances respectively.

3 Experiments

Dataset Semantic paring is crucial to dialogue systems, especially for multi-turn conversations. Additionally, understanding users' intentions and extracting salient requirements play an important role in the dialogue-related semantic parsing. We use a dataset created by Srivastava et al. (2017) as a case study to explore the performance of semantic parsing in dialogue systems. This dataset is collected from an email assistant, which can help users to manage their emails. As shown in Table 3 Users can type some human sentences from the interface. Then the email assistant can automatically convert the natural sentences to the machine-understandable logical forms.

dialog history
...
user: Define the concept " contact "
user: add field " email " to concept " contact "
user: create contact " Mom "
...
logical form
...
(defineConcept (stringNoun " contact "))
(addFieldToConcept contact (stringNoun " email "))
(createInstanceByFullNames contact (stringNoun " mom "))
...

Table 3: A partial conversation from the data

Following Srivastava et al. (2017), we partition the dataset into a training fold (93 conversations) and a test fold (20 conversations) as well. However, this partition might be different from Srivastava et al. (2017), as they only release the raw Email Assistant dataset. The total number of user utterances is 4759, the number of sessions is 113, and the mean/max of the number of utterances per interactive session is 42/273.

3.1 Main Results

Prior to this work, Srivastava et al. (2017) also incorporate the conversational context into a CCG parser (Zettlemoyer and Collins, 2007). CCG requires extensive hand-feature engineering to construct text-based features. However, neural semantic parsers have been demonstrating impressive improvement over various and numerous dataset (Suhr et al., 2018; Dong and Lapata, 2018). Hence, we explore both RNN-based (Bahdanau et al., 2014) and transformer-based (Vaswani et al., 2017) architectures for our attentional seq2seq model, denoted as *RNNS2S* and *Transformer* respectively. Hyperparameters, architecture details, and other experimental choices are detailed in the supplementary material. Unless otherwise mentioned, we use 3 previous utterances as the history. Since there is no validation set, we use 10-fold cross validation over the training set to find the best parameters.

Table 4 demonstrates the accuracy of different models. Our RNNS2S baseline already surpasses the previous SOTA result with a large margin. However, since we use our own partition, this comparison should not be as a reference. Both pointer network and conversational architecture dramatically advance the accuracy. Finally, our transformer model combining these two techniques obtains a new SOTA result.

	Accuracy
Previous methods	
Seq2seq (Srivastava et al., 2017)	52.3
SPCon (Srivastava et al., 2017)	62.3
Our models	
RNNS2S	68.0
RNNS2S + pointer	69.3
RNNS2S + context	69.8
RNNS2S + context + pointer	70.5
Transformer	69.3
Transformer + pointer	72.2
Transformer + context	71.0
Transformer + context + pointer	**73.4**

Table 4: Test accuracy on Email Assistant dataset. **Bold** indicates the best result. SPCon is the best CCG parser with contextual information in Srivastava et al. (2017)

3.2 Analysis

In this section we provide some deep analysis on our models. Since we see the same trend in both RNNS2S and Transformer, we only report the analysis of RNNS2S. The supplementary material reports the analysis of Transformer.

The effects of the copy mechanism We analyze the test data, and count the number of errors that can be rectified by introducing the pointer network for both vanilla and context-dependent seq2seq models. In the test set, we identify that a total of 37 errors made by the seq2seq model and 36 errors made by the seq2seq+context model can be rectified by the copy mechanism. According to Figure 3, our pointer network fixes at least half of the incorrect instances. Clearly, the pointer mechanism cannot solve all copy-related errors. After scrutinizing the system-generated results, we realize that the pointer network tends to retain the copy mode once it is triggered. This phenomenon is consistent with the observations by See et al. (2017). Consequently, the extra copies impinge on the accuracy of the system.

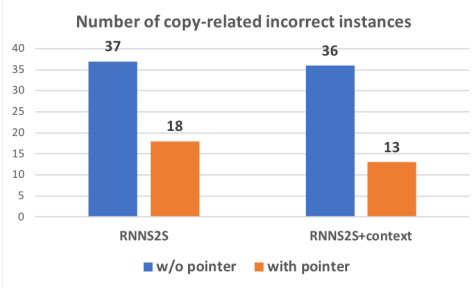

Figure 3: Number of copy-related incorrect instances that can be corrected by a pointer network.

The effects of the context-dependent mechanism. In the experiments, our context-dependent mechanism is shown to be able to address context-related errors, especially when user's input implies a complex and compositional command. These complex commands usually involve a series of complicated actions, as shown in Table 5. According to Table 6, our context-dependent model rectifies 27 out of 68 context-related errors.

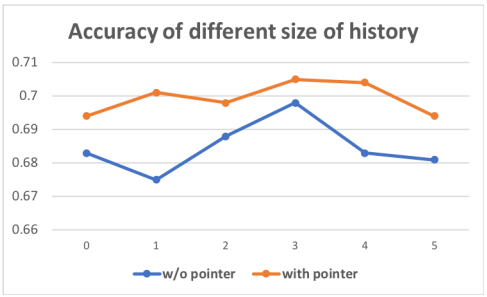

Figure 4: Accuracy of different size of history.

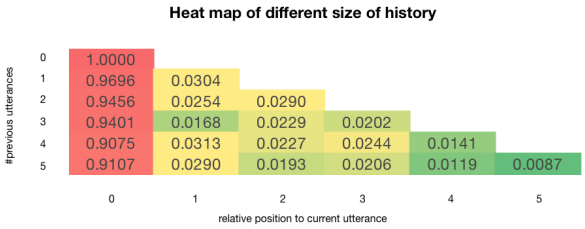

Figure 5: Heat map of different size of history.

Since we notice that previous utterances can also obfuscate the model, we conduct an ablation study over the size of history. As shown in Figure 4, incorporating 3 previous utterances reach the best performance. According to Figure 5, we believe that incorporating 3 previous utterances covers sufficient contextual information. Less than this number, the system cannot better utilize context, while the salient information is contaminated by the extra history. The same behavior is observed in Transformer model. We argue that the size of the effective history would be dependent

utterance: Set recipient to Mom's email . Set subject to hello and send the email
logical form: (doSeq (setFieldFromFieldVal (getProbMutableFieldByFieldName body) (evalField (getProbFieldByInstanceNameAndFieldName inbox body))) (doSeq (setFieldFromFieldVal (getProbMutableFieldByFieldName recipient_list) (evalField (getProbFieldByInstanceNameAndFieldName inbox sender))) (send email)))

Table 5: An example of complex and compositional commands.

	#incorrect
complex command	
RNNS2S	39
RNNS2S + context	20
context dependency	
RNNS2S	29
RNNS2S + context	21

Table 6: Incorrect instances of RNNS2S and context-dependent RNNS2S models in terms of complex commands and context dependency.

on different datasets, but they will demonstrate the same trend.

4 Conclusions

In this work, we explore a neural semantic parser architecture that incorporates conversational context and copy mechanism. These modelling improvements are solidly grounded by our analysis, and they significantly boost the performance of the base model. As a result, our best architecture establish a new state-of-the-art on the Email Assistant dataset. In the future, we would explore other architectural innovations for the system, for example, the neural denoising mechanisms.

5 Acknowledgement

We would like to thank three anonymous reviewers for their valuable comments and suggestions. This work was supported by the Multimodal Australian ScienceS Imaging and Visualisation Environment (MASSIVE).[1] This work is partly supported by the ARC Future Fellowship FT190100039 to G.H.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*.

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

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.

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

Samuel Läubli, Rico Sennrich, and Martin Volk. 2018. Has machine translation achieved human parity? a case for document-level evaluation. *arXiv preprint arXiv:1808.07048*.

Wang Ling, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, Andrew Senior, Fumin Wang, and Phil Blunsom. 2016. Latent predictor networks for code generation. *arXiv preprint arXiv:1603.06744*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Maxim Rabinovich, Mitchell Stern, and Dan Klein. 2017. Abstract syntax networks for code generation and semantic parsing. *arXiv preprint arXiv:1704.07535*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Shashank Srivastava, Amos Azaria, and Tom M Mitchell. 2017. Parsing natural language conversations using contextual cues. In *IJCAI*, pages 4089–4095.

Alane Suhr, Srinivasan Iyer, and Yoav Artzi. 2018. Learning to map context-dependent sentences to executable formal queries. *arXiv preprint arXiv:1804.06868*.

[1] https://www.massive.org.au/

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Luke Zettlemoyer and Michael Collins. 2007. Online learning of relaxed ccg grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687.

Luke S Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 976–984. Association for Computational Linguistics.

A Implementation Details

In RNNS2S model, at the utterance level, a one-layer bidiretional RNNs for the encoder, while the decoder is a two-layer RNNs. We use a one-layer RNNs to represent the conversational information flow. All RNNs use LSTM cells, with a hidden size of 128. The sizes of word embeddings and position embeddings are 128 and 50 respectively. We train our models for 10 epochs by Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.001. The batch size of non-context training is 16, while the context variant is 1.

For Transformer model, we use 3 identical transformer blocks for both encoder and decoder. Within each block, the size of the embeddings is 256, while the feed forward network has 512 neurons. We set the size of heads to 4. The conversational encoder is a one-layer RNNs with the size of 256. The optimizer and training schedule is same as Vaswani et al. (2017), except $warmup\_steps$ = 500. Due to the warmup steps, We train this model for 14 epochs. The batch size is same as that of RNNS2S.

B Analysis of Transformer

The effects of the pointer mechanism According to Figure 6, half of the incorrect instances are fixed by the pointer mechanism.

	#incorrect
complex command	
Transformer	35
Transformer + context	24
context dependency	
Transformer	25
Transformer + context	16

Table 7: Incorrect instances of Transformer and context-dependent Transformer models in terms of complex commands and context dependency.

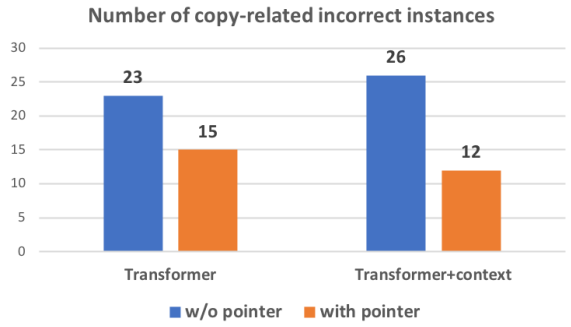

Figure 6: Number of copy-related incorrect instances that can be corrected by a pointer network.

The effects of the context-dependent mechanism. Similarly, incorporating the contextual information is able to address the context-oriented issues by a larger margin (see Table 7).

Finally, as observed in the main paper, having an access to the 3 previous utterances achieves the best performance.

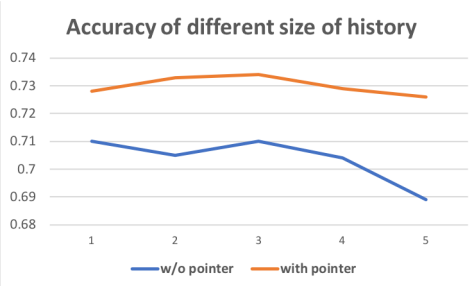

Figure 7: Accuracy of different size of history.