Graph-Based Transformer with Cross-Candidate Verification for Semantic Parsing

Bo Shao,[*1,2] **Yeyun Gong,**[2] **Weizhen Qi,**[2,3] **Guihong Cao,**[4] **Jianshu Ji,**[4] **Xiaola Lin**[1]
[1]Sun Yat-sen University
[2]Microsoft Research Asia
[3]University of Science and Technology of China
[4]Microsoft AI and Research, Redmond WA, USA
shaobo2@mail2.sysu.edu.cn, {yegong, gucao, jianshuj}@microsoft.com,
weizhen@mail.ustc.edu.cn, linxl@mail.sysu.edu.cn

Abstract

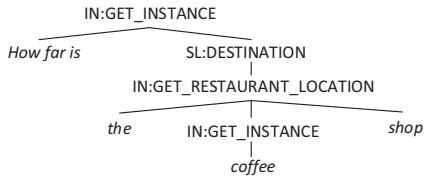
In this paper, we present a graph-based Transformer for semantic parsing. We separate the semantic parsing task into two steps: 1) Use a sequence-to-sequence model to generate the logical form candidates. 2) Design a graph-based Transformer to rerank the candidates. To handle the structure of logical forms, we incorporate graph information to Transformer, and design a cross-candidate verification mechanism to consider all the candidates in the ranking process. Furthermore, we integrate BERT into our model and jointly train the graph-based Transformer and BERT. We conduct experiments on 3 semantic parsing benchmarks, ATIS, JOBS and Task Oriented semantic Parsing dataset (TOP). Experiments show that our graph-based reranking model achieves results comparable to state-of-the-art models on the ATIS and JOBS datasets. And on the TOP dataset, our model achieves a new state-of-the-art result.

Introduction

Semantic parsing is a classic NLP task that has attracted a huge amount of attention recently. It aims at mapping a natural language sentence into a logical form. With the rapid development of natural language processing, semantic parsing has been used in various applications, such as question answering (Kwiatkowski et al. 2011), task-oriented dialog systems (Yih et al. 2015) and interpreting instructions (Artzi and Zettlemoyer 2013).

Recently, many approaches based on the sequence-to-sequence(S2S) model have been successfully used for semantic parsing (Dong and Lapata 2016; Yin and Neubig 2017; Dong and Lapata 2018). In the analysis of (Gupta et al. 2018), the accuracy of exact match on TOP dataset for RNN based sequence-to-sequence model is 75.3%, but beam search with the size of 5 can cover 88.76% golden target sequences. Since the exact match result is computed by the top1 prediction from beam search, it means that the candidates from the beam search have a greater chance of containing the golden target.

To select the correct target from these candidates, it is necessary to use a ranking model to rerank these candidates. There exist many applications that use sequence-to-sequence with reranking methods (Wen et al. 2015; Einolghozati et al. 2019). (Wen et al. 2015) uses recurrent generation model with a convolutional ranker for a dialogue system. In (Einolghozati et al. 2019), they propose a SVM+language model to rerank the candidates generated by generation model. These methods achieve good performance, since the ranking model can capture the global information in questions and decoding candidates simultaneously. However, the general ranking model can not capture the strong hierarchical structures in the logical forms. For example, Figure 1 shows an example of the TOP dataset. The logical form in this dataset is a task oriented representation of a dialog system. It contains a tree structure, which also can be considered as a special graph.



Question: How far is the coffee shop

Logical Form: [IN:GET_DISTANCE How far is [SL:DESTINATION [IN:GET_LOCATION [SL:CATEGORY_LOCATION the coffee shop]]]]

Figure 1: An example of Task Oriented Parsing (TOP) dataset.

In this paper, we propose a ranking model for logical form reranking based on Transformer (Vaswani et al. 2017), compared with other methods based on tree-structure, our graph-based transformer is more scalable, which can handle different semantic graphs in an efficient way. To capture the tree structure information of logical forms, we extend transformer with a Graph-based attention strategy. We incorporate the structure information of the logical forms to multi-head attention in the transformer and generate graph-aware interaction features. This method makes each token

[*]Work done while this author was visiting Microsoft Research.

interact with parent nodes in the semantic tree. Furthermore, to incorporate cross-candidate information, we enable each logical form candidate to interact with the other candidates based on their representations, which is inspired by V-NET (Wang et al. 2018). V-NET is applied on multi-passage reading comprehension dataset. During training, each sample in their task contains several extracted candidate answer spans which are extracted from the given passage. To better select the correct one, V-NET computes the context vector using the selected answers by the boundary model to help the ranking score computation. It is a novel way to aggregate information from all candidates in the ranking model. In our model, different from V-NET, using other answer candidates as a context, we use the similarity score between the current candidate and all other candidates through an interaction strategy to help the ranking score computation.

The main contributions of this paper are:

- We extend the transformer model with a graph-based matching strategy to enable it to capture the structure information of logical forms.
- We design a cross-candidate verification method using the similarity between each candidate and the other candidates to boost the ranking performance.
- We conduct experiments on 3 semantic parsing datasets, TOP, ATIS and JOBS. Our reranking model with a basic sequence-to-sequence generation model achieves a new state-of-the-art result on the TOP dataset and comparable results to the state-of-the-art models on the ATIS and JOBS datasets.

Model

Sequence-to-sequence based models have been successfully applied to semantic parsing. They are trained by annotated <question, logical form> pairs without human designed templates and features. In recent semantic parsing methods (Dong and Lapata 2018), researchers decode the candidate logical forms using beam search method and select top 1 candidate as the decoding result.

These methods have achieved good performance in various semantic parsing tasks. However, we find that beam search can recall most correct logical forms, but some of them can not be ranked in the first place. In this work, we use the reranking based semantic parsing method which first generates candidates using the generation model, and then rerank the candidates by a ranking model.

In the following sections, we first briefly introduce the generation model we used in this work and then we introduce our model for reranking in detail.

Generation Model

The generation model in this paper is based on the general encoder-decoder framework, using a bidirectional RNN-GRU (Cho et al. 2014) layer as an encoder and a unidirectional RNN-GRU layer as a decoder, which has been applied in semantic parsing tasks in (Dong and Lapata 2016; Rabinovich, Stern, and Klein 2017; Dong and Lapata 2018). With the input question of $Q = [x_1, x_2...x_{|Q|}], x_i \in V_q$ the model decodes the target logical form $L = [l_1, l_2, ...l_{|L|}], l_i \in V_l$, where V_q and V_l are the source and target vocabularies. Our baseline model aims to estimate $P(L|Q)$, and the conditional probability can be formulated as follows:

$$p(L|Q) = \prod_{t=1}^{|L|} p(l_t|l_{t-1}, ..., l_1, Q) \tag{1}$$

We will not introduce the implementation of the sequence-to-sequence model in detail because of its popularity these years. Generally, the encoder uses embeddings of each token in question Q as input, and generates hidden state r_i at the ith position of Q. To decode the target sequence with the hidden state r_i, attention mechanism(Bahdanau, Cho, and Bengio 2014) is used in the decoder:

$$\begin{aligned} e_i^t &= u^T \tanh(W_e[r_i; s^t] + b_e) \\ a_i^t &= \frac{e_i^t}{\sum_{j=1}^{|Q|} e_j^t}; c^t = \sum_{i=1}^{|Q|} a_i^t r_i \end{aligned} \tag{2}$$

where r_i is the encoder hidden state and s^t is the decoder hidden state, c^t is the context vector at time stamp t, u^T, W_e b_e are trainable parameters.

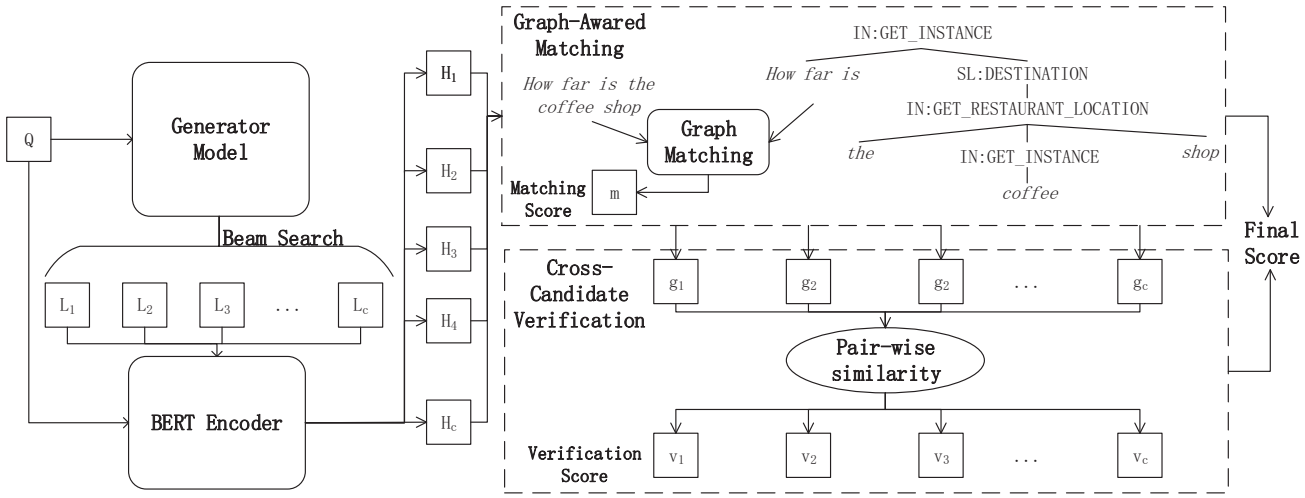
Specifically, to solve the out-of-vocabulary (OOV) words in the target dictionary V_l, we leverage the copy mechanism in (See, Liu, and Manning 2017). We use the attention distribution a_i^t as the copy probability from the source sequence and combine it with the distribution of generating probability in the target dictionary by a gate mechanism.

Reranking Model

In this section, we introduce our Graph-based Transformer with Cross-candidate Verification (GTCV) model. To extract better sentence representations, we integrate BERT to our ranking model with a graph-aware matching strategy and cross-candidate verification.

BERT Encoder We will first briefly introduce the BERT encoder in our model. Given a list of target logical form candidates $LC = [L_1, L_2, ..., L_c]$ from generation model,where L_i is a candidate, and their labels $Y = [y_1, y_2, ..., y_c], y_i \in \{0, 1\}$ indicating whether the candidate is correct, where c is the number of all candidates, and the input question Q. We then make the question and each logical form with the corresponding label as an input of the our model. The tokens of all pairs are packed into a token sequence S as "[CLS] Q [SEP] L_i [SEP]", where [CLS], [SEP] are special tokens to separate the tokens of question Q and the ith logical form candidate L_i. The representation of each token is the sum of three types of embedding, WordPiece embedding (Wu et al. 2016), Position embedding indicating the position of input tokens, and Segment embedding, which is used to indicate the question segment and logical form segment.

Then, the representation of S is fed into the BERT encoder which is a multi-layer bidirectional Transformer(Vaswani et al. 2017). For the implementation of its architecture, readers can refer to (Vaswani et al. 2017; Devlin et al. 2018). BERT encodes the input token sequence S into a context aware representation H_i =



Q:How far is the coffee shop

LF:[IN:GET_DISTANCE How far is [SL:DESTINATION [IN:GET_LOCATION [SL:CATEGORY_LOCATION the coffee shop]]]]

Figure 2: Overview of our method for semantic parsing.

$\{h_0, h_1, ..., h_{|S|-1}\}$ for the ith candidate. Then H_i is used in our graph-based transformer layer.

To directly use BERT in the ranking task, we extract the final hidden state of the first token, h_0, corresponding to the special token "[CLS]". The label probabilities are computed with h_0 by a standard softmax layer. Finally, all of the parameters of the model are fine-tuned to maximize the log-probability of the correct label in the ranking task.

Graph Matching Although BERT has proven its effectiveness for text sequence matching tasks, there still exists a limitation to matching questions and logical form candidates in our task. The two sequences can only interact through a general attention mechanism in the transformer, which is not enough to capture the structure information in complex matching tasks, since it is based on transformer and only uses position embedding to encode the position information. To overcome the limitation, we leverage a hierarchical graph-aware matching method to extend BERT for our task. To better encode the structure information, we add extra n layers of graph-based transformer for graph matching, incorporated with a masked attention mechanism, which uses an adjacent matrix M as mask matrix. This mask matrix contains the structure information of the parsing tree.

For each transformer block, we first apply a multi-head self-attention on input features. We set $Z_0 = H_i$ as the input of the ith candidate, and the attention matrix in each head is computed as follows:

$$A = softmax((W^Q Z_0, W^K Z_0^T)/\sqrt{d_k}) \tag{3}$$

where W^Q and W^K are trainable parameters, d_k is the dimension of Z_0.

This basic attention matrix in the transformer is a word-to-word attention that is not suitable in this task because it makes every token attend to each other and can not incorporate structure information in matching. Thus, we propose a mask attention method based on the adjacency matrix of the parsing tree.

To generate the adjacency matrix, we first use the bracketing matching method to check the validation of candidate parsing trees. If it is an invalid parsing tree, we will discard it. For each logical form, we make the nonterminal tokens as nodes, the brackets to indicate each layer of tree, and the parameters in logical form as leaves. The procedure has been used in previous methods like SEQ2TREE (Dong and Lapata 2016). After that, we first initialize a zero matrix $M \in R^{|L|*|L|}$, then set $M_{i,j}$ to 1, which represents that l_j is the parent of the token l_i. It guarantees that each token in logical form will only interact with its corresponding parents and not with nodes in different sub-trees. After processing all tokens in question q, we will obtain the the adjacency matrix M as a mask matrix.

Then, we combine the mask adjacency matrix with the attention matrix:

$$G_{i,j} = \begin{cases} A_{i,j} M_{i-|Q|-2, j-|Q|-2} & i,j > |Q|+2 \\ A_{i,j} & other \end{cases}$$

And compute the interaction features:

$$Z_i^* = GW^Z Z_i - 1 \tag{4}$$

where W^Z are the trainable parameters and Z_i^* are the interaction features between the question in ith layer and logical form in one head of the multi-head attention mechanism. We concatenate all features from each head as Z_i. We apply the operation n times and acquire the graph aware features Z_n. We then apply the average pooling on Z_n to obtain the final graph based representation Z.

Finally, we combine the interaction feature with the first hidden state h_0 of the BERT encoder:

$$\begin{aligned} g &= [Z : h_0] \\ m &= sigmoid(W^g g + b^g) \end{aligned} \tag{5}$$

where W^g and b^g are trainable parameters, and m is the matching score of the graph matching.

Cross-candidate Verification The BERT encoder and Graph matching method focus on encoding the question and logical form better and on extracting the interaction representation. However, the candidates from beam search method are always similar, which is hard for the reranking task. Furthermore, since the input contains a list of candidates, aggregating the information across all candidates should be useful in this task.

We can observe that each subtree of the correct logical form usually appears in the candidates. If we assume that one of the candidates is the correct one, then each part of this candidates is likely to be covered by other candidates. Thus the confidence for one candidate to be the correct one can be implied by the similarity to other candidates. We propose a method to enable our model to consider all candidates and to verify the candidate through their similarity with each other. To apply this operation, we feed all candidates with the question into the model simultaneously as a list-wise input. For each candidate, we compute their representations g_i for the ith candidate by Eq. 5. g_i can be considered as the representation of each candidate containing the structure, semantic and interaction features. Each candidate computes the similarity from each other with their representations as the supportive information:

$$\begin{aligned} r_j^i &= u_d^T \tanh(W^d[g_i, g_j, g_i g_j] + b^d) \\ d_j^i &= \frac{r_j^i}{\sum_{k=1}^{c} r_k^i}; \\ v_j &= sigmoid(\sum_{k=1}^{c} d_j^k) \end{aligned} \tag{6}$$

where W^d and b^d are trainable parameters, d_j^i represents the similarity between the ith and jth candidates, and for the jth candidate, we use the sum of the normalized similarity score v_j as the final verification score.

Finally, we combine the verification score with the matching score as the final ranking score,

$$p_i = (1 - \alpha) m_i + \alpha v_i \tag{7}$$

where m_i and v_i are the matching score and verification score of the ith candidate, and $\alpha \in [0, 1]$ is used to balance the two parts. Then the loss function can be computed as follows,

$$\mathcal{L} = \sum_{i=1}^{c} -y_i log(p_i) - (1 - y_i) log(1 - p_i) \tag{8}$$

where y_i is the label of ith candidate.

Experiment

Datasets

We conduct our experiment on 3 semantic parsing datasets JOBS, ATIS and TOP.

JOBS is a dataset containing 640 queries annotated from a database of job listings. Questions are paired with Prolog-style queries. We follow the training and test split in (Zettlemoyer and Collins 2012). It contains 500 training and 140 test instances. They have tagged the values for the variables company, degree, language, platform, location, job area, and number.

ATIS is a dataset containing 5410 queries to a flight booking system (Hemphill, Godfrey, and Doddington 1990). The data has been split into 4480 training instances, 480 validation instances, and 450 test instances. Each pair contains a question with the corresponding lambda-calculus expressions and identified values for the variables of date, time, city, aircraft code, airport, airline and number.

TOP[1] is a large scale semantic parsing dataset (Gupta et al. 2018), containing 44,783 annotation question and parsing tree pairs, which are split into 31,279 for training, 4,462 for validation and 9,042 for test. The utterances in this dataset are focused on navigation, events, and navigation to events.

For the TOP dataset, (Einolghozati et al. 2019) remove the samples with intent of "UNSUPPORTED" which represents out-of-domain questions and get a subset of TOP, this dataset contains 28,276 training samples, 4,014 validation samples and 8,191 test samples. We also conduct an experiment on this dataset.

Settings

In our generation model, Glove word embeddings (Pennington, Socher, and Manning 2014) are used as our pre-trained word embeddings.Input sentences are lower-cased. The beam size of the model is 10. We set the dropout rate to 0.5. The dimension of all hidden vectors and word embedding is set to 300. Word vocabulary is not shared between encoder and decoder. Parameters are randomly initialized from a uniform distribution (-0.01, 0.01). We use Adagrad (Duchi, Hazan, and Singer 2011) as optimizer during training, and an early stop strategy is used to decide the training epoch. In our GTCV ranking model, we load the pre-trained model of BERT$_{BASE}$ with small parameters to reduce the training time. The number of Transformer blocks is 12, and the dimension of all hidden states is 768 in our model. The batch size of the model is 32. Dropout is set to 0.1. We set the block number n of our graph-based transformer to 3. The score weight α in our model is set to be 0.1. We use all the 10 candidates from beam search to train and evaluate our ranking model. We optimize our model by an Adam optimizer with an initial learning rate of $3e-4$, β_1 = 0.9, β_2 = 0.999 and γ= 10^{-9}. Gradient accumulation is used in our training, and accumulate step is set to 12.

For TOP, we use the same metrics as used in (Gupta et al. 2018), including exact match accuracy(ACC), labeled bracketing F1 score (Black et al. 1991), and their proposed

[1] http://fb.me/semanticparsingdialog

Model	ACC	F1	P	R	TL-F1	TL-P	TL-R	TV
RNNG (Dyer et al. 2016)	78.51	90.23	90.62	89.84	84.27	84.64	83.91	**100.00**
FAIRSEQ (Gehring et al. 2016)	75.87	88.56	89.25	87.88	82.31	82.92	81.72	99.75
S2S-LSTM (Wiseman and Rush 2016)	75.31	87.69	88.35	87.03	81.15	81.72	80.58	99.94
Transformer (Vaswani et al. 2017)	72.20	86.60	87.09	86.11	78.54	78.99	78.19	99.55
MatchLSTM* (Wang and Jiang 2015)	78.31	89.10	88.91	89.30	84.26	84.80	83.71	99.64
Ours								
Generation	77.40	88.51	88.80	88.23	83.80	84.17	83.44	99.67
BERT*	80.15	89.52	89.70	89.34	85.02	85.43	84.61	99.63
GTCV*	**82.51**	**90.79**	**90.87**	**90.71**	**88.01**	**88.40**	**87.62**	99.78

Table 1: Performance (in percentage) of our proposed model and the state-of-the-art methods. * represents the reranking methods for semantic parsing.

metric Tree Label(TL) and Tree Validation(TV). The first two metrics are commonly used in various semantic tasks. In particular, TL is used to evaluate the subtree structures for non-terminal tokens. TV represents the percentage of predictions that is formed valid trees via bracket matching.

For ATIS and JOBS datasets, we identify the entities and numbers in the input questions and replace them with their type names and unique IDs, which is the same as the preprocessing in (Dong and Lapata 2016).

Results and Analysis on TOP

Table 1 shows the performance of our model and the state-of-the-art methods. RNNG (Dyer et al. 2016) is a top-down transition-based parser and was originally proposed for parsing syntax trees and language modeling. FAIRSEQ (Gehring et al. 2016) and S2S-LSTM (Wiseman and Rush 2016) represent convolution based and LSTM based sequence to sequence model for natural language generation. Transformer (Vaswani et al. 2017) is a recent popular generation model based transformer blockers with multi-head attention mechanism and has been applied on different tasks.These results are reported in (Dyer et al. 2016) as compared baselines. "Generation" represents our generation model without reranking. "MatchLSTM" represents using MatchLSTM proposed in (Wang and Jiang 2015) as ranking model to rerank the candidates. "BERT" denotes using BERT as our ranking model. "GTCV" represents the model proposed in this paper.

From the results, we observe that "GTCV" achieves state-of-the-art performance among most metrics. Compared with "Generation", "GTCV" achieves about 5 points improvement over accuracy which illustrates the effectiveness of our reranking method. From the results of "MatchLSTM", "BERT" and "GTCV", we see that "GTCV" achieves significantly performance improvement which demonstrates the structure information is important for the reranking model.

To evaluate the contribution of each part, we conduct an ablation experiment on TOP. Table 2 shows the results. "w/o Verification" represents our model Without cross-candidate verification. "w/o Graph" denotes our model without graph information. Comparing the results of "GTCV" "w/o Graph" and "w/o Verification", we see that the graph information and cross candidates verification proposed in this paper are effectiveness for the performance improvement.

Methods	**ACC**	**F1**
Generation	77.40	88.51
BERT	80.15	89.52
GTCV	82.51	90.79
w/o Verification	81.16	90.02
w/o Graph Matching	81.26	90.06

Table 2: Ablation Experiment

Method	**ACC**
RNNG+Top 1	81.21
LM	82.80
SVM +LM ranker	84.26
Generation	79.82
GTCV	85.84

Table 3: In domain result.

Table 3 shows the results of our model on the subset of TOP, which represents in-domain questions, proposed in (Gupta et al. 2018). The baseline methods are proposed in (Gupta et al. 2018), "RNNG+Top1" is a generation method. "LM" and "SVM+LM ranker" are re-ranking methods. From the results of "RNNG+Top1" and "Generation", we observe that the performance of our generation model is lower than "RNNG+Top1", while "RNNG+Top1" requires manually defined grammars. Our generation model is more scalable which can be trained without any manually features. In this work, we mainly focus on the reranking model. From the results of "GTCV" and "SVM+LM ranker", we observe that even though the performance of our generation model is lower than RNNG, our model "GTCV" achieves better performance than "SVM+LM ranker".

Parameter Analysis We range the beam size among {1,2,3,5,10}, when the beam size is n, the model will generate n candidates, we compute the recall of these candidates. Figure 3 shows the recall of different beam sizes. From the results, we observe that the recall increase fast when the beam size ranges from 1 to 5, while from 5 to 10, the growth is slowed down, When the beam size equals to 10, We get

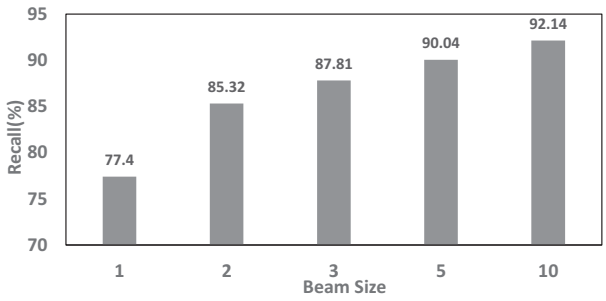


Figure 3: Recall of our baseline sequence-to-sequence model in different beam sizes.

above 92% recall. However the recall for beam size equals to 1 is 77.4%. Thus, there exists a large improvement space using reranking methods.

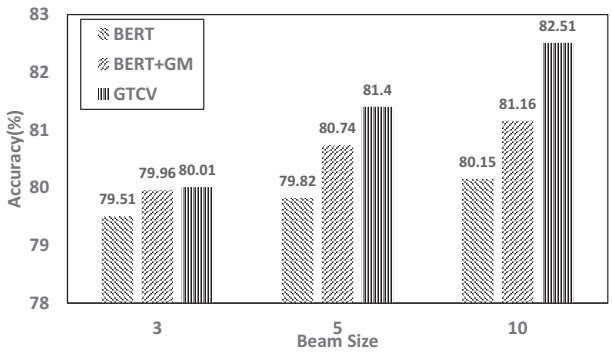


Figure 4: Accuracy of various beam sizes.

To evaluate the robustness of our method, we run our ranking model on different number of candidates. Figure 4 shows the results when the candidate number ranges in {3,5,10}. We find that our model achieves improvement for all these candidate numbers. Furthermore, when the candidate number is 3, since there are few candidates, the effectiveness of the cross candidate verification mechanism is reduced, while the graph-based matching mechanism still achieves significantly performance improvement. We also see that the effectiveness of cross candidate verification increases with the number of candidate increasing.

Case Study Figure 5 shows some cases of top 1 results ranked by GTCV and BERT. In the first case, we observe that the logical form selected by BERT has a branch "SL:CATEGORY_EVENT-the Franch market", while "the Franch market" is not a "CATEGORY_EVENT". Our model GTCV considers the structure information and filters this candidate. In the second case, we observe that the bold branches selected by BERT are mismatch. The content of the candidates selected by BERT and selected by our model are the same, the difference is the structure information. We use this case to illustrate that our model can better distinguish the candidate through structure information.

From the cases, we see that the candidates generated by the generation model are often similar except the structure,

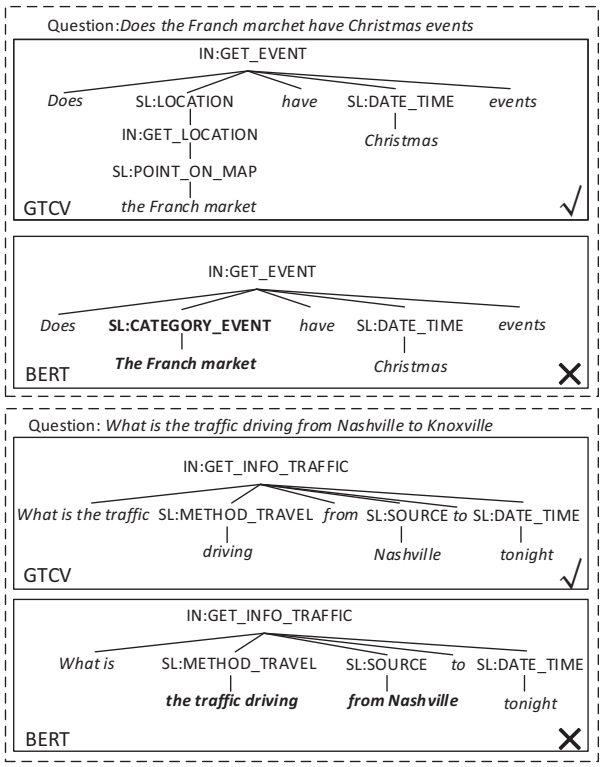


Figure 5: Comparision of Top1 ranking examples in GTCV and BERT

our graph based model captures the structure information, and reranks the candidates better.

Results on ATIS and JOBS

	JOBS	ATIS
FUBL (Kwiatkowski et al. 2011)	-	83.5
GUSP++ (Poon 2013)	-	83.5
DCS+L (Liang, Jordan, and Klein 2013)	90.7	-
TISP (Zhao and Huang 2014)	85.0	84.2
Pointer Network (See, Liu, and Manning 2017)	86.7	83.4
seq2seq (Dong and Lapata 2016)	87.1	84.2
seq2tree (Dong and Lapata 2016)	90.1	84.6
ASN (Rabinovich, Stern, and Klein 2017)	91.4	85.3
ASN+SUP (Rabinovich, Stern, and Klein 2017)	92.9	85.9
Seq2Act (Chen, Sun, and Han 2018)	-	85.5
Coarse2fine(Dong and Lapata 2018)	-	**87.7**
MatchLSTM* (Wang and Jiang 2015)	90.1	82.8
Generation	89.3	82.4
BERT*	90.1	84.8
GTCV*	**92.9**	87.3

Table 4: Accuracy on JOBS and ATIS. * represents the reranking methods for semantic parsing.

We also evaluate our method on the datasets, ATIS and JOBS in Table 4. We observe that our GTCV model achieves comparable result with the state-of-the-art methods, compared with many existing methods. The accuracy of our

GTCV model obtains absolute improvements of 3.6% and 4.9% in both datasets, compared to the generation result. Our model also outperforms MatchLSTM and BERT ranking models. Furthermore, we find that our GTCV with the basic sequence-to-sequence model achieves comparable result to state-of-the-art methods. Specially, since the two datasets are small and the size of test set in jobs only 140, thus the slight improvement on the test set is not so convinced. And the two datasets do not contain the complex structure information as TOP. And Results on different datasets demonstrate the robustness and effectiveness of our ranking model, which is only based on the basic sequence-to-sequence model without specific designed generation model. The experiments show that our approaches can be on various datasets based on general sequence to sequence model and achieve good performance.

Related Work

Semantic Parsing

Various models have been proposed over the years for semantic parsing (Kwiatkowski et al. 2011; Shao et al. 2019; Reddy, Lapata, and Steedman 2014; Artzi and Zettlemoyer 2011; Xiao, Dymetman, and Gardent 2017; Yin and Neubig 2017). (Kwiatkowski et al. 2011) propose a combinatory categorical grammar induction technique for semantic parsing. (Xiao, Dymetman, and Gardent 2017; Yin and Neubig 2017) use syntax information to improve semantic parsing models. (Reddy, Lapata, and Steedman 2014) try to build semantic parsers without relying on logical form annotations through distant supervision. With the rapid development of deep learning models. Most of these transitional methods require experts to design features to represent and rank candidates. Recently, neural semantic parsing methods (Dong and Lapata 2016; Jia and Liang 2016; Chen, Sun, and Han 2018; Dong and Lapata 2018) are proposed to train semantic parsers in an end-to-end neural framework. (Dong and Lapata 2016) proposes a SEQ2TREE model, which captures a hierarchical structure of logical forms. (Guo and Gao 2017) leverages a new attention mechanism to generate a more precise SQL. (Dong and Lapata 2018) proposes a hierarchical decoding process, from coarse sketches to fine target logical forms. On the TOP dataset, various generation model including FAIRSEQ (Gehring et al. 2016), S2S-LSTM (Wiseman and Rush 2016) and Transformer (Vaswani et al. 2017) are used and reported as the baselines in (Gupta et al. 2018).

Graph-based Neural Network

Recently, graph-based neural network has become a hot research topic. Researchers seek to capture more information in graph structure data. SDNE(Wang, Cui, and Zhu 2016) uses finding proper node embedding to reconstruct its neighborhood as part of its objective function. GCN(Kipf and Welling 2016) iteratively aggregates neighbor information from previous layers with a defined convolution operator. GAN(Veličković et al. 2017) leverages attention mechanisms to focus on the most relevant information by attending over nodes' neighborhood. Structure information also helps neural semantic parsing, SEQ2TREE (Dong and Lapata 2016) uses a hierarchical tree decoder to model the compositional nature of meaning representations in logical forms. In (Xu et al. 2018), researchers propose a graph-to-sequence model that uses the graph to represent syntactic information of word order, dependency and constituency.

Conclusion

In this paper, we propose a Graph-based Transformer model for semantic parsing. The model captures the hierarchical structure information of logical forms. To make each candidate interact with other candidates, we also incorporate a cross-candidate verification mechanism into our model. Further, with the Graph-based Transformer model, we build a generator + ranking pipeline. Experimental results on 3 semantic parsing datasets ATIS, JOBS and TOP show the effectiveness and robustness of our model.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grants No.U1711263.

References

Artzi, Y., and Zettlemoyer, L. 2011. Bootstrapping semantic parsers from conversations. In *Proceedings of the conference on empirical methods in natural language processing*, 421–432. Association for Computational Linguistics.

Artzi, Y., and Zettlemoyer, L. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association of Computational Linguistics* 1:49–62.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Black, E.; Abney, S.; Flickenger, D.; Gdaniec, C.; Grishman, R.; Harrison, P.; Hindle, D.; Ingria, R.; Jelinek, F.; Klavans, J.; et al. 1991. A procedure for quantitatively comparing the syntactic coverage of english grammars. In *Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19-22, 1991*.

Chen, B.; Sun, L.; and Han, X. 2018. Sequence-to-action: End-to-end semantic graph generation for semantic parsing. *arXiv preprint arXiv:1809.00773*.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dong, L., and Lapata, M. 2016. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*.

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

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12(Jul):2121–2159.

Dyer, C.; Kuncoro, A.; Ballesteros, M.; and Smith, N. A. 2016. Recurrent neural network grammars. *arXiv preprint arXiv:1602.07776*.

Einolghozati, A.; Pasupat, P.; Gupta, S.; Shah, R.; Mohit, M.; Lewis, M.; and Zettlemoyer, L. 2019. Improving semantic parsing for task oriented dialog. *arXiv preprint arXiv:1902.06000*.

Gehring, J.; Auli, M.; Grangier, D.; and Dauphin, Y. N. 2016. A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344*.

Guo, T., and Gao, H. 2017. Bidirectional attention for sql generation. *arXiv preprint arXiv:1801.00076*.

Gupta, S.; Shah, R.; Mohit, M.; Kumar, A.; and Lewis, M. 2018. Semantic parsing for task oriented dialog using hierarchical representations. *arXiv preprint arXiv:1810.07942*.

Hemphill, C. T.; Godfrey, J. J.; and Doddington, G. R. 1990. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Jia, R., and Liang, P. 2016. Data recombination for neural semantic parsing. In *Proceedings of ACL*, 12–22.

Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kwiatkowski, T.; Zettlemoyer, L.; Goldwater, S.; and Steedman, M. 2011. Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the conference on empirical methods in natural language processing*, 1512–1523. Association for Computational Linguistics.

Liang, P.; Jordan, M. I.; and Klein, D. 2013. Learning dependency-based compositional semantics. *Computational Linguistics* 39(2):389–446.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, 1532–1543.

Poon, H. 2013. Grounded unsupervised semantic parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 933–943.

Rabinovich, M.; Stern, M.; and Klein, D. 2017. Abstract syntax networks for code generation and semantic parsing. *arXiv preprint arXiv:1704.07535*.

Reddy, S.; Lapata, M.; and Steedman, M. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association of Computational Linguistics* 2(1):377–392.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Shao, B.; Gong, Y.; Bao, J.; Ji, J.; Cao, G.; Lin, X.; and Duan, N. 2019. Weakly supervised multi-task learning for semantic parsing. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 3375–3381. AAAI Press.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, S., and Jiang, J. 2015. Learning natural language inference with lstm. *arXiv preprint arXiv:1512.08849*.

Wang, Y.; Liu, K.; Liu, J.; He, W.; Lyu, Y.; Wu, H.; Li, S.; and Wang, H. 2018. Multi-passage machine reading comprehension with cross-passage answer verification. *arXiv preprint arXiv:1805.02220*.

Wang, D.; Cui, P.; and Zhu, W. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM.

Wen, T.-H.; Gasic, M.; Kim, D.; Mrksic, N.; Su, P.-H.; Vandyke, D.; and Young, S. 2015. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. *arXiv preprint arXiv:1508.01755*.

Wiseman, S., and Rush, A. M. 2016. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960*.

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xiao, C.; Dymetman, M.; and Gardent, C. 2017. Sequence-based structured prediction for semantic parsing. US Patent 9,830,315.

Xu, K.; Wu, L.; Wang, Z.; Yu, M.; Chen, L.; and Sheinin, V. 2018. Exploiting rich syntactic information for semantic parsing with graph-to-sequence model. *arXiv preprint arXiv:1808.07624*.

Yih, S. W.-t.; Chang, M.-W.; He, X.; and Gao, J. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base.

Yin, P., and Neubig, G. 2017. A syntactic neural model for general-purpose code generation. *arXiv preprint arXiv:1704.01696*.

Zettlemoyer, L. S., and Collins, M. 2012. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*.

Zhao, K., and Huang, L. 2014. Type-driven incremental semantic parsing with polymorphism. *arXiv preprint arXiv:1411.5379*.