AST Based Sequence to Sequence Natural Language Question to SQL Method

Kai Yang

[1]Department of Computer Science, Yale University

LILY Lab

Introduction

Converting natural language question to SQL queries is an interesting problem and it is one of the key component of Artificial Intelligent agent. However, the current model could only solve very simple SQL queries and due to the limitation of current dataset and evaluation metrics, the real ability of existing models is overestimated. In this paper, we proposed new dataset and evaluation metrics. We also adopt a AST based model and evaluate the model using new dataset and evaluation metrics.

Materials and Methods

We adopt an AST based sequence to sequence model from Yin et al.
For encoder part, we convert our natural language question into word embeddings and feed into bidirectional LSTM encoder.
For decoder part, the RNN decoder each timestamp generate a grammar rule, an abstract syntax tree could construct by applying the rules from top to bottom and left to right. At each timestamp, the hidden state is generated by:

$$s_t = f_{LSTM}([a_{t-1}: c_t: p_t: n_t], s_{t-1})$$

a_{t-1} is the previous action, c_t is the context vector, p_t is current node's parent, n_t is the embedding of current rule.

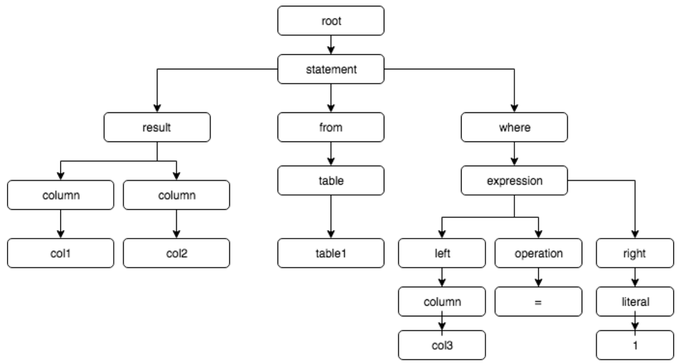

Figure 1. AST For SQL: select col1,col2, from table1 where col3=1;

Derived Grammar Rules From above AST
root->statement
statement->result, from, where
result->column, colum
column->col1
column->col2
from->table
table->table1
where->expression
expression->left, operation, right
left->column
column->col3
operation->=
right->literal
literal->1

Table 1. Derived Grammar Rules

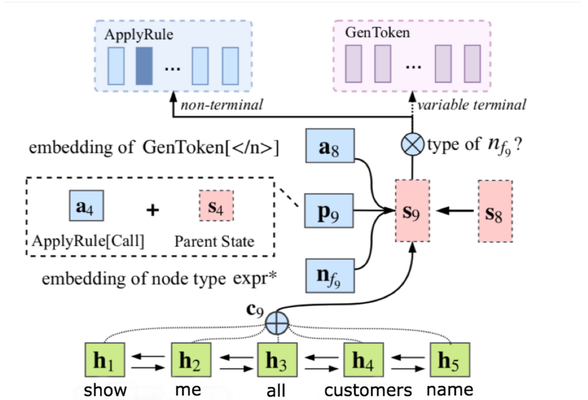

Figure 2. Model Architecture Overview

SQL Components	Accuracy
select	0.216
select_without_agg	0.140
select_agg	0.424
where_expression	0.029
where_operator	0.183
where_nested	0.040
group	0.047
order by	0.250
compound	0.000

Table 2. Test Result on our model

Evaluation and Results

We used an AST based evaluation method. First, we converted generated SQL queries and labeled SQL queries into an AST. After that, we compared the two syntax tree. Also, we divided SQL queries into the following parts and compute accuracy:
a) select columns, b) select all aggregation functions, c) select all without aggregation functions, d) where expressions, e) operations in where, f) nested queries in where, g) group by, h) having, i) order by, k) compound statement: EXCEPT, UNION, INTERSECT
Our dataset contains 4204 training data and 632 test data. Evaluation result is presenting at Table2. From the result, we can see the model is relatively good at predicting aggregation functions in select clause and the model has very limited ability in generating nested queries and compound statement.

Acknowledges

I would like to acknowledge professor Dragomir Radev for his supervise of my project. I would also thanks PhD. Tao Yu for his guidance in the through the progress.

Reference

[1] Yin, Pengcheng, and Graham Neubig. "A syntactic neural model for general-purpose code generation." arXiv preprint arXiv:1704.01696 (2017).