Journal of Computer Science

Original Research Paper

SQL Generation from Natural Language: A Sequence-to-Sequence Model Powered by the Transformers Architecture and Association Rules

[1,2]Youssef Mellah, [1]Abdelkader Rhouati, [1]El Hassane Ettifouri,
[2]Toumi Bouchentouf and [2]Mohammed Ghaouth Belkasmi

[1]NovyLab Research, Novelis, Paris, France
[2]Mohammed First University Oujda, National School of Applied Sciences, LaRSA/SmartICT Lab, Oujda, Morocco

Article history
Received: 12-03-2021
Revised: 21-04-2021
Accepted: 28-04-2021

Corresponding Author:
Youssef Mellah
NovyLab Research, Novelis,
Paris, France
And
LARSA Laboratory, ENSAO,
Mohammed First University,
Oujda, Morocco
Email: ymellah@novelis.io

Abstract: Using Natural Language (NL) to interacting with relational databases allows users from any background to easily query and analyze large amounts of data. This requires a system that understands user questions and automatically converts them into structured query language such as SQL. The best performing Text-to-SQL systems use supervised learning (usually formulated as a classification problem) by approaching this task as a sketch-based slot-filling problem, or by first converting questions into an Intermediate Logical Form (ILF) then convert it to the corresponding SQL query. However, non-supervised modeling that directly converts questions to SQL queries has proven more difficult. In this sense, we propose an approach to directly translate NL questions into SQL statements. In this study, we present a Sequence-to-Sequence (Seq2Seq) parsing model for the NL to SQL task, powered by the Transformers Architecture exploring the two Language Models (LM): Text-To-Text Transfer Transformer (T5) and the Multilingual pre-trained Text-To-Text Transformer (mT5). Besides, we adopt the transformation-based learning algorithm to update the aggregation predictions based on association rules. The resulting model achieves a new state-of-the-art on the WikiSQL DataSet, for the weakly supervised SQL generation.

Keywords: SQL, Text-to-SQL, Sequence-to-Sequence, Transformers Architecture, Multilingual Pre-Trained Text-To-Text Transformer, WikiSQL

Introduction

Semantic Parsing (SP) is one of the most important tasks in NLP, it requires both understanding the meaning of Natural Language (NL) sentences and mapping them to formal meaning representations (Zelle and Mooney, 1996; Panait and Luke, 2005; Clarke *et al*., 2010; Liang *et al*., 2011) often to machine-executable programs, for a range of tasks such as question-answering (Yih *et al*., 2014), robotic control (Matuszek *et al*., 2013) and intelligent tutoring systems (Graesser *et al*., 2005). As a sub-area of SP, we address the problem of mapping natural language utterances to executable relational DB queries, which is known to be difficult due to the flexibility and ambiguity in natural language and the complexity of relational databases.

In database areas (Androutsopoulos *et al*., 1995; Popescu *et al*., 2003; Affolter *et al*., 2019), the general problem was known as "Natural Language Interface to Databases (NLIDBs)", in particular, we are interested in translate natural language questions to SQL, due to the popularity of SQL as the domain-specific language used to query and manage data stored in most available relational databases (Ramakrsihnan *et al*., 1998). Despite the importance of the task, researchers have recently appeared to approach Deep Learning (DL) methods for the crucial problem of NLIDBs.

Translating an NL to SQL is often referenced as "NL-to-SQL" or "Text-to-SQL" (Xu *et al*., 2017; Zhong *et al*., 2017; Shi *et al*., 2018; Yu *et al*., 2018; He *et al*., 2019; Hwang *et al*., 2019; Guo *et al*., 2019). Almost all works operated on achieving good results on well-known Text-to-SQL benchmarks such as ATIS, GeoQuery and WikiSQL (Xu *et al*., 2017; Shi *et al*., 2018; Dong and Lapata, 2018; Hwang *et al*., 2019; He *et al*., 2019).

In this study, we treat the Text-to-SQL task with WikiSQL1[1] (Zhong *et al*., 2017). This DataSet is the first large-scale dataset for Text-to-SQL, with about 80 K human-annotated pairs of Natural Language question and SQL query. WikiSQL is very challenging because tables and questions are very diverse. This DataSet contains about 24K different tables.

There are two leaderboards for the WikiSQL challenge: Weakly supervised (without using logical form during training) and supervised (with logical form during training). On the supervised challenge, there are two results: Those with Execution Guided (EG) inference and those without EG inference.

The previous state-of-the-art weakly supervised model SeqGenSQL+EG (Li *et al*., 2020) achieved 90.5% execution accuracy on the test DataSet. On the supervised challenge, IE-SQL (Ma *et al*., 2020) achieves 87.8% execution accuracy without EG inference on the test dataset and 92.5% execution accuracy with EG inference. In this study, we are interested only in a weakly supervised challenge with execution accuracy as a metric.

In this study, we revisit the Seq2Seq approach, but this time powered by the transformers architecture, precisely using T5 (Raffel *et al*., 2019) and mT5 (Xue *et al*., 2020) language models. The preliminary results show that the prediction of the Aggregation function (AGG) decreases the performance of the model, which brings us to adopt the learning algorithm based on the transformation inspired by (Brill, 1995), to update the aggregation predictions based on association rules, which improve the AGG prediction and the whole model's results.

We organize our paper as follows: In section 2 we review the related work. In section 3 we formulate the problem of Text-to-SQL and describe the WikiSQL DataSet in more detail. In section 4, we describe our methods, then we present the obtained results in section 5. After that, in section 6 we analyze and discuss some errors to improve and finally, we draw the conclusion and future work.

Related Work

Building natural Language Interfaces for Databases (NLIDBs) has been a significant challenge in the SP area. Old works (Warren and Pereira, 1982; Androutsopoulos *et al*., 1995; Popescu *et al*., 2004) focused on rule-based approaches with handcrafted features, then later systems enabled users to query the databases with simple keywords (Simitsis *et al*., 2008; Blunschi *et al*., 2012; Bast and Haussmann, 2015).

The next step was to enable the processing of more complex NL questions by applying a pattern-based approach (Popescu *et al*., 2004; Zheng *et al*., 2017).

Moreover, to improve the precision of natural language interfaces, grammar-based approaches were introduced by restricting users to formulate queries according to certain pre-defined rules (Song *et al*., 2015; Ferré, 2017).

In recent works, that operated on WikiSQL DataSet for training and evaluation; many approaches share a similar encoder-decoder architecture. In this case, information from both the NL and table schema is encoded into a hidden representation by the encoder. Some of those works encode the question with each column name separately (Xu *et al*., 2017; Yu *et al*., 2018; Hwang *et al*., 2019) and other choose encoding the concatenation of the question with columns name (Zhong *et al*., 2017; Dong and Lapata, 2018; Chang *et al*., 2020; Hwang *et al*., 2019; He *et al*., 2019).

There are also some works that do both at different layers (Hwang *et al*., 2019; He *et al*., 2019). Then the hidden representation is decoded with a decoder to a SQL query.

Some early work tried the seq2seq architecture, one step decoding (Zhong *et al*., 2017; Dong and Lapata, 2018), by using advanced Neural Network (NN) architectures to synthesize SQL queries given a user question, precisely the use of a classical encoder-decoder architecture based on Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM). However, it is found challenging in output syntax.

Later more works treat the Text-to-SQL as a classification problem by approaching this task as a sketch-based slot-filling and predicting several parts of the SQL query like SELECT column, WHERE column, WHERE operator, WHERE value, etc. (Xu *et al*., 2017; Yu *et al*., 2018; Hwang *et al*., 2019; He *et al*., 2019). That way, the chance of output syntax problems is reduced.

Then most new works is how to leverage pre-trained language models (Devlin *et al*., 2018; Liu *et al*., 2019b; Yang *et al*., 2019; Liu *et al*., 2019a) and get very good results (He *et al*., 2019; Hwang *et al*., 2019).

“Hybrid Ranking Network” (Lyu *et al*., 2020), is an example of a model which also based on a BERT/RoBERTa pre-trained model and achieved 92% execution accuracy using an annotated logical form and EG inference predictions returning empty results will be dropped and the next most probable prediction is chosen.

The previous state-of-the-art model on the weakly-supervised challenge “SeqGenSQL” on which we are inspired in this study, also based on a pre-trained T5 model, exploring the use of increasing questions with table schema information (column name, type and database content) and the use of automatically augmented training data.

Text-to-SQL Task

Task Definition

The specific semantic parsing problem we study in this study is to map a natural language question to a SQL

[1] https://github.com/salesforce/WikiSQL

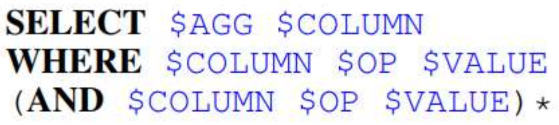
query, which can be executed in a given DB to find the answer to the original question. In particular, we use the currently cross-domain largest natural language questions to SQL DataSet "WikiSQL" (described with more details in the next paragraph) to evaluate our model.

DataSet

We operate on WikiSQL (Zhong *et al*., 2017), a DataSet for Text-to-SQL task which contains a collection of questions, corresponding SQL queries and SQL tables. This is the largest hand-annotated Semantic Analysis DataSet to date, it is larger than other DataSets that have logical shapes, either in terms of the number of tables or number of examples. It provides tree sets: Train, dev and test set.

The WikiSQL dataset consists of 80,654 pairs of SQL questions and queries spread across 24,241 Wikipedia tables. Besides the question in NL, the entry also contains a unique table schema. Each table is present in a single set, train, development or test, which forces models to generalize to invisible tables. The SQL structure of the WikiSQL dataset queries is restricted and always follows the sketch in "Fig. 1".

$COLUMN is a single table column and $AGG is an aggregator function (empty, COUNT, SUM, MAX, MIN, AVG). The WHERE segment is a sequence of conditions. Each $OP is a filtering operator (=, <, >) and the filtering value $VALUE is mentioned in the question. Although the DataSet ships with a "standard" linear order of conditions, the order is irrelevant given the semantics of the WHERE clause (the semantic of "AND"). Figure 2 gives an example from the DataSet.



```
SELECT $AGG $COLUMN
WHERE $COLUMN $OP $VALUE
(AND $COLUMN $OP $VALUE)*
```

Fig. 1: SQL Sketch. The tokens starting with "$" are slots to fill. "*" indicates zero or more AND clauses

Table:

Player	Country	Points	Winnings ($)
Steve stricker	United States	9000	1260000
K.J. Choi	South Korea	5400	756000
Rory Sabbatini	South Africa	3400	4760000
Mark Calcavecchia	United States	2067	289333
Ernie Els	South Africa	2067	289333

Question: What is the points of South Korea player?
SQL: Select point where country = South Korea
Answer: 5400

Fig. 2: Example of WikiSQL DataSet. For given questions and table headers, the model generates corresponding SQL query and retrieves the answer from the table

Methods

We use the original T5&mT5 pre-trained models as our Seq2Seq baseline model.

We notice that the following described methods are generic and can be used for any DataSet handling the Text-to-SQL task.

Input Representation

Given a question Q and a table schema T with columns name $C1$, $C2$, …, Cn, we form the input sequence as follow:

$$[BOS]\ Q\ [SEP]\ C1\ [SEP]\ \ldots\ [SEP]\ [Cn]\ [EOS]$$

where, [*BOS*] and [*EOS*] denote the beginning and the end of the input sequence respectively and [*SEP*] denotes the separator between columns name.

Preprocessing

We only transformed all DataSet text to lowercase format and store the input and the output sequences as Tab-Separated Values in TSV files, the format required for fine-tuning T5&mT5 (we notice that the original format of WikiSQL DataSet files is JSON).

T5&mT5 Fine-Tuning

In recent years, Transfer Learning (TL) has led to a new wave of cutting-edge results in Natural Language Processing (NLP). The power of TL comes from pre-training a model on abundantly available unlabeled text data with a self-supervised task. After that, the model can be refined on smaller labeled data sets, which often results in (much) better performance than training on the labeled data alone. The recent success of transfer learning was sparked in 2018 by ULMFiT (Howard and Ruder, 2018), ELMo (Peters *et al*., 2018) and BERT. The 2019 year saw the development of a wide variety of new methods like GPT (Radford *et al*., 2019), XLNet (Yang *et al*., 2019), RoBERTa, ALBERT (Lan *et al*., 2019), Reformer and MT-DNN (Liu *et al*., 2019a). The pace of progress on the ground has made it difficult to assess the most significant improvements and their effectiveness when combined.

In this sense, we used T5 (for Text-to-Text Transfert Transformer), a language model pre-trained on Colossal Clean Crawled Corpus (C4). This model achieves top results on many NLP benchmarks while being flexible enough to be fine-tuned for a variety of important downstream tasks. We fine-tune T5 on WikiSQL considering the input sequence (question + schema table) and the output SQL query as texts, which is illustrated in "Fig. 3".

We notice that WikiSQL DataSet treats another language besides English, However, T5 is pre-trained model only in English. In fact, it is unable to decode some no English questions. Also within this T5's limits over WikiSQL, it's unable to decode some special characters in the DataSet.

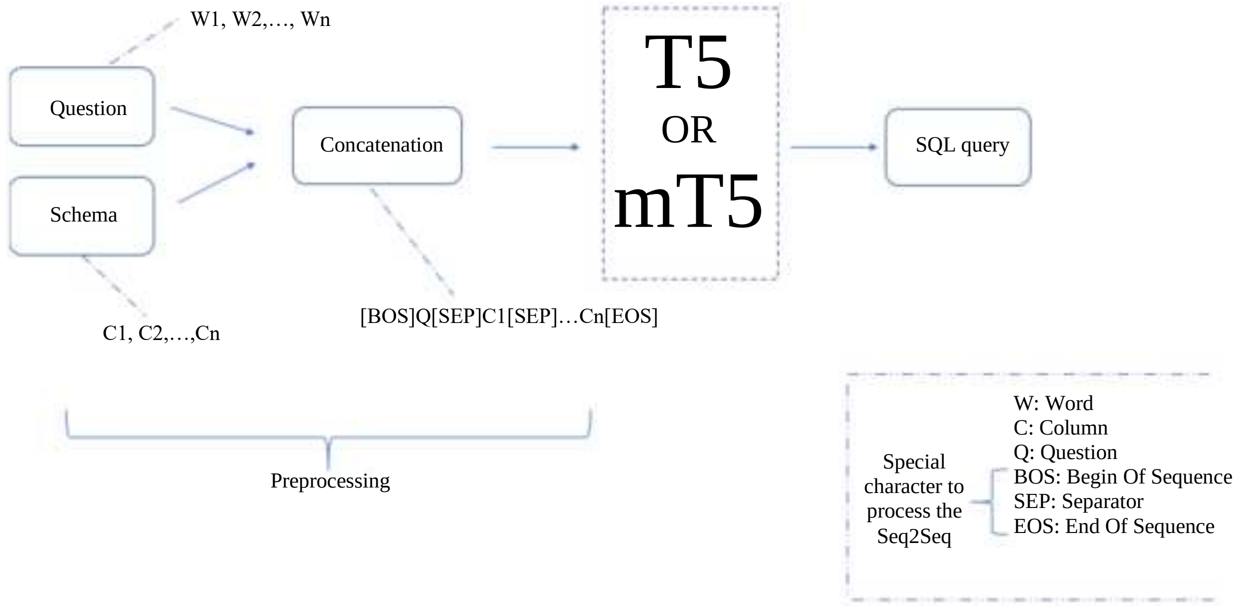


Fig. 3: General Architecture for using T5&mT5 over WikiSQL

To leverage those limits, we fine-tune alternatively mT5 (m for multilingual), which is pre-trained on mC4 DataSet covering 101 languages.

T5&mT5

According to studies and statistics, T5 is more efficient than mT5 on English sequences. Since mT5 allows to decode and give results on non-English and as mentioned below, the WikiSQL dataset also contains non-English sentences with characters not managed by T5, but managed by mT5, we decided to use T5&mT5 together to benefit from the advantages of each pre-trained models. The following algorithm explains how we proceeded.

Algorithm: Procedure pseudo-code for the use of T5&mT5 together over WikiSQL

```
INPUT: Question
OUTPUT: SQL_query
DETERMINE SQL_query depending on the Question
IF Question contains English words only AND not contains special characters
        SQL_Query = Call T5-model
ELSE
        SQL_Query = Call mT5-model
Return SQL_Query
```

Gated Extraction Network

Seq2Seq models suffer from generation completely new words, or not generate all words completely. In our case, the majority of words to predict in the output sequence are present in the input sequence, so in most cases, it is better to copy them directly from the input rather than to try to generate them.

An example of generation new words by T5&mT5 is as follow:

> **Question:** In which country is the city of Netanya
> **Predicted SQL query:** Select country from 1-14937957-1 where city = "net**her**anya"
> **Expected SQL query:** Select country from 1-14937957-1 where city = "netanya"

An example of not generate all complete words by T5&mT5 is as follow:

> **Question:** Are there registration notes on usek.edu.lb?
> **Predicted SQL query:** Select official registration notes from 1-1160660-1 where website = 'usk.edu.lb'
> **Expected SQL query:** Select official registration notes from 1-1160660-1 where website = 'us**e**k.edu.lb'

To handle this problem and encourage extraction from the input, we use a gated extraction T5&mT5 network, similar to (Li *et al*., 2020) and to a Pointer Generation Network (See *et al*., 2017). Between the encoder (Henc) and decoder (Hdec), we implement a cross attention layer, then we create a gate layer from the attention layer to control whether the output should be generated by the decoder or extracted from the encoder.

The cross attention layer is implemented the same way as the T5&mT5 cross attention layer where the score is the product of H_{enc} and H_{dec}, as illustrated in "Fig. 4":

$$HD = Linearization\left(H_{dec}\right);$$
$$HE = V = Linearization\left(H_{enc}\right).$$
$$Score = HD * V$$

The context is then calculated using Score, H_{dec} and softmax function:

$$Context = Func\left(softmax\left(Score\right) * V\right)$$

Based on the hidden state of the Context and the decoder wrapped inside a sigmoid function, we get the final gate as a probability:

$$HdecNormalized = Normalisation\left(HD\right)$$
$$ContextNormalized = Normalisation\left(Context\right)$$

The final step is to merge both generation and extraction using element-wise operation:

$$O\_final = Pext * Oext + \left(1 - Pext\right) * Ogen$$

AGG Prediction Reinforcement

The results of (Hwang *et al*., 2019) indicate that AGG annotations in WikiSQL contain up to 10% errors which negatively affects the prediction of the entire SQL query. In such a case, a learning model fails to train well. So, we decided to improve AGG results compared to the original model, using only simple association signals in the training data. Notably, we adopt a transformation-based learning algorithm (Brill, 1995) to update the aggregation function predictions based on association rules in the form of "change from X0 to X1", (example: Change from COUNT to SUM), considering some word occurrences. The algorithm mine and rank those rules from the training data "Fig. 5".

For example, for the question: What is the total number of assists for players with under 55 games and over 6 assists per game average?

The predicted SQL is: Select count total assists where games inferior 55 and ast avg > 6, while the expected SQL is: Select sum total assists where games inferior 55 and ast avg > 6.

In this case, the rule "Change from COUNT to SUM" is triggered and the predicted SQL query is corrected to: Select sum total assists where games inferior 55 and ast avg > 6; which corresponds to the expected one.

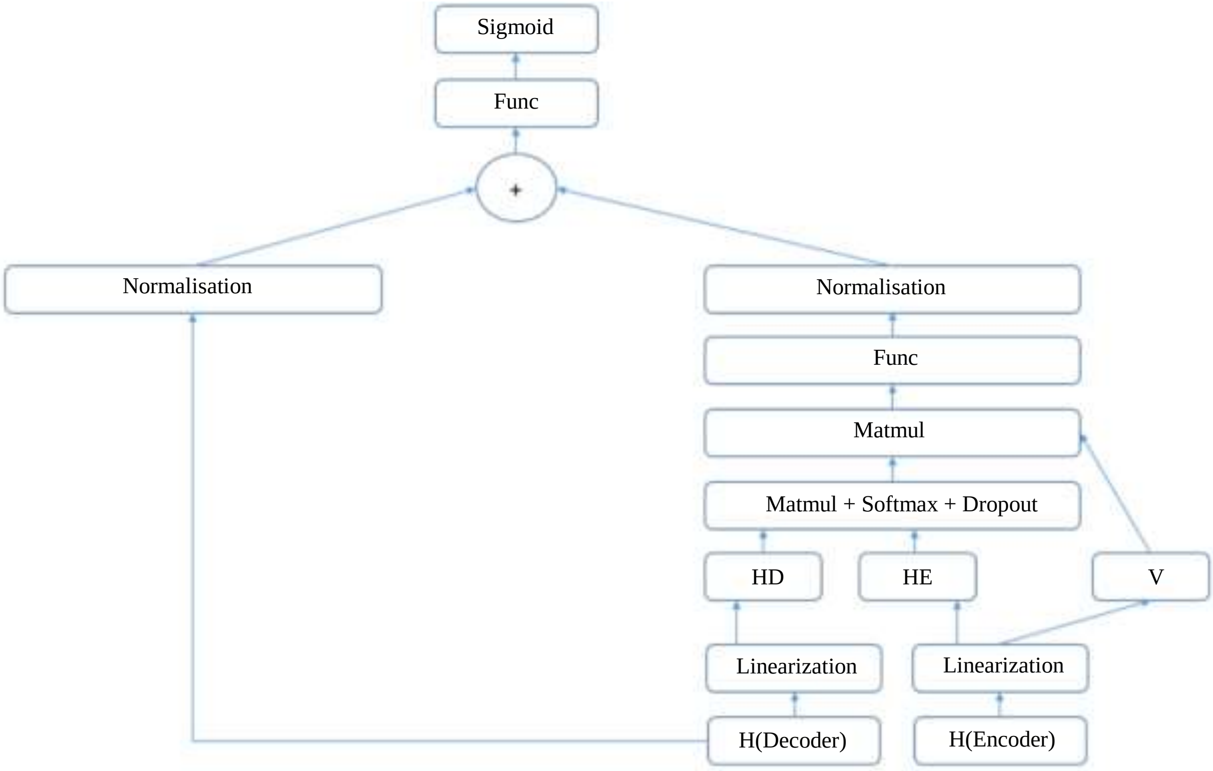


Fig. 4: Operations for gated extraction network

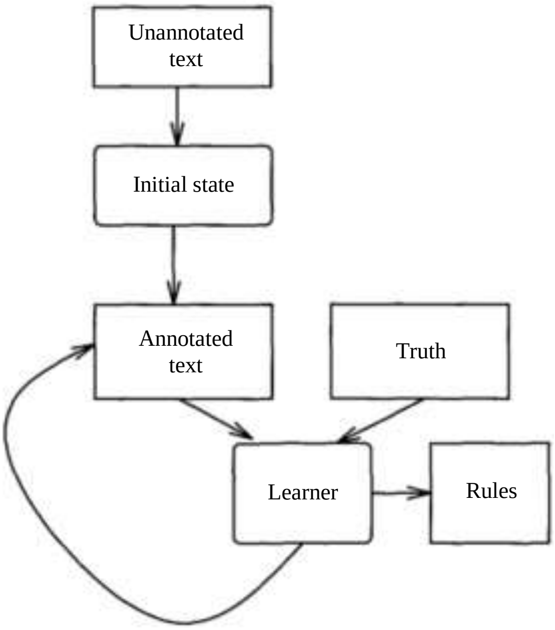


Fig. 5: The mechanism of the transformation-based learning algorithm

Execution Guided Inference

It is always difficult to generate a perfect SQL query. To improve the prediction, Wang *et al*. (2018) introduced Execution Guided (EG) inference.

EG models send generated SQL queries to the SQL database engine and make adjustments if the database engine returns an empty result or run-time errors.

We experimented with beam search using run-time error or empty result during execution.

To apply EG inference, we used beam search by trying to execute each output sequence to the SQL database engine. If the SQL database engine returned a run-time error or an empty result, we drop the current prediction and return the next most efficient output sequence and so on. The algorithm of the EG is as following.

Algorithm for EG

```
Predicts_SQLs = Predict N SQL query (with N > 1)
Final_SQLs = []
For SQL in Predicts_SQLs
    Try
        Result = Run SQL in database
        If Result is Empty
          Continue
        Final_SQLs.append(SQL)
        Break
    Finally
        Continue
## Then we use SQL queries in Final_SQLsfor the evaluation, instead of initials SQL queries in Predicts_SQLs predicted by T5/mT5
```

Experiments and Results

Implementation Details

We used Colab pro with TPU and Google Cloud Storage (GCS) as our environment and Pytorch with Tensorflow for codding.

240 is the length of input tokens and 75 is the length of the output sequence.

For tokenization, we used the default tokenizers T5Tokenizer and MT5Tokenizer of T5&mT5 respectively. And since the raw data splits are stored as JSON files, we first converted them to TSV format to make them parsable in TensorFlow. We also take the opportunity to do a bit of cleaning of the text.

For the loss function, we used CrossEntropyLoss and AdamW (Loshchilov and Hutter, 2017) to optimize the models with default hyperparameters.

We train both T5&mT5 on 35 epochs with the batch size set in 64.

Evaluation and Results

“Table 1” shows the performances of our work. Using T5-base and mT5-base, our baselines models achieve 84.1 and 86.2% respectively, on test data execution accuracy.

Using a gated extraction network, the precision is improved by about 1%, achieving 85.1 and 87.2% using T5&mT5 respectively.

Ensembling T5&mT5 and using them at the same time improve the performance by 5% compared to our baseline, achieving 90% on test data execution accuracy.

By adding association rules, the accuracy is augmented by 0.3% achieving 90.3%.

Finally, by using EG and replacing T5-base and mT5-base by T5-large and mT5-large respectively, the test data execution accuracy improved more and achieves 91.0%. “Figure 6” presents a bar chart for visualizing the improvements of experimentation results on WikiSQL DataSet.

With those techniques, we exceeded the previous top model (Li *et al*., 2020) by 0.5% and achieving the new state of the art in the weakly supervised WikiSQL challenge.

Table 1: Results of our multiple experimentations on WikiSQL DataSet

Model	Test execution accuracy
T5 baseline (T5-base)	84.1
T5-base + gated extraction network	85.1
mT5 baseline (mT5-base)	86.2
mT5 + gated extraction network	87.2
T5-base&mT5-base	90.0
T5-base&mT5-base+ Association Rules	90.3
T5-large&mT5-large + Association Rules	91.0

Table 2: Our best result compared to previous works on weakly-supervised WikiSQL challenge

Model	Dev execution accuracy	test execution accuracy
T5-large&mT5-large + Association Rules (ours)	91.2	91.0
SeqGenSQL+EG (Li *et al*., 2020)	90.8	90.5
SeqGenSQL (Li *et al*., 2020)	90.6	90.3
HardEM (Min *et al*., 2019)	84.4	83.9
LatentAlignment (Wang *et al*., 2018)	79.4	79.3
MeRL Agarwal, 2019	74.9 +/- 0.1	74.8 +/- 0.2
MAPO (Liang *et al*., 2018)	72.2 +/- 0.2	72.1 +/- 0.3
Rule-SQL (Guo *et al*., 2019)	61.1 +/- 0.2	61.0 +/- 0.3

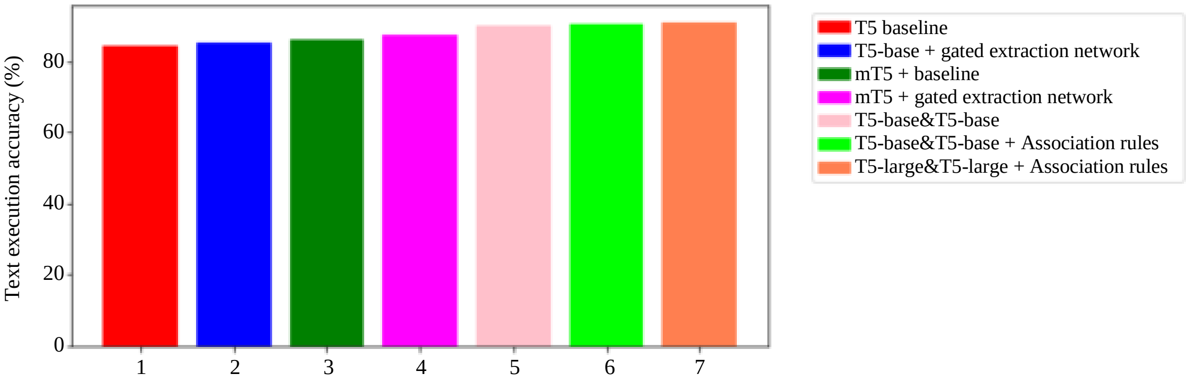


Fig. 6: Chart for visualizing the improvements of experimentation results on WikiSQL DataSt

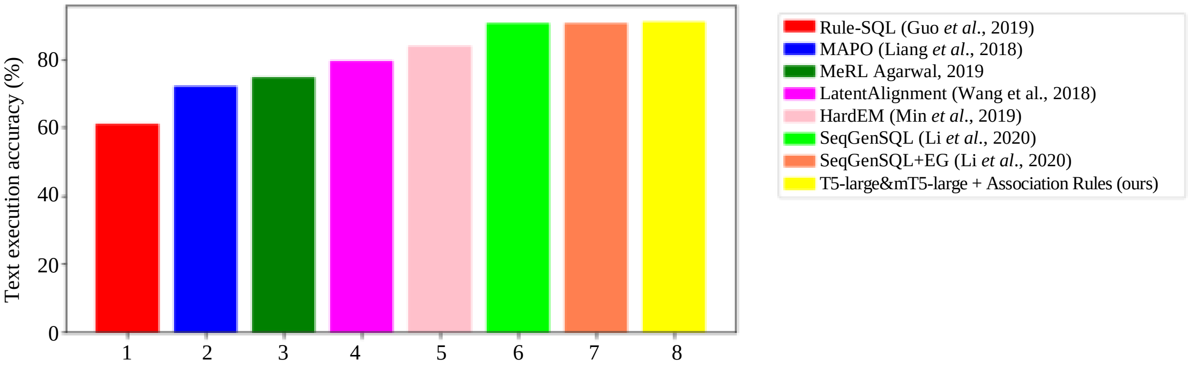


Fig. 7: Our top work compared to previous ones on WikiSQL DataSet

"Table 2" shows our position compared to the top previous works operated on the weakly supervised WikiSQL challenge. We note that the majority of those works uses the database content, so we believe that if we did it to, we can achieves more accuracy. "Figure 7" presents a bar chart for visualizing our top work compared to previous ones, in term of test execution accuracy.

Errors Analysis and Discussion

Even with association rules, analysis of the errors on the test data shows that most of them were in the prediction of bad aggregation functions.

Besides, some questions contain ambiguous words. For example, the word ”total” can be interpreted as a SUM function in SQL and the word "sum" can be interpreted as a COUNT function.

Even with the gated extraction network, the model still predicts new words or predicts an SQL query with fewer characters or words.

We remarked also that some of the questions simply didn't provide sufficient information to compose an accurate SQL statement, particularly, questions that do not contain the name of columns, so in this case, the model predicts usually the wrong column in the SQL query (select column or condition column).

We also note that there are cases where the expected SQL query doesn't match the questions and in some of these cases, our model predicts the SQL query correctly in manual verification, even though they don't match the expected one.

Despite all these constraints, Sequence generation simplifies the process of generating SQL statements and performs one-step prediction for this task with high accuracy. Besides, we still consider the following things for improving the accuracy of the model:

- We assume larger base models could provide even more improvement. I fact, we can use T5-3B and mT5-xl (about * 4 larger than T5-large and mT5-large), but this requires more resources precisely in terms of TPU and RAM
- Improve the gated extraction network
- Carefully designed question augmentation (with more information like data-type, POS), as well as accessing the content of the database and include some rows as input
- Training the model on more data by doing data augmentation on train and dev set

Conclusion

In this study, we present our work for the generation of SQL queries from natural language. We used T5&mT5 for SQL sequence generation and we adopt a transformation-based learning algorithm to update the aggregation function predictions based on simple association rules, which helps to improve the general results. We trained and evaluated our model on WikiSQL DataSet, outperforming all previous works and achieving a new state-of-the-art on the weakly supervised challenge. As future work, we plan first to improve our model over WikiSQL Dataset, then use more complex DataSet like Spider, which treats multi-tables, containing nested SQL queries, with jointures and more components.

Acknowledgment

I would like to thank all members of the Novelis research team, as well as all members of the LARSA laboratory of ENSAO, Mohammed First University.

Author’s Contributions

All authors equally contributed in this work.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

Affolter, K., Stockinger, K., & Bernstein, A. (2019). A comparative survey of recent natural language interfaces for databases. The VLDB Journal, 28(5), 793-819. https://doi.org/10.1007/s00778-019-00567-8

Androutsopoulos, I., Ritchie, G. D., & Thanisch, P. (1995). Natural language interfaces to databases-an introduction. Natural Language Engineering, 1(1), 29-81. https://doi.org/10.1017/S135132490000005X

Bast, H., & Haussmann, E. (2015, October). More accurate question answering on freebase. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (pp. 1431-1440). https://doi.org/10.1145/2806416.2806472

Blunschi, L., Jossen, C., Kossman, D., Mori, M., & Stockinger, K. (2012). Soda: Generating SQL for business users. Proceedings of the VLDB Endowment, 5, 932-943. https://doi.org/10.14778/2336664.2336667

Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. Computational Linguistics, 21(4), 543-565. https://dl.acm.org/doi/abs/10.5555/218355.218367

Chang, S., Liu, P., Tang, Y., Huang, J., He, X., & Zhou, B. (2020, April). Zero-shot text-to-SQL learning with auxiliary task. In Proceedings of the AAAI Conference on Artificial Intelligence, (pp. 7488-7495). Association for the Advancement of Artificial Intelligence. https://doi.org/10.1609/aaai.v34i05.6246

Clarke, J., Goldwasser, D., Chang, M. W., & Roth, D. (2010, July). Driving semantic parsing from the world’s response. In Proceedings of the fourteenth conference on computational natural language learning (pp. 18-27). https://www.aclweb.org/anthology/W10-2903.pdf

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://arxiv.org/abs/1810.04805

Dong, L., & Lapata, M. (2018). Coarse-to-fine decoding for neural semantic parsing. In 56th Annual Meeting of the Association for Computational Linguistics, (pp. 731–742), Association for Computational Linguistics, Melbourne, Australia. https://doi.org/10.18653/v1/P18-1068

Ferré, S. (2017). Sparklis: An expressive query builder for SPARQL endpoints with guidance in natural language. Semantic Web, 8(3), 405-418. https://doi.org/10.3233/SW-150208

Graesser, A. C., Chipman, P., Haynes, B. C., & Olney, A. (2005). AutoTutor: An intelligent tutoring system with mixed-initiative dialogue. IEEE Transactions on Education, 48(4), 612-618. https://doi.org/10.1109/TE.2005.856149

Guo, J., Zhan, Z., Gao, Y., Xiao, Y., Lou, J. G., & Liu, T. (2019). Towards complex text-to-SQL in cross-domain database with intermediate representation. In 57th Annual Meeting of the Association for Computational Linguistics, (pp. 4524-4535), Association for Computational Linguistics, Florence, Italy. https://doi.org/10.18653/v1/P19-1444

He, P., Mao, Y., Chakrabarti, K., & Chen, W. (2019). X-SQL: reinforce schema representation with context. arXiv preprint arXiv:1908.08113. https://arxiv.org/abs/1908.08113

Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, (pp. 328-339), Association for Computational Linguistics, Melbourne, Australia. https://doi.org/10.18653/v1/P18-1031

Hwang, W., Yim, J., Park, S., & Seo, M. (2019). A comprehensive exploration on wikiSQL with table-aware word contextualization. arXiv preprint arXiv:1902.01069. https://arxiv.org/abs/1902.01069

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942. https://arxiv.org/abs/1909.11942

Li, N., Keller, B., Butler, M., & Cer, D. (2020). SeqGenSQL--A Robust Sequence Generation Model for Structured Query Language. arXiv preprint arXiv:2011.03836. https://arxiv.org/abs/2011.03836

Liang, P., Tripp, O., & Naik, M. (2011, January). Learning minimal abstractions. In Proceedings of the 38th annual ACM SIGPLAN-SIGACT Symposium on Principles of programming languages (pp. 31-42). https://doi.org/10.1145/1926385.1926391

Liu, X., He, P., Chen, W., & Gao, J. (2019a). Multi-task deep neural networks for natural language understanding. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, (pp. 4487–4496), Association for Computational Linguistics, Florence, Italy. https://doi.org/10.18653/v1/P19-1441

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019b). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692. https://arxiv.org/abs/1907.11692

Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101. https://arxiv.org/abs/1711.05101

Lyu, Q., Chakrabarti, K., Hathi, S., Kundu, S., Zhang, J., & Chen, Z. (2020). Hybrid ranking network for text-to-SQL. arXiv preprint arXiv:2008.04759. https://arxiv.org/abs/2008.04759

Ma, J., Yan, Z., Pang, S., Zhang, Y., & Shen, J. (2020). Mention Extraction and Linking for SQL Query Generation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), (pp. 6936–6942), Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.563

Matuszek, C., Herbst, E., Zettlemoyer, L., & Fox, D. (2013). Learning to parse natural language commands to a robot control system. In Experimental robotics (pp. 403-415). Springer, Heidelberg. https://doi.org/10.1007/978-3-319-00065-7_28

Min, S., Chen, D., Hajishirzi, H., & Zettlemoyer, L. (2019). A discrete hard EM approach for weakly supervised question answering. arXiv preprint arXiv:1909.04849.

Panait, L., & Luke, S. (2005). Cooperative multi-agent learning: The state of the art. Autonomous Agents and Multi-Agent Systems, 11(3), 387-434. https://doi.org/10.1007/s10458-005-2631-2

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, (pp. 2227–2237), Association for Computational Linguistics, New Orleans, Louisiana. https://doi.org/10.18653/v1/N18-1202

Popescu, A. M., Armanasu, A., Etzioni, O., Ko, D., & Yates, A. (2004). Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics (pp. 141-147). https://doi.org/10.3115/1220355.1220376

Popescu, A. M., Etzioni, O., & Kautz, H. (2003, January). Towards a theory of natural language interfaces to databases. In Proceedings of the 8th international conference on Intelligent user interfaces (pp. 149-157). https://doi.org/10.1145/604045.604120

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI Blog, 1(8), 9. http://www.persagen.com/files/misc/radford2019language.pdf

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21, 1-67. https://www.jmlr.org/papers/volume21/20-074/20-074.pdf

Ramakrsihnan, R., Donjerkovic, D., Ranganathan, A., Beyer, K. S., & Krishnaprasad, M. (1998, July). SRQL: Sorted relational query language. In Proceedings. Tenth International Conference on Scientific and Statistical Database Management (Cat. No. 98TB100243) (pp; 84-95). IEEE. https://doi.org/10.1109/SSDM.1998.688114

See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, (pp. 1073–1083), Association for Computational Linguistics, Vancouver, Canada. https://doi.org/10.18653/v1/P17-1099

Shi, T., Tatwawadi, K., Chakrabarti, K., Mao, Y., Polozov, O., & Chen, W. (2018). IncSQL: Training incremental text-to-SQL parsers with non-deterministic oracles. arXiv preprint arXiv:1809.05054. https://arxiv.org/abs/1809.05054

Simitsis, A., Koutrika, G., & Ioannidis, Y. (2008). Précis: from unstructured keywords as queries to structured databases as answers. The VLDB Journal, 17(1), 117-149. https://doi.org/10.1007/s00778-007-0075-9

Song, D., Schilder, F., Smiley, C., Brew, C., Zielund, T., Bretz, H., ... & Harrison, J. (2015, October). TR Discover: A natural language interface for querying and analyzing interlinked datasets. In International Semantic Web Conference (pp. 21-37). Springer, Cham. https://doi.org/10.1007/978-3-319-25010-6_2

Wang, C., Tatwawadi, K., Brockschmidt, M., Huang, P. S., Mao, Y., Polozov, O., & Singh, R. (2018). Robust text-to-SQL generation with execution-guided decoding. arXiv preprint arXiv:1807.03100. https://arxiv.org/abs/1807.03100

Warren, D. H., & Pereira, F. C. (1982). An efficient easily adaptable system for interpreting natural language queries. American Journal of Computational Linguistics, 8(3-4), 110-122. https://dl.acm.org/doi/abs/10.5555/972942.972944

Xu, X., Liu, C., & Song, D. (2017). SQLNet: Generating structured queries from natural language without reinforcement learning. arXiv preprint arXiv:1711.04436. https://arxiv.org/abs/1711.04436

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., ... & Raffel, C. (2020). mT5: A massively multilingual pre-trained text-to-text transformer. arXiv preprint arXiv:2010.11934. https://arxiv.org/abs/2010.11934

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019). XLNet: Generalized autoregressive pretraining for language understanding. arXiv preprint arXiv:1906.08237. https://arxiv.org/abs/1906.08237

Yih, W. T., He, X., & Meek, C. (2014, June). Semantic parsing for single-relation question answering. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (pp. 643-648). https://doi.org/10.3115/v1/P14-2105

Yu, T., Li, Z., Zhang, Z., Zhang, R., & Radev, D. (2018). TypeSQL: Knowledge-based type-aware neural text-to-SQL generation. 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics, New Orleans. https://arxiv.org/abs/1804.09769

Zelle, J. M., & Mooney, R. J. (1996, August). Learning to parse database queries using inductive logic programming. In Proceedings of the national conference on artificial intelligence (pp. 1050-1055). https://www.aaai.org/Papers/AAAI/1996/AAAI96-156.pdf

Zheng, W., Cheng, H., Zou, L., Yu, J. X., & Zhao, K. (2017, November). Natural language question/answering: Let users talk with the knowledge graph. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 217-226). https://doi.org/10.1145/3132847.3132977

Liang, C., Norouzi, M., Berant, J., Le, Q., & Lao, N. (2018). Memory augmented policy optimization for program synthesis and semantic parsing. arXiv preprint arXiv:1807.02322.

Zhong, V., Xiong, C., & Socher, R. (2017). Seq2SQL: Generating structured queries from natural language using reinforcement learning. arXiv preprint arXiv:1709.00103. https://arxiv.org/abs/1709.00103