Deep Learning Approaches for Text-to-SQL Systems

George Katsogiannis-Meimarakis
katso@athenarc.gr
Athena Research Center
Greece

Georgia Koutrika
georgia@athenarc.gr
Athena Research Center
Greece

ABSTRACT

To bridge the gap between users and data, numerous text-to-SQL systems have been developed that allow users to pose natural language questions over relational databases. Recently, novel text-to-SQL systems are adopting deep learning methods with very promising results. At the same time, several challenges remain open making this area an active and flourishing field of research and development. To make real progress in building text-to-SQL systems, we need to de-mystify what has been done, understand how and when each approach can be used, and, finally, identify the research challenges ahead of us. The purpose of this tutorial is to present recent advances of deep learning techniques for text-to-SQL translation, and to highlight open problems and new research opportunities for researchers and practitioners in the fields of database systems, natural language processing and deep learning.

1 INTRODUCTION

Data is a prevalent part of every business and scientific domain, but its explosive volume and increasing complexity make data querying and exploration challenging even for experts. In an attempt to bridge the gap between users and data, numerous *text-to-SQL systems* have been implemented, both from industry and academia, that enable users to pose *unstructured queries* (using keywords or free-form text) over relational databases [1, 4, 26]. The recent advances on deep neural networks and the creation of two large datasets for training text-to-SQL systems, have led to the emergence of several, novel, text-to-SQL systems that leverage deep learning techniques. These efforts show very promising results. At the same time, several open challenges make this area an active and flourishing field of research and development. It is high time for a systematic study of these solutions.

In this work, we aim at presenting the recent advances in the field of text-to-SQL systems with the adoption of deep learning techniques. We follow a systematic and structured approach. First, we introduce the text-to-SQL problem, explain and categorize its challenges. Then, we present available benchmarks and explain their advantages and shortcomings. We zoom in on the recent advances of deep learning techniques for text-to-SQL translation. We explain the problems they address and their limitations, and we highlight research opportunities on the intersection of database systems, natural language processing and deep learning.

2 THE TEXT-TO-SQL PROBLEM

The text-to-SQL (also known as NL2SQL) problem can be described as follows: *Given a Natural Language Query (NLQ) on a Relational Database (RDB), produce a SQL query equivalent to the NLQ, which is valid for the said RDB.* Several challenges arise, including: ambiguity, schema linking, vocabulary gap and user mistakes.

Ambiguity of natural language queries is one of the most difficult challenges a text-to-SQL system has to cope with. There are several types of ambiguity [3, 27]. For instance, *lexical ambiguity* refers to the case of a single word with multiple meanings (e.g., *"Paris"* can be a city or a person).

On the other hand, *schema linking* is the problem of understanding which parts of the NLQ refer to which parts of the database schema. *Vocabulary gap* refers to the differences between the vocabulary used by the database and the one used by the user. *User mistakes*, such as syntactical or grammatical errors, make the problem even more challenging.

3 TEXT-TO-SQL LANDSCAPE

The problem of translating user queries to SQL has been a holy grail for the database community for over 30 years [4]. In this section, we will give a very brief overview of the earlier approaches, especially those proposed by the database community.

Early database approaches use (a) inverted indexes (like search engines do) to map query keywords to database elements (relations, attributes and values) and (b) the database schema to find how relations in a query should be joined [18]. These approaches use either a schema graph that represents the database relations as nodes and the joins between them as edges [2, 6, 14, 21, 22, 30, 38] or a tuple graph where the nodes are the database tuples [5, 9, 13, 16]. Answers to a query are defined as sub-graphs over the complete graph, comprising a subset of the relations and tuples that contain the query keywords and are connected by the joins between them. NALIR [17] is the first to use a syntactic parse tree to represent a query and map it to the database schema graph. ATHENA [29] employs an ontology to represent a real-world domain (such as finance) and an ontology-to-database mapping, which describes how the ontology elements are mapped to the database objects. DBPal [33] aims at generating join queries based on the information learnt from domain-specific training data, and requires many training examples with different join paths.

4 AVAILABLE BENCHMARKS

Training a deep learning system is a very data-intensive procedure; large amounts of data are required in order to train an accurate model. For this reason, the availability of datasets is the main fuel for the development of deep learning solutions and the text-to-SQL task is no exception. In this section, we will introduce the two major large-scale benchmarks, explain their characteristics as well as highlight their shortcomings.

WikiSQL [39] is *a large crowd-sourced dataset for developing natural language interfaces for relational databases* released along with the Seq2SQL text-to-SQL system. It contains over 25,000 tables gathered from Wikipedia pages and over 80,000 natural language and SQL question pairs, which were created by crowdsourcing. Note that each of WikiSQL's questions is directed to a single table and not to a relational database. This means that the

© 2021 Copyright held by the owner/author(s). Published in Proceedings of the 24th International Conference on Extending Database Technology (EDBT), March 23-26, 2021, ISBN 978-3-89318-084-4 on OpenProceedings.org.

10.5441/002/edbt.2021.90

proposed task is much simpler than the ultimate goal of creating a natural language interface for relational databases. Additionally, the complexity of the queries is very low. There are no JOIN, GROUP BY, UNION, INTERSECTION or other complex SQL elements. We must also note that that WikiSQL contains multiple errors and ambiguities, which might hinder the performance of any model trained on it.

Spider [37] *is a large-scale complex and cross-domain semantic parsing and text-to-SQL dataset* annotated by 11 Yale students. It contains 200 relational databases from 138 different domains along with over 10,000 natural language questions and over 5,000 complex SQL queries. Its queries range from very simple to very hard, using all the common SQL elements, including nested queries. All the above, along with the fact that it was hand-crafted and re-checked are an indicator of its higher quality, compared to WikiSQL, and has led to the development of very promising systems.

5 NL REPRESENTATION

We will now provide an overview of the state-of-the-art techniques for natural language representation in neural networks. The use of neural networks, which can only handle numerical inputs and not raw text, has led to the adoption of word embeddings for numerical word representation. Additionally, in the past few years, the use of language models is blooming, following their rise as an efficient solution for increased performance in NL tasks.

Word embeddings assume that every unique word has a numerical representation that can be different from all other words and at the same time incorporate useful information about the word, and aim at mapping each word to a multidimensional vector. Besides the brute-force creation of one-hot embeddings, researchers have provided highly efficient techniques to create representations that carry the word's meaning and its relationships with other words. Word2Vec [25], GloVe [28] and WordPiece embeddings [34], to name a few, are some famous word embedding techniques that are used in most, if not all, text-to-sql systems.

Language models are a novel and emerging type of pre-trained neural networks for processing NL, that has been shown to excel in NL tasks during the past few years. Note that language models are not a replacement for word embeddings, given that they are neural networks and they still need a way of transforming words to vectors. The way this type of models are created, is that a very large network (10^8 order of magnitude of parameters) is created and is pre-trained on a very large NL dataset (10^9 order of magnitude of words). The pre-trained model is made available for researchers who can then adapt its inputs and outputs to the specific task they aim to solve, and train it for an additional number of epochs on their task-specific dataset. The result is a much stronger model that can reach state-of-the-art performance even without the need of complex architectures [8]. These models have been able to reach such performances due to the use of a neural network architecture that was recently proposed, called the Transformer [31], which excels at handling NL sequences. Some of the most used language models for the text-to-SQL task are BERT [8] and MT-DNN [20].

6 TEXT-TO-SQL DEEP LEARNING APPROACHES

Deep learning systems following the encoder-decoder architecture can be distinguished in three categories, based on the output of their decoder [7]: (a) sequence-to-sequence approaches, (b) grammar-based approaches, and (c) sketch-based slot-filling approaches. In order to better understand the proposed systems, we will now give a taxonomy of deep learning approaches for text-to-SQL, and highlight the main characteristics as well as the advantages and shortcomings of each neural network architecture. Additionally, we will provide an overview of some key systems in each category.

6.1 Sequence-to-sequence approaches

This category includes systems (e.g. [19, 39]) that produce a sequence of SQL tokens and schema elements as their output, with the resulting sequence being the final SQL query prediction, or a major part of it. Essentially, they attempt to transform an input NLQ sequence to an output SQL sequence. This approach is the simplest, but is also very prone to errors. It was adopted by one of the first deep-learning systems for the task at hand, Seq2SQL [39], but later systems steered away from such approaches. The main drawback of sequence-to-sequence architectures is that they do not take the strict grammatical rules of SQL into account when generating a query. The system attempts to learn how a SQL sequence is generated, but at prediction time there are no measures to safeguard from producing syntactically incorrect queries.

Seq2SQL [39] was one of the first neural networks created specifically for the text-to-SQL task and was based on a previous work focusing on generating logical forms using neural networks [10]. Its authors released the WikiSQL dataset along with it, which signified a new era for deep learning research on the text-to-SQL problem. The system predicts an aggregation function and the column for the SELECT clause as classification tasks and generates the WHERE condition clause using a seq-to-seq network. The latter part of the system is burdened with generating parts of the query that can lead to syntactic errors, which is its major drawback. The network architecture combines LSTM and linear layers, and the GloVe embeddings are used to represent the inputs.

6.2 Grammar-based approaches

Grammar-based approaches (e.g., [7, 11, 12, 32]) are an evolution of sequence-to-sequence approaches, and produce a sequence of grammar rules instead of simple tokens as their output. These grammar rules are instructions that, when applied, can create a SQL query. The advantage over sequence-to-sequence approaches is that the possibility for generating an out-of-place token or a syntactically incorrect query is dramatically reduced. This is the most used approach for generating complex SQL queries.

RAT-SQL [32] is a grammar-based text-to-SQL system focusing on the Spider dataset. It is capable of generating complex SQL queries by incorporating three note-worthy features. First, it creates a *question-contextualized schema graph*, i.e. a graph representing the database schema, its tables and columns, as well as the words of the user's question as nodes and the connections between them as edges. The edges between DB elements are created based on the DB schema and the edges between NLQ words and DB elements are created by performing text matching, which is a form of schema linking. Furthermore, it uses a modified Transformer network for *relation aware self-attention*, that is specifically designed to leverage the information of the created graph and its edges. Finally, it follows a method for SQL generation as an abstract syntax tree, by generating a sequence of actions for building the tree, as proposed in [36].

IRNet [12] is another grammar-based system capable of generating complex SQL queries. It uses text-matching techniques to address the schema linking challenge similarly but in a simpler form than RAT-SQL. It uses a complex architecture of linear and recurrent neural networks to process the input, in addition to BERT. After processing the input, it creates an SQL query using the same method as RAT-SQL for generating an abstract syntax tree, with the main difference that the output it produces is in an intermediate language called SemQL designed specifically for this system. Its authors argue that it is easier to generate queries in this language and then transform them to SQL.

6.3 Sketch-based slot-filling approaches

Systems following this approach (e.g., [15, 23, 24, 35]) aim at simplifying the difficult task of generating a SQL query, to the easier task of predicting certain parts of the query (e.g. which of the table columns will appear in the SELECT clause), transforming in this way the SQL generation task to a classification task. In this case, we consider a query sketch with a number of empty slots that must be filled and develop neural networks that predict which element is most probable to fill each slot. A basic prerequisite for such approaches is to have a query sketch that, when filled, will be able to capture the NLQ's intention. As a result, this category of systems is rarely able to produce complex SQL queries.

SQLNet [35] was one of the first sketch-based approaches. It was based on the observation that the way Seq2SQL chose to generate the WHERE clause was prone to errors that could be avoided. For this reason, a query sketch, which could cover every SQL query in the WikiSQL dataset, was developed and separate neural networks were created to fill each slot. All slots are filled by considering a classification task (e.g., which of the six possible aggregation functions is appropriate for the given NLQ) except for the condition value slot which was generated by a seq-to-seq network. Note that in this case the aforementioned seq-to-seq network only generates a value and does not handle SQL tokens, meaning that it is not possible to generate syntactically incorrect queries. Another improvement is the introduction of a *column attention* neural mechanism to the network.

HydraNet [23] focuses on the WikiSQL task and follows a sketch-based approach, using the same sketch as SQLNet, but takes advantage of the BERT language model and achieves much better results.

SQLova [15] is another sketch-based approach focusing on the WikiSQL dataset and leveraging the BERT language model, just as the HydraNet system. Their main difference is that while HydraNet aims to use a very simple network after receiving BERT's output, SQLova employs a large and complex network similar to the one used by SQLNet, while also incorporating BERT into the system. What must be noted is that even though SQLova employs a larger and more complex network than HydraNet, it achieves lower accuracy scores on the WikiSQL dataset.

7 CHALLENGES AND RESEARCH OPPORTUNITIES

While a lot of progress has been made on the text-to-SQL problem, several important issues need to be tackled. Here, we outline some of the most challenging ones.

The need for new benchmarks and in-depth system evaluations is pressing and the database community can help complement the work done by benchmarks such as Spider. New benchmarks are needed that can test the query expressivity (i.e., what types of queries a system can answer) as well as the efficiency and scalability of text-to-SQL systems to bigger and more complex data sets.

Furthermore, there is a need for further research on answer validation. Since in many cases users are not familiar with SQL, the question is how they can confirm that the obtained results match the intention of the NLQ. Another challenge is the universality of the solution, i.e. the system's ability to perform equally well for different databases. It is also important to enable natural language queries in languages other than English, which is the main focus of current efforts. Due to the problem's multidisciplinarity, database, ML, and NLP approaches can join forces to push the barrier further.

8 PRESENTERS

George Katsogiannis-Meimarakis is a research assistant at Athena Research Center in Athens, Greece, where he works on the INODE (Intelligent Open Data Exploration) project, focusing on the text-to-SQL problem. He is a graduate of the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens, where he completed his thesis with the title "Translating Natural Language to SQL using Deep Learning". Currently, he is attending a MSc programme on Data Science and Information Technologies with a specialisation on Artificial Intelligence and Big Data.

Georgia Koutrika is a Research Director at Athena Research Center in Greece. She has more than 15 years of experience in multiple roles at HP Labs, IBM Almaden, and Stanford. Her work focuses on data exploration, recommendations, and data analytics, and has been incorporated in commercial products, described in 14 granted patents and 26 patent applications in the US and worldwide, and published in more than 90 papers in top-tier conferences and journals. She is Editor-in-chief for VLDB Journal, PC chair for VLDB 2023, associate editor for TKDE, and an ACM Distinguished Speaker. *Prior tutorials:* Fairness in Rankings and Recommenders [EDBT20], Recommender Systems [SIGMOD'18, EDBT'18, ICDE'15], Personalization [ICDE'10, ICDE'07, VLDB'05].

ACKNOWLEDGMENTS

This work has been partially funded by the European Union's Horizon 2020 research and innovation program (grant agreement No 863410).

REFERENCES

[1] Katrin Affolter, Kurt Stockinger, and Abraham Bernstein. 2019. A comparative survey of recent natural language interfaces for databases. *VLDB J.* 28, 5 (2019), 793–819.

[2] Sanjay Agrawal, Surajit Chaudhuri, and Gautam Das. 2002. DBXplorer: A system for keyword-based search over relational databases. In *ICDE*. 5–16.

[3] Ambiguity [n.d.]. Ambiguity. https://stanford.io/2YXcECi.

[4] Ion Androutsopoulos, Graeme D. Ritchie, and Peter Thanisch. 1995. Natural language interfaces to databases - an introduction. *Natural Language Engineering* 1, 1 (1995), 29–81. https://doi.org/10.1017/S135132490000005X

[5] Gaurav Bhalotia, Arvind Hulgeri, Charuta Nakhe, Soumen Chakrabarti, and Shashank Sudarshan. 2002. Keyword searching and browsing in databases using BANKS. In *ICDE*. 431–440.

[6] Lukas Blunschi, Claudio Jossen, Donald Kossmann, Magdalini Mori, and Kurt Stockinger. 2012. SODA: Generating SQL for Business Users. *PVLDB* 5, 10 (2012), 932–943.

[7] DongHyun Choi, Myeong Cheol Shin, EungGyun Kim, and Dong Ryeol Shin. 2020. RYANSQL: Recursively Applying Sketch-based Slot Fillings for Complex Text-to-SQL in Cross-Domain Databases. arXiv:cs.CL/2004.03125

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:cs.CL/1810.04805

[9] Bolin Ding, Jeffrey Xu Yu, Shan Wang, Lu Qin, Xiao Zhang, and Xuemin Lin. 2007. Finding top-k min-cost connected trees in databases. In *ICDE*. 836–845.

[10] Li Dong and Mirella Lapata. 2016. Language to Logical Form with Neural Attention. arXiv:cs.CL/1601.01280

[11] Li Dong and Mirella Lapata. 2018. Coarse-to-Fine Decoding for Neural Semantic Parsing. arXiv:cs.CL/1805.04793

[12] Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation. arXiv:cs.CL/1905.08205

[13] Hao He, Haixun Wang, Jun Yang, and Philip S Yu. 2007. BLINKS: ranked keyword searches on graphs. In *ACM SIGMOD* . ACM, 305–316.

[14] Vagelis Hristidis, Luis Gravano, and Yannis Papakonstantinou. 2003. Efficient IR-style Keyword Search over Relational Databases. In *VLDB*. 850–861.

[15] Wonseok Hwang, Jinyeong Yim, Seunghyun Park, and Minjoon Seo. 2019. A Comprehensive Exploration on WikiSQL with Table-Aware Word Contextualization. arXiv:cs.CL/1902.01069

[16] Mehdi Kargar, Aijun An, Nick Cercone, Parke Godfrey, Jaroslaw Szlichta, and Xiaohui Yu. 2014. *MeanKS: Meaningful Keyword Search in Relational Databases with Complex Schema*. ACM. http://ceur-ws.org/Vol-1912/paper20.pdf

[17] Fei Li and H. V. Jagadish. 2014. Constructing an Interactive Natural Language Interface for Relational Databases. *PVLDB* 8, 1 (Sept. 2014), 73–84.

[18] Yunyao Li and Davood Rafiei. 2017. Natural Language Data Management and Interfaces: Recent Development and Open Challenges. In *ACM SIGMOD*. 1765–1770.

[19] Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020. Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Online, 4870–4888. https://doi.org/10.18653/v1/2020.findings-emnlp.438

[20] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-Task Deep Neural Networks for Natural Language Understanding. arXiv:cs.CL/1901.11504

[21] Yi Luo, Xuemin Lin, Wei Wang, and Xiaofang Zhou. 2007. Spark: Top-k Keyword Query in Relational Databases. In *ACM SIGMOD*. 115–126.

[22] Yi Luo, Wei Wang, Xuemin Lin, Xiaofang Zhou, Jianmin Wang, and Keqiu Li. 2011. SPARK2: Top-k Keyword Query in Relational Databases. *IEEE Trans. Knowl. Data Eng.* 23, 12 (2011), 1763–1780. https://doi.org/10.1109/TKDE.2011.60

[23] Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, and Zheng Chen. 2020. Hybrid Ranking Network for Text-to-SQL. arXiv:cs.CL/2008.04759

[24] Jianqiang Ma, Zeyu Yan, Shuai Pang, Yang Zhang, and Jianping Shen. 2020. Mention Extraction and Linking for SQL Query Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 6936–6942. https://doi.org/10.18653/v1/2020.emnlp-main.563

[25] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:cs.CL/1301.3781

[26] Amihai Motro. 1986. Constructing Queries from Tokens. In *ACM SIGMOD*. 120–131. https://doi.org/10.1145/16894.16866

[27] Notes on Ambiguity [n.d.]. Notes on Ambiguity. http://bit.ly/2YTLFeR.

[28] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[29] Diptikalyan Saha, Avrilia Floratou, Karthik Sankaranarayanan, Umar Farooq Minhas, Ashish R. Mittal, Fatma Özcan, IBM Research. Bangalore, and IBM Research. Almaden. 2016. *ATHENA: An Ontology-Driven System for Natural Language Querying over Relational Data Stores*. VLDB. http://www.vldb.org/pvldb/vol9/p1209-saha.pdf

[30] Alkis Simitsis, Georgia Koutrika, and Yannis Ioannidis. 2008. Précis: from unstructured keywords as queries to structured databases as answers. *The VLDB Journal* 17, 1 (2008), 117–149.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. arXiv:cs.CL/1706.03762

[32] Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. arXiv:cs.CL/1911.04942

[33] Nathaniel Weir, Prasetya Utama, Alex Galakatos, Andrew Crotty, Amir Ilkhechi, Shekar Ramaswamy, Rohin Bhushan, Nadja Geisler, Benjamin Hättasch, Steffen Eger, Ugur Çetintemel, and Carsten Binnig. 2020. DBPal: A Fully Pluggable NL2SQL Training Pipeline. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, David Maier, Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo (Eds.). ACM, 2347–2361.

[34] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv:cs.CL/1609.08144

[35] Xiaojun Xu, Chang Liu, and Dawn Song. 2017. SQLNet: Generating Structured Queries From Natural Language Without Reinforcement Learning. arXiv:cs.CL/1711.04436

[36] Pengcheng Yin and Graham Neubig. 2017. A Syntactic Neural Model for General-Purpose Code Generation. arXiv:cs.CL/1704.01696

[37] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2019. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. arXiv:cs.CL/1809.08887

[38] Zhong Zeng, Mong Li Lee, and Tok Wang Ling. 2016. Answering Keyword Queries involving Aggregates and GROUPBY on Relational Databases. *EDBT* (2016), 161–172.

[39] Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. arXiv:cs.CL/1709.00103