Discovering and Integrating Tabular Data

Davood Rafiei[1,*,†], Arash Dargahi Nobari[1,†] and Soroush Omidvartehrani[1,†]

[1]University of Alberta

Abstract

Tabular data are made available from different sources and this data can be used for exploratory analysis or be integrated with other data sources. However, searching for such data and integrating tables from different sources with little information about the schema has been a challenge. Our vision is to develop tools and algorithms to make this process easier. This paper presents some of our recent in this direction.

1. A Scenario

A business intelligence specialist at a bank is in charge of collecting data on local businesses to assess their market value, which is used for loan approvals. There are two primary sources of information available to the specialist: property assessments and company profile databases. Property assessments (referred to as A) are made available, as part of open government data, by the municipal government, which conducts these assessments annually for tax purposes. In addition to property assessments, the specialist can access two company profile databases that provide different information about the companies. One database contains details about the properties owned by each company (referred to as P), while the other focuses on stock valuations (V). The stock valuations include the company symbol, number of shares outstanding, and the share price among other information. The specialist's task is to gather the relevant records from these sources and present them in a clean and concise format for the loan approval officers. By consolidating the property assessment data, company property information, and stock valuations, the specialist creates a comprehensive overview of each business, enabling more informed decision-making during the loan approval process.

2. Challenges

The task involves a number of challenges. Firstly, the open government data consists of an extensive collection of tables, numbering in the tens of thousands. To navigate through this vast amount of data, the specialist requires appropriate search tools to identify the relevant tables and records. The data does not provide the names of the businesses associated with each property. Hence searches cannot be done based on business names. Also, searches based on city names or postal codes may yield many non-relevant tables and rows, making the search process more complex. Secondly, there is a discrepancy in the formatting of property addresses between the open government data (A) and the company profile data (P). Consequently, performing an equi-join of A and P will not work. The specialist faces the challenge of reconciling these differences in order to effectively match the relevant records. Thirdly, the company valuation data (V) includes the stock symbol of each company, rather than the complete company name. This poses difficulties when attempting to join the company names in P with the stock symbols in V. Bridging this gap is a necessary but not straightforward task in data integration. Lastly, data is subject to change over time, and the specialist's responsibilities extend beyond a one-time process. Regular and periodic reviews of the data are necessary to ensure its accuracy and relevance. The specialist must be prepared to undertake this recurring task to maintain up-to-date and reliable information.

By addressing these challenges, the specialist can streamline the data consolidation process and provide loan approval officers with accurate and comprehensive information consistently. We next review some of our work that tackle these challenges.

3. Table Search and Querying

We have developed Bare Table Query Language (in short BareTQL) an interactive framework for querying large collections of tables [1]. Compared to table search approaches in the literature (e.g. [2]), BareTQL offers some distinctive features including (1) the composability and interoperability of operations with little reliance on the schema information of the tables being queried, and (2) the ability to customize search and transform tables. This is achieved by providing a set of algebraic operators over a table collection when little is known or can be assumed about the underlying table schemes. BareTQL achieves this by taking an exploratory approach to search with a

Joint Workshops at 49th International Conference on Very Large Data Bases (VLDBW'23) — TaDA'23: Tabular Data Analysis Workshop, August 28 - September 1, 2023, Vancouver, Canada

*Corresponding author.

drafiei@ualberta.ca (D. Rafiei); dargahi@ualberta.ca (A. Dargahi Nobari); s.omidvartehrani@ualberta.ca (S. Omidvartehrani)

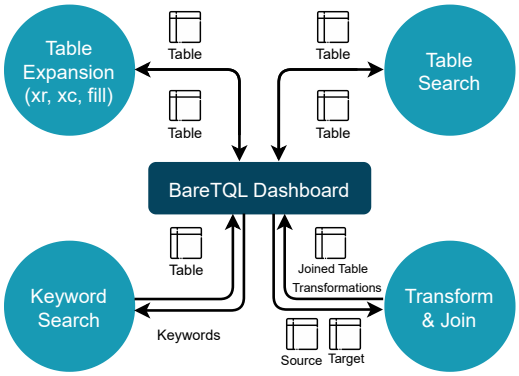

Figure 1: BareTQL operations

focus on what is known already and building on top.

BareTQL supports four classes of operations as shown in Figure 1. A *keyword search* may be used when there is little information about the tables being queried and their structures, whereas a *table search* may be invoked when the user has a table and wants to find more related tables. Through a set of table expansion operators, one can expand a table by adding more rows (xr), adding more columns (xc) and filling missing values (fill). These operations may invoke table search to find the relevant tables, but the relevance ranking can vary for each table expansion operator. In our given scenario, the specialist may search open government data for tables using keywords from property addresses, and filter the results using follow-up operations. The specialist may also provide some examples of addresses and their valuations to find more a more comprehensive list. BareTQL supports joining tables from different sources through the use of transformations. We next discuss our work on automatically finding such transformations.

4. Transforming Tables for Join

When tables are gathered from multiple sources, they rarely conform to the same formatting. We study the problem of transforming tables for joinability under two settings: (1) each transformation is expressed as a sequence of string operations, and (2) transformations are learned in a latent space. Unlike previous works relying on similarity functions for matching (e.g., [3, 4]), learning transformations offer greater insight into formatting differences and broader application possibilities.

4.1. Mappings through string operations

Consider joining tables in sources A and P in our scenario based on property addresses. An arbitrary address in A can be formatted as "12345 78 Ave NW, Apt#202", while the same address in P is represented as "202-12345 78 avenue, NW." We need to transform one formatting to the other before a join. Each transformation may be represented as a sequence of basic string operations such as *substr*, *split* and *splitSubstr*, and the search for a transformation may be conducted by searching over the set of possible transformations and their parameters. The search space for possible transformations grows exponentially with the number of basic operations and the parameters of those operations. In our Common String-based Transformer (CST) [5], the search space is constrained based on common text sequences that are observed in source and target tables. Thanks to its efficient search algorithm, CST improves upon competitive approaches such as Auto-join [6] by a few orders of magnitude in running time.

While string-based transformations are human-readable and interpretable, there have two general limitations: (1) these transformations are syntactic and they usually miss semantic mappings (e.g. synonyms) between sources, (2) learning transformations is a resource-intensive process. Our next work attempts to address these challenges.

4.2. Transformations in a latent space

Consider our scenario again where companies in P are identified by their names, while the same companies in V are identified by their stock symbols. Transforming company names to their stock symbols using string operations is less trivial or meaningful. We have been recently studying the power of transformer models in mapping tabular data. In particular, our Deep Tabular Transformer (DTT) framework [7] learns string-based transformations in a latent space. To start with a basic language understanding, a large language model is adopted for the problem and transformations are learned on top. To train such a model though, one usually needs a large set of labeled data. Our study shows that string transformations can be learned from synthetic data which can be generated in large volumes. DTT employs a decomposer and an aggregator module to deal with disparity in table sizes and to handle large tables.

Although DTT is not fine-tuned on real-world data, our experiments indicate that it delivers outstanding performance on both real-world and synthetic datasets, compared to state-of-the-art baseline methods. Furthermore, the DTT framework is complex enough to observe various patterns in the data, while being small enough to efficiently run on a single GPU.

We are studying ways of improving our mapping functions while keeping them human-readable and interpretable.

References

[1] D. Rafiei, H. Fah, T. Lafrance, A. Dargahi Nobari, Baretql: An interactive system for searching and extraction of open data tables, in: 27th International Conference on Intelligent User Interfaces, 2022, pp. 30–33.

[2] O. Lehmberg, D. Ritze, P. Ristoski, R. Meusel, H. Paulheim, C. Bizer, The mannheim search join engine, Journal of Web Semantics 35 (2015) 159–166.

[3] S. Chaudhuri, K. Ganjam, V. Ganti, R. Motwani, Robust and efficient fuzzy match for online data cleaning, in: Proceedings of the 2003 ACM SIGMOD international conference on Management of data, 2003, pp. 313–324.

[4] J. Wang, G. Li, J. Fe, Fast-join: An efficient method for fuzzy token matching based string similarity join, in: 2011 IEEE 27th International Conference on Data Engineering, IEEE, 2011, pp. 458–469.

[5] A. D. Nobari, D. Rafiei, Efficiently transforming tables for joinability, in: 2022 IEEE 38th International Conference on Data Engineering (ICDE), IEEE, 2022, pp. 1649–1661.

[6] E. Zhu, Y. He, S. Chaudhuri, Auto-join: Joining tables by leveraging transformations, Proceedings of the VLDB Endowment 10 (2017) 1034–1045.

[7] A. D. Nobari, D. Rafiei, Dtt: An example-driven tabular transformer by leveraging large language models, 2023. arXiv:2303.06748.