Università di Genova | DIPARTIMENTO INFORMATICA, BIOINGEGNERIA, ROBOTICA E INGEGNERIA DEI SISTEMI

Computer Science Workshop
PhD program in Computer Science and Systems Engineering

Table Augmentation in Data Lakes

Federico Dassereto, Giovanna Guerrini

Relational-like operations at large scale

Introduction

Data Lakes are large repositories of both structured and unstructured data. A great part of such repositories is made of tables without any schema information [1]. These tables are extremely valuable due to their origin, since enterprises and administrative offices publish them daily. The lack of a common schema for the tables inside Data Lakes makes it difficult to efficiently perform traditional data management operations, such as joins over different tables. In our work, we focus on a discovery scenario. Among the possible discovery operations, we recognize as relevant joinability, unionability and augmentation.

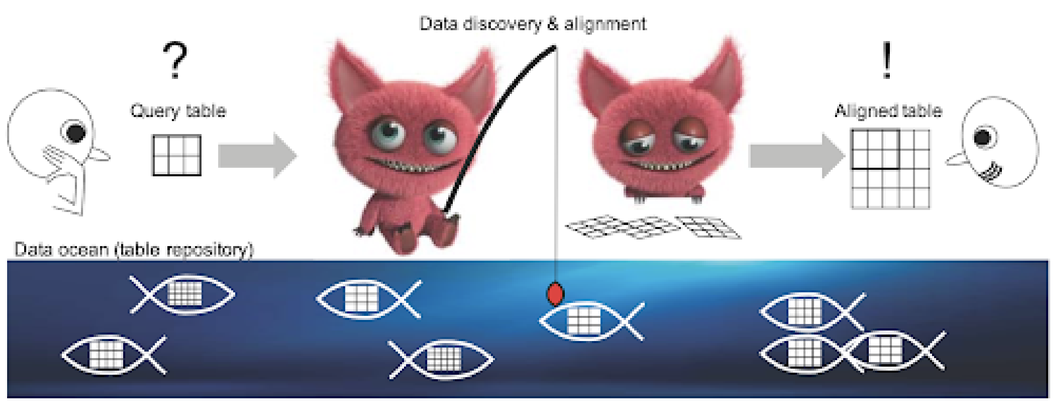

Figure 1: Table-as-a-query paradigm

Existing approaches only focus on retrieve tables whose columns are the best option for joinability, without considering the amount of information that can be added but actually materializing the join [3, 4].

Key Concepts

- Table-as-a-Query: paradigm for data lake exploration, in which the query is a complete table rather than a string representation
- Data Lake: large repository of tables sharing no schema information
- Joinability Search: Searching of tables that can be horizontally concatenated; the two (or more) tables must share a common column
- Unionability Search: Searching of tables that can be vertically concatenated; the two (or more) tables must share a common schema
- Augmentation: A more specific joinability search, in which the most relevant columns are not the most overlapping but the ones *augmenting the information the most* concerning a Machine Learning task such as Classification on Regression

The Problem

Given a data lake $D = (T_1, ..., T_n)$, a query table Q, a join column $Q.J$, a target column $Q.T$, the goal is to provide a ranking of tables $C_i \in D$ such that:

1. $\exists j \in C_i$ s.t. $Q.J \tilde{\bowtie} C_i.j$
2. $\exists C_a \in C_i$ s.t. $corr(C_a, Q.T)$

Where $\tilde{\bowtie}$ denotes a fuzzy join over sets and $corr(\cdot, \cdot)$ denotes a positive correlation among two columns and represents the augmentation of information. The expected output is a ranking $R = (C_1, ..., C_k)$ in which tables are ordered by *augmentation of information* obtained by probing an *index structure*.

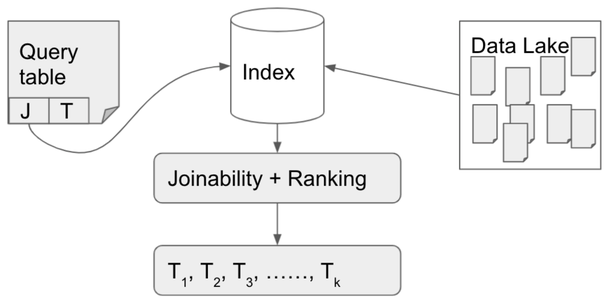

Figure 2: Overview of the schema to solve the problem

Design choices

The ideal solution to quantify the augmentation would be checking every possible pair of columns. Unfortunately, this approach is inefficient both in terms of space and execution time, due to the size of data lakes. Since computing every possible pairs is unfeasible, we need to approximate the measures, by relying on an indexing structure based on a family of hashing functions called learning2hash [2].

- The index store information in a columnar way, by representing each column with a single hash signature
- The index allows for a fast retrieval of candidate tables joining with the join column of the query table
- Probing the index results in a ranking of tables order by augmentation of information, where the augmentation is quantified in terms of information theoretic measurements
- We allow for one-to-one joins
- Joins must be performed among columns that act as key in their tables

Q

J	T
Genoa	1
Berlin	1
Boston	0
NY	0
Rome	1

D_1

J'	C_i
Genoa	Italy
Berlin	Germany
Boston	USA
NY	USA
Rome	Italy

D_2

J'	C_i
Genoa	580k
Berlin	3.6M
Boston	680k
NY	8.4M
Rome	2.8M

$$H(Q.T \mid D_1.C_i) < H(Q.T \mid D_2.C_i)$$

Figure 3: A Query table and two joinable candidates. The comparison among the columns of the candidates and $Q.T$ is unfeasible

The figure below shows an example of two tables, one more relevant than the other. Table D_1 augments the information the most, Since $D_1.C_i$ helps predict whether the cities in the join column $Q.J$ are in Europe or outside. The augmentation is here expressed in terms of conditional entropy, where a lower value represents a better augmentation. Intuitively, it represents the reduction of uncertainty of Y when knowing X.

$$H(Y|X) = -\sum_{x \in X, y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

Forthcoming Research

- Implementation of ad-hoc hashing functions to preserve and capture similarities in columns, depending on their types
- A priori pruning on joinability search and augmentation evaluation to speed up the execution time
- Extension to different kind of joins, one-to-many and many-to-many

References

[1] Renée J. Miller. Open data integration. *Proc. VLDB Endow.*, 11(12):2130–2139, 2018.

[2] Sean Moran. Awesome papers on learning to hash. `https://learning2hash.github.io`, 2017.

[3] Erkang Zhu, Dong Deng, Fatemeh Nargesian, and Renée J Miller. Josie: Overlap set similarity search for finding joinable tables in data lakes. In *Proceedings of the 2019 International Conference on Management of Data*, 2019.

[4] Erkang Zhu, Fatemeh Nargesian, Ken Q Pu, and Renée J Miller. Lsh ensemble: internet-scale domain search. *Proceedings of the VLDB Endowment*, 9(12):1185–1196, 2016.

CONTACTS

Federico Dassereto
federico.dassereto@edu.unige.it

Giovanna Guerrini
giovanna.guerrini@unige.it