Extending RapidMiner with Data Search and Integration Capabilities

Anna Lisa Gentile[1], Sabrina Kirstein[2], Heiko Paulheim[1], and Christian Bizer[1]

[1] University of Mannheim, Germany
[2] RapidMiner, Germany
annalisa@informatik.uni-mannheim.de,skirstein@rapidminer.com,
heiko@informatik.uni-mannheim.de,chris@informatik.uni-mannheim.de

Abstract. Analysts are increasingly confronted with the situation that data which they need for a data mining project exists somewhere on the Web or in an organization's intranet but they are unable to find it. The data mining tools that are currently available on the market offer a wide range of powerful data mining methods but hardly support analysts in searching for suitable data as well as in integrating data from multiple sources. This demo shows an extension to RapidMiner, a popular data mining framework, which enables analysts to search for relevant datasets and integrate discovered data with data that they already know. In particular, we support the iterative extension of data tables with additional attributes. We will demonstrate the usage of the extension with a large corpus of tabular data extracted from Wikipedia.

1 Introduction

The amount of data which is available within organizations and on the public Web has rapidly increased in recent years. Due to the large number of data sources, analysts face the challenge of collecting the data they need, which, although largely available, are difficult to find and integrate. The focus of many data mining projects is therefore increasingly shifting from the actual data analysis to the search for data which is suitable for a particular mining task, as well as their integration. The data mining tools currently available on the market offer analysts a wide variety of high-performance analytical methods, but they hardly support analysts in finding and integrating data. On the other hand, there are data management solutions such as Google Fusion Tables[1] or Microsoft Power Query for Excel[2] which provide for searching relevant data as well as for the manual integration of data from multiple sources. The major drawback for both solutions is that they are not full data mining frameworks and, while offering some data analysis facilities and dashboards, they are not originally intended as data mining tools and therefore lack advanced analytical methods.

[1] https://goo.gl/8uD4OB
[2] https://goo.gl/Cj0Fnr

In this demo, we show our efforts towards filling this gap by designing and implementing data search and data integration functionalities within RapidMiner[3], in order to assist the user in the process of finding relevant datasets for a given data mining task as well as in integrating newly discovered data with data that the user already knows. Our RapidMiner extension implements *SearchJoins*, i.e., a combined operation of searching and joining tables, similar to [1,3,5]. Starting with an initial *query table* and a *extension attribute* name (a column that the user wants to add to the table) our framework (i) retrieves *relevant tables* from previously indexed data sources and (ii) determines schema and instance correspondences between the *query table* and the retrieved *relevant tables*. The discovered tables and the correspondences between these tables and the query table are presented with a novel user interface within RapidMiner, which allows the user to iteratively correct and refine the results.

The main advantages from a user perspective are (i) the transparent search for relevant (table) data over multiple data sources, both on intranet and on the Web; (ii) the integration of retrieved *relevant tables* to the *query table* and (iii) the possibility to visually correct integration results and iterate the process.

In the following we first describe the demo from a user perspective as well as the dataset that we use for the demo (Section 2). We then provide a general description of our *SearchJoin* framework (Section 3).

2 SearchJoin: Demo and Dataset

The *SearchJoin* framework has two main steps: (i) Data Search, to find records that further describe an initial set of given entities; (ii) Data Integration, to interactively extend local tables with additional attributes based on the retrieved data records. During the demo, we will show how to perform a *SearchJoin* within RapidMiner. The user will be able to select a table she wants to enrich, e.g. a set of countries with country names and their capital and she wants to find out the population for each of them.

As table corpus for the *SearchJoin*, we will use a dataset originally consisting of 1.35 million Wikipedia tables, extracted in the WikiTables project [2]. After applying heuristics to detect useful tables (e.g. (i) it is a relational table, (ii) a subject column is detected, (iii) it contains at least 3 columns and 5 rows...) we retain and index 541K tables. Table 1 shows some statistics of the dataset. The dataset is publicly available for download as json files[4].

The user will be free to perform arbitrary queries, i.e. choose any table from the corpus that she wants to enrich and specify the desired expansion attributes. We will also provide a ranked list of frequent classes and headers in the dataset to facilitate the formulation of the query.

[3] RapidMiner (`https://rapidminer.com/`) is recognised as a leader for Advanced Analytics Platforms (`http://goo.gl/Qttrqb`).

[4] All demo material, including a screencast of the demo, can be found at `http://goo.gl/6MPHl0`.

Table 1. Dataset statistics.

Number of tables	541,611
Total number of cell values	56,126,735
...of which string	32,931,915
...of which numeric	18,285,894
...of which date	4,908,539
...of which link	387
Number of cells identified as subjects	9,911,733
Number of headers	158,245

3 SearchJoin: A Table Extension Solution

3.1 SearchJoin: The Framework

The SearchJoin framework is specifically designed to retrieve tables that extend an initial user-provided *query table* and complement it with additional information when available. The search space can span to multiple and diverse sources of tabular data, coming from the intranet or from the Web. These are pre-processed offline, as a batch process, by the *TableIndexer* component. At the current status, the *TableIndexer* gets as input a *TableDataset*, a collection of tables $TD = t_1, t_2, ..., t_n$ and (i) detects if t_i is useful (we use a set of heuristics to discard non relational tables), (ii) pre-processes and semantically enriches all useful tables and (iii) stores the information in two Lucene[5] indexes, one for headers, one for values (to allow fast independent searches on headers or values).

In the pre-processing phase we reuse components from the T2K framework [4] to: (i) identify pseudo key attributes (the subject column); (ii) recognize table header structures; and (iii) identify data types. To identify the subject column we apply the heuristic proposed by [4] of choosing the column of type string with the highest number of unique values (in case of a tie, the left-most column is used). For detecting the headers, we currently assume that the header row is the first non-empty entity. For data type detection we use about 100 manually defined regular expressions to detect string, numeric value, date and link.

The *TableSearch* component incorporates searching and integration facilities. *TableSearch* receives as input a *query table*, where the user indicates which one is the subject column (the column in the table that contains entity names), and a set of extraction attributes (that she wants to complement her table with). It (i) retrieves a set of relevant tables from the connected indexes of tables (pre-compiled by the *TableIndexer*) and (ii) identifies correspondences between those and the query table, both at schema level (which columns in each retrieved table correspond to which columns in the query table) and at instance level (which rows in each retrieved table correspond to which rows the query table). As output *TableSearch* returns all the identified correspondences and a confidence score for each matching.

[5] `https://lucene.apache.org`

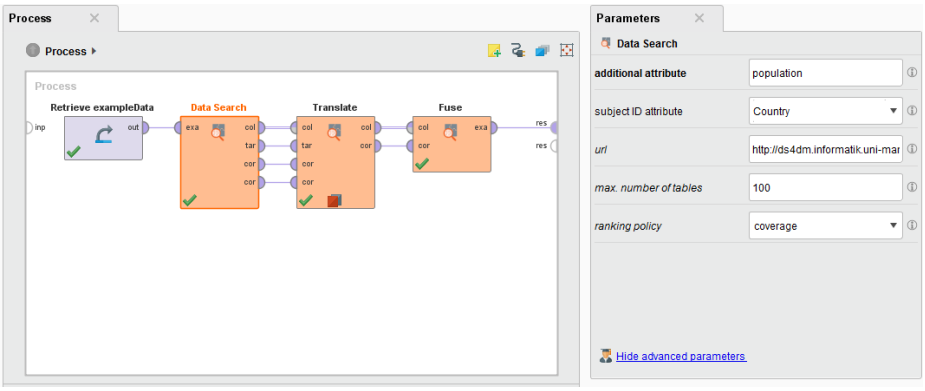

Fig. 1. RapidMiner operators *Data Search*, *Translate* and *Fuse* used to search for new attributes.

3.2 SearchJoin: The RapidMiner Extension

TableSearch is provided as a RESTful service, for which a user interface is provided in RapidMiner as an extension, which is named the *Data Search Extension*. RapidMiner uses so called operators (building blocks for operations on data) to read data, transform data, learn models, apply models and write data and enables business users to build predictive analytics processes in a visual environment. Extensions can define new operators to provide additional functionality.

The new *Data Search Extension* adds three new operators to RapidMiner. The first operator, called *Data Search*, sends a request to *TableSearch* (cf. 3.1). The server response contains the target schema, names of relevant tables, and correspondences between those tables and the query table. The *Data Search* operator fetches each of those tables from the *TableSearch* Engine. The output of the operator is a collection of the relevant tables, as well as the target schema and the correspondences, at schema and at instance level.

The operator *Translate* receives the outputs from the *Data Search* operator and translates each of the relevant tables into the target schema. It delivers a collection of tables in form of the target schema, filled with the corresponding values from the relevant tables that were fetched from the *TableSearch* Engine. The operator *Fuse* uses a pre-defined order of fusion policies to decide for each row of the target schema, which value given from the relevant tables becomes the final value of this row. The result is the user *query table*, extended by an additional attribute. Figure 1 shows a RapidMiner process with the new operators *Data Search*, *Translate* and *Fuse* to add a new attribute.

4 Conclusions and Future Work

In this work, we present a data search and integration framework suitable for data coming from diverse data sources, being local storage, a company intranet

or the Web. The framework is implemented as an Open Source RapidMiner extension. For demonstration purposes, we provide a 541K dataset of tables, but the approach can be applied to any collection of tabular data.

In this prototype the integration of tables is based on strategies from [3] while the retrieval of relevant tables and extension attributes is mainly relying on keyword matching. As future work, we plan to develop a data search method that does not require the extension attributes to be known, but which makes it possible to search the attributes, e.g., based on their correlation with existing local attributes.

Acknowledgments

This paper describes the first public demo of the project DS4DM (Data Search for Data Mining) `http://ds4dm.de`, funded by the German Federal Ministry of Education and Research (BMBF).

References

1. S. Balakrishnan, A. Halevy, B. Harb, H. Lee, J. Madhavan, A. Rostamizadeh, W. Shen, K. Wilder, F. Wu, and C. Yu. Applying WebTables in Practice. *Cidr 2015*, 2015.
2. C. S. Bhagavatula, T. Noraset, and D. Downey. Methods for exploring and mining tables on Wikipedia. *Proc. of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics - IDEA '13*, pages 18–26, 2013.
3. O. Lehmberg, D. Ritze, P. Ristoski, R. Meusel, H. Paulheim, and C. Bizer. The mannheim search join engine. *Web Semantics: Science, Services and Agents on the World Wide Web*, 35, Part 3:159 – 166, 2015. Semantic Web Challenge 2014.
4. D. Ritze, O. Lehmberg, and C. Bizer. Matching html tables to dbpedia. In *Proc. of the 5th International Conference on Web Intelligence, Mining and Semantics*, WIMS '15, pages 10:1–10:6, New York, NY, USA, 2015. ACM.
5. M. Yakout, K. Ganjam, K. Chakrabarti, and S. Chaudhuri. Infogather: Entity augmentation and attribute discovery by holistic matching with web tables. In *Proc. of ACM SIGMOD '12*, pages 97–108, New York, NY, USA, 2012. ACM.