Exposé for diploma thesis

Similarity Search on Tabular Data

Anne Isberner
isberner@informatik.hu-berlin.de

Supervision: Prof. Dr. Ulf Leser

March 11, 2016

Motivation

Tables are commonly used to display multidimensional data in an easy to grasp format. They are used in many different contexts, either by themselves or embedded within full text documents. Tabular data is useful in almost every domain and holds vast amounts of information that remains mostly separated and unconnected from other data sources.

While some tables are structured and annotated similar to tables used in relational databases - and are thereby relatively easy to query for specific results - there are many that are not. Quite often, tabular data lacks labels or identifiers, or they cannot be used to uniquely identify a referenced entity. Many tables are therefore not searchable and finding similar data relies on querying the enclosed text for indicators, even though its connection to the table's data may not be obvious.

Additionally, many authors create different tables on the same base data, either unaware of or deliberately differing from existing alternative visualizations. Finding these different versions, similar tables or the same data in other tables is the goal that this work aims for.

There are multiple interpretations of what a *similar* table actually is. This work will focus on similarity based on both content and structure of a given table's columns. Similarity based on sharing the same subject or topic, based on content alone, has already been examined using text clustering[2] or inclusion dependencies[3]. These works have both influenced this work greatly (see Related Work).

Related Work

So far there have been successful implementations of document based similarity search that yield performances similar to that of a human evaluator[1], but the special structure of tabular data and the mostly unannotated cell contents that oftentimes consist of only a single number or a few words make it especially difficult to apply these traditional text analysis practices to tables.

Clustering of tables in text documents

A recent bachelor's thesis by Irina Glushanok[2] explored the possibility of clustering tables in scientific full text documents. In this approach, the table cells were serialized and interpreted as plain text, tokenized and then lemmatized. Numeric values, arithmetic expressions and empty values were discarded as was the structural data implied by the column and row relationships. Her research suggests that one may use well known

clustering algorithms such as k-means, but would most likely get better results when considering not only the textual contents and contextual information - such as document paragraphs referencing said table - but also deducing from other types of cell contents, e.g. numeric values, as well as the structural information of the table itself.

Web of Tables based on Inclusion Dependencies

Identifying inclusion dependencies between a large amount of tables has been implemented using the *Many*-Algorithm by Fabian Tschirschnitz[3]. The objective was to find possible joins between tables gathered from various online resources and therefore allow the user to visually explore connections between different data sets. The approach is based on first creating a storage efficient representation of table columns that may then be used to quickly determine the existence of inclusion dependencies between different tables' columns.

While this algorithm finds connections between multiple tables based on containment of values in columns, we are interested in finding *similar* columns, including those whose values differ slightly and are not actual inclusion dependencies. Two columns that contain mutually exclusive values from the same topic or vocabulary are still similar, even though they do not qualify as an inclusion dependency. Many approaches used in this work regarding sanitization and the concept of creating column signatures may prove to also be useful for our problem.

TableSeer

TableSeer is an "automatic table extraction and search engine system"[4] proposed by Ying Liu et. al. They describe the detection and extraction of tables from scientific documents and offer search tools to the end user. The results of such a table search are ranked by a custom *TableRank* using a vector space model which considers the *Table Term Frequency - Inverse Table Term Frequency* (TTF-ITTF) based on the table's metadata instead of the *Term Frequency - Inverse Document Frequency* (TF-IDF) approach based on the whole document. Using this yields search results that are based on the table itself instead of the whole document.

Schema matching

Schema matching is the "task of identifying semantically same or similar elements in two different schemas"[5]. Even though we do not have an explicit schema, when we consider a table as a data set, each table column as a schema entry, and each row as a piece of instance data, the concept of schema matching also applies to our problem. Finding matching schema entries (i.e. columns) in other tables therefore indicates structural similarity between those tables. This is especially important on columns that do not offer many other aspects for similarity, such as numeric columns without headers.

Specifically instance-level approaches like the ones used by Match[6] are valuable to this work, such as *constraint-based characterization* of column values using value ranges or patterns. We will use this technique to create column signatures and simplify finding other structurally similar columns.

Another approach is *horizontal matching*[5] using duplicate or very similar data within multiple tables' instance data. One of the requirements is that some data rows exist in both of the compared instances and therefore allows to match their column schemas. This works well even without a given schema, such as is our case, however we will primarily focus on a column-based solution.

Goals

The goal of this work is to create an algorithm to retrieve a ranked list of tables from our corpus that are similar to a given table or column. The ranking itself should be based on contents, context as well as the table's structure and therefore utilize as much information as possible.

Methodology

We will use the same data corpus as Glushanok[2] which contains tables extracted from scientific documents. The tables are available as parts of structured XML documents and will first be preprocessed and sanitized regarding both structure and data. Afterwards the columns of every table will be analysed and given a structural signature. These signatures will be compared to the given tables' column signatures in order to find similarities. Other tables with similar columns will then be ranked according to the number of similar columns and the similar columns' contents. The tables' contexts, especially the referencing paragraph(s), will also be added to the inspected data. The result is a list of tables ranked by similarity which will be returned to the user.

Preprocessing

All tables must be preprocessed and sanitized to increase their comparability. This has to be applied to both structure as well as the contents, since both components are relevant for the algorithm.

Structural Preprocessing

Any column (or row) group, i.e. one cell spanning multiple columns (or rows) must be expanded by duplicating the spanned content across all affected columns. The resulting cells may then also be collapsed again. Combining cells like this leads to a unified representation of data.

Apartment	**Size**	
	m2	**sq ft**
Apt. 1	25	269
Apt. 2	75	807

Apartment	**Size m2**	**Size sq ft**
Apt. 1	25	269
Apt. 2	75	807

Table with multi-row and multi-column cells The same table with unified cells.

Tables that only contain labels in the first column instead of a header row should be transposed, to create a uniform structure where each row may be interpreted as a data set and each column represents a data attribute. However, detecting this state is not trivial but may be deduced from small hints such as text style, e.g. bolded text, or cell content, such as a lone text cell in an otherwise numeric row.

Apartment	Apt. 1	Apt. 2
Size qm	25	75
Size sq ft	269	807

Incorrectly formatted table that needs to be transposed

In general, cells in first row or first column that differ strongly from the type of the remaining cells may indicate a header row. These headers should also be separated from the data itself and instead be treated as

metadata on the respective columns.

Since neither the order of rows nor the order of columns are considered relevant with regards to table similarity, no sorting is applied.

Data Preprocessing

The data found in each table cell will also be preprocessed. Values that represent an empty or null value (e.g. `0`, `0.0`, `null`, `""`, `-`, `n/a`) will be unified into a single representative, while also trying to identify numeric values. This is no trivial task since scientific notation (e.g. `4×10^3`, `4 * 10^3`, `4e3`), written numbers (e.g. `one`, `eleven`), fractions (`3/4`, `¾`), US decimal (e.g. `1,234.56`) and european decimal (`1.234,56`) formats all need to be recognized and normalized. Similarly, string values should also be normalized. There are many possibilities for this, some have already successfully been implemented, such as dictionary-based lemmatization[2]. Additionally, the use of domain specific dictionaries to map synonyms, stemming and filtering stop words may be considered. Since the corpus only includes sources written in English, this approach to text preprocessing should yield comparable results.

Similarity Comparison

Determining the similarity of two tables will be a two-step process: First, based on column structure, and then based on the actual cell contents and the table's context. For each comparable table pair, the *similarity* between every column pair needs to be investigated since columns in similar tables aren't necessarily in the same order.

The normalized tables' columns need to be inspected separately and every column will be assigned a feature set based on its properties, possibly similar to the Bloom filter signature used in Web Tables[3]. However, these will focus only on the metatadata of the column, for example its column header if available, its prominent data type (integer, decimal, string or null), value range (for numeric values) or length range (for other types), value distribution or spread, its size (row count) and possibly others. Each value may be weighted differently to fit the user's search intention, e.g. a larger table containing many rows might still be considered similar to a smaller one, as long as its data is similar. Using these feature sets allows us to determine the structural similarity between two columns using a suitable measure depending on the selected features.

Another important aspect besides the column structure is the column's contents. For columns containing primarily string values, comparing the columns' cell values for textual similarity is essential. Since the order of rows is not relevant to our similarity detection, using a bag of words approach is an effective and well proven solution. For numeric columns, a different approach is needed. However, one could assume that primarily numeric columns that are of a similar structure will also contain similar contents, therefore allowing us to skip this step.

Lastly, we will investigate the table's context. Since this applies equally to all columns, indifferent of the column's primary content type, it is necessary to group the similar columns by their tables. The tables' context similarity will then be evaluated using a bag of words approach.

The final score is then calculated by combining the resulting similarity values for column structure, column content and table context into a single value.

Evaluation

Since similarity is a subjective measure and depends strongly on the user's search goals, the evaluation will be based on multiple selected samples. These should span a wide range of use cases, such as queries using table

inputs and queries using column inputs. The retrieved ranked lists will be inspected regarding the similarity on both the column level as well as the table level and marked as either (closely) similar, dissimilar or unknown. Since the table similarity is strongly influenced by its column similarities, the main focus will lie on the column similarity. Different feature weights and feature selections will be tested regarding structural similary. Sanitization approaches for both textual and numeric cell values will most likely also produce different results and should be examined closely. Comparing these samples to results from other approaches is also a possibility. It should be possible to apply established text analysis to the table's containing document and determine whether our approach yields more accurate or useful, or additional results not found using those techniques.

---

1. Michael Lee, Brandon Pincombe, Matthew Welsh, 2005. An empirical evaluation of models of text document similarity. *Cognitive Science.*↩

2. Irina Glushanok, 2015. Semantische Analyse von Tabellen in Volltexten. Bachelor's Thesis, Humboldt University of Berlin.↩

3. Fabian Tschirschnitz, 2014. Spinning a Web of Tables through Inclusion Dependencies. Master's Thesis, Hasso Plattner Institute.↩

4. Ying Liu, Kun Bai, Prasenjit Mitra, C. Lee Giles, 2007. Searching for Tables in Digital Documents. *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on.* IEEE.↩

5. Alexander Bilke, Felix Naumann, 2005. Schema matching using duplicates. *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on.* IEEE.↩

6. Erhard Rahm, Philip Bernstein, 2001. A survey of approaches to automatic schema matching. *the VLDB Journal*, 10(4), pp.334-350.↩