Towards Odalic, a Semantic Table Interpretation Tool in the ADEQUATe Project*

Tomáš Knap

Semantic Web Company
Neubaugasse 1
1070 Vienna, Austria
tomas.knap@semantic-web.com

Abstract. The goal of the ADEQUATe project is to assess and improve quality of the (tabular) open data being published at two Austrian open data portals – https://www.data.gv.at and https://www.opendataportal.at. The goal of the quality improvement technique described in this paper is to semantically interpret such tabular data and publish them as Linked Data; this basically means to (1) classify columns of the input data using Linked Data vocabularies, (2) link cell values of the input data against Linked Data entities, (3) discover relations among the columns of the input data by searching for evidences of such relations among Linked Data sources, and (4) export such semantically interpreted data as RDF/Linked Data. In this paper, we describe limitations of TableMiner+, one of the tools for semantic table interpretation, with respect to our needs in the ADEQUATe project. Furthermore, we present Odalic, a tool for (semi)automatic semantic table interpretation and Linked Data publishing and describe how it addresses these limitation. We describe lessons learned using Odalic in the ADEQUATe project and also future work planned.

Keywords: Open Data, Linked Data, Table Interpretation, Named entity recognition, Data quality, Data linking, Entity disambiguation

1 Introduction

The advent of Linked Data [1] accelerates the evolution of the Web into an exponentially growing information space where the unprecedented volume of data offers information consumers a level of information integration and linking that has up to now not been possible.

In the recent days, governmental organizations publish their data as open data (most typically as CSV files). To fully exploit the potential of such data, the data publishing process should be improved, so that data are published as Linked Open Data. By publishing data as Linked Data, we increase usefulness of the data by (1) providing global identifiers for things and (2) links to external sources.

* Special thanks go to Václav Brodec, Ištván Satmári, Josef Janoušek, Jan Váňa, and Kateřina Boková who worked on the implementation of the Odalic tool. This work has been supported in part by the Austrian Research Promotion Agency (FFG) under the project ADEQUATe (grant no. 849982).

To semantically interpret CSV files and publish them as Linked Data, it is necessary to (1) classify CSV columns based on its content and context against selected (Linked Data) knowledge base(s) (2) assign RDF terms (HTTP URLs) to the particular processed cell values according to Linked Data principles (HTTP URL identifiers may be reused from one of the existing knowledge bases), (3) discover relations between columns based on the evidence for the relations in the existing knowledge bases, and (4) convert CSV data to RDF/Linked Data properly using data types, language tags, well-known Linked Data vocabularies, etc.

To illustrate such process of semantic table interpretation on an example, suppose that a processed CSV file contains 2 columns about movies – a movie's title and a movie's director. The steps of semantic table interpretation described above should basically (1) classify both columns as containing instances of classes `Movie` and `Director` respectively, 2) convert cell values in the movies' and directors' columns to HTTP URL resources, e.g., the movie "Matrix" may be represented with the HTTP URL `http://dbpedia.org/resource/The_Matrix` representing that movie in the DBpedia knowledge base[1] with all the other attributes and links to related resources, and (3) discover relation `isDirectedBy` between first and second column[2].

The project ADEQUATe is a national Austrian project. The goal of the ADEQUATe project is to provide a set of tools which would assess and improve the quality of the (tabular) open data being published at two Austrian open data portals – `https://www.data.gv.at` and `https://www.opendataportal.at`. One such data quality improvement is to semantically interpret such tabular data and provide them as Linked Data. We do not aim to provide a tool which would automatically interpret all the tabular data from those portals and provide them as Linked Data, as this is really challenging; we rather aim to provide a tool, which would run the semantic table interpretation on demand, incorporate user feedback and then publishes such data as Linked Data.

In this paper, we first describe limitations of the TableMiner+ tool [12], one of the tools for semantic table interpretation. We decided to exploit the TableMiner+ tool, because it outperforms similar algorithms, such as the one proposed by Mulwad et al. [7] or the algorithm presented in [5] and is available under an open license.

Furthermore, we describe Odalic, a tool for semantic table interpretation and publishing of tabular data as Linked Data. We describe the basic architecture of Odalic and then explain how Odalic addresses the limitations of TableMiner+. At the end, we describe lessons learned using Odalic in the ADEQUATe project and outline future work.

The rest of the paper is organized as follows. Section 2 discusses possible approaches for semantic table interpretation and Linked Data publishing and justifies selection of TableMiner+ as the most promising tool to be used in the ADEQUATe project. Section 3 introduces limitation of the TableMiner+ tool. Section 4 describes Odalic and how it address the limitations identified. Section 5 summarizes lessons learned of using Odalic in the ADEQUATe project and outlines future work. We conclude in Section 6.

[1] `http://dbpedia.org`

[2] Note: The classes `Movie` and `Director` and the relation `isDirectedBy` mentioned above should be represented by HTTP URLs and reused from a well known Linked Data vocabulary

2 TableMiner+ and Related Work

TableMiner+ [12] is an algorithm and tool for semantic table interpretation and Linked Data publishing. TableMiner+ consumes a table as the input. Further, it (1) discovers subject column of the table (the 'primary key' column containing identifiers for the rows), (2) classifies columns of the table to concepts (topics) available in Freebase, (3) links (disambiguates) cell values against Linked Data entities in Freebase, and (4) discovers relations among the columns by trying to find evidence for the relations in Freebase. TableMiner+ originally used Freebase as its knowledge base; as the authors in [12] claim, Freebase was the largest knowledge base and Linked Data set in the world, containing over 2.4 billion facts about over 43 million topics (e.g., entities, concepts), significantly exceeding other popular knowledge bases such as DBpedia and YAGO [10]. Nowadays, Freebase is shut down and being migrated into the Wikidata project[3]. TableMiner+ is available under an open license – Apache License v2.0.

Limaye et al. [5] model table components (e.g. headers of columns, cells) and their interdependence using a probabilistic graphical model, which consists of two components: *variables* that model different table components, and *factors* modeling (1) the compatibility between the variable and each of its candidate and (2) the compatibility between the variables believed to be correlated. For example, given a named entity column, the header of the column is a variable that takes values from a set of candidate concepts; each cell in the column is a variable that takes values from a set of candidate entities. The task of inference amounts to searching for an assignment of values to the variables that maximizes the joint probability [12].

Mulwad et al. [7] argue that computing the joint probability distribution in Limaye's method [5] is very expensive. Built on the earlier work by Syed et al. [11] and Mulwad et al. [6][8], they propose a lightweight semantic message passing algorithm that applies inference to the same kind of graphical model.

When comparing approach of Limaye et al. and Mulwad et al. with TableMiner+ approach, as the authors [12] state, TableMiner+ approach is fundamentally different since it (1) adds a subject column detection algorithm, (2) deals with both named entity columns and literal columns, while Mulwad et al. only handle named entity columns, (3) uses an efficient approach bootstrapped by sampled data from the table while Mulwad et al. and also Limaye et al. build a model that approaches the task in an exhaustive way, which is not efficient, (4) uses different methods for scoring and ranking candidate entities, concepts and relations; and (5) models interdependence differently which, if transformed to an equivalent graphical model, would result in fewer factor nodes.

In [2], the authors present an approach for enabling the user-driven semantic mapping of large amounts of tabular data using MediaWiki[4] system. Although we agree that user's feedback is important when judging about the correctness of the suggested concept for classification or suggested entity for disambiguation, and completely automated solutions semantically interpreting and publishing tabular data as Linked Data are very challenging, the approach in [2] relies solely on the user-driven mappings, which expects too much effort from the users.

[3] http://www.wikidata.org
[4] http://mediawiki.org

Open Refine[5] with RDF extension[6] provides a service to disambiguate cell values to Linked Data entities, e.g., from DBpedia. Nevertheless, the disambiguation is not interconnected with the classification as in case of, e.g., the TableMiner+ approach introduced in [12], so either a user has to manually specify the concept (class) restricting the candidate entities for disambiguation or all entities are considered during disambiguation, which is inefficient. Furthermore, the disambiguation phase is based just on the comparison of labels, without taking into account the context of the cell – further row cell values, column values, column header, etc.

3 Limitations of the TableMiner+ Tool

We decided to exploit TableMiner+ as a tool for semantically interpreting tabular (CSV) data obtained from the Austrian open data portals and publishing them as Linked Data, because it outperforms similar algorithms, such as the one proposed by Mulwad et al. [7] or the algorithm presented in [5] and is available under an open license.

Nevertheless, we were not able to use TableMiner+ in the ADEQUATe project directly as it has several limitations; those limitations are discussed below and grouped under 6 broader topics:

1. **User Interface:** There is no user interface in TableMiner+, which would allow to (1) configure semantic table interpretation tasks, e.g., input files to be processed, knowledge bases used, the way how the data is exported as Linked Data, and (2) which would allow users to provide feedback.
2. **User Feedback:** TableMiner+ does not allow users to provide feedback to the suggested semantic table interpretation – a user cannot specify, e.g., an alternative classification other then the one suggested by the algorithm, a user cannot manually select custom candidates by either searching the knowledge base or enriching the knowledge base, etc. Such user feedback must be also taken into account by the semantic table interpretation algorithm in the subsequent executions of the algorithm as a set of constraints.
3. **Supported Input Data:** TableMiner+ focuses on processing of tables obtained from HTML pages. Nevertheless, in the ADEQUATe project, we need to semantically interpret CSV data. Furthermore, TableMiner+ works the best for tables, where each line describes one resource, e.g., one project, one person etc., with no nested resources; however, this is not always the case in the ADEQUATe project, where we have to deal with files containing, e.g., information about countries and cities at the same time. Lastly, TableMiner+ has no support for interpreting and publishing statistical data as Linked Data, which is a common type of data used in the ADEQUATe project. In case of statistical data, the relation discovery phase is a bit specific and also the resulting RDF/Linked Data format is specific – resulting Linked Data should be published as RDF Data Cubes[7].

[5] `http://openrefine.org/`
[6] `https://github.com/fadmaa/grefine-rdf-extension`
[7] `https://www.w3.org/TR/vocab-data-cube/`

4. **Supported Knowledge Bases:** TableMiner+ does not support multiple knowledge bases for semantic table interpretation, so it has to always run against one knowledge base; on the other hand, in ADEQUATe we would like to run semantic table interpretation against ADEQUATe knowledge base and, e.g., DBpedia at the same time. TableMiner+ also did not support at the beginning general SPARQL based knowledge bases, such as DBpedia – it used Freebase as the only knowledge base supported out of the box. Lastly, setting up a configuration for a knowledge base in TableMiner+ usually means to adapt multiple configuration files, which is prone to errors and also quite complex.
5. **Support for Linked Data Publishing:** The process of publishing data as Linked Data in TableMiner+ is fragile and it does not follow any standard, which would, e.g., automatically generate RDF data out of table annotations. We would like to use, e.g., CSV on the Web metadata standard[8] for that. There is also no support for publishing statistical data as RDF Data Cubes[9].
6. **Semantic Table Interpretation Algorithm:** Performance of the TableMiner+ algorithm is not sufficient – tables with ten columns and tens of lines are usually processed tens of minutes. Furthermore, the TableMiner+ algorithm is not robust enough - minor errors during the algorithm's execution cause the whole semantic table interpretation process ends with an error.

4 Odalic

Being motivated by the limitations of TableMiner+ above, we propose Odalic, a tool for semantic interpretation of tabular data (CSV files) and publishing of such data as Linked Data.

Odalic builds on top of TableMiner+. Nevertheless, since we wanted to provide user interface to semantic interpretation processes, so that users can see results of and provide feedback to these processes, we had to change the way how semantic table interpretation library (TableMiner+) is provided. Odalic exposes the functionality of the TableMiner+ tool as a set of services (it was a Java library before), so that we can decouple business logic of the tool from the planned user interface. Main components of Odalic are as follows (see also Figure 1 below):

- **Odalic User Interface:** User interface allowing users to define/run semantic table interpretation processes, export data as Linked Data, review results, provide and persist user feedback.
- **Odalic Server:** The server (Java Web application) wraps the extended version of TableMiner+ algorithm (called Odalic Core) and provides REST API and business logic for the operations needed by the user interface of Odalic.

As depicted in Figure 1, Odalic can also be integrated with UnifiedViews[10], an ETL tool for management of RDF data processing tasks [4]. The main idea of the UnifiedViews integration is that Odalic semantic table interpretation tool may be used in a

[8] https://www.w3.org/TR/tabular-data-primer/

[9] https://www.w3.org/TR/vocab-data-cube/

[10] http://unifiedviews.eu, http://www.semantic-web-journal.net/system/files/swj1490.pdf

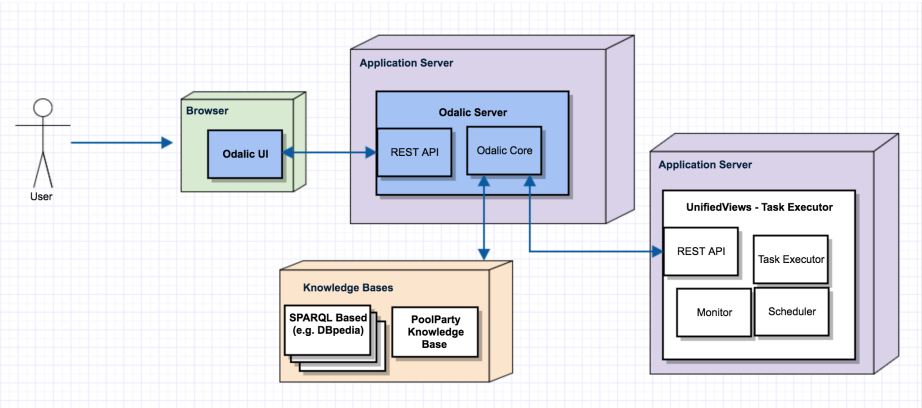

Fig. 1. Odalic

longer RDF data processing task starting, e.g., by fetching the original tabular data, providing such data to Odalic tool, refining the produced data by a series of SPARQL Update queries, merging with further external data and preparing resulting RDF data mart. UnifiedViews may schedule such RDF data processing task to run, e.g., every month. In these use cases, Odalic user interface is meant to be used for fine-tuning the configuration of the Odalic semantic table interpretation process before it can be automated by the UnifiedViews tool.

Odalic is published as an open source at GitHub[11]; it may be easily installed via Docker image[12]. In the next sections, we will discussed the limitations of TableMiner+ from Section 3 and how these limitations are addressed by the Odalic tool.

4.1 User Interface

User interface of Odalic is a single page web application implemented using AngularJS[13]. In this section, we present only excerpts from the Odalic user interface.[14]

Figure 2 shows a configuration of a semantic table interpretation task, which allows users to (1) select an input CSV file (and configure it, e.g., to specify delimiters, quotes, encoding of that file etc.), and (2) specify one or more knowledge bases used for a semantic table interpretation; one knowledge base must be always selected as the primary one – such knowledge base may be also extended with the new proposed classes/entities.

After the task is successfully executed, users may review results of a semantic table interpretation process. Figure 3 displays the input CSV data in a grid of interactive cells, which allows users to inspect the chosen and alternative classifications for the columns and disambiguations for the cell values, and provide feedback – e.g., users

[11] https://github.com/odalic/
[12] https://github.com/odalic/odalic-docker
[13] https://angularjs.org/
[14] Full demo of Odalic will be presented at the workshop.

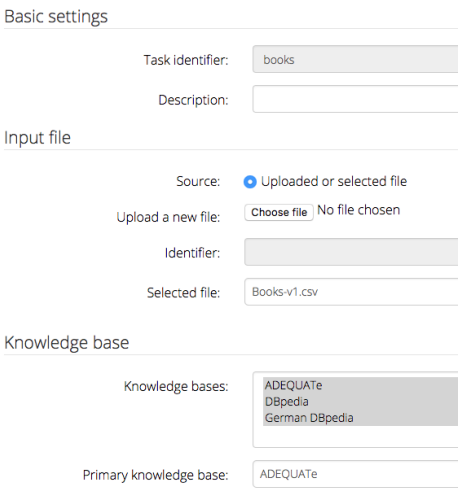

Fig. 2. Odalic Semantic Table Processing Task Definition

Fig. 3. Odalic Classification and Disambiguation and User Feedback

can delete the suggested class and propose a completely new one which is then written to the knowledge base. Figure 4 shows a dynamic graph of nodes representing the columns of the original input data and suggested relations between them, which can be reviewed/adjusted by a user – for example, a user may change the suggested predicate, add/remove relations, etc. There is also a view, which allows users to select for which columns the relations to other columns should be discovered.

4.2 User Feedback

Odalic allows users to provide feedback to the suggested semantic table interpretation; the supported user activities are as follows:

– A user may select an alternative classification/disambiguation/relation (than the one suggested by the Odalic Core algorithm).

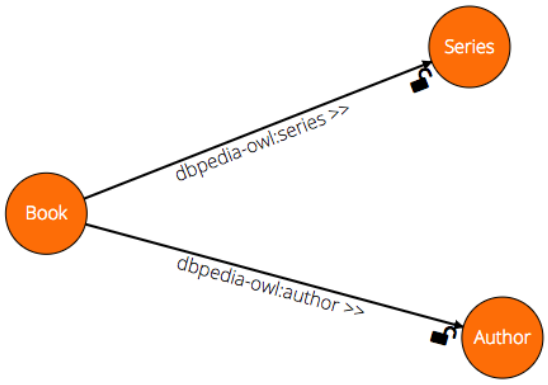

Fig. 4. Odalic Relations Discovery and User Feedback

- A user may search knowledge bases for an alternative classification/disambiguation/relation (which was even not suggested by the Odalic Core algorithm).
- A user may propose new classification/disambiguation/relation, which is not yet available in the knowledge base; for that, a knowledge base has to support SPARQL Update queries, or other means to change the knowledge base content.
- A user may delete any unwanted classification/disambiguation/relation.
- A user may customize a subject column – a column for which relations with other columns are discovered. User may also specify multiple subject columns for one table, covering the use cases when each row does not represent information about just a single resource, but rather multiple resources, e.g., countries and cities.
- User may force the algorithm to semantically interpret certain column not marked automatically by the Odalic Core algorithm as a named entity column (Odalic Core semantically interprets only named entity columns, e.g., numerical columns are ignored by default);
- User may flag certain named entity column as being ignored by the Odalic Core algorithm (as a result, although that column is a named entity column, such column is not semantically interpreted by the Odalic Core algorithm).

Odalic incorporates any user feedback provided into the subsequent executions of the Odalic Core algorithm – the feedback does not simply overrule the original results of Odalic Core algorithm; instead, it forms a set of constraints that the Odalic Core algorithm is able to work with and use to influence other suggested classification/disambiguation/relation results. For example, a manually set cell disambiguation provided by a user has still its score evaluated and it affects how the respective column is classified.

4.3 Supported Input Data

TableMiner+ focuses on tables obtained from HTML pages and takes into account also surrounding texts around those tables. On the other hand, Odalic has to support CSV files as these are the most common types of files used in the ADEQUATe project; however, Odalic does not take into account any further contextual data as part of the Odalic Core algorithm[15]

Furthermore, TableMiner+ works best for tables, where each line describes one resource, e.g., one project, one person etc., with no nested resource. Nevertheless, this is not always the case in the ADEQUATe project. So we added an option, which allows users to manually specify multiple subject columns, and extended Odalic Core algorithm to support that; as a result, relations may be searched for all such subject columns specified.

Odalic also allows user to specify that certain input CSV file should be processed as statistical data; as a result, the standard relation discovery process is skipped and the assigned types of columns (classes) are used to identify possible candidates for dimensions (every named entity column as a dimension) and measures (every non-named entity column is a measure) in the resulting data. A user can further customize those sets of dimensions and measures and export data as an RDF Data Cube[16].

4.4 Supported Knowledge Bases

TableMiner+ did not support at the beginning general SPARQL based knowledge bases – it used Freebase as the only knowledge base supported out of the box. Odalic on the other hand supports any knowledge base accessible via SPARQL query language and it also supports PoolParty knowledge base, a proprietary knowledge base used in the ADEQUATe project[17]. Furthermore, Odalic can be easily extended to support other types of knowledge bases, it is just necessary to implement interfaces for searching/enriching such knowledge bases. We also added to Odalic Core, in cooperation with the original author of TableMiner+, Ziqi Zhang, support for DBpedia knowledge base.

While the Odalic Core algorithm works exclusively with one knowledge base at a time, Odalic server can aggregate results from multiple executions of the Odalic Core algorithm on top of different knowledge bases. As a result, Odalic can run semantic table interpretation against multiple knowledge bases, one by one, and than aggregate results and provide them to the user. For example, Figure 3 shows top suggestions (if any) for classifications/disambiguations from two knowledge bases, each knowledge base using a different color.

Knowledge bases in Odalic can be configured in the Odalic user interface, which simplifies the configuration and makes it more robust – we can avoid cases when errors occurred due to typos in the knowledge base configuration files. Furthermore, a knowledge base configuration was also simplified, by, e.g., automatically detecting predicates,

[15] CKAN catalogs with ADEQUATe data contain, e.g., dataset's metadata, which may be used in the future as further contextual information.

[16] `https://www.w3.org/TR/vocab-data-cube/`

[17] `https://www.poolparty.biz/`

which may hold titles of resources – such predicates are used when searching knowledge base for the classification/disambiguation/relation candidates.

4.5 Support for Linked Data Publishing

Odalic Core saves the suggested semantic table interpretation result, optionally with the user feedback provided, as a set of (JSON) annotations over the original CSV file, which is done based on W3C CSV on the Web metadata standard[18].

Then, there are multiple formats in which a user may export the resulting data. First, a user may download an extended version of the original CSV file (hereafter `extended CSV file`), which contains the original CSV data and also an extra column for each named entity column in the original CSV file, which contains disambiguations for the corresponding column. Second, a user may download Linked Data/RDF data (either in the Turtle format[19] or JSON-LD format[20]). Linked Data/RDF data is produced automatically in a way described by the W3C CSV on the Web metadata standard by applying annotations over the original CSV file to the extended CSV file.

4.6 Semantic Table Interpretation Algorithm

Implementation of the Odalic Core algorithm (which is based on the TableMiner+ algorithm) was made more resilient against connection disruptions by ignoring the minor issues and just logging them; for example, if there is an error in the knowledge base lookup for cell's label, it does not cause the whole semantic table interpretation process to fail. We did not manage to solve the performance issues – performance improvements of the Odalic Core algorithm are our future work and are discussed a bit more in the next section.

5 Odalic: Lessons Learned and Future Work

In this section we describe lessons learned from using Odalic in the ADEQUATe project and also describe future work directions.

In [3], we tested precision and recall of the Odalic Core algorithm on top of the ADEQUATe data (see details in [3]) and we showed that classification and disambiguation has reasonable precision for certain types of columms, e.g., cities, districts, states, organizations. Nevertheless, for certain columns/cell values, the precision was rather low, which was caused mainly by missing evidence for the cell values in the target knowledge base.[21]

We also observed there is a high correlation between precision of the disambiguation and classification, which is caused by the fact that candidate classes of a column are based on the classes of the disambiguated entities for the cells in the given column.

[18] `https://www.w3.org/standards/techs/csv#w3c_all`
[19] `https://www.w3.org/TR/turtle/`
[20] `https://json-ld.org/`
[21] It was still Freebase knowledge base in that paper.

Nevertheless, if only one or two different cell values are disambiguated for the given column, or the disambiguated cell values do not agree on a common class or set of classes, it does not make much sense to classify that column (as Odalic Core currently does), as the classification will be in most cases misleading; in these cases, it is better to report that there is not enough evidence for the classification/disambiguation.

Furthermore, a class for a column is basically computed based on the classes associated with the disambiguated entities in that column. Thus, if entities are associated with more classes (usually forming a hierarchy), the Odalic Core algorithm tends to prefer more generic classes as winning classes, as they are associated with more disambiguated entities. The Odalic Core algorithm can mitigate such behavior by taking into account class hierarchies and preferring more specific classes.

As already discussed in Section 3, the Odalic Core algorithm in its current version does not take into account any contextual data about the processed CSV files. Nevertheless, such approach can be improved in the future as CKAN portals hosting those processed CSV files include also, e.g., descriptions of datasets.

Furthermore, recall of the relations discovered by the Odalic Core algorithm is quite low. The reason for that is that in order to suggest a relation between two columns, the Odalic Core algorithm has to find an evidence for such a relation for at least one row in the input data – but this is usually difficult. The idea is to use alternative methods to recommend relations, e.g., to look at the class of a subject column and distribution of the values within the potential object column and based on that try to suggest suitable predicate between those subject and object columns [9].

Performance of the Odalic Core algorithm has to be improved. Even for small input CSV files (tens of rows, ten columns) and DBPedia knowledge base, the semantic interpretation process can run for tens of minutes if the candidates from the target knowledge base are not pre-cached already – the reason for that is that there are numerous SPARQL queries to fetch relevant data from the target knowledge base and, quite surprisingly, also the caching itself (which is done using Solr) takes long. For example, for a file about Youth centers in Austria[22], which contains approximately 70 rows and 10 columns, the semantic table processing runs almost 20 minutes if the candidates from the knowledge base are not pre-cached. Surprisingly, the caching itself (pushing documents to Solr) takes approximately 10minutes – half of that time. If all the candidates from DBpedia are already pre-cached, the process can run much faster, but still it takes approximately 1 minute, which shall be improved. To speed up the semantic table interpretation process, we will investigate an option to pre-load the whole knowledge base, e.g., to Solr, tracking for each resource `R` important labels of that resource `R` and also labels of resources to which one can get via object properties of resource `R`. Also we will analyze the algorithm further to improve also the performance in case of already pre-cached data.

We also plan to provide a usability study of Odalic, to show to which extent the user interface of Odalic is reasonably designed, by observing how efficiently users can solve various semantic table interpretation tasks using Odalic. Nevertheless, for now, we have to say that the ability to provide user feedback to the results of the semantic

[22] `http://data.ooe.gv.at/files/cms/Mediendateien/OGD/ogd_dirBGD/Jugendzentren.csv`

table interpretation is crucial, as the algorithm itself has certain limitations and also the knowledge base is far from covering all the named entities in the input CSV files.

6 Conclusions

TableMiner+ is a tool for semantically interpreting tabular data and publishing them as Linked Data. In this paper, we described the limitations of the TableMiner+ tool with respect to the needs in the ADEQUATe project. Further, we described the basic architecture of Odalic, a tool we developed as an extension of TableMiner+, and explained how Odalic addresses these limitations of TableMiner+, e.g., by providing user interface for users to provide user feedback to the suggested results of the semantic table interpretation, or by improving Linked Data publishing. We described lessons learned from using Odalic in the ADEQUATe project and outlined future work.

References

1. C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1 – 22, 2009.
2. I. Ermilov, S. Auer, and C. Stadler. Csv2rdf: User-driven csv to rdf mass conversion framework. *Proceedings of the ISEM '13, September 04 - 06 2013, Graz, Austria*, 2013.
3. T. Knap. *Increasing Quality of Austrian Open Data by Linking Them to Linked Data Sources: Lessons Learned*, pages 243–254. Springer International Publishing, Cham, 2016.
4. T. Knap, P. Hanecak, J. Klimek, C. Mader, M. Necasky, B. v. Nuffelen, and P. Skoda. UnifiedViews: An ETL Tool for RDF Data Management. *Semantic Web Journal - to appear*, 2017.
5. G. Limaye, S. Sarawagi, and S. Chakrabarti. Annotating and searching web tables using entities, types and relationships. *PVLDB*, 3(1):1338–1347, 2010.
6. V. Mulwad, T. Finin, and A. Joshi. Generating linked data by inferring the semantics of tables. In *Proceedings of the First International Workshop on Searching and Integrating New Web Data Sources - Very Large Data Search, Seattle, WA, USA, September 2, 2011*, pages 17–22, 2011.
7. V. Mulwad, T. Finin, and A. Joshi. Semantic message passing for generating linked data from tables. In *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I*, pages 363–378, 2013.
8. V. Mulwad, T. Finin, Z. Syed, and A. Joshi. Using linked data to interpret tables. In *Proceedings of the First International Workshop on Consuming Linked Data, Shanghai, China, November 8, 2010*, 2010.
9. S. Neumaier, J. Umbrich, J. X. Parreira, and A. Polleres. *Multi-level Semantic Labelling of Numerical Values*, pages 428–445. Springer International Publishing, Cham, 2016.
10. F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM.
11. Z. Syed, T. Finin, and A. Joshi. Wikipedia as an ontology for describing documents. In *Proceedings of the Second International Conference on Weblogs and Social Media*. AAAI Press, March 2008.
12. Z. Zhang. Effective and efficient semantic table interpretation using tableminer+. *Semantic Web Journal*, 2016.