Základní informace

Jméno projektu	Open Data Linker and Classifier
Zkratka	Odalic
Vedoucí	Tomas Knap <knap@ksi.mff.cuni.cz>
Konzultanti	Ziqi Zhang <ziqi.zhang@sheffield.ac.uk>
Anotace	The goal of the project is to 1) improve TableMiner+ algorithm for converting tabular data to Linked Data, 2) provide REST API for operating the algorithm by third party applications, and 3) provide user interface for operating the algorithm by users, so that users may configure the algorithm, run the conversions, provide feedback to the results of the conversions, maintain knowledge bases used by the algorithm etc. Odalic tool will be tested on tabular data (CSV files) obtained from 2 Austrian open data catalogs [4, 6].

Motivace

The advent of Linked Data [12] 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 that has not been possible until now. Consumers can now integrate information for use in a myriad of alternative end uses.

In the recent days, governmental organizations publish their data as open data (predominantly as CSV files). To fully exploit the potential of such data, the publication process should be improved, so that data are published as Linked Open Data. When elevating open data to Linked Data, we increase usefulness of the data by providing global identifiers for things and we enrich the data with links to external sources. To transform CSV files to Linked Data, it is necessary to:

1. classify CSV columns based on their content and context against existing knowledge bases;
2. assign globally unique HTTP URL identifiers to the particular cell values according to Linked Data principles - such 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 data properly using data types, language tags, well-known Linked Data vocabularies, etc.

To introduce an illustrative example of transforming CSV files to Linked Data, assume that the published CSV file contains names of the movies in the first column and names of the directors of these movies in the second column. The conversion of CSV files into Linked Data should automatically:

1. classify the first and the second column as those containing instances of classes 'Movie' and 'Director';
2. convert cell values in the movies' and directors' columns to HTTP URL resources, e.g. instead of using 'Matrix' as the name of the movie, URL 'http://www.freebase.com/m/02116f' may be used, pointing to Freebase knowledge base [13] and standing for 'Matrix' movie with further attributes of that movie and links to further resources;
3. discover relations between the columns, such as the relation 'isDirectedBy' between first and second column (the classes 'Movie' and 'Director' and the relation 'isDirectedBy' mentioned

above should be reused from some well know Linked Data vocabulary).

4. Produce RDF data, e.g., in this case, the following RDF triples serialized in Turtle syntax [14]:
   <http://www.freebase.com/m/02116f> a x:Movie .
   <http://www.freebase.com/m/02xxxx> a x:Director .
   <http://www.freebase.com/m/02116f> x:isDirectedBy <http://www.freebase.com/m/02xxxx> .

TableMiner+ [3] is an algorithm for (semi-)automatic conversion of tabular data to Linked Data. TableMiner+ consumes a table as the input. Further, it:

1. discovers the subject column of the table (the 'primary key' column containing the main subject the row is about),
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+ uses Freebase as its knowledge base; as the authors in [3] also mention, since Freebase is going to be shut down, it should be replaced with a different knowledge base.

Popis projektu

The goal of the project is to:

1. improve TableMiner+ algorithm [3], in particular to improve the classification, disambiguation and relation discovery parts of TableMiner+ algorithm. The improved version of TableMiner+ is from now on called **Odalic Core**.
2. provide user interface for interacting with Odalic Core (e.g. providing feedback to the results of Odalic Core, or describing how RDF data should be produced),
3. provide API, so that other applications may work with Odalic Core easily, and
4. provide executor, so that executions of conversions of tabular data to RDF data format may be automated.
5. provide support for additional knowledge bases (used by Odalic Core algorithm for classification, disambiguation and relation discovery).

The modules of Odalic are:

- **Odalic User interface (UI)**
  - Allows users to interact with Odalic Core algorithm: to submit CSV file to Odalic Core, observe results of the algorithm, provide feedback, re-execute the Odalic Core algorithm, describe how RDF data should be produced from results of the Odalic Core algorithm and present the data. Users may also save configuration for the whole process for later execution.
- **Odalic Server**
  - **Odalic Core** improving classification, disambiguation and relation discovery of TableMiner+.
  - **Odalic API**, so that UI and third party applications may operate with Odalic Core algorithm.

        - **Executor**, which allows to execute Odalic Core conversion tasks set through the UI or API.
        - **Odalic Knowledge Base Management**:
            - To store knowledge bases used.
            - To allow management of knowledge bases.
- **Odalic RDF Store (RDF database)**
    - To store knowledge bases used and information about external knowledge bases.
    - To store configurations of the transformations.
    - To store imported and locally available knowledge bases.
    - To store feedback from the users.

Odalic (Odalic Core and also the whole framework) will be tested on tabular data (CSV files) obtained from the Austrian open data catalogs [4, 6] - it must be able to process all valid CSV files in these 2 portals and produce reasonable output.

The project will be hosted on GitHub and Maven based.

Core User Requirements:

- User may load tabular files (CSV files) and use them as input to Odalic Core algorithm.
    - User may select files/folders to be processed or directly upload a file.
- User may configure Odalic Core algorithm.
    - User may select which knowledge base(s) Odalic Core will use (from the set of supported knowledge bases).
        - User may also specify which knowledge base should be used as the primary one for reusing URIs in the resulting RDF data.
    - User may adjust similarity metrics and their configuration used during classification/disambiguation/relation discovery of Odalic Core algorithm.
- User may execute Odalic Core from UI on the selected input.
- User may observe the results of Odalic Core.
    - User is provided with the classification/disambiguation/relation discovery results:
        - User is provided with top k candidates and the best candidate as computed by Odalic Core.
        - User is also provided with basic statistics - number of concepts classified, number of entities classified in each column, number of relations discovered.
- User may provide feedback - adjust the classification/disambiguation/relation discovery results of Odalic Core algorithm:
    - User may select the preferred alternative candidate concept/entity/relation.
    - User may specify new custom concept/entity/relation.
    - User may just denote that certain suggested classification/disambiguation/relation candidate is wrong.

        - System will incorporate such feedback in further runs of the algorithm (for this task).
- User may re-adjust configuration of Odalic Core algorithm, such as the set of knowledge based used.
- User may re-execute Odalic Core algorithm and the next run of the algorithm has to incorporate feedback from the user and (if any) changed configuration (e.g. different set of knowledge bases).
- User may save the configuration of the CSV->RDF data conversion so that it can be executed or re-executed later:
    - System must save the configuration in machine readable format, e.g. JSON-LD, RDF data format serializations. Every such saved configuration has its own Linked Data identifier.
    - The configuration includes:
        - URL of the configuration (identifier),
        - description of the conversion,
        - configuration for Odalic Core algorithm (knowledge bases used, similarity metrics, feedback from the user),
        - inputs to which it was applied (if any),
        - settings of the outputted RDF data format (if available).
    - The configuration must be complete, so that it is possible to execute the same conversion later in the future to the same/similar inputs.
    - User may export the configuration of the conversion in machine readable format.
- User may define the shape of the resulting data RDF data (classified, disambiguated, with relations discovered).
    - User may be presented with a preview of the resulting RDF data (presented on a sample row) derived from the data resulting from Odalic Core algorithm run, which may be further adjusted (URI prefixes, used predicates, etc.).
        - System then applies the prepared template to all rows to produce RDF data
        - User may specify whether URIs should be reused from the (prefered) knowledge base or whether URIs for concepts/entities/relations should be generated according to a certain template and then linked to classified concept/disambiguated entities/discovered relations, e.g. by using owl:sameAs RDF predicates.
    - System will support generation of statistical data in RDF data cube format (as these are typical files appearing within [4,6]):
        - So user may specify which columns are dimensions of the data cube, which columns are measures of the data cube and then the RDF data cube is prepared using standard RDF data cube [11].
    - User may save the generated RDF data either to a file (or files), which he or she may download, or to a preconfigured RDF store.
- Administrator may manage knowledge bases Odalic Core can work on top of:
    - Administrator may select which knowledge bases Odalic Core should use by default.

        - Administrator may add more knowledge bases:
            - by importing them in a file (RDF format - RDFS/OWL),
            - by providing URL to their SPARQL endpoint.
        - Administrator may remove knowledge bases.
        - Administrator may browse knowledge bases (using SPARQL endpoint of RDF store or any other suitable UI).
- User may get list of all conversion configurations previously defined and saved :
    - For each conversion configuration, user may get:
        - Description of the conversion.
        - Configuration for Odalic Core algorithm (knowledge bases, similarity metrics, feedback from the user).
        - Inputs to which it should be applied
            - The system needs to cache the input files.
        - Settings of the outputted RDF data format (if available).
    - The list of conversions is automatically prefiltered for each user so that he can see the conversions defined by him.
    - Administrator may see all conversions.
- User may monitor executions of conversions to see the list and status of the executed conversions:
    - For each conversion execution, user may get:
        - configuration
        - produced outputs
        - status of the conversion - whether it was done, or is in progress
    - User may see list of executions executed by him.
    - Administrator may see all executions.
- User must log in before he can save the configuration, see the list of executions or manage his configurations and executions.

Odalic Server

In the following sections we discuss in detail parts of Odalic Server module, in particular:

1. Odalic API
2. Odalic Knowledge Base Management
3. Odalic Core
    a) classification/disambiguation algorithm
    b) relation discovery

1. Odalic API

API will be implemented as REST API if possible, supported API calls:

- Configurations:

        - get list of conversions,
        - get details of the conversion - description & configuration, info about inputs, outputs.
    - Executions:
        - execute Odalic Core algorithm with a defined configuration on certain input,
        - get list of executed conversions,
        - get details about the executed conversion - status, result, output produced.
    - Knowledge Bases:
        - import new knowledge base
            - either by importing information about the SPARQL endpoint or by importing the file with the knowledge base.
        - export knowledge bases.
        - get list of knowledge bases defined.
        - get list of default knowledge bases.

Odalic will support authentication, so that Odalic API is secured in a way that only applications/users knowing the name (and password) can operate API services. Every user will have its own authentication name (and optionally password) and based on that, his executions may be displayed/operated.

In terms of authorization, there are two roles Odalic will support - User and Administrator. User works with his resources (configurations of tasks, executions), Administrator may work with any resource and manage use of resources by other users.

2. Odalic Knowledge Bases Management

We can distinguish two types of knowledge bases used to realize classification, disambiguation and relation discovery - general purpose knowledge bases (e.g. DBpedia.org [7], WikiData [8], other datasets in the Linked Open Data Cloud [5]) and focused knowledge bases (an example could be a dataset with all schools in the given country, all streets in the given city etc.). The choice of focused knowledge bases depends heavily on the processed data.

Currently, TableMiner+ supports Freebase as a knowledge base. Nevertheless, Freebase is deprecated and a different knowledge base must be used. Odalic Core algorithm must be extended to support different knowledge bases -- the goal is to add support for at least two general knowledge bases and at least 5-10 focused knowledge bases (we plan to access these knowledge bases uniformly - using SPARQL protocol). User of Odalic must be able to access these knowledge bases via the UI.

Odalic will support any RDFS/OWL knowledge base which is available via (remote) SPARQL endpoint. Also Odalic will support imports of any RDFS/OWL ontology from a file to the internal Odalic

RDF store.

Odalic Core will be extended to take into account the hierarchy of topics within the knowledge base (if knowledge base supports such hierarchy), so that the most concrete concept are prefered during classification, which improves performance and increases precision of the algorithm.

3. Odalic Core

In Odalic Core, two extensions are planned as part of the project:

a) regarding classification/disambiguation algorithm and
b) regarding relation discovery

3a) Classification/Disambiguation Algorithm

The improvements of the classification/disambiguation algorithm are guided by the currently discovered issues in the existing TableMiner+ algorithm, such as:

1. no use of focused knowledge bases, which are most relevant when classifying data,
2. no use of feedback provided by the users to the classification/disambiguation algorithm,
3. too many queries to the knowledge bases during disambiguation caused by ineffective restriction of the searched entities in the knowledge bases,
4. too many false positives in case of lower evidence for the disambiguated cells/classified columns or CSV files providing low context for the classified columns/disambiguated cells.

To address (1), focused knowledge bases relevant for the data obtained from Austrian data portals should be used. To address (2) Odalic User Interface will allow user to provide feedback, which is then incorporated by the Odalic Core algorithm.
To address (3), special heuristics will be proposed. For example, the algorithms for disambiguation typically take into account the context of the disambiguated cell, such as the row and column in the table the cell is part of. Nevertheless, there is no differentiation among the meaning of the other columns' cells forming the context of the examined row; for example, if I would disambiguate name of a city, the information about 'locality' (state, country) is really important to reduce the number of entities probed in the knowledge bases and increase precision. To address (4), algorithm should be extended to work reasonably in the case of lower evidence for the disambiguated cells/classified columns or processing CSV files providing low context for the classified columns/disambiguated cells. For example, when there are too few records in the CSV file, or too few cells were disambiguated, it does not make much sense to try to classify the column. Similarly, if the cell values are too short and there is no other named entity column clearly clarifying the context of these short values, it does not make much sense to disambiguate.

3b) Relation Discovery Algorithm

The planned improvements of the relation discovery algorithm are guided by the currently discovered issues in the existing TableMiner+ algorithm, such as:

1. no use of focused knowledge bases, which are most relevant when classifying data,
2. no use of feedback provided by the users to the relation discovery,
3. inappropriate comparison of potentially relevant relations in the knowledge bases,
4. selection of the label for the relation in case of more evidences for the relation in more knowledge bases.

To address (1), focused knowledge bases relevant for the data obtained from Austrian data portals should be used. To address (2) Odalic User Interface will allow user to provide feedback, which is then incorporated by the Odalic Core algorithm.
To address (3), comparison of potentially relevant relations in the knowledge bases has to be improved. Currently, when searching for relation evidence in the selected knowledge bases, the values within potentially related CSV columns are compared with RDF triples containing subject column of the CSV table as the subject of the triple. Nevertheless, object of the triple in the knowledge base may be a resource (URI), not just plain literal, and in that case different comparison is needed than just comparing similarity of the cell value and URI of the resource. Also the relation discovery algorithm should take into account that subject column in the CSV file may be also an object of the triple, not just subject. To address (4), various knowledge bases should vote for the best label/predicate selected for the given relation.

Platforma, technologie

The system consist of the three main modules - **Odalic User Interface**, **Odalic Server**, and **Odalic RDF Store**.

Odalic Server will be Java Web application, which run in an application server, such as Tomcat. Odalic User Interface will be implemented as Web UI using JavaScript. The particular technology and framework used for UI will be decided after detailed analysis.

For Odalic RDF store, we will use open source version of certain RDF database, such as OpenLink Virtuoso. The particular RDF database which we will use will be determined after initial analysis.

Odhad náročnosti

Expected number of team members: 5
Deadline for realization: In 9 months
Plan:

- 1st and 2nd month – detailed analysis of the requirements and architecture
    - All team members (5 students)
- 3rd to 6th month – implementation
    - Odalic Server
        - Odalic Core - 2 students
        - API, Odalic Knowledge Base Management - 1 student
    - Odalic User Interface
        - 2 students
    - Milestones

        - 4th month – alfa version
        - 6th month – beta version
- 7th to 9th month – finalization, documentation, testing, bug-fixing
    - Testing of Odalic on top of valid CSV files from Austrian portals
    - All team members (5 students)

Vymezení projektu

Projekt je zaměřen na následující oblasti (zaškrtněte vyhovující):

Diskrétní modely a algoritmy	
	diskrétní matematika a algoritmy
	geometrie a matematické struktury v informatice
	optimalizace
Teoretická informatika	
	Teoretická informatika
Softwarové a datové inženýrství	
x	softwarové inženýrství
x	vývoj software
x	webové inženýrství
x	databázové systémy
	analýza a zpracování rozsáhlých dat
Softwarové systémy	
	systémové programování
	spolehlivé systémy
	výkonné systémy
Matematická lingvistika	
	počítačová a formální lingvistika
	statistické metody a strojové učení v počítačové lingvistice
Umělá inteligence	
	inteligentní agenti
	strojové učení
	robotika
Počítačová grafika a vývoj počítačových her	
	počítačová grafika
	vývoj počítačových her

Poznámky

The project is consulted with Ziqi Zhang from University of Sheffield, the author of the original

TableMiner+ algorithm. Also for this reason, all materials are written in English.

References

[1] Frank Manola and Eric Miller. RDF Primer. W3C Recommendation 10 February 2004. [http://www.w3.org/TR/2004/REC-rdf-primer-20040210/]

[2] Christian Bizer, Tom Heath, Tim Berners-Lee: Linked Data - The Story So Far. Int. J. Semantic Web Inf. Syst. 5(3): 1-22 (2009) [http://tomheath.com/papers/bizer-heath-berners-lee-ijswis-linked-data.pdf]

[3] Ziqi Zhang, Effective and Efficient Semantic Table Interpretation using TableMiner+, Submitted to Semantic Web Journal. [http://www.semantic-web-journal.net/system/files/swj1111.pdfTableMiner+]

[4] Official Austrian open data catalog [https://www.data.gv.at/]

[5] Richard Cyganiak et al. Linked Open Data Cloud. Online: http://lod-cloud.net/

[6] Open data catalog, https://www.opendataportal.at/

[7] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, Christian Bizer: DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. Semantic Web Journal, Vol. 6 No. 2, pp 167–195, 2015.

[8] Wikidata.org Online: https://www.wikidata.org/

[9] Linked Open Vocabularies. Online. http://lov.okfn.org/dataset/lov/

[10] Ivan Ermilov. Web Tables Automatic Property Mapping - TAIPAN. Online: https://github.com/AKSW/TAIPAN

[11] RDF Data Cube https://www.w3.org/TR/vocab-data-cube/

[12] C.Bizer,T.Heath,andT.Berners-Lee.LinkedData-TheStorySoFar.InternationalJournal on Semantic Web and Information Systems, 5(3):1 – 22, 2009.

[13] Freebase knowledge base, http://freebase.com

[14] Turtle, W3C Recommendation, https://www.w3.org/TR/2014/REC-turtle-20140225/