SAIM - Manual

Klaus Lyko

March 25, 2013

Contents

Chapter 1

Introduction

Links between instance are of central importance for the Semantic Web and for a large number of tasks, such as data integration, federated querying and knowledge retrieval. Finding those links is the subject of Link Discovery (LD). Two main problems arise when trying to discover links between data sets.

First, naive solutions have to compare all instances of a source knowledge base with all instances of the target knowledge base, thus having a quadratic time complexity. Second, one have to guarantee the quality of those links. To address the first problem time efficient algorithms and frameworks such as SILK[1] and LIMES[2] have been developed to reduce the number of comparisons. For the second problem both supervised (e.g. [3, 4]) and unsupervised (e.g. [5]) machine learning approaches have been developed.

SAIM [2] encompasses solutions for both problems within one interface. It is based upon algorithms implemented in the LIMES[3] framework which have been shown to outperform state-of-the-art in previous work w.r.t. time-complexity [1]. SAIM offers an intuitive GUI for experts to manually create Link Specifications while implementing the supervised and unsupervised machine learning algorithms EAGLE [3] and COALA [4] respectively, which have been shown to lead to highly accurate link specifications.

This manual will cover the complete workflow of SAIM. Starting in chapter 2 from scratch it will cover the process of defining the endpoints(2.1), class restrictions(2.2) and properties(2.3) of the knowledge bases to link. Chapter 3 describes how experts can build link specifications manually. Chapter 4 and 5 discusses the unsupervised and supervised learning algo-

[1]`https://www.assembla.com/spaces/silk/`

[2]SAIM stands for (Semi-)Automatic Instance Matcher.

[3]`http://limes.sf.net/`

rithms respectively. Finally, chapter 6 covers additional features of SAIM such as saving and loading link specifications, uploading RDF dumps.

Chapter 2

Source and Target Endpoint definition

Figure 2.1 shows SAIM upon startup. As all no link specification is defined yet the 3 buttons at the bottom are deactivated and the accordion panel is empty. To start a Link Specification we go to the menu `Link Specification`. Figure 2.2 shows the available menu entries:

1. `Start new configuration` Start a new Link Specification from scratch.
2. `Import Link Specification` To Import Link Specifications. It is covered in section 6.2.

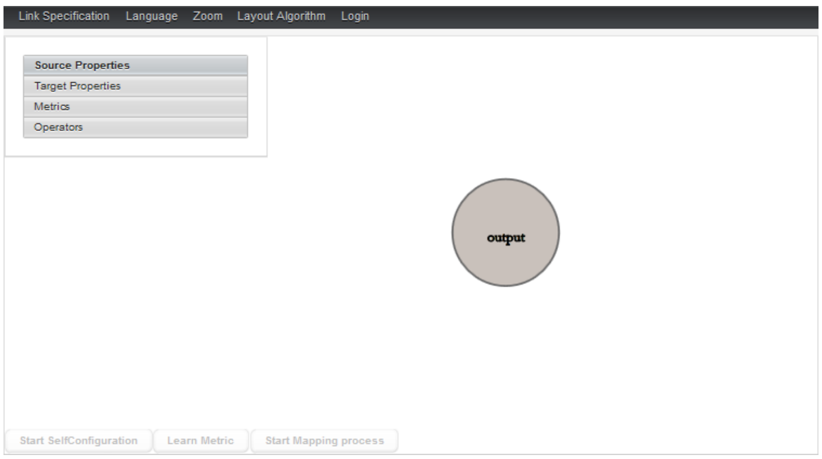


Figure 2.1: The main window of SAIM is divided into three parts: A <u>menu bar</u> at the top, an <u>accordion</u> panel to the left and the <u>metric panel</u> at the center.

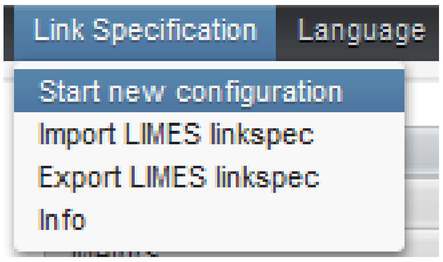


Figure 2.2: The `Link specification` menu.

3. `Export Link Specification` Let you save Link Specifications on you local machine. It is covered in section 6.2.

4. `Info` Will open a window with basic information about us.

2.1 Endpoints

To start a link specification we have to specify the RDF data sources dubbed *Source* and *Target Endpoint*. To do so go in the menu bar to `Link Specification` >> `Start new configuration` as depicted in figure 2.2. This will open a Window as shown in figure 2.3.
There are fields to be set:

1. `Endpoint URL` The URL of the public accessible SPARQL endpoint.

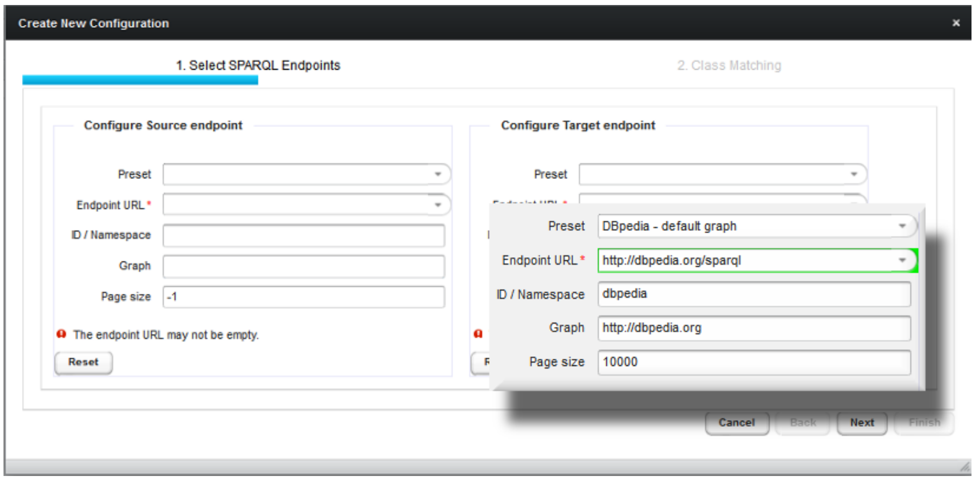
2. `ID/Namespace` identifies the endpoint. Should be a string differentiating the endpoint as this entry is used be certain class matchers later.

3. `Graph` (optinal) URI of a specify RDF graph.

4. `Page Size` (optional) for pagination of SPARQL queries.

It is possible to use predefined Presets from a list, such as DBPedia as depicted in the top layer of figure 2.3. SAIM also supports RDF local dumps on the Server. We have loaded the OAEI2010[1] Benchmark datasets Person1, Persons2 and Restaurants. These datasets are accessible via the `Preset` list. SAIM supports the upload of RDF dumps in diffferent formats. Please refer to section 6.3 for details.
If a public accessible SPARQL endpoint is used instead, SAIM tests its

[1]Ontology Alignment Evaluation Initiative: `http://oaei.ontologymatching.org/2010/`


Create New Configuration

1. Select SPARQL Endpoints

2. Class Matching

Configure Source endpoint

Preset	
Endpoint URL *	
ID / Namespace	
Graph	
Page size	-1

The endpoint URL may not be empty.

Reset

Configure Target endpoint

Preset	DBpedia - default graph
Endpoint URL *	http://dbpedia.org/sparql
ID / Namespace	dbpedia
Graph	http://dbpedia.org
Page size	10000

Figure 2.3: The Panel to specify the SPARQL endpoints. The front layer shows an example configuration to use DBPedia as an endpoint.

availability by issueing a simple SPARQL query. If the test for reachability is successful SAIM indicates it by a green margin arround the `Endpoint URL` field as indicated in the top layer of figure 2.3.

If both source and target knowledge bases are specified we confirm it by clicking the `Next` button. The step is the class setting.

Note, that we will use the Person11 and Person12 datasets of the Person1 OAEI2010 Benchmark as source and target endpoints throughout this manual.

2.2 Class restriction

Goal here is to specify the `rdf:type` of the instances we want to match. Figure 2.4 shows the Class Matching Step.

Per default a simple string based comparison off all available classes is computed. Upon finish the user can choose a match in the list at the top of the class matching window. The more sophisticated but time consuming link based comparison could be started manually. This method tries to link `rdf:types` via availlable `owl:sameAs` statements. If large knowledge bases such as DBPedia are used to query all availlable classes may take some time, so please stay patient. Once this is done, the user can also navigate through the class and subclass hierachy of an endpoint via the tree at the bottom.

More details

Once two instance types are selected the next and final step of the

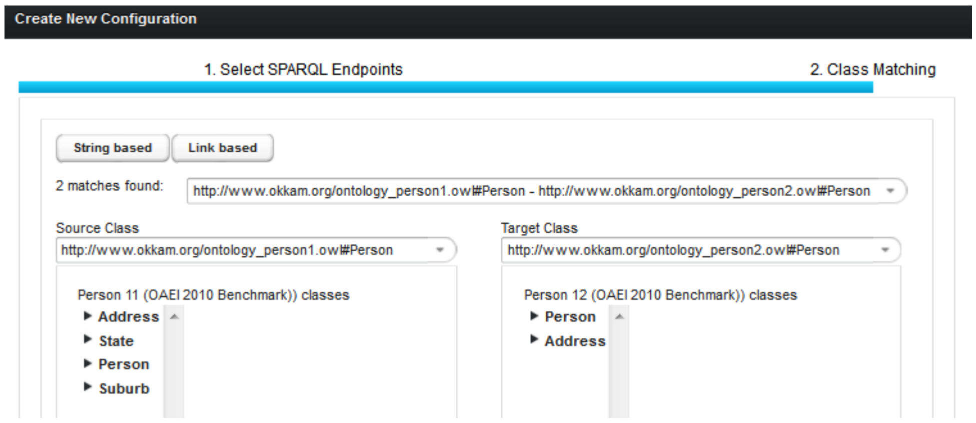

Create New Configuration
1. Select SPARQL Endpoints
2. Class Matching
String based
Link based
2 matches found:
http://www.okkam.org/ontology_person1.owl#Person - http://www.okkam.org/ontology_person2.owl#Person
Source Class
http://www.okkam.org/ontology_person1.owl#Person
Target Class
http://www.okkam.org/ontology_person2.owl#Person
Person 11 (OAEI 2010 Benchmark)) classes
Address
State
Person
Suburb
Person 12 (OAEI 2010 Benchmark)) classes
Person
Address

Figure 2.4: Class Matching step. Goal is to specify the `rdf:type` restrictions both source and target instances have to fulfill. The user can manually navigate through the available classes via the class trees or use calculated class matches via the list at the top. The two buttons `String based` and `Link based` at the top will start a new automatic matching process.

endpoint specification is to choose the properties the Link Specifications can rely on.

2.3 Property Matching

Goal of this step is to specify sets of properties for the source and target instances. Note, that the use specifies property matches. These matches will later have an influence on Link Specification learning algorithms based on Genetic Programming. These algorithms will only consider comparisons between these links. So if you're uncertain about corresponding properties we encourage you to choose all possible property pairs. However, the property matching will only restrict learning methods. The user will be free to later manually compare any source property with any target propertiy.

Once again this step is assisted by automatic computed property matches. On default a string based comparison of the property names is computed. But the user can also start a more sophisticated yet complex matching algorithm at the top of the property matching panel. Once all property pairs were defined, we submit the settings by clicking the `Finish` button. We will return to the main page of SAIM. As

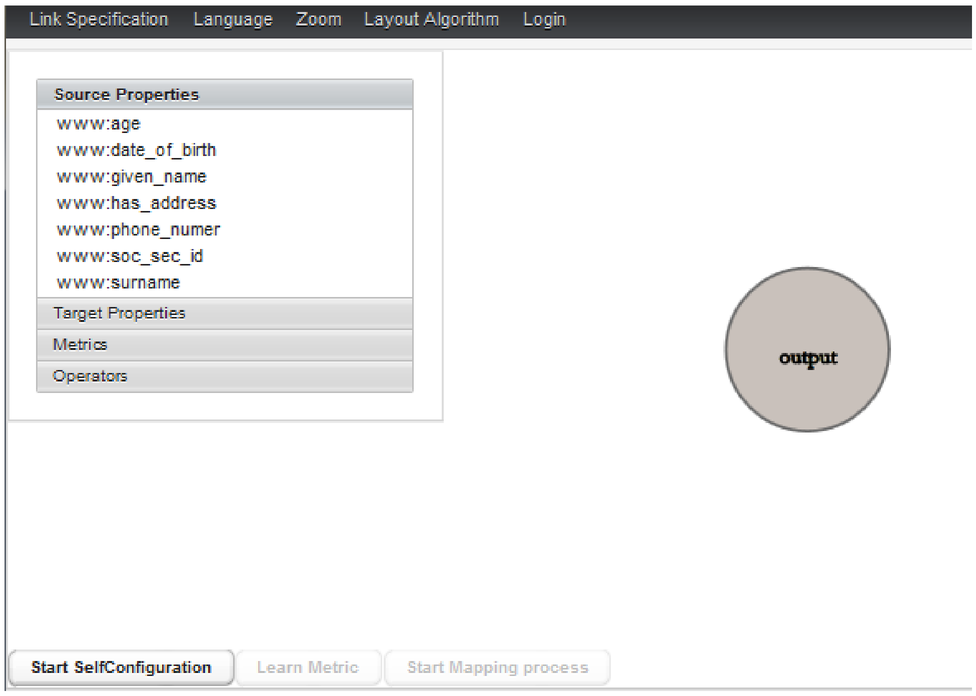


Figure 2.5: The main window of SAIM after specifying the endpoints. The accordion lists of source and target properties on the left hand are filled according to the propties the user selected in the Property Matching step described in section 2.3. This sceenshot is based on the running example Person1 dataset of OAEI2010, depicted in appendix A.2

Chapter 3

Manual metric definition and execution

The main window of SAIM enables the user to manually specify a Link Specification without knowing the LIMES grammar in detail. For a formal specification please refer to [3].

A Link Specification relies on a specification of atomic similarity measures and similarity measures. Generally, a similarity measure m is a function such that $m : S \times T \rightarrow [0, 1]$, where S (source) and T (target) are set of enteties. We call a measure atomic when it relies on exactly one similarity measure σ (e.g., trigrams similarity for strings) to compute the similarity of a pair $(s, t) \in S \times T$. We call a link specification atomic if it compares the value of a measure m with a threshold θ, thus returning the pairs (s, t) that satisfy the condition $\sigma(s, t) \geq \theta$.
A link specification $spec(m, \theta)$ is either an atomic link specification or the combination of two link specifications via specification operators such as `AND` (intersection of the set of results of two specs), `OR` (union of the result sets), `XOR` (symmetric set difference), or `DIFF` (set difference).

Measures and Operators

SAIM supports the following atomic similarity measures two compare the values of two properties:

- `Cosine` A Vector based String Similarity.
- `Trigrams` The Dice Coefficient based on common trigrams.
- `Levensthein` aka Edit distance.

- `Jaccard` A Token-based coefficient defined as the size of the intersection divided by the size of the union.
- `Overlap`
- `Euclidean` The euclidean distance of two numbers.

To build complex Link Specifications SAIM supports complex measures:

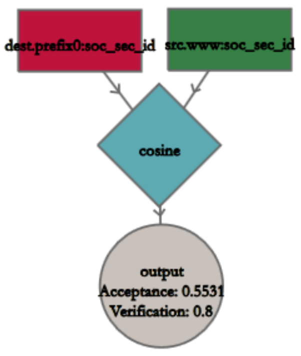
- `MIN`(m_1, m_2) Metric operator, combines the matches of m_1 and m_2 based on the minimum similarity.
- `MAX`(m_1, m_2) Metric operator, combines the matches of m_1 and m_2
- `ADD`(m_1, m_2) A linear combination, based on coefficients.
- `MULT`(m_1, m_2) A linear combination, based on coefficients.
- `AND`(m_1, m_2) Intersection of the set of results of two specs (based on (local) thresholds).
- `OR`(m_1, m_2) Union of the set of results of two specs (based on (local) thresholds).
- `XOR`(m_1, m_2) Symmetric set difference of the set of results of two specs (based on (local) thresholds).
- `DIFF`(m_1, m_2) Set difference of the set of results of two specs (based on (local) thresholds).

Buildung a Link Specification

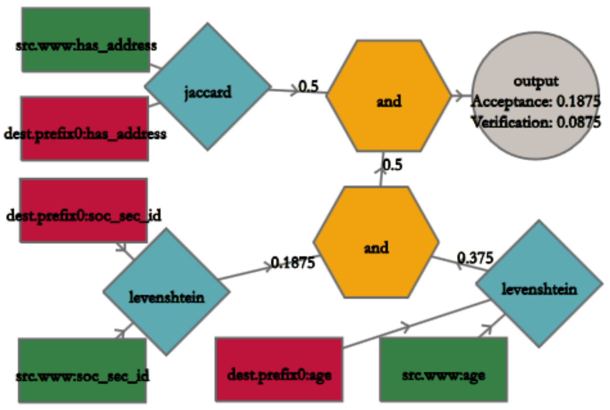
Figure 3.1 shows an example of an atomic and a complex Link Specification.

In order to build a atomic Link Specification as depicted in 3.1a the user has to do:

1. Choose a source property from the accordion panel (e.g. `www:soc_sec_id` as in 3.1a)
2. Choose a target property from the accordion panel (e.g. `pefix0:soc_sec_id` as in 3.1a)
3. Choose a atomic similarity measure from the accordion panel (e.g. `cosine` as in 3.1a)



(a) Atomic Link Specification



(b) Complex Link Specification

Figure 3.1: Examples of Link Specifications. Basesd upon example A.2.

4. Connect both properties with the measure (right-click on the property, choose *Link to* from the context menu, left-click on the measure)

5. Connect the measure with the output node (right-click on the measure, choose *Link to* from the context menu, left-clcik on the Output node).

6. Double-click the Output node and set/adjust the thresholds.

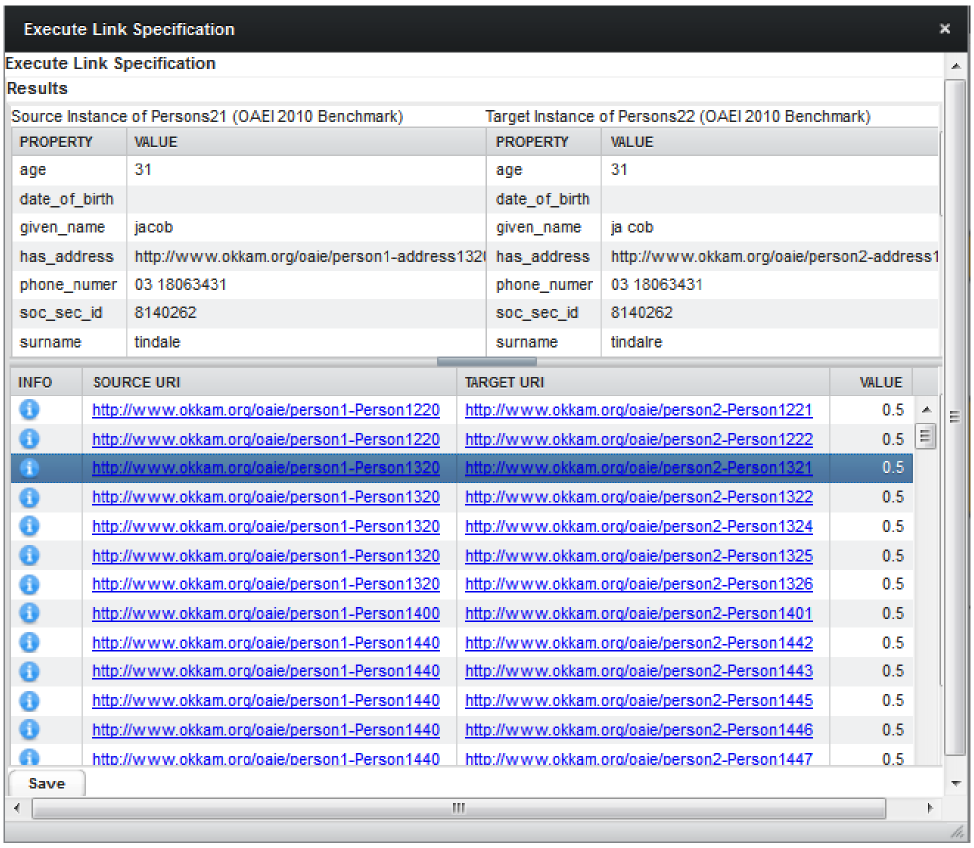
Note that several nodes in the Link Specification tree have parameters to be set. To do so **double-click** the node.

- Property nodes: Select preprocessing functions for property.
- Operator nodes: Set parameters: local thresholds/coefficients for child specs.
- Output Node: global acceptance(verification) threshold.

Executing a Link Specification

To run a Link Specification click the button `Start Mapping process`. First the specified metric is checked if it is complete and valid. If everything is ok, the mapping process is started and a new window shows the progress. The results will be presented in the Mapping Window as depicted in figure 3.2.
The **table** at the bottom shows instance pairs of the specified source and

target endpoint which fulfill the Link Specification. Clicking on a row will present the fetched data of botch matched instances above the table. Clicking on a link will try to dereferentiate the corresponding instance by its URI in a separat browser tab. The `value` column holds the similarity value of the matches.
To **download** the complete mapping click the `Save` button beneath the table. This will open a sub window, to choose different serialization format. As of now, we support Turte, N-Triple and Tab-seperated files.



Execute Link Specification

Execute Link Specification

Results

Source Instance of Persons21 (OAEI 2010 Benchmark)		Target Instance of Persons22 (OAEI 2010 Benchmark)	
PROPERTY	VALUE	PROPERTY	VALUE
age	31	age	31
date_of_birth		date_of_birth	
given_name	jacob	given_name	ja cob
has_address	http://www.okkam.org/oaie/person1-address132	has_address	http://www.okkam.org/oaie/person2-address1
phone_numer	03 18063431	phone_numer	03 18063431
soc_sec_id	8140262	soc_sec_id	8140262
surname	tindale	surname	tindalre

INFO	SOURCE URI	TARGET URI	VALUE
ℹ	http://www.okkam.org/oaie/person1-Person1220	http://www.okkam.org/oaie/person2-Person1221	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1220	http://www.okkam.org/oaie/person2-Person1222	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1320	http://www.okkam.org/oaie/person2-Person1321	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1320	http://www.okkam.org/oaie/person2-Person1322	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1320	http://www.okkam.org/oaie/person2-Person1324	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1320	http://www.okkam.org/oaie/person2-Person1325	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1320	http://www.okkam.org/oaie/person2-Person1326	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1400	http://www.okkam.org/oaie/person2-Person1401	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1440	http://www.okkam.org/oaie/person2-Person1442	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1440	http://www.okkam.org/oaie/person2-Person1443	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1440	http://www.okkam.org/oaie/person2-Person1445	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1440	http://www.okkam.org/oaie/person2-Person1446	0.5
ℹ	http://www.okkam.org/oaie/person1-Person1440	http://www.okkam.org/oaie/person2-Person1447	0.5

Figure 3.2: Mapping window presents results (Mapping) of a Link Specification. This are the results of executing the Link Specification of 3.1b.

Chapter 4

Selfconfiguration

SAIM also supports unsupervised machine learning method of Link Specifications, called Selfconfiguration. Once all endpoints are specified as described in chapter 2 this option is available. These machine learning methods rely on so-called Psuedo F-Measures. .
To start click the `Start Selfconfiguration` on the main window of SAIM.

cite EUCLID paper

Mesh based Selfconfiguration

There are several deterministic selfconfiguration algorithms available trying to discover high quality Link Specifications baes upon different classifier. Other parameters to adjust are:

- Beta $\beta \in [0,1]$ value of the Pseudo F-measure.
- Number of iterations to learn.
- Minimal coverage over all instances for a property.

Selfconfiguration based on Genetic programming

Besides the parameters described above the user can set additional parameters for the Genetic Programming approach. These are:

- Number of generations to evolve
- Number of individuals in a population
- Corssover and mutation probality

Whereas, the first two influence the runtime of the algorithm.

Chapter 5

Improving results through learning

SAIM also implements supervised learning algorithms presented in [3]. If all endpoints are specified as described in chapter 2 and an intial Lnk Specification has been set either manually or through self configuartion as decribed in chapter 3 and 4 respectively. The initial Link Specification is required to retrieve initial training data to be annoted by the user.
All learning algorithms are based on Genetic Programming. Beside the typical parameters like Population size and number of generations to evolve and mutation propability the user can set the `number of inquiries` presented each iteration. To start the learning process the user can choose between active and batch learning approaches. In order to get good results we recommend a higher number of inquiries. There are 3 active learning approaches implemented. The basic EAGLE active learning approach [3] and two enhancements using correlation features to better estimate the quality of examples to be classified. One is based on clustering the other uses a graph based features of spreading activation [4].
Either way, the user evaluates the examples through checkboxes added to a equivalent table representation of the possible links as depicted in figure 3.2. Again, the number of generations to evolve and the population size have the most influence at runtimes.
To inspect the best Link Specification learned the user can click on the `Get best solution` button.

Chapter 6

Additional features

6.1 Caching

6.2 Import and save Link Specifications

6.3 Uploading RDF dumps

Bibliography

[1] Axel-Cyrille Ngonga Ngomo. On link discovery using a hybrid approach. Journal on Data Semantics, 1:203 – 217, December 2012.

[2] Axel-Cyrille Ngonga Ngomo and Sören Auer. LIMES - A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data. In Proceedings of IJCAI, pages 2312–2317, 2011.

[3] Axel-Cyrille Ngonga Ngomo and Klaus Lyko. Eagle: Efficient active learning of link specifications using genetic programming. In Elena Simperl, Philipp Cimiano, Axel Polleres, Óscar Corcho, and Valentina Presutti, editors, Proceedings of ESWC, volume 7295 of Lecture Notes in Computer Science, pages 149–163. Springer, 2012.

[4] Axel-Cyrille Ngonga Ngomo, Klaus Lyko, and Victor Christen. Coala - correlation-aware active learning of link specifications. In Elena Simperl, Philipp Cimiano, Axel Polleres, Óscar Corcho, and Valentina Presutti, editors, Proceedings of ESWC, Lecture Notes in Computer Science, page to be published. Springer, 2013.

[5] Andriy Nikolov, Mathieu D'Aquin, and Enrico Motta. Unsupervised learning of data linking configuration. In Elena Simperl, Philipp Cimiano, Axel Polleres, Óscar Corcho, and Valentina Presutti, editors, Proceedings of ESWC, volume 7295 of Lecture Notes in Computer Science, pages 119–133. Springer, 2012.

Appendix A

A.1 The menu bar

A.2 Person1 Running example: sample Configuration

This is a sample configuartion we use as a running example throughout this manual. We link the Person11 and Person12 dataset of the Persons1 OEI2010 Benchmark.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE LIMES PUBLIC "LIMES"
"http://konradhoeffner.de/files/limes.dtd">
<LIMES>
  <PREFIX>
    <NAMESPACE>
         http://www.okkam.org/ontology_person2.owl#
       </NAMESPACE>
    <LABEL>prefix0</LABEL>
  </PREFIX>
  <PREFIX>
    <NAMESPACE>
         http://www.okkam.org/ontology_person1.owl#
       </NAMESPACE>
    <LABEL>www</LABEL>
  </PREFIX>
  <PREFIX>
    <NAMESPACE>
         http://www.w3.org/1999/02/22-rdf-syntax-ns#
```

```
        </NAMESPACE>
    <LABEL>rdf</LABEL>
</PREFIX>
<SOURCE>
    <ID>Person 11 (OAEI 2010 Benchmark))</ID>
    <ENDPOINT>SAIM\EPStore\person11.nt</ENDPOINT>
    <GRAPH>http://dbpedia.org</GRAPH>
    <VAR>?src</VAR>
    <PAGESIZE>-1</PAGESIZE>
    <RESTRICTION>
          ?src rdf:type www:Person
        </RESTRICTION>
    <PROPERTY>www:surname</PROPERTY>
    <PROPERTY>www:phone_numer</PROPERTY>
    <PROPERTY>www:soc_sec_id</PROPERTY>
    <PROPERTY>www:has_address</PROPERTY>
    <PROPERTY>www:date_of_birth</PROPERTY>
    <PROPERTY>www:given_name</PROPERTY>
    <PROPERTY>www:age</PROPERTY>
    <TYPE>N3</TYPE>
</SOURCE>
<TARGET>
    <ID>Person 12 (OAEI 2010 Benchmark))</ID>
    <ENDPOINT>
        SAIM\EPStore\person12.nt
        </ENDPOINT>
    <GRAPH/>
    <VAR>?dest</VAR>
    <PAGESIZE>-1</PAGESIZE>
    <RESTRICTION>
          ?dest rdf:type prefix0:Person
        </RESTRICTION>
    <PROPERTY>prefix0:surname</PROPERTY>
    <PROPERTY>prefix0:phone_numer</PROPERTY>
    <PROPERTY>prefix0:soc_sec_id</PROPERTY>
    <PROPERTY>prefix0:has_address</PROPERTY>
    <PROPERTY>prefix0:date_of_birth</PROPERTY>
    <PROPERTY>prefix0:given_name</PROPERTY>
    <PROPERTY>prefix0:age</PROPERTY>
    <TYPE>N3</TYPE>
</TARGET>
```

```
<METRIC>levenshtein(src.www:soc_sec_id,
```