SLINT+ Results for OAEI 2013 Instance Matching

Khai Nguyen[1] and Ryutaro Ichise[2]

[1] The Graduate University for Advanced Studies, Japan
nhkhai@nii.ac.jp
[2] National Institute of Informatics, Japan
ichise@nii.ac.jp

Abstract. The goal of instance matching is to detect identity resources, which refer to the same real-world object. In this paper, we introduce SLINT+, a novel interlinking system. SLINT+ detects all identity linked data resources between two given repositories. SLINT+ does not require the specifications of RDF predicates and labeled matching resources. SLINT+ performs competitively at OAEI instance matching campaign this year.

1 Presentation of the system

The problem of detecting instances co-referring to the same real-world object is positively important in data integration. It is useful for reducing the heterogeneity and warranting the consistency of data. The asynchronous development of linked data increasingly requires the specific linked data instance matching algorithms to connect existing instances and newly added instances.

In linked data, the differences of data representation appear not only in object values, but also in RDF predicates, which specify the meaning of objects properties. This issue is popularly found in different data repositories or even the same repository but different domains. Also, it separates the current solutions into two groups: schema-dependent and schema-independent. The first approach uses the descriptions of RDF predicate as the guide for matching while the second approach does not. In addition, the schema-independent approach consists of two minor branches: supervised learning and unsupervised learning, which involve with using or not using the labeled data of identity instances.

We develop SLINT [3] and its extension SLINT+ [2]. These systems are schema-independent and training-free. SLINT+ is used for instance matching track of OAEI 2013.

1.1 State, purpose, general statement

SLINT+ is a flexible schema-independent linked data interlinking system. SLINT+ can interlink various data sources, and is independent on the schema of data sources. By detecting appropriate predicate alignments without supervised learning, SLINT+ does not require expensive curation on data examples.

The principle of SLINT+ is similar to previous data interlinking systems. There are two main phases in the interlinking process of SLINT+: candidate generation and

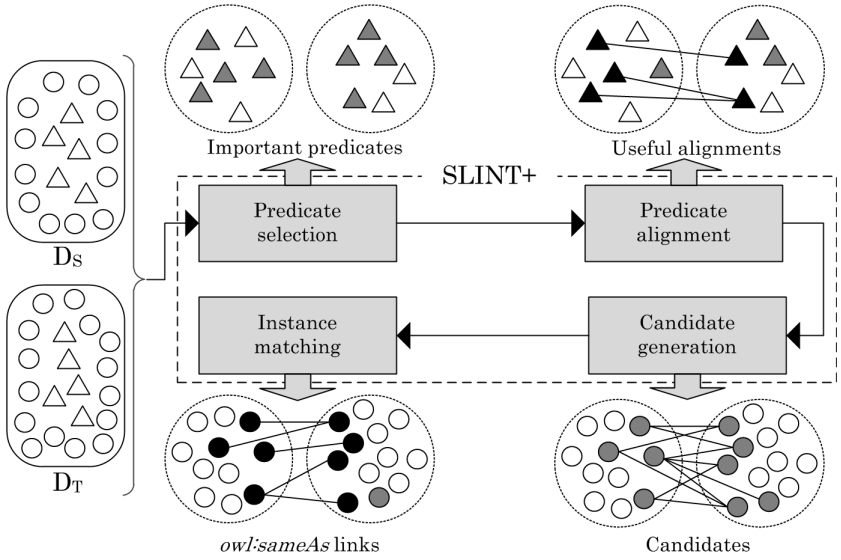

Fig. 1. The interlinking process of SLINT+

instance matching. The first phase separates similar instances into different groups in order to reduce the number of the pending pairs. The second phase will determine which candidate is really identity. With the schema-independent goal, we add two steps into SLINT+: predicate selection and predicate alignment. The mission of these new steps is to find the predicate alignments specifying the same properties of instances.

1.2 Specific techniques used

The architecture of SLINT+ is depicted in Fig.1. D_S and D_T are the source and target data. The predicate selection step finds the important predicate by some statistical measures based on RDF objects involving with each predicate. The predicate alignment step matches important predicates and selects the reasonable alignments. This step can be recognized as an instance-based ontology matching task. The candidate generation step picks up similar instances, which are predicted to be identity. The final step, instance matching, compares suggested candidates and produces the interlinking result. In the following sections, we describe the details of each step in order of the process.

1.2.1 Predicate selection

This is the first step of the interlinking process. It collects the important predicates of each input data sources. Important predicates are expected to be used by a large portion of instances and stored specific information of each instance. Thus, an important predicate should have high frequency and diver RDF objects.

We use coverage and discriminability as the metrics to evaluate the importance level of each predicate. These metrics are the extensions from [4]. Equation (1) and (2) in turn

are the formulas of coverage $Cov(p_k)$ of predicate p_k and its discriminability $Dis(p_k)$. The notation $< s, p, o >$ stands for subject, predicate, and object of a RDF triple. x, D, and f are the instance, data source, and frequency of RDF object, respectively.

$$Cov(p_k) = \frac{|\{x|x \in D, \exists t =< s, p_k, o >\in x\}|}{|D|} \tag{1}$$

$$\begin{aligned} Dis(p_k) &= \frac{Var(p_k) \times H(p_k)}{Var(p_k) + H(p_k)} \\ Var(p_k) &= \frac{|O_{p_k}|}{|\{t|\exists x \in D, t =< s, p_k, o >\in x\}|} \\ H(p_k) &= -\sum_{o_i \in O_{p_k}} \frac{f(o_i)}{\sum_{o_j \in O_{p_k}} f(o_j)} \times \log \frac{f(o_i)}{\sum_{o_j \in O_{p_k}} f(o_j)} \\ O_{p_k} &= \{o|\exists x \in D, t =< s, p_k, o >\in x\} \end{aligned} \tag{2}$$

A predicate is important if its coverage, discriminability, and the harmonic means of them are greater than given thresholds α, β, and γ, respectively. We select two sets of important predicates, from two input data sources. In the next step, we align these sets and find the useful predicate alignments.

1.2.2 Predicate alignment

In this step, we firstly group the predicates by their type. The type of a predicate is determined by the dominant type of its RDF objects. There are five predicate types used in SLINT+: string, URI, double, integer, and date. Secondly, we combine the type-similar predicates of source and target data to get raw predicate alignments. Confidence is the evidence for evaluating the usefulness of raw alignments. The confidence is estimated using the intersection of all RDF objects described by the predicates of each alignment. Equation (3) describes the confidence of the alignment between predicates p_i and p_j.

$$\begin{aligned} conf(p_i, p_j) &= \frac{|R(O_{p_i}) \cap R(O_{p_j})|}{|R(O_{p_i}) \cup R(O_{p_j})|} \\ O_{p_i} &= \{o|\exists x \in D_S, < s, p_i, o >\in x\} \\ O_{p_j} &= \{o|\exists x \in D_T, < s, p_j, o >\in x\} \end{aligned} \tag{3}$$

By using function R, the string, URI, and double are compared indirectly. For string and URI, R collects lexical words from given texts and links. For double, R rounds the values into two decimal points precision. For the remaining types, R uses the original values without transformation.

Only useful alignments whose confidence is greater than a threshold will be kept for the next steps. This threshold is computed by averaging the confidence of non-trivial alignments. An alignment is considered as non-trivial if its confidence is higher than threshold ϵ, a small value. In the next steps, useful alignments will be used as the specification for comparing instances.

1.2.3 Candidate generation

The goal of candidate generation is to limit the number of instances to be compared. SLINT+ performs a very fast comparison for each pair of instances. The result of this comparison is a rough similarity between instances. It is consolidated from their shared RDF objects, without any consideration for each predicate alignment. That is, two compared RDF objects can associate with two predicates having no alignments selected. Equation (4) is the rough similarity of instances x_S and x_T. In this equation, A is the set of filtered predicate alignments; R is the preprocessing procedure, as used in predicate alignment step; $w(O, S)$ and $w(O, T)$ are the weight of shared value O in each data source D_S and D_T, respectively. The weight of string and URI values is estimated by the TF-IDF score while that of remaining types is fixed to 1.0.

$$
\begin{aligned}
rough(x_S, x_T) = & \sum_{O \in R(O_S) \cap R(O_T)} w(O, D_S) \times w(O, D_T) \times sum(p_S) \times sum(p_T) \\
& O_k = \{o | \exists x \in D_k, < s, p_k, o > \in x\} \\
& sum(p_S) = \sum_{(p_S, p) \in A} conf(p_S, p) \\
& sum(p_T) \sum_{(p, p_T) \in A} conf(p, p_T)
\end{aligned}
\tag{4}
$$

Although the rough similarity is computed for each pair of instances, in technical aspect, it can easily be accumulated by passing through each repository only one time in combination with matrix representation for all rough similarities.

Candidates are the pairs whose similarity satisfies two conditions. First, it must be higher than the average similarity of all others λ, because not every instance in source data has the identities in target data. Second, it must be relatively higher than the similarity of the others, in which each instance of the current pair participating. In addition, we multiply the maximum similarity with a damping factor ζ to avoid the single pairing assumption.

1.2.4 Instance matching

For each candidate, we re-compute the similarity and then select identity pairs based on this measure. The similarity of two instances is calculated from the shared values in RDF objects, which are described by each pair of useful predicate alignments. The confidence of each alignment is used as the weight for the similarity. Concretely, the similarity function is given in equation (5). In this equation, R is similar with the previous steps; and $corr$ is the similarity function for RDF objects declared by p_S and p_T. $corr$ function works variously for different type of data. For string and URI, it computes the TF-IDF cosine similarity. For double and integer, it returns the inverted disparity. For date, it simply performs the exact matching.

$$sim(x_S, x_T) = \frac{1}{W} \times \sum_{<p_S, p_T> \in A} conf(p_S, p_T) \times corr(R(O_S), R(O_T)),$$
$$O_k = \{o | \exists x \in D_k, < s, p_k, o > \in x\} \quad (5)$$
$$W = \sum_{<p_S, p_T> \in A} conf(p_S, p_T)$$

Similar to selecting candidate in the previous step, we use η and θ as the same functions and estimations with λ and ζ in the candidate generation step.

The instance matching step closes the standard interlinking process of SLINT+. Next, we describe the configuration for participating OAEI 2013 instance matching.

1.3 Adaptations made for the evaluation

The instance matching track this year requests participant to connect instances between DBpedia and an anonymous synthesis repository. There are five test cases with different difficulty levels. To simplify the experiment, our aim is not using different parameters for each test case. Therefore, we select the configuration that conciliate the results and use it for all test cases. We installed the parameters of SLINT+ by testing the system on training data. We set α, β, γ to 0.01, ϵ = 0.01, ζ = 0.20 and θ = 0.75. We slightly modify the use of η and θ. Instead of using average value of similarities, we permanently set η and λ to 0.

We use BM25 weighting scheme as an alternative for TF-IDF modified cosine in computing the string similarity in instance matching step. To improve the quality of string matching, we remove the words whose frequency is under 0.25.

In addition, we use some unsupervised transformation on the data before inputting to the system. For test case #2, we leverage the information stored in linked instances. In order to obtain adequate properties for matching, we recover the hidden data by dereferencing the linked instances provided in RDF objects. For test case #3, #4, and #5, we use machine translation to get the English version of French strings stored in the target data. SLINT+ currently does not support multilingual matching. Translation from other languages into English is a prerequisite.

1.4 Link to the system, parameters file, and provided alignments

SLINT+ is available to be downloaded at http://ri-www.nii.ac.jp/SLINT/oaei2013.html. We also provide the parameters and the set of alignments for OAEI 2013 instance matching on this page.

2 Results

In this section, we report the experiment result of SLINT+. The results of candidate generation and instance matching are separately reported. To evaluate the final result

Table 1. Candidate generation result

Test case	Number of candidates	Pair completeness	Reduction ratio
#1	2675	0.995	0.986
#2	3602	1.000	0.994
#3	5000	0.995	0.971
#4	5438	0.987	0.968
#5	7177	0.991	0.967

Table 2. Instance matching result

Test case	Recall	Precision	F1
#1	0.979	0.979	0.979
#2	0.998	1.000	0.999
#3	0.913	0.935	0.924
#4	0.909	0.912	0.910
#5	0.880	0.875	0.878

Table 3. Total runtime

Test case	Runtime (in millisecond)
#1	409
#2	421
#3	342
#4	283
#5	353

of data interlinking process, we use the conventional measures: Recall, Precision, and F1 score. To evaluate candidate generation, we use Pair completeness and Reduction ratio [4]. Pair completeness expresses how many actual identity pairs are selected to the candidate set. Reduction ratio is the compression level of instance matching pool comparing with all possible instances pairs between given data sources. We also report the runtime of SLINT+. The experiment was conducted on a computer running with core 2 quad 2.66 GHz CPU, 4 GB of RAM, and Windows 7 64 bit version.

The result of candidate generation, instance matching, and time consumption are given in Table 1, Table 2, and Table 3, respectively.

The results of candidate generation are very good when reserving at least 98.7% of correct alignments and nearly 100% on test case #2. In addition, the largest number of candidate is only 7177, which reduces 96.7% of total instance pairs.

The final results of SLINT+ are generally good. Comparing Recall and Pair completeness on each test, they are similar on test case #1 and #2 and about 7% different on remaining test cases. In addition, SLINT+ performs a stable interlinking since the precision and recall are equivalent for all test cases.

The main reason for the lower result on test case #3, #4, and #5 comes from the transformation of string values. We temporarily used machine translation to convert strings from French to English before conducting the matching process. A better translation strategy may boost the Recall of SLINT+ on these test cases.

The runtime of SLINT+ is very short. It takes about 350 milliseconds for SLINT+ to finish each test case. This is a promising indication for designing a scalable system based on SLINT+ in the future.

3 General comments

3.1 Comments on the OAEI 2013 test cases

Comparing with recent years, the test cases of this time are more interesting and challenging. It is very reasonable when OAEI organizers publish the offline data for keeping the same input for all participants. In addition, it is effective that the test cases can inspect the advantages of each system.

The test cases for this year assume that every instance in source data has an identical one in target data. In our opinion, it could be more efficient if there is a test case that does not imply this assumption. Besides that, the various sizes of data can help evaluate the performance of participants.

3.2 Comments on the OAEI 2013 measures

Since most interlinking systems generate potentially identity instances before matching, we suggest to evaluate this step in separation with instance matching, as also was recommended in [1]. There are many recognized measures in assessing the quality of this step, such as recall, and reduction ratio as we reported in this paper.

4 Conclusion

We introduced SLINT+, a schema-independent and training-free linked data interlinking system. SLINT+ performs four steps interlinking including predicate selection, predicate alignment, candidate generation, and instance matching. SLINT+ gives a promising result at the campaign this year.

Implementing a graph matching algorithm is our objective in improving SLINT+. Since linked data is basically a graph, leveraging linking characteristics between instances will result more confident matching quality. Improving SLINT+ to a scalable system is also our current goal.

References

1. Euzenat J (2012). A modest proposal for data interlinking evaluation. In: ISWC'12 7th Workshop on Ontology Matching, pp. 234.
2. Nguyen K, Ichise R, Le B (2012). Interlinking linked data sources using a domain-independent system. In: 2nd Joint International Semantic Technology, pp.113-128, LNCS 7774.
3. Nguyen K, Ichise R, Le B (2012). SLINT: A schema-independent linked data instance matching system. In: ISWC'12 7th Workshop on Ontology Matching, pp. 1-12.
4. Song D, Heffin J (2011). Automatically generating data linkages using a domain-independent candidate selection approach. In: ISWC' 11, pp. 649-664.