A Relational Learning Approach for Collective Entity Resolution in the Web of Data

Gustavo de Assis Costa[1,2] , José Maria Parente de Oliveira[2]

[1] Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Brazil
[2] Divisão de Ciência da Computação, Instituto Tecnológico de Aeronáutica, Brazil
Praça Mal. Eduardo Gomes, 50, Vila das Acácias, São José dos Campos - SP, Brazil
{gacosta, parente}@ita.br

Abstract. The integration of different datasets in the Linked Data Cloud is a key aspect to the success of the Web of Data. To tackle this problem most of existent solutions have been supported by the task of entity resolution. However, many challenges still prevail specially when considering different types, structures and vocabularies used in the Web. Another common problem is that data usually are incomplete, inconsistent and contain outliers. To overcome these limitations, some works have applied machine learning algorithms since they are typically robust to both noise and data inconsistencies and are able to efficiently utilize nondeterministic dependencies in the data. In this paper we propose an approach based in a relational learning algorithm that addresses the problem by statistical approximation method. Modeling the problem as a relational machine learning task allows exploit contextual information that might be too distant in the relational graph. The joint application of relationship patterns between entities and evidences of similarity between their descriptions can improve the effectiveness of results. Furthermore, it is based on a sparse structure that scales well to large datasets. We present initial experiments based on BTC2012 datasets.

Keywords: Entity resolution, Semantic web, Linked data, Machine Learning, Relational Learning.

1 Introduction

Following the trend of the World Wide Web, a considerable amount of data has been published in the Web of Data. As a result, the challenges in handling all this data has grown at the same rate.

RDF datasets are usually created from some data conversion process, importing the same existing issues of databases as, e.g., outliers, duplication, inconsistency, and other, like schema heterogeneity. These kind of restrictions poses an hindrance to the effective integration and sharing of linked data.

The key point of LOD[1] (Linked Open Data) is in dataset integration. One way to materialize what is expected from this scenario would be to provide interlinkage of entities, since these are key elements for data representation. Unconnected descriptions of the same thing can be obtained from different sources. This way the semantic value that could be obtained with the link between different datasets would be lost.

In the last years, many works have addressed the task of Entity Resolution (ER) which deals with extracting, matching and resolving entity mentions in structured and unstructured data. In linked data research community it is recognized as a prominent issue. Also known as record linkage, de-duplication, co-reference resolution, instance matching, among others, it has been used to look for interrelationships, previously unknown, between different representations of the same real world entity.

In Big "Linked" Data era, the need for high quality entity resolution is only growing. We are inundated with more and more data that needs to be integrated, aligned and matched before further utility can be extracted [1].

From this scenario, some posed challenges deserve attention. The first is deal with semi-structured data. Different semantic description structures can be employed to refer to the same element, e.g., the description of entities of the same type. A second challenge is related to noise in the data. As already stated, there are various problems related to literal descriptions of data and existent solutions adopt metrics that still cannot resolve problems like attributes without value.

Our approach is based in a statistical approximation method that captures joint evidence of similarity related to values of descriptions of entities and any relationship correlation through existing entity-entity predicates. The overall process is composed of four steps: 1) Preprocessing, 2) Pair-wise string similarities 3) Similarity evidences modeled as a matrix entity-attribute 4) Relationships between entities modeled as a tensor. Finally, the problem is formulated as a coupled matrix and tensor factorization. This work will be supported by applying an extended version of RESCAL [2] model, a tensor factorization model for relational learning.

The main contributions of this approach are as follows:

1. We use an extension of RESCAL model by exploring different types of descriptions of an entity. Aiming to increase the number of evidences to be used in the model, thereby increasing the effectiveness of the approach, we propose to perform similarity computation with three types of literals: attributes, URI infixes and predicates.
2. We propose an approach of collective entity resolution [3] that leverages the global interactions model of a RDF graph. It computes a global latent-component representation of the entities and local interaction-models of the latent variables for each predicate. When considering the co-occurrence of different entities, we can perform joint analysis based not only on the representations of an entity, but also on the representations of other entities that are related to this first. Thus, it is possible to increase the accuracy of the results even considering noisy and inconsistent data.

[1] http://linkeddata.org/

3. Besides comparative analysis with other solutions, we show initial experiments that demonstrate the improved effectiveness results of our approach in large datasets.

The remainder of this paper is structured as follows: Section 2 presents an architectural overview of the approach, while in Section 3 we present the process of computing and modeling similarity evidences into the entity-attribute matrix that will be coupled in the factorization, Section 4 describes the statistical relational learning model and its extension, while experimental results are reported in Section 5. Section 6 summarizes related works and in Section 7 we conclude and outline our future work.

2 Overall approach

Figure 1 depicts an overview of our approach. The literal information for each entity in all datasets are extracted. There are some discriminative literal descriptions that, considering the triple structure, extracts the most significant information of each element:

- Attribute values. Correspond to features of an entity (e.g. name/label, birth date, profession). Most approaches explore these values due to the precision when identifying an entity;
- URI infix. In Papadakis et. al. [4] experiments results showed that approximately 66% of the 182 million URIs of a dataset follow a common pattern: the Prefix-Infix(-Suffix) scheme. Each component of this form plays a special role: the Prefix part contains information about the source (i.e., domain) of the URI, the Infix part is a sort of local identifier, and the optional Suffix part contains either details about the format (e.g., .rdf and .n3), or a named anchor.
- Predicate: We will use the last token (normalized) of the URI, e.g., “has spouse” for "fb:has_spouse".

Considering two datasets A and B with *n* and *m* entities respectively, and a brute force algorithm, there will be at least *n* x *m* comparisons between instance pairs. It is impractical, especially when dealing with real world datasets, with millions of triples or even larger. To overcome this problem, we perform a preprocessing step to obtain the possible matching pairs. An inverted index is built for instances of some key words in the descriptions to efficiently determine potential candidates. The entities sharing the same keys in the index are considered to be candidate matching instances.

After literal information extraction, we apply the similarity metrics for the candidates and generate the entity-attribute matrix, with its entries corresponding to an entity having or not certain attribute or an infix in its URI description. In the same way, entity-entity relations from datasets are mapped to tensor, with which we perform the coupled factorization. The factor-matrix A computed in the above process can be interpreted as an embedding of the entities into a latent-component space that reflects their similarity over all relations in the domain of discourse.

In order to retrieve entities that are similar to a particular entity ***e*** with respect to all relations in the data, we compute a clustering in the latent-component space. Initially, however, we normalize the rows of A, such that each row represents the normalized participation of the corresponding entity in the latent components. From feature vectors corresponding to each entity (matrix rows) it is possible to create clusters of similar entities, since matrix A represents entities by their participation in the latent components. The clustering will be determined by the entities' similarity evidences in the relational domain.

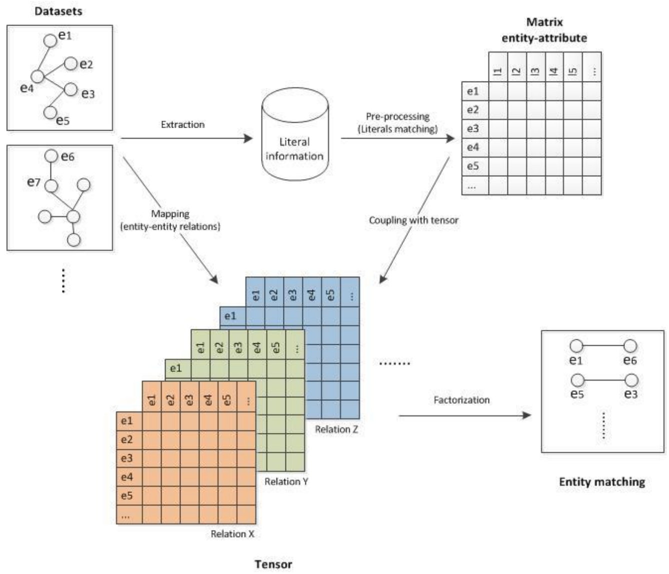

Fig. 1. An overview of the proposed framework

3 Computing and modeling similarity evidences

We aim to extract the most evidences about entities since the performance of our approach is greatly influenced by the quality of feature extraction. This section describes the essential step for increasing efficiency results obtained from the RESCAL factorization model.

3.1 Information Extraction

We try to extract the greatest number of literal information from the RDF triples. The literals can be grouped in three types: a) Attribute values; b) URI Infixes and c) Predicates. Attribute values can be a feature, a description or even an associated event, that when considered jointly, can uniquely identify an entity. It's important to point out that not always we have all the values informed or even they can be inconsistent.

A second type of literal that can be of relevance to the process are the URI infixes of subjects or objects within triples. As stated in [4], it could be expected that Infixes of URIs, which are more source-independent than the Prefixes and the Suffixes, can contain the most discriminative information for the similarity task within a URI. Despite the high heterogeneity in the Prefixes of the URIs, the Infix remains the same. The Suffix is optional and can be ignored when matching URIs. Table 1 illustrates an example with two URIs that refer to the same person but are syntactically different. Specifically in this case, even the infixes are different, necessarily requiring the application of a string similarity metric.

Prefix	Infix	Suffix
http://liris.cnrs.fr	/olivier.aubert	/foaf.rdf#me
http://bat710.univ-lyon1.fr	/˜oaubert	/foaf.rdf#me

Fig. 2. Example of two URIs that refer to the same person

By definition, a predicate is the second part of an RDF statement and defines the property for the subject of the statement. Unlike a subject or object, a predicate must always be a URI. From an empirical analysis it appears that property matching is not trivial since the datasets were usually designed with their own ontologies. Nevertheless, if we make analyzes of synonymy it is possible to overcome results obtained only with similarity metrics. Some other linguistic facts, like polysemy and homonymy, are not treated because the isolated weight of these phenomena have little significance when considering all the contextual facts involved in the likelihood estimation of similarity in the model.

3.2 Strategies for evaluating and modeling the similarity

Some different similarity metric functions can be used for different types of literal information. Each of the literals have its own characteristics and thus we consider the application of different metrics and strategies to analyze string similarity.

For attribute values we use TF-IDF. An attribute can contain a great diversity of values, with different sizes and types. When performing the algorithm we will treat each string/label relative to entities as a set of values (documents). The TF-IDF weight is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus.

As URI infixes are normally composed of little strings to represent a single identifier, we decided to use Edit-Distance metric. It is a metric that measures the distance between two words as the minimum number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other.

Finally, it's important to highlight that predicates frequently involve verbs, which can appear in a wider variety of forms than nouns. They also contain often more functional words, such as articles and prepositions. In this case we try to overcome the difficulty by using a combination of the metric of Edit-distance with the Wordnet to perform a jointly analysis that considers besides the string similarity, an analysis of synonyms between the two descriptions. In this strategy, when the similarity metric result is below to some predefined threshold, we apply the analysis of synonymy.

After processing all the similarities we have to model the evidences in the entity-attribute matrix. The matrix D of size *n x l* is composed of *n* entities at rows and *l* literal evidences at columns. A matrix entry D_{ij} = 1 denotes that an entity have certain attribute value. Otherwise, if the entity does not have this attribute it will be set to 0. If two entities share the same attribute, but with different attribute values, they will not share the same entry in the table, i.e, each of them will have its distinct entry set to 1. In the sense of normalizing a set of similar values according to the metric, each column will be represented by one canonical value randomly chosen from the set, i.e., whether the values are slightly different, there will be just one entry . Although URI infixes and predicates are not strictly attribute values, we will handle them the same way as attributes. As a result, the matrix will contain all the literal evidences obtained from the performance of different metric and strategies of similarity.

4 Statistical relational learning model

We now present the model that we applied in our approach. Firstly, we need to identify the key elements of the model which in turn are the entities and its relations. The entities are given by the set of all resources, classes and blank nodes in the data, while the set of relations consists of all predicates that include relationships between entities. Once these elements were extracted from datasets we set out to the transformation of them into a tensor representation.

A tensor is a multidimensional array. More formally, an *N-way* or N_{th}-order tensor is an element of the tensor product of N vector spaces, each of which has its own coordinate system. A third-order tensor has three indices as shown in Figure 3. A first-order tensor is a vector, a second-order tensor is a matrix, and tensors of order three or higher are called higher-order tensors[5].

Assuming that our relational domain consists of *n* entities and *m* relation types, data is modeled as a three-way tensor X of size $n \times n \times m$, where the entries on two modes (dimensions) of the tensor correspond to the combined entities of the domain of discourse and the third mode holds the m different types of relations.

A tensor entry $X_{ijk}= 1$ denotes the fact that the *k-th* relation (*i-th* entity, *j-th* entity) exists. Otherwise, for non-existing or unknown relations, X_{ijk} is set to zero.

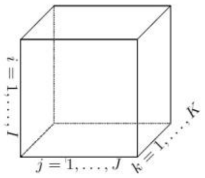

Fig. 3. Tensor model for relational data. i and j represent entities and k the relationships.

Besides the modeling aspect, the motivation to use tensor factorization is due to its power of prediction when used as machine learning task, as is done in SVD method for example. The process of factorization decomposes an observed matrix/tensor into latent (or hidden) factors. Latent factors can be interpreted as new features that have been invented to describe the data.

In RESCAL, learning is performed using the latent components of the model (Fig. 4). The approach employs the rank-r factorization as follows, where each segment is factored as X_k

$$X_k \approx A R_k A^T \textit{ where } k = 1, ..., m \tag{1}$$

A is a *n x r* matrix containing the components of the latent representation of the entities in the domain and R_k is an asymmetric matrix *r x r* modeling the interactions of the components of the k_{th} latent predicate. The rows of the factor matrices *A* and *R* can be considered latent-variable representations of entities that explain the observed variables X_{ij}, the columns can be considered the invented latent features and the entries of the factor matrices specify how much an entity participates in a latent feature.

The factor-matrices *A* and R_k are computed by solving a regularized minimization problem [2] applying an alternating least squares algorithm (RESCAL-ALS), which updates *A* and R_k iteratively until a convergence criterion is met (linear regression). In order to retrieve entities that are similar to a particular entity *e* with respect to all relations in the data, it is sufficient to compute a ranking of entities by their similarity to *e* in A.

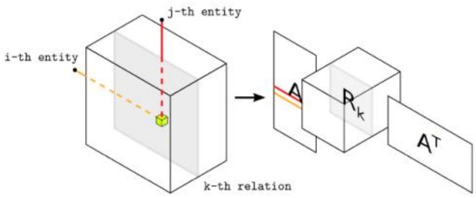

Fig. 4. Illustration of data representation and factorization in the model

Once this model assumes that two of the three modes are defined by entities, the process becomes limited to RDF resources. So we used an extension of the model, coupling the entity-attribute matrix with the tensor (Fig. 5) aiming to perform the factorization [6, 7].

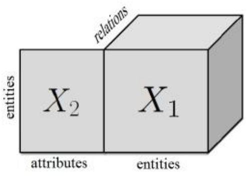

Fig. 5. A Tensor coupled with a matrix of attributes.

If we include all the literal evidences in the tensor, a huge amount of entries would be wasted, which would lead to an increased runtime since a significantly larger tensor would have to be factorized. So, the idea is to add the predicate-value pairs to a separate entity-attributes matrix D and not to the tensor X. The entity-attributes matrix D is then factorized into

$$D \approx AV \qquad (2)$$

where A is the entities' latent-component representation of the model and V is an *r x l* matrix, which provides a latent-component representation of the literals. To include this matrix factorization as an additional constraint on A in the tensor factorization of X, it is necessary to adjust the minimization problem.

In figure 6 we show an illustration that depict an example. The latent-component representations of entities **A** and **B** will be similar to each other in this example, as both representations reflect that their corresponding entities are related to the same object (wikipedia page) and attribute value. Because of this and their own similarity evidences, **C** and **D** will also have similar latent-component between their representations. Consequently, the latent feature vector of **A** will yield similar values to the latent feature vector of **B** and as such the likelihood of matching can be predicted correctly. The attribute values are only considered here due to the extension of the model.

Considering that a_i and a_j denote the i-th and j-th row of A and thus are the latent-component representations of the i-th and j-th entity, the products

1) $a_{\text{fb:m.05mwy8}}^{T} R_{\{\text{spouse\_s, isMarriedTo}\}}\ a_{\text{fb:m.0pc9q}}$,
2) $a_{\text{fb:m.05mwy8}}^{T} R_{\{\text{spouse\_s, isMarriedTo}\}}\ a_{\text{yago:Luiz\_Inácio\_Lula\_da\_Silva}}$,
3) $a_{\text{yago:Marisa\_Leticia\_Lula\_da\_Silva}}^{T} R_{\{\text{spouse\_s, isMarriedTo}\}}\ a_{\text{fb:m.0pc9q}}$
4) $a_{\text{yago:Marisa\_Leticia\_Lula\_da\_Silva}}^{T} R_{\{\text{spouse\_s, isMarriedTo}\}}\ a_{\text{yago:Luiz\_Inácio\_Lula\_da\_Silva}}$

along with the similarities evidences obtained, will contribute to get likelihood of A and B representing the same real world entity.

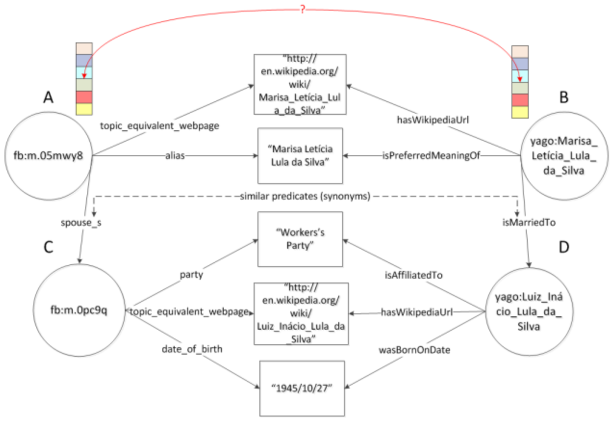
?
A
B
C
D
fb:m.05mwy8
yago:Marisa_Letícia_Lula_da_Silva
fb:m.0pc9q
yago:Luiz_Inácio_Lula_da_Silva
"http://en.wikipedia.org/wiki/Marisa_Letícia_Lula_da_Silva"
"Marisa Letícia Lula da Silva"
"Workers's Party"
"http://en.wikipedia.org/wiki/Luiz_Inácio_Lula_da_Silva"
"1945/10/27"
topic_equivalent_webpage
alias
hasWikipediaUrl
isPreferredMeaningOf
similar predicates (synonyms)
spouse_s
isMarriedTo
party
isAffiliatedTo
topic_equivalent_webpage
hasWikipediaUrl
date_of_birth
wasBornOnDate

Fig. 6. Illustration with representations of the same real world entities in Freebase and YAGO. The red line indicates the wanted matching.

5 Experiments

We report the experimental results on benchmark datasets in OAEI 2010 and 2011. We used these two editions of the campaign aiming to get more comparative results. All algorithms have been implemented in Python, NumPy, RDFLib, NLTK and Scikit-learn, respectively, and were evaluated on 3,8 GHz Intel (R) Core i7 CPU machine with 4 cores and 64 GB RAM, except where otherwise noted.

The first dataset is a small real dataset, which includes two collections of RDF data files concerning persons (denoted by Person1 and Person2, respectively) and one collection about restaurants. OAEI 2010 organizers provided reference mappings for each collection, where each mapping contains two URIs from different data files that denote the same person or restaurant. The goal of our evaluation on the dataset is two-fold. First, we want to test various values for the parameters in our approach and apply the best ones to the experiments. Second, we can compare the results obtained with other systems on the same dataset.

We compared the results of our approach (Relational) with other four entity/coreference resolution systems, namely ASMOV, CODI, LN2R and RiMOM, which also submitted their results on the same dataset to OAEI. ASMOV and CODI employed similarity-based matchers to obtain coreferent URIs and performed logical inference to remove inconsistent results. LN2R integrated a knowledge-based matcher to find semantically coreferent URIs and adopted a similarity propagation algorithm to generate similarities. RiMOM is a purely similarity-based system, which integrated many matchers to exploit a range of characteristics for both concepts and instances. All of them can only deal with pairwise instances, which are precisely called instance matching systems.

The comparison results on F-Measure is depicted in Table 1. From the table, we can observe that our approach achieved the best F-Measure in average on the dataset. In particular, our Precision result is quite good , because we extracted a sensible number of evidences for the resolution process.

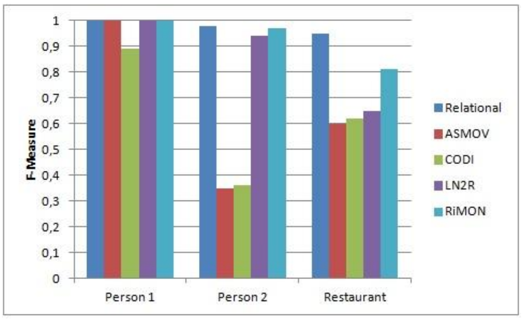

Table 1. F-Measure comparison among approaches on the benchmark test

In the second dataset, we compared our Relational approach with AgreementMaker, SERIMI, and Zhishi.Links, and that results were obtained from

OAEI 2011 Instance Matching Campaign. AgreementMaker and Zhishi.Links are approaches that can be used only in one domain. Table 2 shows the comparative results with our approach. According to results, we got very much higher precision and recall if compared with other systems. Relational obtained good performance specifically on D4 and D5 datasets.

Dataset	Relational			Agree.Maker			Zishi.Links			SERIMI		
	PR	**RC**	**F1**	**PR**	**RC**	**F1**	**PR**	**RC**	**F1**	**PR**	**RC**	**F1**
D1	0.98	0.97	0.97	0.79	0.61	0.69	0.92	0.91	0.92	0.69	0.67	0.68
D2	0.97	0.95	0.96	0.84	0.67	0.74	0.90	0.93	0.91	0.89	0.87	0.88
D3	1.0	0.96	0.98	0.98	0.80	0.88	0.97	0.97	0.97	0.94	0.94	0.94
D4	0.98	0.95	0.96	0.88	0.81	0.85	0.90	0.86	0.88	0.92	0.90	0.91
D5	0.97	0.96	0.96	0.87	0.74	0.80	0.89	0.85	0.87	0.92	0.89	0.91
D6	1.0	1.0	1.0	0.97	0.95	0.96	0.93	0.92	0.93	0.93	0.91	0.92
D7	1.0	1.0	1.0	0.90	0.80	0.85	0.94	0.88	0.91	0.79	0.81	0.80
H-mean	0.97	0.97	0.97	0.92	0.80	0.85	0.93	0.92	0.92	0.89	0.88	0.89

Table 2. Our relational approach compared with other systems results on OAEI2011 dataset

In the next step we proceed to perform preliminary tests with our approach with large datasets, more specifically with some datasets from BTC 2012 (Billion Triples Challenge) [8], respectively RESTL and Freebase. These datasets contains approximately 122 million RDF n-quads triples. Firstly, we remove provenance information, duplicate triples, RDF blank nodes as well as reification statements. In the first round, it was generated a total of 675,244 *sameAs* links. The precision achieved was of 82% and the recall was of 75%. In the second round approximating 1 million links was generated, but the precision has dropped to 74% and the recall was of 68%. After all, the memory capacity was one of the big barriers we had to face, which showed a major drawback of the approach. Although the sparse nature of the matrix and tensor, we still had the problem of scale, dealing with millions of entities at the same time. Even so, we believe that the relational learning approach could result in the selection of the most likely mappings. Although, it is important to note that conducting more experiments is needed to deepen the discussion.

6 Related work

The problem of entity resolution has emerged as an important task to the Web of Data. This same task has been exploited in many different research arenas.

Some approaches are based on domain specific solutions. One first tool is SILK - Link Discovery Framework [9]. It is a well-known tool for publishing and managing RDF relationship between two RDF datasets. The framework provides a declarative language in which different string similarity metrics, defined by the user, can be manually combined. Raimond et al. [10] addresses the problem in the domain of music, modeling datasets as graphs, performing mapping between the graphs. Sleeman and Finin [11] and Shi et al. [12] present solutions for solving FOAF entities using logical constraints. Another group of solutions are domain independent.

Among them, there are some papers that addresses logical constraints like functional/inverse functional properties and cardinality as key aspects in their solutions [13, 14, 15]. Using this properties is not sufficient to find traces of similarity in LOD. Hogan et. al. [16] tries to find more inverse functional properties with a statistical method. However, datasets must share the same vocabulary.

Some papers focus on improving the efficiency on the matching. Ngomo and Auer [17] present LIMES framework in order to circumvent the scalability problem by applying the method of triangulation in metric spaces. Song and Heflin [18] generate candidates by indexing some key words of instances. To our knowledge, as our work, few papers [4, 19, 20] focus on improving the effectiveness of matching with RDF data.

7 Conclusions

In this paper we presented a new approach for collective entity resolution supported by a linear regression statistical model. We use an algorithm that models the problem in a tensor, a prominent mathematical structure that fit nicely to the dyadic structure of RDF triples. The tensor model is one of the key strength of the approach, as it allows to include the influence of all the relationship patterns from a dataset. The experimental results on the datasets showed that, as expected, the relationship patterns can improve the results, considering that as much more evidences the better the effectiveness. However this has the cost that most of the solutions have to confront, the noise nature of data. We believe that our approach has achieved reasonably good results due to the similarity strategies we used.

In the near future, we intend to investigate ways of parallelizing both the preprocessing/extraction step and the factorization step, all based on the MapReduce paradigm. We assume that for the preprocessing step we can improve our approach by not discarding some false positives and for the factorization step we can slice data to be processed in each node of a cluster.

References

1. L. Getoor and A. Machanavajjhala, "Entity resolution: theory, practice and open challenges," *Proc. VLDB Endow.*, vol. 5, no. 12, pp. 2018–2019, Aug. 2012.
2. M. Nickel, V. Tresp, and H.-P. Kriegel, "A Three-Way Model for Collective Learning on Multi-Relational Data," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, New York, NY, USA, 2011, pp. 809–816.
3. I. Bhattacharya and L. Getoor, "Collective entity resolution in relational data," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, Mar. 2007.

4. G. Papadakis, G. Demartini, P. Fankhauser, and P. Kärger, “The Missing Links: Discovering Hidden Same-as Links Among a Billion of Triples,” in *Proceedings of the 12th International Conference on Information Integration and Web-based Applications &#38; Services*, New York, NY, USA, 2010, pp. 453–460.
5. T. G. Kolda and B. W. Bader, “Tensor Decompositions and Applications,” *SIAM Review*, vol. 51, no. 3, pp. 455–500, Aug. 2009.
6. Y. K. Yilmaz, A.-T. Cemgil, and U. Simsekli, “Generalised Coupled Tensor Factorisation,” in *Proceedings of Neural Information Processing Systems (NIPS)*, Granada, SPAIN, 2011, pp. 2151–2159.
7. M. Nickel, V. Tresp, and H.-P. Kriegel, “Factorizing YAGO: scalable machine learning for linked data,” in *Proceedings of the 21st international conference on World Wide Web*, New York, NY, USA, 2012, pp. 271–280.
8. A. Harth, *Billion Triples Challenge data set*. 2012.
9. J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov, “Discovering and Maintaining Links on the Web of Data,” in *Proceedings of the 8th International Semantic Web Conference*, Berlin, Heidelberg, 2009, pp. 650–665.
10. Y. Raimond, C. Sutton, and M. Sandler, “Automatic Interlinking of Music Datasets on the Semantic Web,” presented at the Linked Data on the Web (LDOW2008), 2008.
11. J. Sleeman and T. Finin, “Learning Co-reference Relations for FOAF Instances.,” in *ISWC Posters&Demos*, 2010, vol. 658.
12. L. Shi, D. Berrueta, S. Fernández, L. Polo, and S. Fernańdez, “Smushing RDF instances: are Alice and Bob the same open source developer?,” in *Personal Identification and Collaborations: Knowledge Mediation and Extraction*, Karlsruhe, Germany, 2008, vol. 403.
13. E. Ioannou, O. Papapetrou, D. Skoutas, and W. Nejdl, “Efficient semantic-aware detection of near duplicate resources,” in *Proceedings of the 7th international conference on The Semantic Web: research and Applications - Volume Part II*, Berlin, Heidelberg, 2010, pp. 136–150.
14. A. Hogan, A. Zimmermann, J. Umbrich, A. Polleres, and S. Decker, “Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora,” *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 10, no. 0, pp. 76–110, Jan. 2012.
15. W. Hu, Y.-Z. Qu, and X.-Z. Sun, “Bootstrapping Object Coreferencing on the Semantic Web,” *Journal of Computer Science and Technology*, vol. 26, no. 4, pp. 663–675, 2011.
16. A. Hogan, A. Polleres, J. Umbrich, and A. Zimmermann, “Some entities are more equal than others: statistical methods to consolidate Linked Data,” in *Proceedings of the Workshop on New Forms of Reasoning for the Semantic Web: Scalable & Dynamic (NeFoRS2010)*, Heraklion, Greece, 2010.
17. A.-C. Ngomo and S. Auer, “LIMES A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data,” in *IJCAI*, 2011, pp. 2312–2317.
18. D. Song and J. Heflin, “Automatically generating data linkages using a domain-independent candidate selection approach,” in *Proceedings of the 10th international conference on The semantic web - Volume Part I*, Berlin, Heidelberg, 2011, pp. 649–664.
19. G. Papadakis, E. Ioannou, T. Palpanas, C. Niederee, and W. Nejdl, “A Blocking Framework for Entity Resolution in Highly Heterogeneous Information Spaces,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 12, pp. 2665–2682, Dec. 2013.
20. C. Böhm, G. de Melo, F. Naumann, and G. Weikum, “LINDA: Distributed Web-of-data-scale Entity Matching,” in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, New York, NY, USA, 2012, pp. 2104–2108.