Towards Joint Inference for Complex Ontology Matching

Christian Meilicke and **Jan Noessner** and **Heiner Stuckenschmidt**
Research Group Data and Web Science, University of Mannheim

Abstract

In this paper, we show how to model the matching problem as a problem of joint inference. In opposite to existing approaches, we distinguish between the layer of labels and the layer of concepts and properties. Entities from both layers appear as first class citizens in our model. We present an example and explain the benefits of our approach. Moreover, we argue that our approach can be extended to generate correspondences involving complex concept descriptions.

Introduction

In open systems, different parties will define different ontologies, in the following referred to as $\mathcal{O}_1$ and $\mathcal{O}_2$, to describe the same or overlapping domains. Ontology matching systems implement algorithms to create links between concepts and properties defined in $\mathcal{O}_1$ and $\mathcal{O}_2$ (Euzenat and Shvaiko 2007). These links are called correspondences, a set of correspondences is called an alignment. Typical examples for correspondences, if interpreted in a strict way, are equivalence axioms between concepts from $\mathcal{O}_1$ and $\mathcal{O}_2$.

The matching process can be divided into the phase of generating matching hypotheses, and the phase of selecting the final alignment from these hypotheses. With respect to the second phase, it has been proposed to select a coherent alignment that is optimal with respect to the confidence values of the incoming hypotheses (Niepert, Meilicke, and Stuckenschmidt 2010; Albagli, Ben-Eliyahu-Zohary, and Shimony 2012). Such approaches are based on the strict distinction between the first and the second phase.We propose a model that weakens the border between those two phases. In particular, we do not add correspondences as hypotheses to the optimization problem, but hypotheses about the meaning of tokens that appear in the labels.

Motivation

Figure 1 depicts fragments of the concept hierarchy of $\mathcal{O}_1$ and $\mathcal{O}_2$. Both ontologies are based on a similar conceptualization. However, $\mathcal{O}_1$ uses the word `Contribution`, while $\mathcal{O}_2$ uses the word `Paper`. This choice is reflected in the whole fragment shown and might also appear at other

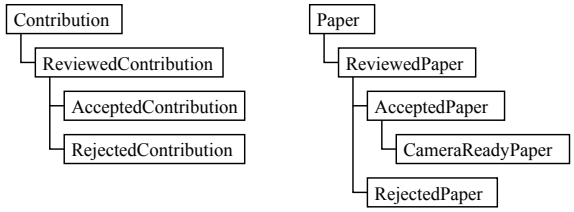

Figure 1: Fragments of ontologies $\mathcal{O}_1$ (left) and $\mathcal{O}_2$ (right).

labels in $\mathcal{O}_1$ or $\mathcal{O}_2$, e.g., a label `reviewContribution` might be used to describe a property in $\mathcal{O}_1$.

Now suppose that a matching system cannot detect that `Contribution` and `Paper` have the same meaning. This will also have an impact on the analysis of the compound labels. The outcome of the first phase will be an empty set, and the same will be the case for the whole process.

However, each of the concepts in $\mathcal{O}_1$ has an equivalent counterpart in $\mathcal{O}_2$. Thus, an alignment $\mathcal{A}$ should consist of four correspondences. Several observations can be treated as evidence for $\mathcal{A}$. The modifiers `Submitted`, `Accepted`, and `Rejected` appear in concept labels in both $\mathcal{O}_1$ and $\mathcal{O}_2$. Moreover, $\mathcal{A}$ is coherent and obeys to the subsumption hierarchy of $\mathcal{O}_1$ and $\mathcal{O}_2$. The only reason for not generating $\mathcal{A}$ is based on the missing link between the words `Contribution` and `Paper`. We show how to model the matching problem as a problem of joint inference taking this evidence into account in order to detect the missing link and to generate the final set of correspondences.

Approach

In the following, we distinguish explicitly between the layer of ontological entities (concepts and properties) and the layer of tokens that appear in concept and property labels. Let *i#X* refer to a concept X in $\mathcal{O}_i$, e.g., *1#AcceptedPaper*, and let *i:X* refer to a label or a part of a label that describes the ontological entities in $\mathcal{O}_i$, e.g., *1:Accepted*.

It has already been shown in (Niepert, Meilicke, and Stuckenschmidt 2010) how to model the second phase as optimization problem. Niepert et al. transform the problem to a Markov logic network (MLN) and compute via a maximum a-posteriori query the most probable state of the MLN. In Markov logic (Richardson and Domingos 2006) real-valued

weights are assigned to first-order clauses. Intuitively, the higher the weight of a clause, the less probable is a possible state violating groundings of said clause. The most probably state corresponds finally to the chosen alignment. According to (Niepert, Meilicke, and Stuckenschmidt 2010) the MLN can be constructed as follows.

(A) Add literals weighted with confidences to model the set of matching hypotheses generated in the first phase.

(B) Model the terminological hierarchies including disjointness and domain/range restrictions in the MLN.

(C) Add hard constraints, using an infinite weight, to ensure that the final solution is a coherent one-to-one alignment.

(D) Add soft constraints to make a solution more probable that does not change the hierarchy of $\mathcal{O}_1$ and $\mathcal{O}_2$.

In our modeling approach we stick to (B), (C), and (D) and replace (A) by a set of formulae introduced in the following. We first add weighted literals to model the similarity between tokens. We add, for example, (1) and (2) to express our confidence that two tokens have (or do not have) a similar meaning.

$$\mathit{sim}(\mathit{1{:}Accepted}, \mathit{2{:}Accepted}), 1.0 \quad (1)$$

$$\mathit{sim}(\mathit{1{:}Contribution}, \mathit{2{:}Paper}), -1.0 \quad (2)$$

Now we need to establish a connection between ontological entities and token entities. With respect to our example, we want to say that tokens appear in the role of a headnoun or as a modifier within a label of a certain concept.

$$\mathit{headnoun}(\mathit{1{:}Contribution}, \mathit{1\#AcceptedContribution}) \quad (3)$$

$$\mathit{modifier}(\mathit{1{:}Accepted}, \mathit{1\#AcceptedContribution}) \quad (4)$$

Two types of constraints are now required to model dependencies between the two layers.

(I) If the tokens assigned to the concepts have a similar meaning, then these concepts should be matched.

(II) If two concepts are matched on each other, then their labels should have a similar meaning.

At first glimpse it might be unclear why to introduce (II) as the counterpart of (I). By adding (II), any solution that results in $\mathcal{A}$ will also force *sim*(*1:Contribution*, *2:Paper*) to be in the solution as well as the *sim* literals related to the corresponding modifiers. The negative weight of *sim*(*1:Contribution*, *2:Paper*) needs to be overruled by (B), (C), and (D) and by the similarity of the other tokens. Thus, we have defined a problem of joint inference. A solution to this problem both results in an alignment between ontological entities and contains statements related to the meaning of the tokens that appear in the labels.

We have implemented a first prototype based on the described approach. It generates $\mathcal{A}$ if applied to our example ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$. We observed positive results in further experiments including larger, only partially overlapping ontologies. Furthermore, our experiments have shown that all constraints are required, i.e., if we drop one of the modeling components, the solution is incomplete or incorrect. The solution of the optimization problem does not only contain the generated correspondences, but also the ground atoms on the token level that resulted in the final choice. For example, *1#AcceptedContribution* is matched on *1#AcceptedPaper*, while at the same time the solution contains the information that *1:Contribution* and *2:Paper* are assumed to have a similar meaning.

Complex Matching

Suppose now that there are no named concepts to refer to accepted contributions in $\mathcal{O}_1$. Instead of that, there exists a property *1#acceptedBy* and *1#Contribution* $\sqcap$ $\exists$*1#acceptedBy*.$\top$ is equivalent to the named concept *2#AcceptedPaper*.

We can extend our approach to generate such a complex correspondence. For that purpose, we have to describe the linguistic roles of tokens more precisely. Moreover, we have to state that the verb form in the label of *1#acceptedBy* and the modifier in the label of *2#AcceptedPaper* stem from the same word. Then we have to add a set of rules to describe the relation between the formulae that we added and the complex correspondence we want to generate. This approach can be extended in several directions. We might for example state that a token is the passive voice of another token. This information can then be exploited to detect correspondences between a property and an inverse property.

Note that similar approaches have already been proposed for generating complex matches (Ritze et al. 2010). However, these techniques have been designed as a set of rules that are always triggered if certain conditions hold. Contrary to this, we exploit the various dependencies between logical reasoning and a fine-grained analysis related to the linguistic roles of labels in the context of an optimization problem.

Final Remarks

To our knowledge, the proposed approach is a novel contribution to the field of ontology matching. An exception can be seen in the ontology system S-Match (Giunchiglia, Shvaiko, and Yatskevich 2004). S-Match splits concept labels into tokens that are re-combined into complex concept descriptions. This is exploited by transforming the matching problem into a SAT problem where such concept descriptions are treated as clauses. However, the complexity of the concepts is limited to union and intersection derived from labels that contain `or`, `and` or similar clues for logical connectives. Moreover, the solution generated is not optimal with respect to a well-defined criteria taking soft and hard constraints into account.

Currently, our prototype requires a specific input format to generate the formulae on the token level. Moreover, the ideas discussed in the section on complex matching are not yet implemented. In the future we will extend our prototype and test it extensively on larger ontologies from various fields. This will help us to better understand the impact of different modeling styles and to learn how to cope with scalability problems that might emerge.

References

Albagli, S.; Ben-Eliyahu-Zohary, R.; and Shimony, S. E. 2012. Markov network based ontology matching. *Journal of Computer and System Sciences* 78(1):105–118.

Euzenat, J., and Shvaiko, P. 2007. *Ontology matching*. Springer Heidelberg.

Giunchiglia, F.; Shvaiko, P.; and Yatskevich, M. 2004. S-match: an algorithm and an implementation of semantic matching. In *The semantic web: research and applications*. Springer. 61–75.

Niepert, M.; Meilicke, C.; and Stuckenschmidt, H. 2010. A probabilistic-logical framework for ontology matching. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, 1413–1418.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine learning* 62(1):107–136.

Ritze, D.; Völker, J.; Meilicke, C.; and Šváb-Zamazal, O. 2010. Linguistic analysis for complex ontology matching. In *Proceedings of the 5th International Workshop on Ontology Matching*, 1–12. CEUR-WS.org.