Contextual Ontology Alignment of LOD with an Upper Ontology: A Case Study with Proton

Prateek Jain[1,2], Peter Z. Yeh[2], Kunal Verma[2], Reymonrod G. Vasquez[2], Mariana Damova[3], Pascal Hitzler[1], and Amit P. Sheth[1]

[1] Kno.e.sis Center, Wright State University, Dayton, OH, USA
[2] Accenture Technology Labs, San Jose, CA, USA
[3] Ontotext AD, Sofia 1784, Bulgaria

Abstract. The Linked Open Data (LOD) is a major milestone towards realizing the Semantic Web vision, and can enable applications such as robust Question Answering (QA) systems that can answer queries requiring multiple, disparate information sources. However, realizing these applications requires relationships at both the schema and instance level, but currently the LOD only provides relationships for the latter. To address this limitation, we present a solution for automatically finding schema-level links between two LOD ontologies – in the sense of ontology alignment. Our solution, called BLOOMS+, extends our previous solution (i.e. BLOOMS) in two significant ways. BLOOMS+ 1) uses a more sophisticated metric to determine which classes between two ontologies to align, and 2) considers contextual information to further support (or reject) an alignment. We present a comprehensive evaluation of our solution using schema-level mappings from LOD ontologies to Proton (an upper level ontology) – created manually by human experts for a real world application called FactForge. We show that our solution performed well on this task. We also show that our solution significantly outperformed existing ontology alignment solutions (including our previously published work on BLOOMS) on this same task.

1 Introduction

The Linked Open Data (LOD) is a major milestone towards realizing the Semantic Web vision. A key differentiator of LOD from previous approaches is that data providers are actually creating links across these data sets, which has led to a number of innovative applications spanning multiple, disparate information sources [4]. One missing facet of LOD so far is that these ever-growing ontologies are linked to each other mainly at the instance-level. There are very few schema-level linkages – i.e. links between class hierarchies such as rdfs:subClassOf relations.

A number of researchers [14,13,18][1] (including some of the co-authors of this paper) have argued that without schema-level linkages the LOD cloud will not have semantic-enough information to enable more ambitious, reasoning-based applications of Semantic Web such as Question Answering and Agent-based information brokering. Existing

[1] `http://semtech2010.semanticuniverse.com/sessionPop.cfm?confid=42&proposalid=2854`

efforts to develop these types of applications primarily utilize manually created schema-level links between LOD ontologies. For example, *FactForge* enables querying across various LOD ontologies, and utilizes manually developed schema-level mappings of LOD ontologies to an upper level ontology called *Proton* [7].

We believe that manual creation of schema-level mappings across LOD ontologies is not a viable solution given the size of the LOD and the rate at which it is growing. A more automated solution is needed in order for applications such as *FactForge* to effectively scale to (and keep up with) the size of LOD. To this effect, we previously introduced a solution, called Bootstrapping-based Linked Open Data Ontology Matching System (BLOOMS) [13] for automatically finding schema-level links between LOD ontologies. Our previous solution performed well on this task compared to existing solutions such as [12,8,15,16], but there is significant room for improvement.

In this paper, we present a solution called BLOOMS+ which extends our previous solution in two significant ways. BLOOMS+ 1) uses a more sophisticated metric to determine which classes between two ontologies to align, and 2) BLOOMS+ considers contextual information to further support (or reject) an alignment. We present a comprehensive evaluation of BLOOMS+ using schema-level mappings from various LOD ontologies to Proton (an upper level ontology), created manually by human experts. We show that BLOOM+ performed well on this task. We also compare BLOOMS+ to existing ontology alignment solutions (including our previously published work on BLOOMS) on this same task, and show that BLOOMS+ outperformed these solutions. Finally, we present an ablation study, which shows why BLOOMS+ performed well.

The rest of the paper is organized as follows. We first describe the knowledge requirements for BLOOMS+, and why we selected Wikipedia to satisfy these requirements. We then present the BLOOMS+ approach, followed by a comprehensive evaluation of BLOOMS+ and existing solutions. Finally, we present related works along with conclusions and future work.

2 Related Work

To the best of our knowledge, the only other work which exploits contextual information for the purpose of ontology matching has been described in [9]. However, their approach is different from ours as they rely on background knowledge from online ontologies, whereas we rely on a noisy loose categorization of Wikipedia for performing the contextual match. Further, their process relies on identification of contextual relationship using the relationships encoded in the ontologies.

Research in the area of 'Ontology Matching' is very closely related to our body of work. In [10,5] the authors present a survey in the area of ontology matching.[2]. The survey work also categorizes the techniques on the basis of external knowledge source utilized by ontology matching systems. While typically, systems utilize a structured source of information such as dictionaries or upper level ontologies, In our previous work in [13] we have presented an approach which exploits a generic and noisy

[2] The ontology matching portal at `http://www.ontologymatching.org/` gives a good review of the state-of-the-art research in this area

categorization system such as Wikipedia in the context of ontology matching. Previously, Wikipedia categorization has been utilized for creating and restructuring taxonomies [20,19].

Another body of related work is identification and creation of links between LOD cloud data sets. In [17] ontology schema matching was used to improve instance co-reference resolution. This helps in cleaning up the data and improving the quality of links at the instance level, but the issue of identifying appropriate relationships at the schema level has not been addressed. The voiD Framework [1] along with the SILK Framework [22] automate the process of link discovery between LOD datasets at the instance level. At the schema level, a notable effort for creating a unified reference point for LOD schemas is UMBEL [3], which is a coherent framework for ontology development which can serve as a reference framework.

3 Knowledge Requirements

BLOOMS+ requires a knowledge source to align two ontologies. The minimum requirements for this knowledge source are:

1. The knowledge source is organized as a class hierarchy where links between classes in this hierarchy capture super and subclass relationships.
2. The knowledge source covers a wide range of concepts and domains, so it can be widely applicable – especially given the wide range of domains covered by the LOD Cloud.

Many knowledge sources – such as WordNet [11], FrameNet [2], SNOMED [6], etc. – satisfy the first requirement, but they fail to satisfy the second. For example, many classes in WordNet and FrameNet are very generic, and hence may have limited utility when aligning domain specific LOD schemas such as Music and Census. SNOMED, on the other hand, captures classes specific to the medical domain, and can be useful for aligning life science LOD schemas. However, it will have limited utility in aligning LOD schemas outside of life science.

BLOOMS+ uses Wikipedia – in particular the category hierarchy in Wikipedia. Although the Wikipedia category hierarchy is not a formal class hierarchy, it still reflects a taxonomy structure. Wikipedia categories roughly correspond to classes in a class hierarchy, and the super and subcategory relationships between these categories roughly correspond to super and subclass relationships. Wikipedia also covers a wide range of categories (over 10 million categories), across many domains. This satisfied the second requirement. Moreover, our previous research [13] has shown that the Wikipedia category hierarchy is effective in aligning LOD schemas.

4 Approach

BLOOMS+ aligns two ontologies through the following steps. BLOOMS+ first uses Wikipedia to construct a set of category hierarchy trees for each class in the source and target ontologies. BLOOMS+ then determines which classes to align by extending BLOOMS in two significant ways. BLOOMS+ 1) uses a more sophisticated measure to compute the similarity between source and target classes based on their category hierarchy trees; and 2) computes the contextual similarity between these classes to further support (or reject) an alignment. Finally, BLOOMS+ aligns classes with high similarity based on the class and contextual similarity.

4.1 Construct BLOOMS+ Forest

BLOOMS+ constructs a set of category hierarchy trees – we call a *BLOOMS+ Forest* F for each class C from the source and target ontologies. For each C, BLOOMS+ tokenizes (and stems) the name of C, and removes stop words from the name.

BLOOMS+ uses the resulting terms as a search string to retrieve relevant Wikipedia pages using Wikipedia search web service.[3] BLOOMS+ treats each page as a possible sense s_i of C and constructs a category hierarchy tree we call a BLOOMS+ tree T_i – for s_i via the following steps.

1. The root of the tree is s_i.
2. The immediate children of s_i are all Wikipedia categories that s_i belongs to.
3. Each subsequent level includes all unique, direct super categories of the categories at the current level.

BLOOMS+ imposes a limit on the depth of the tree being constructed, and defaults this limit to 4. Based on empirical observation depths beyond 4 typically include very general categories (e.g. “Humanities”), which are not useful for alignment. The resulting tree is then added to F.

4.2 Compute Class Similarity

BLOOMS+ compares each class C in the source ontology with each class D in the target ontology to determine their similarity. This is done by comparing each $T_i \in F_C$ with each $T_j \in F_D$ where F_C and F_D are the BLOOMS+ forests for C and D respectively. For each source tree T_i, BLOOMS+ determines its overlap with the target tree T_j.

However, simply counting the number of common nodes the approach used by BLOOMS is insufficient for the following reasons:

– Common nodes that appear deeper in the tree are more generic (and hence less discriminative). They can appear in many BLOOMS+ trees, which can result in false alignments. These nodes should be given less importance when computing the overlap between two trees (and hence the similarity between two classes).
– A large tree can be unfairly penalized because it must have more nodes in common with another tree in order to have a high similarity score. Hence, we need to avoid bias against large trees when computing the overlap.

[3] `http://en.wikipedia.org/w/api.php`

Table 1. Common nodes between the two trees in Figure 1, and their depth. The first column gives the common nodes between the two trees rooted at *Record Label* and *Music Industry*. The second column gives the depth (the distance from root) of these nodes in the BLOOMS+ tree rooted at *Record Label* – i.e. the source tree.

Common Nodes	**Node Depth**
Music_industry	1
Music; Industries; Cultural_economics	2
Industry; Other_special_topics_(economics); Cultural_studies; Economic_systems; Entertainment; Performing_arts; Sound	3

To address these issues, BLOOMS+ uses the following equation to compute the overlap between two BLOOMS+ trees (and hence the similarity of their corresponding classes).

$$Overlap(T_i, T_j) = \frac{log\Sigma_{n \in T_i \cap T_j}(1 + e^{d(n)^{-1} - 1})}{log 2|T_i|} \quad (1)$$

where $n \in T_i \cap T_j$ are the common nodes between the source and target tree; and $d(n)$ is the depth of a common node n in T_i. The exponentiation of the inverse depth of a common node gives less importance to the node if it is generic, and the log of the tree size avoids bias against large trees. This equation ranges from 0.0 to 1.0 where 0.0 indicates no similarity and 1.0 indicates maximum similarity.

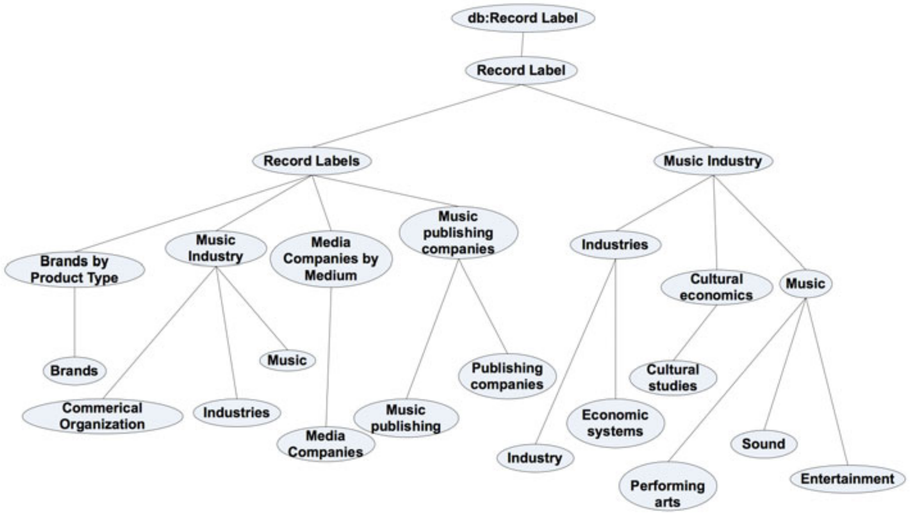
For example, let's assume BLOOMS+ needs to determine whether to align the source class *RecordLabel* from DBpedia with the target class *MusicCompany* from Proton. BLOOMS+ first constructs the BLOOMS+ forests for *RecordLabel* and *MusicCompany*, and Figure 1 shows a BLOOMS+ tree from each forest. BLOOMS+ then identifies the common nodes between these trees, and the depth of these nodes in the tree for the source class (see Table 1).

Finally, the class similarity (see above equation) between *RecordLabel* and *MusicCompany* w.r.t the two BLOOMS+ trees in Figure 1 is 0.79.

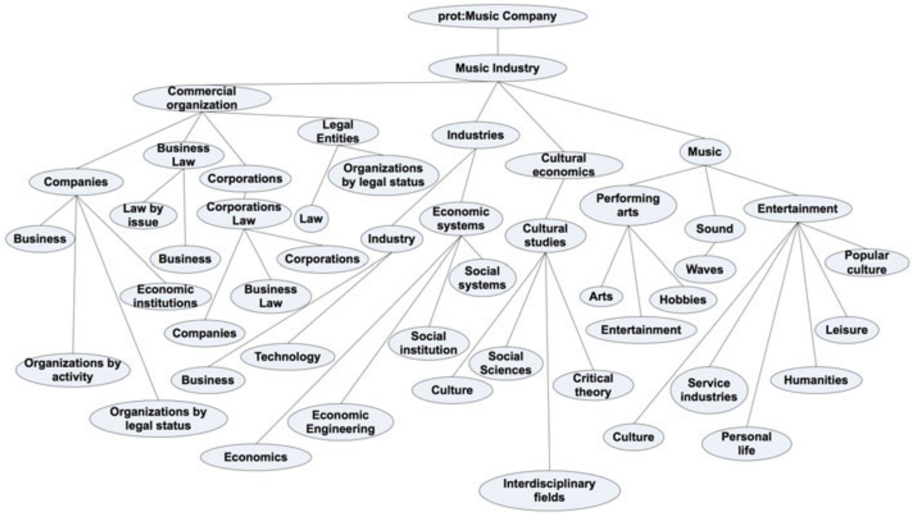
4.3 Compute Contextual Similarity

BLOOMS+ computes the contextual similarity between a source C and target D class to further determine whether these classes should be aligned. A good source of contextual information is the superclasses of C and D from their respective ontologies. If these superclasses agree with each other, then the alignment between C and D is further supported and hence should be given more preference. Otherwise, the alignment should be penalized. For example, the class *Jaguar* might be aligned to the class *Cat*, which seems like a reasonable alignment. However, if *Jaguar* has superclasses such as *Car* and *Vehicle*, and *Cat* has superclasses such as *Feline* and *Mammal*, then the alignment should be penalized because its contextual similarity is low.

BLOOMS+ implements the intuition above in the following way. For each pair wise class comparison (C, D), BLOOMS+ retrieves all superclasses of C and D up to a specified level, which BLOOMS+ defaults to 2. The two sets of superclasses – we'll refer to as $N(C)$ and $N(D)$ – are the neighborhoods of C and D respectively.



(a) BLOOMS+ tree for *RecordLabel* with sense *Record Label*



(b) BLOOMS+ tree for *MusicCompany* with sense *Music Industry*

Fig. 1. BLOOMS+ trees for Record Label 1(a) and Music Company 1(b)

For each BLOOMS+ tree pair (T_i, T_j) between C and D, BLOOMS+ determines the number of superclasses in $N(C)$ and $N(D)$ that are supported by T_i and T_j respectively. A superclass $c \in N(C)$ is supported by T_i if either of the following conditions are satisfied:

- The name of c matches a node in T_i.[4]
- The Wikipedia article (or article category) corresponding to c – based on a Wikipedia search web service call using the name of c – matches a node in T_i.

The same applies for a superclass $d \in N(D)$.

BLOOMS+ computes the overall contextual similarity between C and D with respect to T_i and T_j using the harmonic mean, which is instantiated as:

$$CSim(T_i, T_j) = \frac{2R_C R_D}{R_C + R_D} \tag{2}$$

where R_C (and R_D) are the fraction of superclasses in $N(C)$ (and $N(D)$) supported by T_i (and T_j). We chose the harmonic mean to emphasize superclass neighborhoods that are not well supported (and hence should significantly lower the overall contextual similarity).

Returning to our example, BLOOMS+ needs to compute the contextual similarity for *RecordLabel* and *MusicCompany*. Assuming a level of 2, the neighborhood of *RecordLabel* includes the DBpedia superclasses of *Company* and *Organization*. Both superclasses are supported by the BLOOMS+ tree for *RecordLabel* (see Figure 1(a)), so $R_{RecordLabel}$ is $\frac{2}{2}$. Similarly, the neighborhood of *MusicCompany* includes the Proton superclasses of *CommercialOrganization* and *Organization*. Both superclasses are supported by the BLOOMS+ tree for *MusicCompany* (see Figure 1(b)), so $R_{MusicCompany}$ is also $\frac{2}{2}$. Finally, the overall contextual similarity (see above equation) is 1.0, so BLOOMS+ should give more preference to this alignment.

4.4 Compute Overall Similarity

BLOOMS+ computes the overall similarity between classes C and D w.r.t. BLOOMS+ trees T_i and T_j by taking the weighted average of the class (see Section 4.2) and contextual (see Section 4.3) similarity.

$$O(T_i, T_j) = \frac{\alpha Overlap(T_i, T_j) + \beta CSim(T_i, T_j)}{2} \tag{3}$$

where α and β are weights for the concept and contextual similarity respectively. BLOOMS+ defaults both α and β to 1.0 to give equal importance to each component.

BLOOMS+ then selects the tree pair $(T_i, T_j) \in F_C \times F_D$ with the highest overall similarity score and if this score is greater than the alignment threshold H_A, then BLOOMS+ will establish a link between C and D. The type of link is determined as follows:

- If $O(T_i, T_j) = O(T_j, T_i)$, then BLOOMS+ sets C owl:equivalentClass D.

[4] We define this match as either a direct string match or a substring match.

- If $O(T_i, T_j) < O(T_j, T_i)$, then BLOOMS+ sets C rdfs:subClassOf D.
- Otherwise, BLOOMS+ sets D rdfs:subClassOf C.

Returning to our running example, the overall similarity score between *RecordLabel* and *MusicCompany* is 0.895 (i.e. $\frac{0.79+1.0}{2}$), and BLOOMS+ will establish a link between these classes – assuming the alignment threshold is 0.5. Finally, BLOOMS+ sets *RecordLabel* rdfs:subClassOf *MusicCompany* because $O(T_{\text{Music Industry}}, T_{\text{Record Label}}) > O(T_{\text{Record Label}}, T_{\text{Music Industry}})$.

5 Evaluation

We evaluated the following claims to show that our approach (i.e. BLOOMS+) is effective for ontology alignment over LOD schemas.

Claim 1: BLOOMS+ can outperform state-of-the-art solutions on the task of aligning LOD ontologies.
Claim 2: BLOOMS+ performs well because it accounts for two critical factors when computing the similarity between two classes – 1) the importance of common nodes between the BLOOMS+ trees of the two classes, and 2) bias against large trees.
Claim 3: The performance of BLOOMS+ can be further improved by using contextual information.

5.1 Data Set

We used a real world data set for our evaluation. This data set contains schema-level mappings from three LOD ontologies to Proton, an upper level ontology, with over 300 classes and 100 properties, designed to support applications such as semantic annotation, indexing, and search[21]. The three LOD ontologies include:

- **DBpedia:**[5] The RDF version of Wikipedia, created manually from Wikipedia article infoboxes. DBpedia consists of 259 classes ranging from general classes (e.g. Event) to domain specific ones (e.g. Protein).
- **Freebase:**[6] A large collection of structured data collected from multiple sources such as Wikipedia, Chefmoz, and MusicBrainz. Freebase consists of over 5 million topics and entities, classified into a class hierarchy.
- **Geonames:**[7] A geographic data set with over 6 million locations of interest, which are classified into 11 different classes.

These mappings were systematically created by Knowledge Engineers (KEs) [7] at OntoText for a real world application called *FactForge*[8], which enables SPARQL query over the LOD cloud. The KEs created these mappings, i.e. equivalence and subclass relationships between LOD and Proton classes, based on the definition of the classes and their usage. A total of 544 mappings were created from the three LOD ontologies to Proton (373 for DBpedia, 21 for Geonames, and 150 for Freebase). Table 2 shows examples of these mappings.

[5] http://downloads.dbpedia.org/3.5.1/dbpedia_3.5.1.owl.bz2
[6] http://www.freebase.com/schema
[7] http://geonames.org
[8] http://factforge.net/

Table 2. Sample mappings of LOD ontologies to PROTON

Ontology	Class	PROTON Class	Relationship
DBpedia	OlympicResult	Situation	subClassOf
Geonames	Class	LandRegion	subClassOf
Freebase	Event	Event	equivalentClassOf

These mappings provide a good gold standard for our evaluation because:

- The mappings were created by an independent source for a real world use case – unlike existing benchmarks which were created primarily for evaluation purposes. Hence, these mappings reflect the types of relationship that are needed in practice.
- The mappings were created by knowledge engineers through a systematic process [7] and hence are of high quality.
- The mappings cover a diverse set of LOD ontologies. For example, DBpedia and Freebase cover diverse domains such as entertainment, sports, and politics. While Geonames covers only geographic information.

5.2 Experimental Setup

To evaluate Claim 1, we measured the precision and recall of the mappings – from the three LOD ontologies to Proton generated by BLOOMS+. To obtain these measures, we applied BLOOMS+ to each LOD-Proton ontology pair to generate mappings whose overall similarity exceeded an alignment threshold of 0.85 (see Section 4.4). We defined this threshold by systematically analyzing which threshold level produced the best f-measure score. We then compared the resulting mappings for each LOD-Proton ontology pair to their respective gold standard, and said that a mapping between two classes is correct if the gold standard also established a mapping between these two classes using the same relationship i.e. equivalence or subclass. Finally, we defined precision as the number of correct mappings over the total number of mappings generated by BLOOMS+, and recall as the number of correct mappings over all mappings in the gold standard.

We also compared the performance of BLOOMS+ to existing solutions that performed well for LOD ontology alignment, as reported in [13]. These solutions include:

- **BLOOMS:** This is the solution that BLOOMS+ extends [13].
- **S-Match:** This solution utilizes three matching algorithms – basic, minimal, and structure preserving – to establish mappings between the classes of two ontologies [12].
- **AROMA:** This solution utilizes the association rule mining paradigm to discover equivalence and subclass relationships between the classes of two ontologies [8].

To ensure a fair comparison, we used the above methodology to measure precision and recall for each solution, and to define the alignment threshold. The best alignment threshold for BLOOMS is 0.6. The performance of AROMA was not affected by the alignment threshold. It had identical performance for all threshold levels between 0.1 to 1.0. S-Match does not support an alignment threshold. Instead, it returns two sets

Table 3. Results for various solutions on the task of aligning LOD schemas to PROTON. Legend: S-Match-M=Result of S-Match Minimal Set, S-Match-C=Result of S-Match Complete Set, Prec=Precision, Rec=Recall, F=F-Measure PRO=PROTON Ontology, FB=Freebase Ontology, DB=DBpedia Ontology, GEO=Geonames Ontology.

Linked Open Data and Proton Schema Ontology Alignment												
	DB-PRO			GEO-PRO			FB-PRO			Overall		
System	Rec	Prec	F	Rec	Prec	F	Rec	Prec	F	Rec	Prec	F
AROMA	0.19	0.59	0.28	0.04	$\frac{8}{1000}$	0.01	0.31	0.49	0.38	0.22	0.37	0.28
S-Match-M	0.26	0.05	0.08	0.04	$\frac{6}{1000}$	0.01	0.2	0.05	0.08	0.23	0.05	0.08
S-Match-C	0.33	$\frac{3}{1000}$	$\frac{6}{1000}$	0.04	0.009	0.01	0.3	0.4	0.34	0.31	$\frac{4}{1000}$	0.007
BLOOMS	0.48	0.19	0.27	0.04	$\frac{6}{1000}$	0.01	0.28	0.32	0.3	0.42	0.19	0.26
BLOOMS+ No Context	0.77	0.59	0.67	0.04	$\frac{5}{1000}$	0.01	0.48	0.65	0.55	0.66	0.45	0.54
BLOOMS+	0.73	0.90	0.81	0.04	$\frac{5}{1000}$	0.01	0.49	0.59	0.54	0.63	0.55	0.59

Table 4. Sample of **correct** mappings from LOD ontologies to PROTON generated by BLOOMS+

Ontology	**LOD Class**	**PROTON Class**	**Relationship**
DBpedia	RecordLabel	MusicCompany	subClassOf
Geonames	Country	Nation	equivalentClassOf
Freebase	Military_command	Position	subClassOf

of mappings – 1) a minimal set and 2) a complete set, which can be derived from the minimal one. We report both sets in our evaluation.

To evaluate Claims 2 and 3, we created a version of BLOOMS+ without contextual information we call *BLOOMS+ NO-CONTEXT*. The only difference between BLOOMS+ NO-CONTEXT and BLOOMS is the measure used to compute the similarity between two classes (and hence allows us to evaluate Claim 2). The only difference between BLOOMS+ NO-CONTEXT and BLOOMS+ is the use of contextual information (and hence allows us to evaluate Claim 3). We used the above methodology to measure precision and recall for BLOOMS+ NO-CONTEXT, and we set the alignment threshold to 0.85. The evaluation components related to this work are available for download on BLOOMS+ project page.[9]

5.3 Results and Discussion

Table 3 shows the results for all solutions evaluated. Table 4 and Table 5 show examples of correct and incorrect mappings respectively generated by BLOOMS+ from the three LOD ontologies to Proton.

BLOOMS+ performed significantly better than all other solutions in our evaluation on both precision and recall for two LOD-Proton ontology pairs ($p < 0.01$ for χ^2 test in all cases). BLOOMS+ performed well because it utilizes 1) a rich knowledge source – i.e. Wikipedia – to determine the similarity between the classes of two ontologies and 2)

[9] http://wiki.knoesis.org/index.php/CBLOOMS

Table 5. Sample of **incorrect** mappings from LOD ontologies to PROTON generated by BLOOMS+

Ontology	**LOD Class**	**PROTON Class**	**Relationship**
DBpedia	Writer	Message	subClassOf
Geonames	Feature	Art	subClassOf
Freebase	Military_command	Event	subClassOf

contextual information from both Wikipedia and the ontologies being aligned. Hence, these results support our first claim that BLOOMS+ can outperform the state-of-the-art on the task of aligning LOD ontologies.

Interestingly, no solution performed well on aligning Geonames with Proton. The only mapping found by BLOOMS+ (and the other solutions) is the class *Country* in Geonames is equivalent to the class *Nation* in Proton. The key reasons for the poor performance include: 1) Geonames has a small number of classes (and hence very limited contextual information) and 2) the names of the classes in Geonames are often vague and ambiguous (e.g. *Code* and *Feature*), which made it difficult to compute their similarity.

BLOOMS+-NO-CONTEXT performed significantly better than BLOOMS w.r.t the overall precision and recall ($p < 0.01$ for χ^2 test on both precision and recall). We attribute this improvement to the only difference between the two solutions. BLOOMS+-NO-CONTEXT uses a more sophisticated measure to compute the similarity between two classes. This measure considers the importance of common nodes between the BLOOMS+ trees of two classes, and avoids bias against large trees. This result supports our second claim that BLOOMS+ performs well because it considers the importance of common nodes and avoids bias against large trees when computing the similarity between two classes.

BLOOMS+ performed significantly better than BLOOMS+-NO-CONTEXT w.r.t to the overall precision ($p < 0.01$ for χ^2 test). Although BLOOMS+ had lower overall recall, this difference was not statistically significant according to the χ^2 test. Moreover, BLOOMS+ had a higher overall f-measure score. We attribute this result to the only difference between these two solutions. BLOOMS+ uses contextual information, and BLOOMS+-NO-CONTEXT does not. Hence, this result supports our third claim that the use of contextual information can further improve performance – in particular precision and f-measure.

6 Conclusion and Future Work

We presented a solution – called BLOOMS+ – for performing ontology alignment. We evaluated BLOOMS+ using schema-level mappings from three LOD ontologies to Proton – created manually by human experts for a real world application called *Fact-Forge* – and showed that BLOOMS+ performed well on this task. We also applied state-of-the-art ontology alignment solutions (including our previously published work on BLOOMS) to this same task, and showed that BLOOMS+ significantly outperformed these solutions on both precision and recall. We also showed that our solution performed well because:

- BLOOMS+ uses a rich knowledge source – i.e. Wikipedia – to determine the similarity between the classes of two ontologies;
- BLOOMS+ accounts for two critical factors when computing he similarity between two classes – 1) the importance of common nodes between the BLOOMS+ trees of the two classes, and 2) bias against large trees.
- BLOOMS+ uses contextual information from both Wikipedia and the ontologies being aligned to further support (or reject) an alignment.

To the best of our knowledge, BLOOMS+ is the only system which utilizes the contextual information present in the ontology and Wikipedia category hierarchy for the purpose of ontology matching.

We plan to utilize BLOOMS+ for the purpose of LOD querying – as outlined in [14] – which requires significant tool support for LOD schema matching in order to scale and keep up with the growth of the LOD cloud. With BLOOMS+, we have made a important step towards solving this bottleneck, and we hope to tackle the problem of querying of LOD cloud next. Finally, we are investigating additional techniques to further improve BLOOMS+ such as incorporating additional contextual information and utilizing other knowledge sources in addition to Wikipedia.

Acknowledgement. The first author was visiting Accenture Technology Labs as a research intern when this work was performed. This work is also funded from NSF Award:IIS-0842129, titled "III-SGER: Spatio-Temporal-Thematic Queries of Semantic Web Data: a Study of Expressivity and Efficiency". Pascal Hitzler acknowledges support by NSF Award:IIS-1017225, titled "TROn - Tractable Reasoning with Ontologies". Mariana Damova acknowledges support by project "RENDER", FP7-ICT-2009-5, Contract no.: 257790 [10].

References

1. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets – On the Design and Usage of voiD, the 'Vocabulary of Interlinked Datasets'. In: WWW 2009 Workshop on Linked Data on the Web (LDOW 2009), Madrid, Spain (2009)
2. Baker, C., Fillmore, C., Lowe, J.: The berkeley framenet project. In: Proceedings of the 17th International Conference on Computational Linguistics, vol. 1, pp. 86–90. Association for Computational Linguistics, Morgan Kaufmann Publishers (1998)
3. Bergman, M.K., Giasson, F.: UMBEL ontology, volume 1, technical documentation. Technical Report 1, Structured Dynamics (2008), http://umbel.org/doc/UMBELOntology_vA1.pdf
4. Bizer, C., Heath, T., Berners-Lee, T.: Linked data – the story so far. International Journal On Semantic Web and Information Systems 5(3), 1–22 (2009)
5. Choi, N., Song, I.Y., Han, H.: A survey on ontology mapping. SIGMOD Rec. 35(3), 34–41 (2006)
6. Cornet, R., de Keizer, N.: Forty years of SNOMED: a literature review. BMC medical informatics and decision making 8(suppl. 1) (2008), http://www.biomedcentral.com/1472-6947/8/S1/S2

[10] www.render-project.eu

7. Damova, M., Kiryakov, A., Simov, K., Petrov, S.: Mapping the Central LOD Ontologies to PROTON Upper-Level Ontology. In: Shvaiko, P., Euzenat, J., Giunchiglia, F., Stuckenschmidt, H., Mao, M., Cruz, I. (eds.) Proceedings of the Fifth International Workshop on Ontology Matching. CEUR Workshop Proceedings (November 2010)
8. David, J., Guillet, F., Briand, H.: Matching directories and OWL ontologies with AROMA. In: CIKM 2006: Proceedings of the 15th ACM International Conference on Information and Knowledge Management, pp. 830–831. ACM, New York (2006)
9. Euzenat, J., d'Aquin, M., Sabou, M., Zimmer, A.: Matching ontologies for context. Technical Report NEON/2006/D3.3.1/0.8, INRIA (2007)
10. Euzenat, J., Shvaiko, P.: Ontology matching. Springer, Heidelberg (2007)
11. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database (Language, Speech, and Communication). illustrated edition edn. The MIT Press, Cambridge (1998), `http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/026206197X`
12. Giunchiglia, F., Autayeu, A., Pane, J.: S-Match: an open source framework for matching lightweight ontologies (2010)
13. Jain, P., Hitzler, P., Sheth, A.P., Verma, K., Yeh, P.Z.: Ontology Alignment for Linked Open Data. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 402–417. Springer, Heidelberg (2010), `http://data.semanticweb.org/conference/iswc/2010/paper/218`
14. Jain, P., Hitzler, P., Yeh, P.Z., Verma, K., Sheth, A.P.: Linked Data is Merely More Data. In: Brickley, D., Chaudhri, V.K., Halpin, H., McGuinness, D. (eds.) Linked Data Meets Artificial Intelligence, pp. 82–86. AAAI Press, Menlo Park (2010)
15. Li, J., Tang, J., Li, Y., Luo, Q.: RiMOM: A dynamic multistrategy ontology alignment framework. IEEE Transactions on Knowledge and Data Engineering 21, 1218–1232 (2009)
16. Mascardi, V., Locoro, A., Rosso, P.: Automatic Ontology Matching via Upper Ontologies: A Systematic Evaluation. IEEE Transactions on Knowledge and Data Engineering 22(5), 609–623 (2010)
17. Nikolov, A., Uren, V.S., Motta, E., Roeck, A.N.D.: Overcoming schema heterogeneity between linked semantic repositories to improve coreference resolution. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds.) ASWC 2009. LNCS, vol. 5926, pp. 332–346. Springer, Heidelberg (2009)
18. Parundekar, R., Knoblock, C., Ambite, J.L.: Linking and building ontologies of linked data. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 598–614. Springer, Heidelberg (2010), `http://data.semanticweb.org/conference/iswc/2010/paper/334`
19. Ponzetto, S.P., Navigli, R.: Large-scale taxonomy mapping for restructuring and integrating wikipedia. In: Boutilier, C. (ed.) Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, pp. 2083–2088 (2009)
20. Ponzetto, S.P., Strube, M.: Deriving a large scale taxonomy from Wikipedia. In: AAAI 2007: Proceedings of the 22nd National Conference on Artificial Intelligence, pp. 1440–1445. AAAI Press, Menlo Park (2007)
21. Terziev, I., Kiryakov, A., Manov, D.: Base upper-level ontology (bulo) Guidance. Technical report, Ontotext Lab, Sirma Group, Deliverable of EU-IST Project IST-2003-506826 (2004), `http://proton.semanticweb.org/D1_8_1.pdf`
22. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 650–665. Springer, Heidelberg (2009)