COMET: A Contextualized Molecule-Based Matching Technique

Mayesha Tasnim[1,3]*, Diego Collarana[2,3], Damien Graux[3], Mikhail Galkin[3], and Maria-Esther Vidal[4,5,6]

[1] RWTH Aachen, Germany
[2] University of Bonn, Germany
[3] Fraunhofer Institute for Intelligent Analysis and Information Systems, Germany
[4] TIB Leibniz Information Centre for Science and Technology, Germany
[5] Universidad Simón Bolívar, Venezuela
[6] L3S Research Centre, Leibniz University of Hannover, Germany
`mayesha.tasnim@rwth-aachen.de`, `collaran@cs.uni-bonn.de`, `maria.vidal@tib.eu`
{`damien.graux,mikhail.galkin`}`@iais.fraunhofer.de`

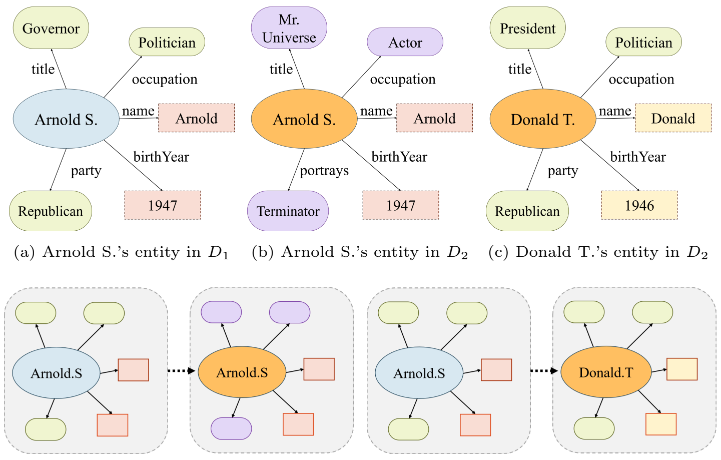
Abstract. Context-specific description of entities –expressed in RDF– poses challenges during data-driven tasks, e.g., data integration, and context-aware entity matching represents a building-block for these tasks. However, existing approaches only consider inter-schema mapping of data sources, and are not able to manage several contexts during entity matching. We devise COMET, an entity matching technique that relies on both the knowledge stated in RDF vocabularies and context-based similarity metrics to match *contextually equivalent* entities. COMET executes a novel 1-1 perfect matching algorithm for matching contextually equivalent entities based on the combined scores of semantic similarity and context similarity. COMET employs the Formal Concept Analysis algorithm in order to compute the context similarity of RDF entities. We empirically evaluate the performance of COMET on a testbed from DBpedia. The experimental results suggest that COMET is able to accurately match equivalent RDF graphs in a context-dependent manner.

Keywords: Data Integration · RDF Knowledge Graphs · RDF Entities.

1 Introduction

The semi-structuredness nature of the RDF data model allows for naturally representing entities of the same type with different properties, as well as for the encoding of multiple contexts of an entity within the same graph. This feature of RDF is of paramount importance for addressing the data complexity challenge of variety—a dominant dimension of data in the Big Data era [6]. Nevertheless, enabling diverse representations of the same entity poses new challenges during the matching of RDF graphs, particularly, because specific contexts need to be considered for an effective identification of similar entities [2]. To perform a

* This research was supported by the European project QualiChain (number 822404).



(a) Arnold S.'s entity in D_1 (b) Arnold S.'s entity in D_2 (c) Donald T.'s entity in D_2

(d) Integration using similarity (e) Integration using similarity and context

Fig. 1: **Motivation Example.** Top row shows three entities across two datasets. The bottom row shows two matching scenarios, the left one not considering context during entity matching, and the right one taking context into consideration.

contextually RDF entity matching, the specific context under which a matching is being performed must be taken into account.

We motivate our work using an example of a context-based entity matching scenario using RDF entities representing persons. *Arnold Schwarzenegger* is a person with an extensive career in both politics and acting. Consequently, there is data available regarding both his career in politics and his achievements in the movie industry. Consider an integration system aiming to match data about American politics. Dataset D_1 contains information about Arnold Schwarzenegger and his political career. In another dataset D_2, there exists information about Arnold's acting career, e.g., the movies he has acted in and the roles he has portrayed. The same dataset D_2 also contains information about other celebrities, like Donald Trump, President of the United States. These entities are presented in Figure 1a, 1b and 1c, respectively. In a typical integration scenario where context is not considered, entities are matched to the ones that are most similar to them. In such a case, Arnold Schwarzenegger's entity from D_1 will be matched with the entity in D_2 containing information about his acting career, as shown in Figure 1d. However, in the context of politics, Arnold's political career is more similar to Donald Trump's than his own career in acting. They are politicians of almost the same age who are both supporting the Republican party. In a political context, their careers are far more similar than when Arnold's post as the Governor of California is compared with his portrayal of the Terminator in Terminator 2. Therefore, when context of American politics is considered, the entity of Arnold S. from D_1 should be matched with Donald T. entity from D_2.

To support scenarios where the inclusion of *context* in entity matching makes results more relevant for real-world scenarios, we present the **CO**ntextualized **M**olecul**E**-based matching **T**echnique (COMET). COMET is a context-aware entity matching method that employs a two-fold approach for both: (1) identifying contextually similar entities, and (2) matching them into a 1-1 set of entities. We present the results of an empirical evaluation that illustrate the benefits of the techniques implemented in COMET.

The remainder of the paper is structured as follows: Section 2 summarizes the related work and compares COMET with the state of the art. Then, we define the COMET approach in terms of its main characteristics and architecture in Section 3. A comprehensive evaluation of the COMET approach is reported in Section 4. Finally, Section 5 summarizes our main conclusions and future work.

2 Related Work

The problem of entity matching between RDF graphs has been extensively treated by the Semantic Web community. As a result, a vast amount of approaches and frameworks have been developed. In the task of identifying whether the given entities refer to the same real-world entity, growing attention in the relational databases field is given to crowdsourcing mechanisms [11]. Reporting impressive results, such approaches, however, might struggle in sophisticated domains with multiple contexts due to a lack of human experts who could reliably provide necessary example or training data.

Entity matching is particularly important in data integration scenarios and integration frameworks tackle this problem. Knoblock et al. [7] propose KARMA, a framework for integrating a variety of data sources including databases, spreadsheets, XML, JSON, and Web APIs. KARMA implements a hybrid approach that relies on supervised machine algorithms for identifying mapping rules from structured sources to ontologies; these mapping rules should solve entity matching problems. Schultz et al. [10] describe the Linked Data Integration Framework (LDIF). LDIF is oriented to integrate RDF datasets from the Web and provides a set of independent tools to support interlinking tasks. LDIF provides an expressive mapping language for translating data from various vocabularies to a unified ontology. LDIF tackles the problem of identity resolution by defining linking rules using the SILK tool [5]. Based on the defined rules, SILK identifies `owl:sameAs` links among entities of two datasets. ODCleanStore [8] relies as well on SILK to perform instance matching and provides a custom data fusion modules to merge the data of the discovered matches. Other efforts to produce a unified view from RDF graphs are often combined with federated SPARQL query engines, e.g., ANAPSID [1], MULDER [4]. Albeit effective in query planning, such engines process raw tuples coming from endpoints and therefore employ only basic entity matching of those raw tuples according to join operators.

The above-mentioned entity matching and integration approaches aim at mapping different data sources with possibly varying schema, i.e., they perform inter-schema mapping. Context-based entity matching could only be supported on a superficial level via filtering query results without applying much of inherent semantics. Contrary, COMET considers diverse criteria during entity matching, e.g., entity similarity and contextual knowledge. In consequence, COMET provides the building blocks for context-based semantic integration mechanisms.

3 The COMET Approach

Grounded on the entity matching component from the data integration technique proposed by Collarana et al. [3], we propose COMET, an entity matching approach to identify and synthesize contextually equivalent RDF entities. Thus, a solution to the *problem of contextually matching entities* is provided.

Problem Definition

RDF Molecule [3] – If $\Phi(G)$ is a given RDF Graph, we define RDF Molecule M as a subgraph of $\Phi(G)$ such that,

$$M = \{t_1, \ldots, t_n\} \quad \forall i, j \in \{1, \ldots, n\}\big(subject(t_i) = subject(t_j)\big)$$

Where $t_1, t_2, \ldots, t_n$ denote the triples in M. In other words, an RDF Molecule M consists of triples which have the same subject. That is, it can be represented by a tuple *M = (R, T)*, where R denotes the URI of the molecule's subject, and T denotes a set of property and value pairs *p = (prop, val)* such that the triple *(R, prop, val)* belongs to *M*. For example, the RDF molecule for *Arnold Schwarzenegger* is (dbr:Arnold-Schwarzenegger, { (`dbo:occupation`, *Politician*), (`dbp:title`, *Governor*)}). An RDF Graph $\Phi(G)$ described in terms of RDF molecules is defined as follows:

$$\Phi(G) = \{M = (R, T) | t = (R, prop, val) \in G \wedge (prop, val) \in T\}$$

Context – We define a context C as any Boolean expression which represents the criteria of a system. Two entities, such as an RDF molecule M_1 and M_2, can be either similar or not similar with respect to a given context. That is, C is a Boolean function that takes as input two molecules M_1 and M_2 and returns `true` if they are similar according to system context, and `false` otherwise. Below is an example of context C, modeled after the example presented in Figure 1, where two molecules are similar if they have the same occupation. If $P = (p, v)$ is the predicate representing the occupation property of a molecule, then context

$$C(M_1, M_2) = \begin{cases} \texttt{true}, & \text{if } P \in M_1 \wedge P \in M_2. \\ \texttt{false}, & \text{otherwise.} \end{cases}$$

Depending on the requirements of the integration scenario, this context can be any Boolean expression.

Semantic Similarity Function – Let M_1 and M_2 be any two RDF molecules. Then *semantic similarity function Sim_f* is a function that measures the *semantic similarity* between these two molecules and returns a value between [0,1]. A 0 expresses that the two molecules are completely dissimilar and 1 expresses that the molecules are identical. Such a similarity function is defined in GADES [9].

Contextually Equivalent RDF Molecule – Let $\Phi(G)$ and $\Phi(D)$ be two sets of RDF molecules. Let M_G and M_D be two RDF molecules from $\Phi(G)$ and $\Phi(D)$ respectively. Then, M_G and M_D are defined as contextually equivalent iff (i) They are in the same context. That is, $C(M_1, M_2) = \texttt{true}$ and, (ii) They have the highest similarity value, i.e., $Sim_f(M_G, M_D) = max(\forall_{m \in \Phi(D)} Sim_f(M_G, m))$

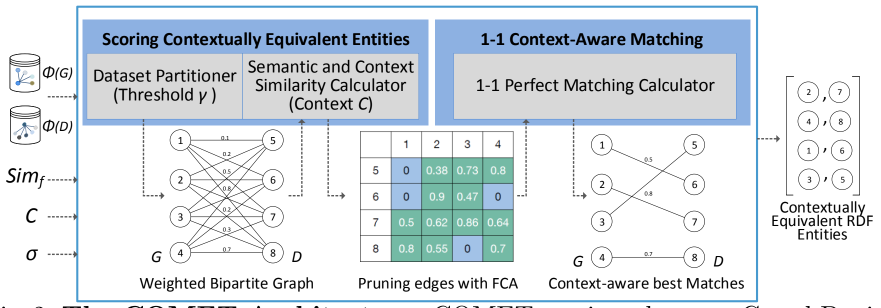
Let F_c be an idealized set of *contextually integrated* RDF molecules from $\Phi(G)$ and $\Phi(D)$. Let θ_C be a homomorphism such that $\theta_C : \Phi(G) \cup \Phi(D) \rightarrow F_c$. Then there is an RDF Molecule M_F from F_c such that $\theta(M_D) = \theta(M_G) = M_F$. From the motivation example, this means that the molecule of *Arnold Schwarzenegger*, the politician is *contextually equivalent* to the molecule of *Donald Trump* as they are similar *and* they satisfy the context condition of having the same occupation.

In this work, we tackle the problem of explicitly modeling the context and then, matching RDF molecules from RDF graphs that are both highly-similar and equivalent in terms of this context. This problem is defined as follows: given RDF graphs $\Phi(G)$ and $\Phi(D)$, let M_G and M_D be two RDF molecules such that $M_G \in \Phi(G)$ and $M_D \in \Phi(D)$. The system is supplied a context parameter C, which is a boolean function evaluating if two molecules are in the same context. It is also supplied a similarity function Sim_f, which evaluates the semantic similarity between M_G and M_D.

The problem of creating a contextualized graph Φ_C consists of building a homomorphism $\theta_C : \Phi(G) \cup \Phi(D) \rightarrow F_c$, such that for every pair of RDF molecules belonging to Φ_C there are none that are *contextually equivalent* according to system context C. If M_G and M_D are contextually equivalent molecules belonging to F_c, then $\theta_C(M_G) = \theta_C(M_D)$, otherwise $\theta_C(M_G) \neq \theta_C(M_D)$.

Architecture We propose COMET, an approach to match contextually equivalent RDF graphs according to a given context, thus providing a solution to the problem of *contextually matching* RDF graphs. Figure 2 depicts the main components of the COMET architecture. COMET follows a two-fold approach to solve the problem of entity matching in RDF graphs in a context-aware manner: First, COMET computes the similarity measures across RDF entities to create a bipartite graph; Finally, COMET executes a context-aware 1-1 perfect matching algorithm for matching RDF entities based on the combined scores of similarity and context, by employing Formal Concept Analysis to validate context.

Building a bipartite graph. The COMET pipeline receives two RDF graphs $\Phi(G), \Phi(D)$ as input, along with context parameter C, and a similarity function Sim_f. COMET first constructs a bipartite graph between the sets $\phi(G)$ and $\phi(D)$. The *Dataset Partitioner* employs a similarity function Sim_f and ontology O to compute the similarity between RDF molecules in $\phi(G)$ and $\phi(D)$ assigning the similarity score as edge weight in the bipartite graph. COMET supports a variety of similarity functions including simple string similarity. However, as showed in [3], semantic similarity measures are advocated (in the implementation of this work we particularly use GADES [9]).


Scoring Contextually Equivalent Entities
Dataset Partitioner (Threshold γ)
Semantic and Context Similarity Calculator (Context C)
1-1 Context-Aware Matching
1-1 Perfect Matching Calculator
Φ(G)
Φ(D)
Sim_f
C
σ
G
D
Weighted Bipartite Graph
Pruning edges with FCA
Context-aware best Matches
Contextually Equivalent RDF Entities

	1	2	3	4
5	0	0.38	0.73	0.8
6	0	0.9	0.47	0
7	0.5	0.62	0.86	0.64
8	0.8	0.55	0	0.7

Fig. 2: **The COMET Architecture.** COMET receives datasets G and D, similarity function Sim_f and context C. Output is a set of matched RDF entities.

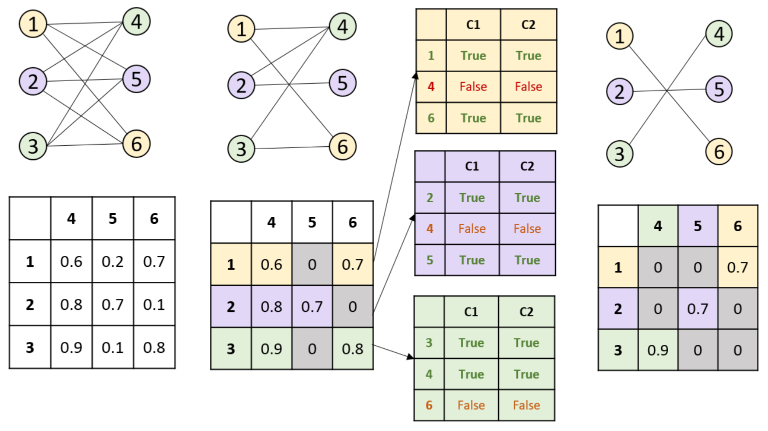
After similarity computation of the RDF molecules, the result of the similarity function is tested against a threshold γ to determine entity similarity (the similarity thresholds minimum acceptable score). Thus, edges are discarded from the bipartite graph whose weights are lower than γ.

1-1 Context-Aware Perfect Matching Calculator. The main contribution of the COMET pipeline is a novel 1-1 context-aware perfect matching calculator, which validates and prunes pairs of RDF molecules that do not comply with the input context C, making COMET a context-aware approach. For identifying contextually equivalent RDF entities, the *Context Validator* component employs the Formal Concept Analysis (FCA) algorithm. FCA[1] is the study of binary data tables that describe the relationship between objects and their attributes. Applying this context validation step over the RDF molecules ensures only contextually relevant tuples are kept. In COMET, context is modeled as any boolean function. Two molecules are matched if they satisfy this condition, otherwise they are not matched. The algorithm by V. Vychodil [12] is applied in COMET; it performs formal concept analysis to compute formal concepts within a set of objects and their attributes. This algorithm is extended in our approach for validating complex *boolean conditions*. A typical formal concept analysis takes as input a binary data table. The rows of this table correspond to object, while the columns denote if an object contains a certain attribute.

In our approach, instead of using objects and attributes, we replace the attributes with a *boolean condition* C. This is the same as the context condition C used in our approach. For example, the context C from the motivating example can be broken down into $C = C_1 \wedge C_2$ where $C_1 =$ "contains property `dbo:occupation`", and $C_2 =$ "has the same value for property `dbo:occupation`". The execution of the FCA algorithm remains unchanged by this adaptation since the format of the input to FCA is still a binary matrix.

When applied to RDF molecules, formal concept analysis returns a set of formal concepts $< M, C >$ where M is a set of all the molecules that contain all conditions contained in C. Thus, the molecules that do not meet the context condition are pruned. In Figure 3 an example of context validation is demon-

[1] https://en.wikipedia.org/wiki/Formal_concept_analysis



	4	5	6
1	0.6	0.2	0.7
2	0.8	0.7	0.1
3	0.9	0.1	0.8

	4	5	6
1	0.6	0	0.7
2	0.8	0.7	0
3	0.9	0	0.8

	C1	C2
1	True	True
4	False	False
6	True	True

	C1	C2
2	True	True
4	False	False
5	True	True

	C1	C2
3	True	True
4	True	True
6	False	False

	4	5	6
1	0	0	0.7
2	0	0.7	0
3	0.9	0	0

(a) Bipartite graph (b) Context validation using Formal Concept Analysis (c) Perfect 1-1 matches

Fig. 3: **Context Validation.** A 1-1 matching algorithm over a bipartite graph that validates context of entities using FCA; 1-1 perfect matches are generated.

strated. Edges in a bipartite graph are filtered according to a threshold value γ as detailed in the previous section. Next, the remaining edges are validated by constructing an FCA matrix according to context condition C. The FCA algorithm returns the edge satisfying the context conditions. The edges that do not satisfy the context condition are discarded.

COMET solves the problem of *context-aware entity matching* by computing a 1-1 weighted perfect matching between the sets of RDF molecules. The input of the 1-1 weighted perfect matching component is the weighted bipartite graph created on the previous step. Since each weight of an edge between two RDF molecules corresponds to a combined score of semantic similarity and context equivalence value, we call this a 1-1 context-aware matching calculator. Finally, the Hungarian algorithm is utilized to compute the matching.

4 Empirical Evaluation

4.1 Effectiveness Evaluation

We conducted an empirical evaluation to study the effectiveness and performance of COMET in solving the entity matching problem among RDF graphs. We address the following research questions: **(RQ1)** "Is COMET able to perform entity matching with regard to context more accurately than MINTE [3] entity matching component?", and **(RQ2)** "Does the content of the dataset with respect to the context condition affect the accuracy of COMET?"

Practically, COMET is implemented in Python and hosted in GitHub[2] along with the datasets and logs used in this evaluation. For the COMET pipeline we

[2] https://github.com/RDF-Molecules/COMET

	Experiment: Effectiveness					
Configuration	**A**		**B**		**C**	
Datasets	*A1*	*A2*	*B1*	*B2*	*C1*	*C2*
Molecules	1000	1000	1000	1000	1000	1000
Triples	70,660	70,660	70,776	70,776	71,124	71,124
Context	$C(M_{D1}, M_{D2}) = \texttt{true}$, if `dbo:occupation` match					

Table 1: **Benchmark Description**. Datasets used in the evaluation including: number of RDF molecules (M), number of triples (T), evaluated contexts (C).

use the semantic similarity measure *GADES* [9] *GADES* examines both string similarity and hierarchy similarity by making use of graph neighbourhoods.

As a baseline, we compare the effectiveness of COMET against the MINTE pipeline proposed by Collarana et al.[3]. Towards **(RQ1)** and **(RQ2)** we design an experiment to measure the *precision*, *recall* and *f-measure* of COMET in comparison to MINTE, while supplying both the pipelines with datasets of different compositions of molecules with respect to context to observe the effect of contextual content on the effectiveness of COMET.

Although each experiment has different datasets and gold standards, we use the same metrics for all the experiments: *Precision*, *Recall*, and *F-meaure*. *Precision* measures what proportion of the performed integrations are actually correct. That is, *precision* is the fraction of RDF molecules that has been identified as contextually equivalent by COMET (C), which intersects with the Gold Standard (GS). On the other hand, *recall* measures the overall proportion of integrated RDF molecules that were identified correctly. That is, *recall* is measured by the fraction of correctly identified similar molecules with respect to the Gold Standard,i.e., $Precision = \frac{|C \cap GS|}{|C|}$ and $Recall = \frac{|C \cap GS|}{|GS|}$. *F-measure* is the harmonic mean of *Precision* and *Recall.*

The datasets contain 1,000 people entities from DBpedia. In order to test the effect of contextual data content on the accuracy of COMET, three pairs of datasets *(A1, A2)*, *(B1, B2)* and *(C1, C2)* are generated using configurations *A*, *B* and *C*: In configuration A, every molecule *a1* in dataset A1 has **2** *highly similar* molecules *a2* and *a3* in dataset A2, such that *a2* satisfies context condition, but *a3* does not. That is, $C(a1, a2) = \texttt{true}$ and $C(a1, a3) = \texttt{false}$. In configuration B, every molecule *b1* in dataset B1 has **3** *highly similar* molecules *b2*, *b3* and *b4* in dataset B2, such that *b2* and *b3* satisfy the context but *b4* does not. For configuration C, every molecule *c1* in dataset C1 has **4** *highly similar* molecules in dataset C2, two of which satisfy context, and two that do not.

Every pair of datasets are synthesized as follows: First, molecules from the original set of 1,000 DBpedia person entities are duplicated according to the configuration condition to create n number of highly similar molecules in the second dataset. Then predicates inside the similar molecules are randomly edited and deleted to create some variation of similarity. The predicates are then edited to ensure that the correct number of similar molecules in the second dataset satisfy the context according to the original dataset.

	COMET			**MINTE**		
Configuration	***Precision***	***Recall***	***F-Measure***	***Precision***	***Recall***	***F-Measure***
A	1.0	1.0	1.0	0.54	0.54	0.54
B	0.708	0.708	0.708	0.449	0.449	0.449
C	0.558	0.558	0.558	0.408	0.408	0.408

Table 2: Effectiveness evaluation of COMET.

Similar to the motivation example shown in Figure 1, the context C used in this experiment checks if two molecules have the same value for the predicate `dbo:occupation`. The Gold Standard contains matches between molecules that (i) Satisfy the context condition and, (ii) Are highest in similarity among all other molecules. For every pair of datasets belonging to the three configurations (i.e., A, B and C), there is a corresponding Gold Standard G_A, G_B and G_C. The datasets, gold standard and the experiment code are all available on GitHub.

Table 1 describes the dataset used during our evaluations. This experiment was conducted on MINTE and COMET once for each pair of datasets *(A1, A2)*, *(B1, B2)* and *(C1, C2)*, with the context condition requiring that every pair of matched molecules must have the same value for `dbo:occupation` property. The threshold value γ for this experiment is applied at 97th percentile in every case. Then, we compare against the Gold Standard G_A, G_B and G_C for configurations A, B and C, respectively; Precision and Recall are calculated each time.

4.2 Discussion of Observed Results

Based on the values of Precision, Recall, and F-measure reported in Experiment 1 (Table 2), we can positively answer **(RQ1)**, i.e., COMET is able to perform contextual entity matching more effectively than MINTE, and answer **(RQ2)** by illustrating how the contextual content of the dataset affects accuracy. In every case, COMET performs better than MINTE, since MINTE does not take context into consideration during its 1-1 perfect matching whereas COMET does. Moreover, a decrease in Precision and Recall of COMET with the increase of highly similar molecules occurs since COMET has a lesser chance of identifying the perfect match with a higher number of options to choose from. On the other hand, in case of *configuration A*, Precision and Recall is perfect since the dataset supplies only one highly similar molecule that also meets the context. Thus, the highest values of precision and recall demonstrate that in an ideal condition with only one perfect option, COMET will always find the correct match.

5 Conclusions and Future Work

We presented COMET, an approach to match contextually equivalent RDF entities from different sources. COMET executes a 1-1 perfect matching where contextually equivalent RDF molecules are identified according to a combined score of semantic and context similarity. COMET utilizes the Formal Concept Analysis algorithm to decide whenever two RDF molecules are contextually equivalent. The behavior of COMET was empirically studied on two real-world RDF graphs under different context configurations. Observed results suggest that COMET is able to effectively identify and match contextually equivalent entities. COMET makes use of a very simple definition of context conditions, modeling context as a boolean function of entities. In future, context can be modeled in a more generalized way, e.g., a probabilistic function. We plan to evaluate a multi-threading version of the FCA algorithm that may enable the implementation of this work on large datasets. Finally, FCA context parameters will be empowered by materializing implicit facts in RDF knowledge graphs via a reasoner.

References

1. M. Acosta, M. Vidal, T. Lampo, J. Castillo, and E. Ruckhaus. ANAPSID: an adaptive query processing engine for SPARQL endpoints. In *International Conference on The Semantic Web*, pages 18–34, 2011.
2. W. Beek, S. Schlobach, and F. van Harmelen. A contextualised semantics for owl: sameAs. In *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC*, pages 405–419, 2016.
3. D. Collarana, M. Galkin, I. T. Ribón, M. Vidal, C. Lange, and S. Auer. MINTE: semantically integrating RDF graphs. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics, WIMS*, pages 22:1–22:11, 2017.
4. K. M. Endris, M. Galkin, I. Lytra, M. N. Mami, M. Vidal, and S. Auer. MULDER: querying the linked data web by bridging RDF molecule templates. In *Database and Expert Systems Applications 2017*, pages 3–18, 2017.
5. R. Isele and C. Bizer. Active learning of expressive linkage rules using genetic programming. *Journal of Web Semantics*, 23:2–15, 2013.
6. H. V. Jagadish, J. Gehrke, A. Labrinidis, Y. Papakonstantinou, J. M. Patel, R. Ramakrishnan, and C. Shahabi. Big data and its technical challenges. *Commun. ACM*, 57(7):86–94, 2014.
7. C. A. Knoblock, P. A. Szekely, J. L. Ambite, A. Goel, S. Gupta, K. Lerman, M. Muslea, M. Taheriyan, and P. Mallick. Semi-automatically mapping structured sources into the semantic web. In *Proceedings of the 9th Extended Semantic Web Conference ESWC, May 27-31, Heraklion, Crete, Greece*, pages 375–390, 2012.
8. J. Michelfeit and T. Knap. Linked data fusion in ODCleanStore. In *ISWC Posters and Demonstrations Track*, 2012.
9. I. T. Ribón, M. Vidal, B. Kämpgen, and Y. Sure-Vetter. GADES: A graph-based semantic similarity measure. In *SEMANTICS - 12th International Conference on Semantic Systems, Leipzig, Germany*, pages 101–104, 2016.
10. A. Schultz, A. Matteini, R. Isele, P. N. Mendes, C. Bizer, and C. Becker. LDIF - a framework for large-scale linked data integration. In *International World Wide Web Conference*, 2012.
11. V. Verroios, H. Garcia-Molina, and Y. Papakonstantinou. Waldo: An adaptive human interface for crowd entity resolution. In *International Conference on Management of Data*, pages 1133–1148, 2017.
12. V. Vychodil. *A new algorithm for computing formal concepts.* na, 2008.