SWARM: An Approach for Mining Semantic Association Rules from Semantic Web Data

Molood Barati[1], Quan Bai[1] and Qing Liu[2]

[1] Auckland University of Technology, New Zealand,
[2] The Commonwealth Scientific and Industrial Research Organization (CSIRO), Australia
mbarati@aut.ac.nz; qbai@aut.ac.nz; q.liu@csiro.au

Abstract. The ever growing amount of Semantic Web data has made it increasingly difficult to analyse the information required by the users. Association rule mining is one of the most useful techniques for discovering frequent patterns among RDF triples. In this context, some statistical methods strongly rely on the user intervention that is time-consuming and error-prone due to a large amount of data. In these studies, the rule quality factors (e.g. Support and Confidence measures) consider only knowledge in the instance-level data. However, Semantic Web data contains knowledge in both instance-level and schema-level. In this paper, we introduce an approach called SWARM (Semantic Web Association Rule Mining) to automatically mine Semantic Association Rules from RDF data. We discuss how to utilize knowledge encode in the schema-level to enrich the semantics of rules. We also show that our approach is able to reveal common behavioral patterns associated with knowledge in the instance-level and schema-level. The proposed rule quality factors (Support and Confidence) consider knowledge not only in the instance-level but also schema-level. Experiments performed on the DBpedia Dataset (3.8) demonstrate the usefulness of the proposed approach.

Keywords: Semantic Web data, Association Rule Mining, Ontology, Knowledge Discovery

1 Introduction

The Semantic Web is an effort to make knowledge on the Web both human-understandable and machine-readable [1]. Semantic Web data is normally structured in triple formats called Resource Description Framework (RDF). By emerging RDF/S, OWL and SPARQL standardization, the number of large KBs such as YAGO, DBpedia and Freebase [3] is growing so fast. Although these KBs suffer many issues such as incompleteness and inconsistencies, they already contain millions of facts which raise new opportunities for data mining community. In recent years, researchers have been working on developing methods and tools

[3] http://freebase.com

for mining hidden patterns from Semantic Web data that promise more potential for Semantic Web applications [2]. In this regard, association rule mining is one of the most common Data Mining (DM) techniques for extracting frequent patterns.

There are several methods in mining associations from large RDF-style KBs. Most existing methods focus on Inductive Logic Programming (ILP) to mine association rules. ILP usually requires counterexamples. AMIE [3][4] is a multi-threaded approach where the KB is kept and indexed in the memory. High memory usage is one of the drawbacks of this approach. This method is restricted to a complete ontology structure. To compute support and confidence values, the method only considers knowledge in the instance-level and removes *rdf:type* relations from datasets. A recent statistical approach for mining association rules in RDF data is [5]. It automatically generates three forms of $s_i \Rightarrow s_j$, $p_i \Rightarrow p_j$, and $o_i \Rightarrow o_j$ rules. This approach does not require counterexamples. However, the method discovers the sequence of subjects, predicates, or objects which are correlated independently. Additionally, the rule quality factors (Support and Confidence) only assess instance-level data.

In comparison with [5], we propose a statistical approach to automatically mine rules from RDF data. Our approach is based on the methodology that adapts association rule mining to RDF data. The rules reveal common behavioural patterns associated with knowledge in the instance-level and schema-level. Consider the RDF triples shown in Table 1. From the triples, our approach generates the following rule:

$$\{Person\}\text{: } (instrument,\ Guitar) \Rightarrow (occupation,\ Songwriter)$$

Table 1. RDF triples from DBpedia

Subject	**Predicate**	**Object**
John Lennon	instrument	Guitar
John Lennon	spouse	Yoko Ono
John Lennon	occupation	Songwriter
George Harrison	instrument	Guitar
George Harrison	occupation	Songwriter
Jimmy Carter	office	President of the USA
Jimmy Carter	party	Democratic
Bill Clinton	office	President of the USA
Bill Clinton	party	Democratic
George W. Bush	office	President of the USA
George W. Bush	party	Republic
John Lennon	*rdf:type*	dbo:Person
George Harrison	*rdf:type*	dbo:MusicalArtist
George Harrison	*rdf:type*	dbo:Person
Jimmy Carter	*rdf:type*	dbo:Person
Bill Clinton	*rdf:type*	dbo:Person
George W. Bush	*rdf:type*	dbo:Person

The above rule shows that most of the time persons who play a musical instrument such as Guitar, they are probably Songwriters. Mining such regularities help us to gain a better understanding of Semantic Web data. In order to elaborate the semantics of the rules, our approach considers *rdf:type* and *rdf:subClassOf* relations in the ontology. As seen in Table 1, both *John Lennon* and *George Harrison* are guitarists and songwriters. Consider Figure 2 as a small fragment of DBpedia ontology. *George Harrison*[4] is an instance of Musical Artist while *John Lennon* [5] belongs to the Person class. Regarding the concept of hierarchy in the ontology, if Musical Artist class is a subclass of Artist class and the Artist class is a subclass of Person class, then *George Harrison* belongs to the Person class as well. But *John Lennon* is not an instance of Musical Artist class. In the context of Semantic Web data, it is not reasonable to interpret the discovered rules without considering such relationships between instance-level and schema-level. As far as we know, the proposed approach in [3][4][5] do not cover such issues on mining Semantic Web data.

Under this motivation, in this paper, we proposed a novel approach called SWARM (Semantic Web Association Rule Mining) to automatically mine and generate semantically-enriched rules from RDF data. The main contribution of this paper is threefold:

1. The SWARM is an approach that automatically mines association rules from RDF data without the need of domain experts.
2. The SWARM measures the quality of rules (Support and confidence) by utilizing knowledge not only in the instance-level but also schema-level.
3. The SWARM reveals common behavioural patterns associated with knowledge in the instance-level and schema-level.

The remainder of this paper is organized as follows. Section 2 gives a general overview of the related works in this context. In Section 3, the SWARM approach is introduced in detail. Both framework architecture and algorithms are presented in this section. Section 4 shows the experimental results. Finally, the conclusion and future work are presented in Section 5.

2 Related works

In the following, we discuss state-of-the-art approaches in the context of Semantic Web data mining.

Logical Rule Mining. Most related research on mining Semantic Web data relies on ILP techniques. ALEPH [6] is an ILP system implemented in the Prolog. WARMeR [7] used a declarative language to mine association rules correspondent to conjunctive queries from a relational database. Galárraga, et al. [3] proposed a multi-threaded approach called AMIE for mining association rules from RDF-style KBs. Galárraga, et al. [4] extends AMIE to AMIE$^+$ using pruning

[4] http://dbpedia.org/page/George_Harrison
[5] http://dbpedia.org/page/John_Lennon

and query rewriting techniques. Similar to AMIE, [8] proposed another approach for extracting horn rules. The proposed methods in [3][4][8] measures the quality of the discovered rules using instance-level data. However, instances in a rule might belong to different classes of the ontologies. To express a broader meaning of the rules, the SWARM considers instance-level data along with *rdf:type* and *rdf:subClassOf* relations in the schema-level.

Association Rule Mining. Association rule mining was originally proposed for shopping basket problems [9]. It reflects high correlation between multiple objects and extracts interesting relationships between data [10]. Nebot and Berlanga [11] proposed a rule mining approach over RDF-based medical data. Transactions have been generated using mining patterns developed by SPARQL queries. This approach heavily relies on the domain experts. Namely, the user should have background knowledge of vocabularies used in the ontology. In comparison to [11], SWARM approach does not require the domain experts.

Abedjan and Naumann [5][12] developed an approach to identify schema and value dependencies between RDF data using six different Configurations. Any part of Subject-Predicate-Object (SPO) statement can be considered as a *context*, which is used for grouping one of the two remaining parts of the statement as the *target* of mining. This approach mines three forms of $s_i \Rightarrow s_j$, $p_i \Rightarrow p_j$, and $o_i \Rightarrow o_j$ rules. The discovered rules shows the correlation among subjects, predicates or objects independently. In comparison with this approach, SWARM generates common behavioural patterns associated with knowledge in instance-level and schema-level.

3 The SWARM approach

In this section, we describe a detailed view of SWARM approach along with the definitions. The overall framework is shown in Figure 1. The main goal of SWARM approach is to tie instance-level to schema-level to attach more semantics to the rules. The SWARM generates Semantic Association Rules from RDF data.

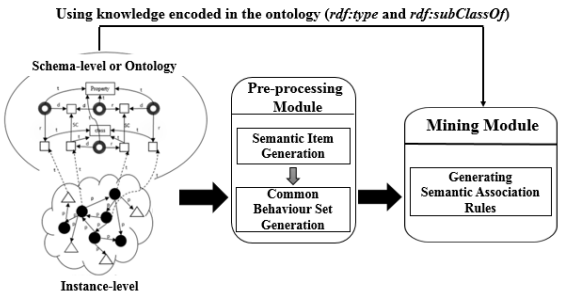

Fig. 1. The SWARM framework

The RDF triples are automatically processed via Pre-processing Module consisting of two sub-modules: Semantic Item Generation and Common Behaviour Set Generation. The Mining Module receives Common Behaviour Sets to generate Semantic Association Rules. The SWARM approach evaluates the importance of rules by using *rdf:type* and *rdf:subClassOf* relations in the ontology. The proposed rule quality factors (Support and confidence) consider knowledge not only in the instance-level bust also schema-level.

3.1 Pre-processing Module

The concept of association rule mining was first introduced in [9]. Let $I = \{i_1, i_2, ..., i_n\}$ be a set of items and $D = \{t_1, t_2, ..., t_m\}$ be a set of transactions. Each transaction contains a subset of items in I. An association rule represents a frequent pattern of the occurrence of some items in transactions. In addition, association rules reveal behavioural patterns of some particular entities. For example, in the traditional shopping basket problem, the rule {butter,bread} $\Rightarrow$ {milk} shows a behavioural pattern of customers. Namely, if a customer buys butter and bread together, she is likely to buy milk as well.

Traditional association rule mining algorithms are suited for homogeneous repositories, where items and transactions play significant roles in the mining process [13]. However, most Semantic Web data are not transactional data, and there exists no items or transactions. To generate association rules in the context of Semantic Web data, we need to model such notions.

As mentioned earlier, Semantic Web data are normally structured in triple format. The assertion of a triple (i.e., subject, predicate, object) indicates a meaningful relationship between entities (subject and object) provided by the predicate. A triple can also be considered as the description of one particular behaviour of entities.

For example, suppose we have a triple $t1$ in Table 1: *(John Lennon, instrument, Guitar)*. If we consider the subject in $t1$ (i.e., John Lennon) as the entity, the other two elements in the triple (i.e., instrument Guitar) can be considered the description of a particular behaviour of John Lennon. Based on this concept, in the SWARM approach, we target at exploring behavioural patterns among entities. Under this motivation, we define Semantic Item and Common Behaviour Set to summarize common behaviours of entities.

3.2 Semantic Item Generation

Consider the example presented in the previous paragraph with triple $t4$ in Table 1: *(George Harrison, instrument, Guitar)*. These two subjects in $t1$ and $t4$ {*John Lennon*, *George Harrison*} have a common activity, i.e., (*instrument Guitar*). Namely, playing guitar is a common behaviour taken by a group of entities, i.e., {*John Lennon*, *George Harrison*}. In the SWARM approach, such combinations are represented as Semantic Items.

Definition 1 *(Semantic Item).* A Semantic Item si is a 2-tuple, i.e., $si = (es, pa)$. es is an Element Set of si. It contains a list of subjects, i.e., $\{s_1, s_2, ..., s_n\}$.

pa is a Pair of si. Corresponding with the content in es, pa contains a combination of predicate-object, i.e., (p, o).

According to Definition 1, triples in a triple store can be converted to a set of Semantic Items *i.e.* $SI = \{si_1, si_2, ..., si_n\}$. Each Semantic Item contains a Pair, which can be considered as a common behaviour taken by entities in the Element Set.

Example 1: Consider the triples shown in Table 1. Table 2 shows some of the Semantic Items generated by Definition 1. For example, the Element Set of si_2 including $\{JohnLennon, GeorgeHarrison\}$ represents all subjects that contain $(occupation, Songwriter)$ as a Pair.

Table 2. Semantic Items

Semantic Items	
si_1	{John Lennon, George Harrison}(instrument, Guitar)
si_2	{John Lennon, George Harrison}(occupation, Songwriter)
si_3	{Jimmy Carter, Bill Clinton, George W. Bush}(office, President of the USA)
si_4	{Jimmy Carter, Bill Clinton}(party, Democratic)

3.3 Common Behaviour Set Generation

As introduced in the previous subsection, a Semantic Item indicates a common behaviour (i.e., the Pair) taken by a group of entities in the Element Set. We define a Common Behaviour Set that represents all common activities taken by similar groups of entities in the Element Sets.

Definition 2 *(Common Behaviour Set).* A Common Behaviour Set *cbs* contains a set of Semantic Items with similar Element Sets, i.e., $\{(es, pa)_1, (es, pa)_2, ..., (es, pa)_n\}$. Items can be aggregated into the same *cbs*, if the similarity degree of their *Element Sets* are greater than or equal to Similarity Threshold *SimTh*. The Similarity Degree of *Element Sets* can be calculated by using Equation 1.

$$sim(es_a, es_b, ..., es_m) = \frac{|es_a \cap es_b \cap ... \cap es_m|}{|es_a \cup es_b \cup ... \cup es_m|} \tag{1}$$

According to Definition 2, the *cbs* is a set of Semantic Items aggregated through the similarity of entities in their *Element Sets*. Namely, a *cbs* shows a collection of common occurrence of some activities taken by entities in the Element Sets.

Example 2: Table 3 shows Common Behaviour Sets generated by Semantic Items in Table 2. Semantic Items si_1, si_2, and Semantic Items si_3, si_4 generate Common Behaviour Sets cbs_1 and cbs_2, when the $SimTh$ among Element Sets is greater than or equal to 50%.

Table 3. Common Behaviour Sets

Common Behaviour Sets	
$cbs1$	{John Lennon, George Harrison}(instrument, Guitar)
	{John Lennon, George Harrison}(occupation, Songwriter)
$cbs2$	{Jimmy Carter, Bill Clinton, George W. Bush}(office, President of the USA)
	{Jimmy Carter, Bill Clinton}(party, Democratic)

3.4 Mining Module

To generate Semantic Association Rules, we need to have a notion of frequency. As we discussed in the previous subsection, each particular Common Behaviour Set *cbs* is a unique set and reveals the common occurrence of some activities taken by entities (subjects) in its Element Sets. In fact, it is a particular form of transaction in the context of Semantic Web data. Under this motivation, we first generate Semantic Association Rules from Common Behaviour Sets and then we evaluate the quality of rules (Support and confidence measures) by extracting knowledge encoded in the ontology.

Definition 3 *(Semantic Association Rule).* A Semantic Association Rule r is composed by two different sets of Pairs pa_{ant} and pa_{con}, where pa_{ant} is called Pairs of Antecedent and pa_{con} is called Pairs of Consequent. pa_{ant} is a set including the number of Pairs in cbs_j, i.e., $\{pa_1, ..., pa_n\}$. pa_{con} is a set including the remaining number of Pairs in the cbs_j, i.e., $\{pa_{n+1}, ..., pa_m\}$. Rule r contains a common Rule's Element Set res where res is a set including union of the Element Sets in cbs_j, i.e., $\{es_1 \cup ... \cup es_m\}$. Each Element Set es_i is a set of instances, i.e., $es_i = \{ins_1, ins_2, ..., ins_k\}$. We indicate a rule r with the antecedent and consequent by an implication

$$res\text{: } pa_{ant} \Longrightarrow pa_{con}$$

where res is a common Rule's Element Set containing $\bigcup_{si_i \in cbs_j} si_i.es$ and pa_{ant}, $pa_{con} \in cbs_j$ and $pa_{ant} \cap pa_{con} = \emptyset$.

Table 4 shows two examples of rules generated from Common Behaviour Sets in Table 3. For example, rule r_1 contains a common Rule's Element Set res generated by union of Element Sets in cbs_1, i.e., $\{John\ Lennon, George\ Harrison\}$. The antecedent and the consequent of r_1 holds the Pair *(instrument, Guitar)* and *(occupation, Songwriter)*, respectively.

Our goal is to measure the quality of rules by using knowledge in the instance-level and schema-level. *rdf:type* is basically an RDF property that ties an instance to a class in the ontology. In the traditional association rule mining, all instances often have one type (class) of actors, i.e., shopping customers. Namely, particular activities have always done by customers. However, here instances in the Rule's Element Set may belong to different types/classes in the ontology. Consider all instances in the Rule's Element Set of rule r_1, i.e., *John Lennon* and *George*

Harrison. Figure 2 shows a small fragment of DBpedia ontology. In this ontology, *George Harrison* belongs to the Musical Artist, while *John Lennon* is an instance of Person class. As we discussed in the introduction section, in the context of Semantic Web data, it does not make sense to measure the quality of rules by only considering knowledge in the instance-level. This observation leads us to assess *rdf:type* and *rdfs:subClassOf* relations in the schema-level. In this paper, we focus on interpreting rules through having a single ontological structure, e.g. DBpedia ontology. Furthermore, we assume that each instance belongs to a single class in the ontology.

Table 4. Semantic Association Rules

Semantic Association Rules	
r_1	$\{John\ Lennon, George\ Harrison\}$: $(instrument, Guitar) \Rightarrow (occupation, Songwriter)$
r_2	$\{Jimmy\ Carter, Bill\ Clinton, George\ W.\ Bush\}$: $(office, President\ of\ the\ USA) \Rightarrow (party, Democratic)$

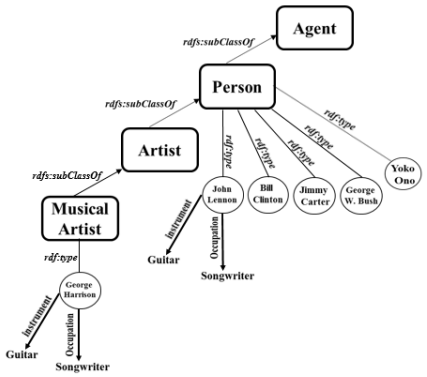

Fig. 2. A fragment of the DBpedia ontology

Figure 3 shows three different hierarchical structures of an ontology. As previously mentioned, in Figure 3 (a), if Class c_1 is subclass of Class c_3 through middle Class c_2 ($c_1 \subseteq c_3$), then the Instance I_a belongs to c_3 as well. However, in Figure 3 (b), Class c_1 and Class c_5 are not in the same hierarchy ($c_1 \nsubseteq c_5$). Even if we consider Class c_3 as a lowest common class for c_1 and c_5, we reduce their semantics. Because classes on the upper levels illustrate more general descriptions to compare with Lower level classes which provide more special descriptions. Therefore, in case that classes are not in the same hierarchy, we just consider the Lowest Level Class (LLC) for each instance in a Rule's Element

Set. For example, in Figure 3 (b), I_a and I_b belong to c_1 and c_5, respectively. In Figure 3 (c), in the LLC, I_a and I_b belong to c_2, while I_c is an instance of c_9. Consider again instances in the Rule's Element Set r_1. Based on our assumption that each instance belongs to a single class, the LLC for both *George Harrison* and *George Harrison* is the Person class.

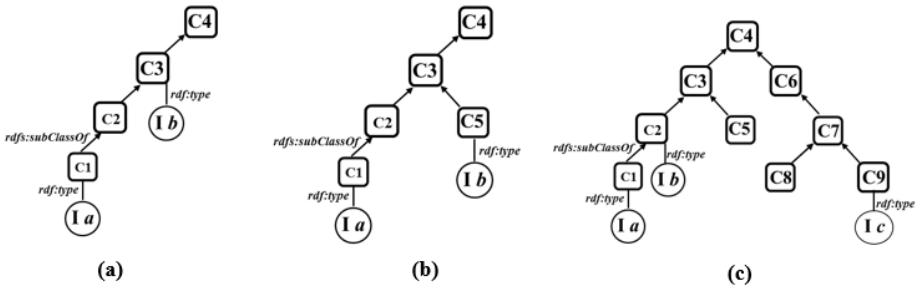

Fig. 3. Examples of different hierarchical structures of an ontology

Support. Consider the Semantic Association Rule r in the form of res : $pa_{ant} \Longrightarrow pa_{con}$. The support $Sup\ (r)$ is defined as:

$$Sup(r) = \frac{|\bigcup_{ins_j \in c_i \wedge ins_j \in res_k \wedge ins_j.pa_{ant_k}} c_i|}{|\bigcup_{ins_j \in c_i \wedge ins_j \in res_k} c_i|} \tag{2}$$

The numerator of support fraction is the total number of instances of Class c_i that contains pa_{ant} as the Pairs. The denominator is the total number of instances of c_i.

Example 3: Regarding three different schemas shown in Figure 3, rules generated from Schema a, b, and c are $r_a = \{I_a, I_b\}$: $pa_{ant} \Rightarrow pa_{con}$, $r_b = \{I_a, I_b\}$: $pa_{ant} \Rightarrow pa_{con}$, and $r_c = \{I_a, I_b, I_c\}$: $pa_{ant} \Rightarrow pa_{con}$, respectively. The supports of rules can be calculated by the following fractions:

$$Sup(r_a) = \frac{|c_3 \cap pa_{ant}|}{|c_3|}$$
$$Sup(r_b) = \frac{|(c_1 \cup c_3) \cap pa_{ant}|}{|c_1 \cup c_3|}$$
$$Sup(r_c) = \frac{|(c_2 \cup c_9) \cap pa_{ant}|}{|c_2 \cup c_9|}$$

Example 4: The support of rule r_1 in Table 4 can be calculated by the following fraction. The numerator of support fraction shows the total number of instances belong to *Person* class that contain *instrumentGuitar* as a Pair. Based on the existing ontology shown in Figure 2, there is only two instances that contain *instrumentGuitar* as a Pair. The denominator of the fraction also is total number of instances belong to the *Person* class which is six in this example (*Sup.*=0.33).

$$Sup(r_1) = \frac{|Person \cap instrument\ Guitar|}{|Person|}$$

Confidence. Consider the Semantic Association Rule r in the form of $res : pa_{ant} \Longrightarrow pa_{con}$. The confidence $Conf\ (r)$ is defined as:

$$Conf(r) = \frac{|\bigcup_{ins_j \in c_i \wedge ins_j \in res_k \wedge ins_j.pa_{ant_k} \wedge ins_j.pa_{con_k}} c_i|}{|\bigcup_{ins_j \in c_i \wedge ins_j \in res_k \wedge ins_j.pa_{ant_k}} c_i|} \quad (3)$$

The numerator of confidence fraction is the total number of instances of Class c_i that contains pa_{ant} and pa_{con} as the Pairs. The denominator of the fraction is the total number of instances of c_i that contains pa_{ant} as the Pairs.

Example 5: The numerator of confidence fraction of rule r_1 shows the total number of instances belong to the *Person* class that contain *instrumentGuitar* and *occupationSongwriter* as the Pairs. The denominator of the fraction also is the total number of instances belong to the *Person* class along with *instrument-Guitar* as a Pair (*Conf.*=1.0). The rule shows that most of the time persons who play Guitar, they probably work as Songwriters. The rule shows that at least 50% of instances in the Rule's Element Set satisfy the rule.

$$Conf(r_1) = \frac{|Person \cap instrument\ Guitar \cap occupation\ Songwriter|}{|Person \cap instrument\ Guitar|}$$

Example 6: Rule r_2 in Table 4 shows that most of the time people who are President of the USA, they are probably members of Democratic party (*Sup*=0.5, *Conf.*=0.66).

4 Experiments

4.1 Overview

Dataset. As a proof of concept, we ran the SWARM on DBpedia (3.8) [6]. The DBpedia datasets usually provide the A-Box and T-Box in two separate files: *Ontology Infobox Properties* and *Ontology Infobox Types*. The *Ontology Infobox Properties* provides instance-level data and the *Ontology Infobox Types* contains triples in the form of (*subject*, *rdf* : *type*, $ClassName$). The $ClassName$ declares the name of classes for each subject in the DBpedia ontology. For example, *Anton Drexler* belongs to the Politician, Person, and Agent classes. In this paper, we filtered out the *Ontology Infobox Types* based on the person class and its subclasses. In the DBpedia ontology, the Person class contains 26 subclasses. By using triples filtered from *Ontology Infobox Types*, we extracted about 50,000 triples of *Ontology Infobox Properties*. We also removed some triples with literals (numbers and strings) from the subset dataset. Literal values such *Birthdate* information are less interesting for rule mining.

Goal. The main goal of this research is to automatically tie instance-level to schema-level to attach more semantics to the rules. To the best of our knowledge,

[6] http://dbpedia.org/services-resources/datasets/data-set-38/downloads-38

this issue has not yet been considered by the existing methods. In comparison to [5][12] that mentioned their approach is more granular in considering predicate correlations and object correlations independently, our approach is able to automatically mine common behavioural patterns associated with knowledge in the instance-level and schema-level.

Table 5. Semantic Association Rules from Person class (*SimTh=60%*)

Rule	Rule's Element Set	Semantic Association Rule	Sup.	Conf.
r_1	{*Alfred Russel Wallace*, *Charles Darwin*, *Andrew Wiles*, *Robert Bunsen*}	{*Scientist*}: (*knownFor*, *Natural selection*) ⇒ (*award*, *Copley Medal*), (*award*, *Royal Medal*)	0.01	1.0
r_2	{*Lodewijk Asscher*, *Eberhard van der Laan*}	{*Politician*}: (*residence*, *Amsterdam*), (*party*, *Labour Party (Netherlands)*) ⇒ (*residence*, *Netherlands*)	0.04	1.0
r_3	{*Augustine of Hippo*, *Saint Titus*, *Bernard of Clairvaux*, *Athanasius of Alexandria*}	{*Saint*}: (*veneratedIn*, *Lutheranism*) ⇒ (*veneratedIn*, *Anglican Communion*)	0.04	0.87
r_4	{*Amyntas I of Macedon*, *Alcetas I of Macedon*, *Alexander I of Macedon*, *Alcetas II of Macedon*, *Perdiccas II of Macedon*}	{*Person*}: (*title*, *King of Macedon*) ⇒ (*religion*, *Religion in ancient Greece*)	0.02	1.0

Table 6. Semantic Association Rules from Person class (*SimTh=80%*)

Rule	Rule's Element Set	Semantic Association Rule	Sup.	Conf.
r_1	{*Afonso VI of Portugal*, *Peter II of Portugal*}	{**BritishRoyalty**}: (*birthPlace*, *Ribeira Palace*), (*parent*, *Luisa of Guzman*), (*parent*, *John IV of Portugal*) ⇒ (*restingPlace*, *Royal Pantheon of the House of Braganza*)	0.04	1.0
r_2	{*Alfonso V of Aragon*, *John II of Aragon*}	{**BritishRoyalty**}: (*parent*, *Ferdinand I of Aragon*), (*birthPlace*, *Medina del Campo*) ⇒ (*parent*, *Eleanor of Alburquerque*)	0.04	1.0

Evaluations. Table 5 represents some Semantic Association Rules of Person class generated by *SimTh=60%*. For example, rule r_1 shows that the Scientists who are known for Natural selection theory, they were probably awarded with the Copley and Royal Medals. Note that in this table, at least 60% of instances of Element Sets satisfy rules. Rule r_2 illustrates that Politicians who are residents of Amsterdam and works in the Labour Party of Netherlands, they are probably residents of Netherlands. Rule r_3 shows that the Saints who venerate in Lutheranism which is a major branch of Protestant Christianity, they are more

likely to be venerated in Anglican Communion as well. Rule r_4 represents that people who were Kings of Macedon, they probably had Ancient Greek religion.

Table 6 also shows some rules generated by *SimTh=80%*. Based on the DBpedia ontology, we identify some inconsistent patterns. Rule r_1 represents that some members of British Royal family who were born in Ribeira Palace and whose parents are Luisa of Guzman and John IV of Portugal, they more likely to be buried in the Royal Pantheon of the House of Braganza. Although the instances of Rule's Element Set r_1 satisfy the rule, none of them belongs to the British Royal family. In fact, they are members of Portugal Royal family. Rule r_2 also suffers from the same issue as r_1 does. In the DBpedia ontology, all royalties belong to the BritishRoyalty and PolishKing classes. The ontology does not define any other classes for these instances. Such inconsistencies between ontology definitions and underlying data lead to an ambiguous interpretation. Sometimes the ontology has been created independently before actual data usage. In the case of DBpedia project, revising existing class definitions might be helpful to obtain a better understanding of data.

We observe that the generated rules tend to have low support rates. The intuition behind this is that the denominator of support fraction usually contains the total number of instances of particular classes. In the real world KBs, the number of instances is too large and it leads to low support rates. Figure 4 shows the number of strong Semantic Association Rules with different minimum Similarity Thresholds. The *SimTh* has a direct effect on the number of generated rules. As seen in Figure 4, the number of high confidence rules from 0.6 to 0.8 has been decreased by increasing the *SimTh*. The reason that SWARM discovers a large number of rules with confidence 1.0 is because of filtering mechanism for generating Common Behaviour Sets. Note that this approach is implemented in the Eclipse Java with 3.20GHz Intel Core i5 processors and 16 GB memory. The time complexity of SWARM algorithm belongs to the O(n^2) class (including the time for generating the Semantic Items, Common Behaviour Sets, and mining Semantic Association Rules by utilizing the instance-level and schema-level knowledge).

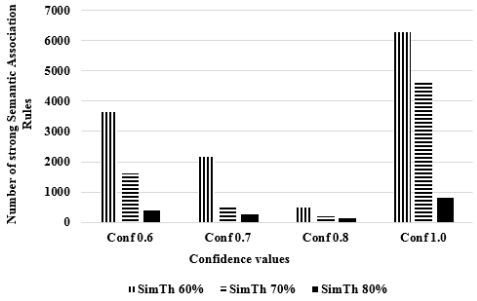

Fig. 4. Number of strong Semantic Association Rules with different minimum Similarity Thresholds

5 Conclusion

In this paper, we propose an approach to automatically mine Semantic Association Rules from Semantic Web data by utilizing knowledge in the instance-level and schema-level. We believe that this type of learning will become important in the future of Semantic Web data mining especially for re-engineering ontology definitions. In comparison with the existing methods [5][12] that evaluate the quality of rules by only using instance-level data, the SWARM approach takes advantage of *rdf:type* and *rdfs:subClassOf* relations to interpret the association rules. Future work will aim to test the approach on different classes of DBpedia Ontology. In order to produce more precise rules, we target at developing this approach wherein each instance of a Rule's Element Set belongs to the multiple classes in the ontologies.

References

1. C. Bizer, T. Heath, and T. Berners-Lee, "Linked data-the story so far," *International journal on semantic web and information systems*, vol. 5, no. 3, pp. 1–22, 2009.
2. S. Kabir, S. Ripon, M. Rahman, and T. Rahman, "Knowledge-based data mining using semantic web," *IERI Procedia*, vol. 7, pp. 113–119, 2014.
3. L. A. Galárraga, C. Teflioudi, K. Hose, and F. Suchanek, "Amie: association rule mining under incomplete evidence in ontological knowledge bases," in *Proceedings of the 22nd international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2013, pp. 413–422.
4. L. Galárraga, C. Teflioudi, K. Hose, and F. M. Suchanek, "Fast rule mining in ontological knowledge bases with amie+," *The VLDB Journal*, vol. 24, no. 6, pp. 707–730, 2015.
5. Z. Abedjan and F. Naumann, "Improving rdf data through association rule mining," *Datenbank-Spektrum*, vol. 13, no. 2, pp. 111–120, 2013.
6. S. Muggleton, "Inverse entailment and progol," *New generation computing*, vol. 13, no. 3-4, pp. 245–286, 1995.
7. B. Goethals and J. Van den Bussche, "Relational association rules: Getting w armer," *Pattern Detection and Discovery*, pp. 145–159, 2002.
8. B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," *arXiv preprint arXiv:1412.6575*, 2014.
9. R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *ACM SIGMOD Record*, vol. 22, no. 2. ACM, 1993, pp. 207–216.
10. J. Han, H. Cheng, D. Xin, and X. Yan, "Frequent pattern mining: current status and future directions," *Data Mining and Knowledge Discovery*, vol. 15, no. 1, pp. 55–86, 2007.
11. V. Nebot and R. Berlanga, "Finding association rules in semantic web data," *Knowledge-Based Systems*, vol. 25, no. 1, pp. 51–62, 2012.
12. Z. Abedjan and F. Naumann, "Amending rdf entities with new facts," in *The Semantic Web: ESWC 2014 Satellite Events*. Springer, 2014, pp. 131–143.
13. J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *ACM Sigmod Record*, vol. 29, no. 2. ACM, 2000, pp. 1–12.