Learning Meta-Descriptions of the FOAF Network

Gunnar AAstrand Grimnes, Pete Edwards, and Alun Preece

Computing Science Dept.
King's College
University of Aberdeen
AB24 3UE Scotland
{ggrimnes,pedwards,apreece}@csd.abdn.ac.uk

Abstract. We argue that in a distributed context, such as the Semantic Web, ontology engineers and data creators often cannot control (or even imagine) the possible uses their data or ontologies might have. Therefore ontologies are unlikely to identify every useful or interesting classification possible in a problem domain, for example these might be of a personalised nature and only appropriate for a certain user in a certain context, or they might be of a different granularity than the initial scope of the ontology. We argue that machine learning techniques will be essential within the Semantic Web context to allow these unspecified classifications to be identified. In this paper we explore the application of machine learning methods to FOAF, highlighting the challenges posed by the characteristics of such data. Specifically, we use clustering to identify classes of people and inductive logic programming (ILP) to learn descriptions of these groups. We argue that these descriptions constitute re-usable, first class knowledge that is neither explicitly stated nor deducible from the input data. These new descriptions can be represented as simple OWL class restrictions or more sophisticated descriptions using SWRL. These are then suitable either for incorporation into future versions of ontologies or for on-the-fly use for personalisation tasks.

1 Introduction

If a Semantic Web as set in out [1] becomes reality one might imagine that machine learning will no longer be required, as by manipulation of logical statements about semantic resources and their descriptions anything can be inferred and understood. However, we believe that no matter how wide-spread and extensive the Semantic Web becomes, every little fact is still unlikely to be explicitly stated, and in addition agents will need to know personalised facts, that although not generally true may be true in a certain context for a certain user. The fundamental organising structure of the Semantic Web is that of a set of inter-related classes; individual resources are members of one or more classes. A large part of the challenge in defining RDF Schemas [2] and OWL [3] ontologies is to identify a sufficiently rich set of classes to capture the various kinds of resources that exist. Inevitably, in any such effort, some potentially interesting and useful classes will be left unidentified. In some cases, they may be alternative ways of viewing an existing conceptualisation (for example, dividing people into subsets according to the kind of work they do, instead of by their gender). In some cases, these

may reflect conceptualisations that are true only locally, in a particular context for a specific user (for example, the class of all restaurants liked by a particular individual).

We believe that machine learning techniques have enormous potential to discover such unspecified classes. In some cases, the designers of schemas and ontologies may be prompted to add the new classes into future versions of their conceptualisations. In other cases, the discovered classes may be created on-the-fly in order to derive some inference or perform some task.

For our discussion of learning from the Semantic Web, we assume it is using the standard RDF representation based on *(subject, object, predicate)* triples [4]. We identify two types of Semantic Web data:

- "Semantic forests" - these consist of many small, disconnected, shallow resource trees. The structure of such a forest is isomorphic to that of an XML document. Real world examples of such semantic forests include meta-data using Dublin Core[1], or the use of RDF Site Summary (RSS)[2]. See Figure 1(left).
- "Semantic webs" - consist of a large graph of resources linking to each other, with no clear distinction where one resource description ends and another begins. Such data cannot easily be expressed in pure XML, and a richer representational language, like RDF, is really needed. The only significant real-world example we are aware of to date is Friend of a Friend (FOAF)[3]. See Figure 1(right).

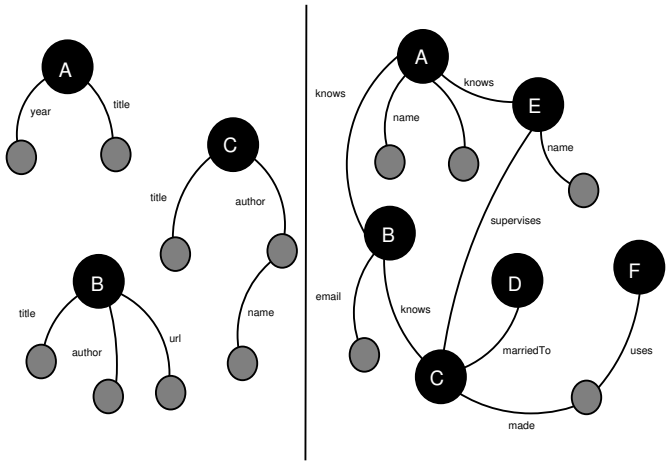

Fig. 1. Schematic Illustration of Semantic Forests and Semantic Webs.

Currently semantic forest data is pre-dominant. The number of true semantic webs should increase as the Semantic Web develops, however we do not expect semantic

[1] http://dublincore.org
[2] http://web.resource.org/rss/1.0/
[3] http://www.foaf-project.org

forests to disappear altogether. These different types of data present us with a challenge: How should Semantic Web learners deal with them? How can the different structures be exploited to improve learner performance? How should the learning outcome be represented to be as useful as possible in both the original context (where it was learned) as well as being portable to other scenarios?

In this paper we describe our experiences with aggregating, pre-processing and learning from FOAF data. Details on our experiments with semantic forest data can be found in [5]. We present experiments conducted using a hierarchical agglomerative clustering [6] algorithm to identify groups of people, followed by the application of the ILP system Aleph [7] to learn descriptions of these clusters. We evaluate some sample learned descriptions to see if they "make sense" as newly-discovered classes of individuals (either in a local or global context). Finally we discuss related work in this area and conclude with a summary of what we have achieved to date and our future plans.

2 Friend of a Friend

The Friend of a Friend (FOAF)[4] project aims to create a Web of machine-readable home-pages describing people, the links between individuals and the things they create and do. The FOAF ontology is described using the Ontology Web Language (OWL) [3]. To join the FOAF world all one has to do is generate a FOAF profile describing oneself and publish it on the Web. The profile must adhere to the ontology and could either be generated by hand, or more often, by copy, paste and edit of other people's FOAF, or by semi-automated tools such as FOAF-a-matic[5]. Part of an example profile is shown in Figure 2 [6]. This example illustrates several important things about FOAF:

- The *foaf:knows* property points to other people known by this person, creating a networked community.
- People in the FOAF world don't need URIs, they are identified through their *foaf:mbox* (or *foaf:mbox_sha1sum*), i.e. the email address. In the FOAF ontology these are identified as *owl:inverseFunctionalProperty*, meaning they uniquely identify a person.
- *foaf:knows* properties do not take the value of the URI of other people's FOAF, instead they point to an anonymous RDF node of type *foaf:Person*, which contains the *foaf:mbox* of the other person. Whether two anonymous nodes represent the same person can then be decided based on the *foaf:mbox* values; merging these nodes is known as "smushing"[7].
- *foaf:mbox_sha1sum* is used to disguise email-addresses for privacy reasons. The use of a checksum rather than just omitting the value allows people to confirm that the address actually does belong to a person.
- Other FOAF files are linked through *rdfs:seeAlso*, allowing Semantic Web bots to crawl through FOAF space.

[4] http://www.foaf-project.org/
[5] http://www.ldodds.com/foaf/foaf-a-matic.html
[6] The full example can be found at: http://www.csd.abdn.ac.uk/∼ggrimnes/foaf.rdf
[7] http://rdfweb.org/topic/Smushing

```
<foaf:Person>

 <foaf:mbox rdf:resource="mailto:ggrimnes@csd.abdn.ac.uk" />
 <foaf:name>Gunnar AAstrand Grimnes</foaf:name>
 <foaf:homepage rdf:resource="http://www.csd.abdn.ac.uk/~ggrimnes" />
 <foaf:workplaceHomepage rdf:resource="http://www.csd.abdn.ac.uk"/>
 <foaf:projectHomepage rdf:resource="http://www.csd.abdn.ac.uk/research/agentcities"/>
 <foaf:groupHomepage rdf:resource="http://www.csd.abdn.ac.uk/research/agentsgroup" />
 <foaf:phone rdf:resource="tel:+441224272835" />

 <foaf:depiction rdf:resource="http://www.csd.abdn.ac.uk/~ggrimnes/gfx/me.jpg" />

 <foaf:interest rdf:resource="http://www.w3.org/2001/sw/" />
 <foaf:interest rdf:resource="http://www.agentcities.net" />

 <foaf:made rdf:resource="http://www.csd.abdn.ac.uk/research/AgentCities/GraniteNights" />

 <contact:nearestAirport>
  <airport:Airport rdf:about='http://www.daml.org/cgi-bin/airport?ABZ' />
 </contact:nearestAirport>

 <foaf:knows><foaf:Person>
     <foaf:mbox rdf:resource="mailto:maym@foobar.lu" />
     <rdfs:seeAlso rdf:resource="http://martinmay.net/foaf.rdf"/>
  </foaf:Person></foaf:knows>
 <foaf:knows><foaf:Person>
     <foaf:mbox rdf:resource="mailto:apreece@csd.abdn.ac.uk" />
  </foaf:Person></foaf:knows>
 <foaf:knows><foaf:Person>
     <foaf:mbox rdf:resource="mailto:pedwards@csd.abdn.ac.uk" />
  </foaf:Person></foaf:knows>
 <foaf:knows>
  <foaf:Person foaf:name="Sonja A Schramm">
     <foaf:mbox_sha1sum>
      83276f91273f2900cf0b6657b3708b736276ef81
     </foaf:mbox_sha1sum></foaf:Person>
 </foaf:knows>

 <rdfs:seeAlso rdf:resource="http://www.csd.abdn.ac.uk/~ggrimnes/codepict.rdf" />
 <rdfs:seeAlso rdf:resource="http://www.csd.abdn.ac.uk/research/agentsgroup/foaf.rdf" />

</foaf:Person>

<rdf:Description rdf:about="">
 <wot:assurance rdf:resource="foaf.rdf.asc" />
</rdf:Description>
```

Fig. 2. Parts of Example FOAF File.

– The *wot:assurance* property at the bottom of the file points to a signature of this file, signed with the person's PGP key, providing a secure way to know who made these statements. This provides a basis for the trust layer of the Semantic Web architecture.

2.1 Topology

The FOAF project began around 1999, but only gained significant momentum in the past two years, due to the increased awareness of the Semantic Web and the existence

of FOAF visualisation tools such as Foafnaut[8] and the FOAF co-depiction project[9]. The co-depiction project allows searches to be made for pictures depicting multiple people, effectively proving their *foaf:knows* relationship, in addition is allows one to visually document the link from oneself to famous people, for example Bill Clinton or Frank Sinatra, increasing the fun-factor and "instant-gratification" of creating a FOAF profile. For our experiments we used a FOAF crawl from September 2003[10], which contains 9097 nodes of type person. When smushed, this is equivalent to 8908 people, of which 1980 people know at least one other person, i.e. they are not leaf-nodes of the FOAF knows-graph. The data consists of 147527 triples, using 201 different namespaces, and 1066 distinct properties (compared to 49 in the FOAF ontology). Many of these properties are not widely used, only 116 are used more than 100 times.

Within the FOAF graph one can typically identify groups of people that are very "close" in real-life. For example, the people within a single research group. In such a group there are many interconnecting *foaf:knows* links, and the level of detail about each person is similar since their profiles are often generated by copy and pasting one person's FOAF, or because they are all generated by the same person or from a database. A group often has only a very narrow connection to the rest of the FOAF graph, or in certain cases no connection at all. In the Aberdeen Computing Science FOAF graph for example most people know each other, but as shown in Figure 3, the only link to the rest of the FOAF network is through a single person node.

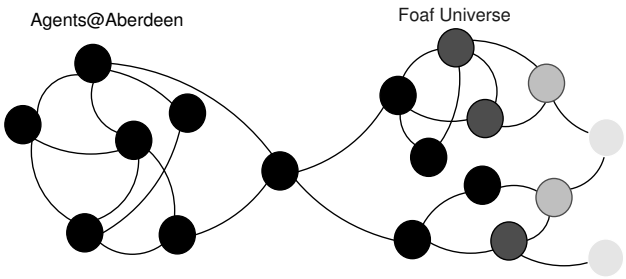

Fig. 3. FOAF Group with Narrow Connection to FOAF World.

2.2 Problems

While the heterogeneity and distributed nature of FOAF is clearly a good thing and makes a data-set based on FOAF very realistic, it does introduce a number of problems when one attempts to reason with or learn from the data. These are summarised below:

[8] http://jibbering.com/foaf/foafnaut.svg
[9] http://swordfish.rdfweb.org/discovery/2001/08/codepict/
[10] http://jibbering.com/foaf/dumps/

– Human errors. The majority of FOAF content is manually generated using a text editor, causing several types of human error:
  - Simple typing mistakes, i.e. *foaf:knosw*.
  - Using properties with the wrong namespace, e.g. *rdf:seeAlso* vs. *rdfs:seeAlso*.
  - Misunderstanding or misinterpretation of the FOAF ontology, e.g. using *foaf:mbox* with the email address as a literal string as opposed to an *rdf:resource* with a *mailto:* link.
– Weaknesses and/or inconsistencies of the FOAF ontology:
  - *foaf:mbox* vs *foaf:mbox_sha1sum*. Both properties are declared as *owl:inverseFunctionalProperty* as detailed above. However, nowhere is it formally declared that *foaf:mbox_sha1sum* is the Secure Hash Algorithm[11] checksum of the *foaf:mbox* property. The intention is of course that a node with a *foaf:mbox_sha1sum* matching the checksum of another's *foaf:mbox* should be smushed together. At the moment this must be hard-coded in an application specific manner.
  - No standard way of expressing interest. *foaf:interest* has range *foaf:Document*, and most commonly points to the URL of a page about the concept. Again, the use of literals vs. *rdf:resource* is inconsistent, but the main problem is that people use different URLs for the same concept. For example: http://www.w3.org/RDF/, http://rdfweb.org, http://rdfweb.org/, http://www.rdfweb.org/.
– Level of detail varies greatly. Our initial experiments with learning from FOAF returned several rules based simply on the presence of an attribute, such as *foaf:groupHomepage*, rather than the value of the attribute.

2.3 Enriching FOAF

The Advance Knowledge Technologies (AKT) project[12] aims to tackle a number of challenges of knowledge management, and as a show-case has created an ontology for representing academic researchers and their organisations. An RDF dataset conforming to this ontology has been created by "screen-scraping" the Web pages of UK based research institutions. The lack of detail in the FOAF data could be addressed by enriching it using the information available in the AKT RDF repository. However, this would involve further complicating the learning task by including yet another ontology. We therefore decided to map the instances from the AKT ontology to FOAF, as the ontologies have very similar domains; the majority of the mappings were straightforward, such as :

[*akt:has-email-address* ⇒ *foaf:mbox*]

Others were more complicated, for instance:

[*rdf:type akt:Professor-In-Academia* ⇒
(*rdf:type foaf:Person* & *foaf:title 'Professor'*)].

[11] http://www.w3.org/PICS/DSig/SHA1_1_0.html
[12] http://www.aktors.org

rdf__type(A, *'akt__Professor-In-Academia'*):-
  rdf__type(A,*'foaf__Person'*),
  foaf__title(A, *'Dr'*).

Fig. 4. Ontology Mapping Excerpts.

For the sake of re-usability these mappings were represented in OWL using *owl:equivalentProperty* for the trivial mappings and our own RDF mapping of RuleML[13] for the more sophisticated rules, like the rule shown in Figure 4.

3 RDF & ILP

Encouraged by our earlier experiences with learning from semantic forests using ILP [5], the next step was to explore the application of these techniques to FOAF data. The mapping of RDF to Prolog is straightforward. Figure 5 illustrates a fragment of the FOAF profile in Figure 2 converted to Prolog, some things to note about the representation are:

- Namespace handling. Namespaces of properties were converted to prefixes in Prolog, with *namespace* predicates giving the mapping from prefixes to actual namespaces. For example *foaf:mbox* becomes:

  foaf__mbox(A, *'mailto:ggrimnescsd.abdn.ac.uk'*).
  namespace(*'foaf'*, *'http://xmlns.com/foaf/0.1/'*).
- RDF types. For each class in the ontology Prolog rules are created to determine if a resource is a member of the specific class, or any sub-class thereof. This allows RDF types to be mapped to ILP internal types, used for limiting which predicates may be applied to a given resource, reducing the search-space dramatically. Figure 6 contains an example.
- Normalisation and inference over interests. In our initial experiments with the FOAF data we attempted to fix the inconsistent *foaf:interest* problem by "smushing" nodes that represented the same concept, for example *http://rdfweb.org/foaf/* and *http://www.foaf-project.org/*. In addition super/sub-concept links were created between concepts such as *http://www.debian.org/* and *http://www.linux.org/*, and some general nodes that did not appear in the original data, e.g. *#ProgrammingLanguages* were added. Our preliminary experiments demonstrated that the ILP learner did not use these generalisations, probably due to the low number of *foaf:interest* links actually appearing in the data. As a result, these extra rules were not included in our full experiments.

An additional advantage of using ILP with RDF is that converting the learned results back into RDF is trivial, given some way of representing Horn clause rules in RDF, e.g. the Semantic Web Rule Language (SWRL) [8].

[13] http://www.csd.abdn.ac.uk/∼qhuo/program/generaltool_sources/ruleml.rdfs

rdf__type(*'genid:002'*, *'foaf__type'*).
foaf__name(*'genid:002'*, *'Gunnar AAstrand Grimnes'*).
foaf__mbox(*'genid:002'*, *'ggrimnes@csd.abdn.ac.uk'*).
foaf__knows(*'genid:002'*, *'genid:003'*).
foaf__mbox(*'genid:003'*, *'apreece@csd.abdn.ac.uk'*).
...

Fig. 5. FOAF Fragment converted to Prolog.

```
foaf__Person(A):-
  instanceOf(A,'http://xmlns.com/foaf/0.1/Person').
foaf__Document(A):-
  instanceOf(A,'http://xmlns.com/foaf/0.1/Document').

instanceOf(A,B):-rdf__type(A,B).
instanceOf(A,B):-rdf__subClassOf(B,C),instanceOf(A,C).
instanceOf(A,unknown):-nonvar(A).

castAsfoaf__Person(A,A):-foaf__Person(A).

:-modeb(*,foaf__interest(+foaf__Person, -foaf__Document)).
:-modeb(1,castAsfoaf__Person(+resource,-foaf__Person)).
:-modeb(1,castAsResource(+foaf__Person,-resource)).
```

Fig. 6. RDF Type Inference in Prolog.

4 Learning from FOAF

For our experiments with FOAF data we used Aleph [7]. Before attempting to learn from the FOAF data it was pre-processed by first smushing it, and then removing any duplicate properties resulting from this merger. Aleph was initially configured to use any of the predicates appearing in the input data when constructing a hypothesis, only restricted by the RDF typing as detailed above. However, as there were 1066 predicates in the full dataset, this was too much for Aleph to deal with and we moved to only using a subset based on the most frequent occurring predicates. The 15 most frequently predicates are shown in Figure 7, and the preliminary experiments were done with these, excluding *rdf:type* as it is applied to every person node. However, this did not give very good results and we moved instead to using the 100 most frequent predicates; for space reasons that list is not re-produced here.

Initial exploratory experiments with Aleph highlighted problems with the scale of the FOAF dataset. Even with a very small subset of the full data (less than 10% of the people in the full crawl), the search-space was still far too large, and Aleph was unable to make any generalisations over the data. To reduce the size of the search space the problem was broken into sub-problems by first applying a clustering algorithm and then feeding each cluster to Aleph separately. Such an operation does make sense in the context of FOAF, as there are often clusters of people, reflecting real-life groups, e.g. research groupings, where the group membership may not be explicitly stated. To perform the clustering step a hierarchical agglomerative clustering algorithm (HAC) [6] has been employed. HAC is a greedy bottom-up clusterer which works by initially

Frequency	Property
1244	http://www.w3.org/1999/02/22-rdf-syntax-ns#type
1120	http://jibbering.com/foaf/jim.rdf#isKnownBy
1119	http://xmlns.com/foaf/0.1/knows
908	http://xmlns.com/foaf/0.1/mbox_sha1sum
906	http://xmlns.com/foaf/0.1/name
846	http://www.w3.org/2000/01/rdf-schema#seeAlso
419	http://xmlns.com/foaf/0.1/depiction
392	http://xmlns.com/foaf/0.1/surname
344	http://xmlns.com/foaf/0.1/firstName
327	http://purl.org/dc/elements/1.1/title
273	http://xmlns.com/foaf/0.1/codepiction
266	http://xmlns.com/foaf/0.1/mbox
246	http://xmlns.com/foaf/0.1/nick
236	http://xmlns.com/foaf/0.1/homepage
230	http://purl.org/dc/elements/1.1/description

Fig. 7. 15 Most Frequent Predicates in FOAF.

creating one cluster for each individual, then repeatedly merging the two closest clusters until there is only one left or some threshold for similarity is reached. Our version of HAC computes the distance between two clusters as the average distance between each of the individuals in the two clusters.

Initially, a modified version of Hamming distance [9] is used as similarity metric; modifications were as follows: each RDF property appears as an attribute of the instance, as does each (property, value) pair. All properties and values are treated as nominal, including anonymous nodes. Although there could have been some scope for treating datatyped properties as ordinal, few of the properties used in FOAF are typed, so dealing with extra complexity was unlikely to pay off. The intention behind this similarity metric is that two people that both have a certain property, say *foaf:interest*, are more similar than two people who do not share any attributes, but less similar than two people who have the same value for *foaf:interest*. Initial clustering experiments with this similarity metric were unsuccessful, manual inspection of the clusters showed that they did not in any way re-produce groups of people a human might have identified. It was clear that considering only direct attributes of the Person node was flawed. A similarity metric is needed that can take into account the FOAF graph immediately around a person, and her position in the bigger graph, not just the immediate attributes. We are not aware of any work to date on the subject of similarity metrics for RDF data. However, in [10] a similarity metric for conceptual graphs is presented. Conceptual graphs are a data-structure commonly used for natural language processing. They consist of a network of concept nodes, representing entities, attributes, or events and relation nodes between them. A simple conceptual graph representing *John loves Mary* is as follows:

$$[\text{John}] \Leftarrow (\text{subj}) \Leftarrow [\text{love}] \Rightarrow (\text{obj}) \Rightarrow [\text{Mary}]$$

The similarity metric developed is based on the idea of the Dice coefficient [11], but incorporating a combination of two complementary sources of similarity: the *conceptual similarity* and the *relational similarity*, i.e. the overlap of nodes and the overlap of edges within the two graphs. Full details of the similarity metric can be found in [10].

Conceptual graphs and RDF instances are sufficiently similar that the same similarity metric should be appropriate in both cases. The similarity metric is designed to

work on separate graphs, and in order to apply it to RDF we had to modify the algorithm to extract a sub-graph around each person. The RDF graph is traversed in either direction from the person node, i.e. triples were considered where the node in question is either the subject or the object. To limit the size of the sub-graph the number of triples traversed is limited. Trial and error showed that the optimal sub-graphs for clustering were obtained if traversal was allowed two triples forwards and one backwards. Note that these traversals may **not** be in any order, and backward-traversals are only permitted from the initial node. Consider for instance Figure 8, the subgraph for the person "Gunnar Grimnes" would include one backwards traversal, i.e. the "Comp. Sci. Dept" node, but not the "University of Aberdeen" node. It would also include two forward traversals, to both the anonymous node and the literal title node.

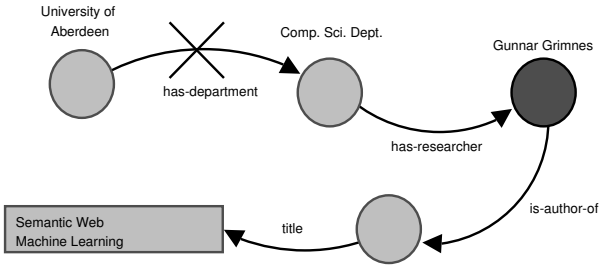

Fig. 8. Example of Extracting Person Subgraph.

Clustering with this similarity metric gave acceptable results. For example the algorithm was able to discover clusters of people from different research-groups at Aberdeen. Aleph was then applied to the generated clusters (as detailed above), to learn a concise description of each cluster.

5 Results

The descriptions presented here were learned using the method detailed above on a subset of 869 of the total FOAF people.

Initially experiments were conducted using the 100 most frequent predicates for both clustering and rule learning; these included *foaf:knows* and its generated inverse predicate. With these settings 219 clusters had rules generated by Aleph, out of the total 825 clusters generated. Most of the learned rules were of the type:

member(A) :-
jibbering__isKnownBy(A,*'http://norman.walsh.name/knows/who#norman-walsh'*). (This rule covered all the 216 people in the cluster it described.)

Specifically, out of the 219 rulesets 177 used either *foaf:knows* or its inverse. It was apparent that the *foaf:knows* relation was so predominant in the FOAF data that it overshadowed everything else. Rules using *foaf:knows* are not very re-usable, and do not

really express general classifications of the data. Therefore, the next experiments were conducted using the same 100 predicates, but removing *foaf:knows* and its inverse. The results for these experiments were much more interesting, as the lack of *foaf:knows* forced Aleph to generate rules using the other predicates. In addition, clustering and learning from the FOAF data excluding *foaf:knows* was much more efficient, the clustering step alone took only a third of the time when done without *foaf:knows*.

For space reasons we will not present all the learned descriptions here, but only discuss a selection of rules describing interesting clusters. Figure 9 shows for each rule the following: the size of the cluster; a recall measure (the number of instances covered by the rule); the false negatives (the members of the cluster not covered by this rule); and the false positives (people covered by this rule who are not a member of the cluster).

#	Rule	Cluster Size	Recall	False Neg.	False Pos.
1	**member**(A) :- **trust__trustsHighly**(B,A).	8	8	0	0
2	**member**(A) :- **foaf__groupHomepage**(A,*'http://www.aktors.org'*).	13	13	0	0
3	**member**(A) :- **pim__nearestAirport**(A, *'http://www.daml.org/cgi-bin/airport?ABZ'*).	12	12	0	2
4	**member**(A) :- **dc__creator**(B,A), **dc__format**(B, *'application/postscript'*).	17	15	2	0
5	**member**(A) :- **dc__creator**(B,A), **dc__title**(B, *'Managing Reference: Ensuring Referential Integrity of Ontologies for the Semantic Web'*).	8	8	0	0

Fig. 9. Selected FOAF Cluster Description Rules.

Rule 1 appears interesting as it characterises people who are highly trusted by someone. The *trust:trustedHighly* predicate comes from the namespace *http://www.mindswap.org/∼golbeck/web/trust.daml#*, and is part of an effort to extend the FOAF network with the concept of trust; one application being email filtering [12]. Although the rule at first sight looks meaningful it becomes apparent that there are very few people in (our subset of) the FOAF world that use this particular predicate. The rule has no false positives, so the 8 people in this cluster represent all the uses of this predicate in our data, and in fact 7 of them are trusted by the primary author of the paper cited above. If the use of FOAF for enabling a Web of trust becomes more popular in the future such a rule would indeed be interesting. However, the question that must be raised is whether such a rule is too general to be useful, or if it is sufficient to know that a person is highly trusted by someone. In a local context this might be significant, but in a global context such knowledge is unlikely to have any great utility.

Rule 2 describes the Advanced Knowledge Technologies group at Aberdeen; these people were originally described using the AKT ontology, but the descriptions were converted to FOAF as described earlier. Based on the subset of data used for our experiments this rule perfectly describes the AKT group of people. However, in the real world there are other AKT groups at other institutions, and although this rule describes a meaningful group of people, namely the group of all members of the AKT project, it is not specific to the local Aberdeen group.

Rule 3 describes the Aberdeen Agents research group (Agents@Aberdeen), the airport referenced in the rule being Aberdeen airport, Dyce (ABZ). Although the people

in the AKT research group are also situated in Aberdeen, the AKT ontology has no information regarding location, and so the FOAF description is lacking this information. A better description of the agent group might have been :

member(A) :- **foaf__groupHomepage**(A,
'http://www.csd.abdn.ac.uk/research/agentsgroup/').

However, as there is one person who is a member of both the AKT and the Agents research group, and in this experiment was inserted into the AKT cluster, this rule would have had an additional false positive, and rule 3 was chosen in its place.

Rule 4 describes a cluster of people who have all created a postscript document. There are 4 postscript documents in our dataset, and the 15 people covered by this rule are the authors of these documents; all the people and documents are from the same UK institution. This is a good example of the bizarre clusters and rules that sometimes occur within the FOAF space. Although a human might perhaps identify this rule as less significant than for instance one using *foaf:groupHomepage*, Aleph has no way of making the distinction. This rule also illustrates how RDF created by copy and paste or from a database can produce artificial clusters, based on the use of certain schemas or predicates particular to that cluster.

Rule 5 describes a cluster made up of 8 people who co-authored a paper. This is clearly a meaningful cluster, although quite small and the scope for using it for classification is limited. This rule is also interesting because the actual publication (Variable B) is sometimes given a URI, and sometimes just referred to as an anonymous node. Aleph must therefore use another clause to identify it. This is analogous to the FOAF use of *foaf:mbox* to identify people. However, in the FOAF case we can pre-process the data and smush the nodes because *foaf:mbox* is declared to be inverse functional in the ontology; this illustrates how background knowledge in an ontology can facilitate learning.

6 Related work

Improving learning performance by taking advantage of the structure that is inherit in data that is marked up using XML is discussed in [13]. XML documents are represented as ordered and labeled trees and the authors present an algorithm called XMiner to extract the most frequent sub-trees for a given class; these are then converted into rules for classifying new instances. The authors demonstrate that their classifier out-performs information retrieval or association rule classifiers when learning from XML data.

Exploiting structure and semantics for learning is also discussed in [14], where ontologies are used to enrich plain text and do feature selection and aggregation. The aim being to improve clustering results. The authors also use semantic meta-data about Web pages to perform web-mining; a user's navigational path through a site becomes a path through semantic concepts, which might be more comprehensible than the raw access-log. The paper also includes a brief discussion of applying ILP to Semantic Web data, highlighting the challenge of solving the scalability problems of ILP to make it usable on the Semantic Web.

Alani et al [15] uses ontologies to detect Communities of Practice (COP) that are only implicitly expressed. For instance, two people might not have a direct relation, but they might have written a paper together. The detection is based on analysis of the graph of people and properties and allows weights to be attached to possible relations. For example, it is more significant that two people have written a paper together, than the fact that they subscribe to the same journal. Experiments are performed using the same AKT ontology used for the work described in this paper. In [16], the COP detection mechanism is combined with ontologies to generate an initial user-profile for a hybrid-recommender system called Quickstep. Analysis of a user's publications taken from her homepage are used to determine interest weights for concepts in the ontology, the user is then matched with similar users in the COP and their combined profiles are in turn matched with the concept weights of research papers to recommend papers of interest, even to new users of the system. Middleton et al also present experiments comparing ontology supported recommendations to those made without ontological inference. This work is continued further in [17], where a new system called Foxtrot which includes profile visualisation, email notification and user feedback.

The European Elena project[14] aims to demonstrate the feasibility of smart spaces for education, and is using Semantic Web technology to achieve this. In [18] RDF metadata for educational resources is combined with an RDF profile to provide a personalised and adaptive view of a hypermedia learning-space.

7 Conclusion

In this paper we have shown how clustering and inductive logic programming can be used to learn descriptions of groups of people from FOAF data. We believe that the type of descriptions that have been shown to have been learned in this paper identify interesting classifications in the data that were not initially specified in the RDF schemas or OWL ontologies. Additionally, these new classes could be integrated back into the original ontologies or instance data, for example by expressing them either as OWL descriptions (using restrictions on property values) or using the Semantic Web Rule Language [8].

Evaluating the information value of the newly-learned descriptions must ultimately be done by a human, since the semantics can never truly be understood by a machine. However, some steps can be taken to filter out the less useful descriptions. We have shown that removal of the *foaf:knows* relation eliminated the generation of very specific clusters surrounding a particular person. These clusters were less useful as it appears in general very hard to generalise a *foaf:knows* relationship any further. Moreover, rules without literal values have less information content than rules specifying a value. Although to an ILP algorithm the presence of a predicate looks significant, we must consider the open world assumption of RDF, in that other FOAF profiles outside our current dataset may also make use of the predicate, which might render classification by this rule incorrect.

We are planning to extend this work by conducting experiments in which we integrate the learned knowledge into the original data, and then re-run the clustering and

[14] http://www.elena-project.org/

learning steps, effectively creating a form of feedback learning. Our hope is that by using the additional descriptors it should be possible to learn even richer and more interesting classifications.

Any attempt to apply machine learning techniques to the Semantic Web will have the problem of scale, and even in the limited domain of FOAF we had to limit our experimental study to a relatively small dataset to gain acceptable performance from the clustering and ILP steps. We will conduct further research into how the scalability of the learning algorithms can be improved to a level where we can at least learn from the whole FOAF crawl. However, for a global Semantic Web of interconnected information, like FOAF, the scalability challenge is huge.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001)
2. Brickley, D., Guha, R.V.: Resource description framework (rdf) schema specification. W3c recommendation, World Wide Web Consortium (2000)
3. McGuinness, D.L., van Harmelen, F.: Web ontology language (owl): Overview. W3c working draft, World Wide Web Consortium (2003)
4. Lassila, O., Swick, R.R.: Resource description framework (rdf) model and syntax specification. W3c recommendation, World Wide Web Consortium (1999)
5. Grimnes, G.A., Edwards, P., Preece, A.: Learning from semantic flora and fauna. In: AAAI, Submitted to Semantic Web Personalization Workshop. (2004)
6. Vorhees, E.: Implementing agglomerative hierarchical clustering algorithms for use in document retrieval. In: Information Processing & Management. Volume 22. (1986) 465–476
7. Srinivasan, A.: The Aleph Manual. (2001)
8. Horrocks, I., Patel-Scheider, P., Boley, H., Tabet, S., Groshof, B., Dean, M.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML. DARPA DAML Program. (2003)
9. Hamming, R.: Error detecting and error correcting codes. Bell System Techincal Journal **29** (1950) 147–160
10. Montes-y-Gómez, M., Gelbukh, A., López-López, A.: Comparison of conceptual graphs. In: Lecture Notes in Artificial Intelligence. Volume 1793. Springer Verlag (2000)
11. Rasmussen, E.: Clustering algorithms. In Frakes, W., Baeza-Yates, R., eds.: Information Retrieval: Data structures & Algorithms, Prentice Hall (1992)
12. Golbeck, J., Parsia, B., Hendler, J.: Trust networks on the semantic web. In: Proceedings of Cooperative Intelligent Agents 2003, Helsinki, Finland (2003)
13. Zaki, M.J., Aggarwal, C.C.: Xrules: An effective structural classifier for xml data. In: 9th International Conference on Knowledge Discovery and Data-mining. (2003)
14. Berendt, B., Hotho, A., Stumme, G.: Towards semantic web mining. In: International Semantic Web Conference. (2002)
15. Alani, H., Dasmahapatra, S., O'Hara, K., Shadbolt, N.: Identifying communities of practice through ontology network analysis. In: IEEE IS. (2003) 18–25
16. Middleton, S., Alani, H., Shadbolt, N., De Roure, D.: Exploiting synergy between ontologies and recommender systems. In: 11th International WWW Conference, Semantic Web Workshop. (2002)
17. Middleton, S., Shadbolt, N., Roure, D.D.: Ontological user profiling in recommender systems. In: ACM Transactions on Information Systems. Volume 22(1). (2004) 54–88
18. Dolog, P., Henze, N., Nejdl, W., Sintek, M.: Towards the adaptive semantic web. In: 1st Workshop on Principles and Practice of Semantic Web Reasoning. (2003)