

Towards a Goal Driven Learner for the Semantic Web

Gunnar AA. Grimnes

Alun Preece

Pete Edwards

Computing Science Department, University of Aberdeen, Aberdeen, AB24 3UE, UK
{ggrimnes,pedwards,apreece}@csd.abdn.ac.uk

1. INTRODUCTION

In our recent research [1] we have argued that when the Semantic Web (SW) becomes a reality, there will be a need for machine learning methods that can operate in a Semantic Web framework. Although the SW will deliver ontology based structured information, we argue that not every piece of useful knowledge will be explicit or inferable; and that learning will help to identify this knowledge. In addition, such an interconnected knowledge-web should be easy to learn from, compared to the current unstructured World Wide Web, so we also hypothesise that SW learning should outperform learning from unstructured text.

2. A GOAL DIRECTED LEARNING AGENT

In our research to date we have demonstrated two points. Firstly, to benefit from the added structure and semantics of SW data a symbolic and knowledge-rich learning algorithm is required. We have successfully used the Inductive Logic Programming (ILP) system Aleph [5]. Secondly, even with the SW still in its infancy, it is already far too large to process in its entirety. Even for subsets of the SW, such as FOAF¹, *trivial* operations such as crawling, smushing (anonymous node disambiguation) and RDFS inference require huge amounts of memory, time and storage space. More advanced processing, such as learning, is impossible within a reasonable timeframe.

These two issues have lead us to reconsider our methods for SW learning. As global reasoning is not possible, a learning agent needs an intelligent way of choosing a subset of the known resources to be used for learning. Also, when a suitable subset has been identified, an intelligent agent should narrow the search-space further by concentrating on particular predicates for constructing an hypothesis. Both of these methods are clearly dependent on the task the agent is trying to achieve.

To solve these challenges, we have devised a new goal-driven approach to learning from the SW, where each step

¹<http://www.foaf-project.org/>

of the learning process is influenced by the agent's goal [3]. In the following discussion we assume goals to be in the form of partially instantiated RDF graphs, much like queries. (As classified by Ram and Leake, these goals are *knowledge goals*, an agent could also have *task goals* [4], but these are not discussed here.) The internal processes of such an agent are shown in Figure 1.

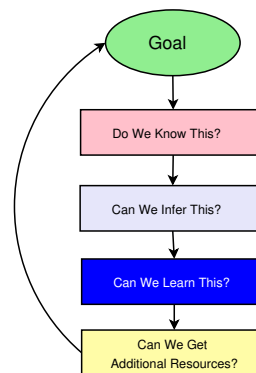


Figure 1: The Goal-Driven Architecture.

A fundamental assumption for this architecture is that retrieval of resources from the SW is expensive. Therefore, the agent keeps a cache of SW data previously retrieved, and will always initially attempt to fulfil goals using this cache. The information seeking process is also recursive, if the agent determines that a particular piece of missing information could help inference or learning, it may formulate this as a new goal and launch the process again. We believe that this type of recursive information seeking, allowing the agent to take a more planned approach to learning [2], has potential to be more efficient than a less directed approach.

When given a goal an agent will first check if the local cache can already answer it. If it fails, the agent will attempt to make inferences to answer this query. We are taking a very naïve approach to inference: we will use a simple forward/backward chaining rule system, much like the inference engine of the SW toolkit Jena². Our system is implemented in Prolog and bounded by the number of nodes explored and/or time, and is thus not complete. We envisage the agent having several types of rules at its disposal; fundamentally it should know RDFS and OWL rules,

²<http://jena.sourceforge.net/inference/index.html>

but may also have domain specific rules and rules learned in previous iterations.

If inference fails to fulfil the current goal, the agent will attempt to learn it. At the time of writing, this means that the agent will call out to Aleph. Running Aleph is very expensive, and it is therefore essential that the agent has some guidance to determine whether learning might yield results. At the moment this takes the form of hard coded heuristics, for example, the agent may notice that all known email addresses are unique to a single individual, and conclude that attempting to learn email addresses is unlikely to be successful.

If the agent believes it is worth attempting to learn it will set up an Aleph run, using heuristic guidance to choose positive and negative examples depending on the current goal. Any learned rules are added to the local cache and may be re-used for later queries.

Finally, if all other steps have failed the agent may consult additional resources on the SW. Determining which resources and what parts might be helpful is a very difficult problem, and again we are taking a naïve approach. We will use a simple service description language allowing us to manually annotate data-sources, either by stating that a data-source covers a particular namespace, or by giving a triple pattern. The agent also knows how to fetch OWL ontologies, and will also attempt to fetch RDFS ontologies from their namespace URIs. This setup should allow the agent to match the characteristics of the current goal with the data-source descriptions to determine what resource to retrieve. When new information has been added to the local cache the information seeking process restarts.

3. ILLUSTRATIVE APPLICATION

We plan to explore and evaluate this approach by building a tool for exploring movies and their relations. Imagine a journalist writing a weekly movie-magazine trivia column. Our tool will allow him or her to explore relations between particular groups of movies, for instance, they may wish to compare movies filmed in Britain in the 70s with those of the 80s. The initial goal of the agent would then be $[(?x \text{ imdb:period } \text{ imdb:70s}), (?x \text{ imdb:filmedIn } \text{ :Britain})]$. The agent will first consult its local cache for matching movies. Imagine that the country information is missing for a particular movie, but we know it is filmed in London. However, the domain rule allowing us to infer that London is in Britain is missing. The agent will then attempt to learn a rule that could help, and in this case it would know that other movies filmed in Britain make good positive examples, while movies filmed elsewhere make negative ones. Also, it will instruct Aleph to only use the predicates known about the movie under consideration when constructing its hypothesis. Aleph would then be able to construct a rule like $[(?x \text{ imdb:filmedIn } \text{ :London}) \Rightarrow (?x \text{ imdb:filmedIn } \text{ :Britain})]$. If learning also fails, for instance if there are no movies that are explicitly stated to be filmed in both London and Britain, the agent would consult the annotated data-sources it has access to: the IMDb data-source and the CIA Factbook. The IMDb data-source provides information about the imdb namespace, but this is already consulted for this movie; the CIA Factbook is annotated as containing triples matching $[(?x \text{ geo:locatedIn } \text{ ?y})]$, the agent determines this could help the inference step, fetches the resource and the process starts again. Eventually, the agent will have lists

“Sci-fi films starring an actor who has guest-starred in the TV-series The Troubleshooters.”

$\text{likes}(A):-\text{imdb_genres}(A,\text{Sci-Fi}), \text{imdb_cast}(A,B),$
 $\text{imdb_tvGuestIn}(B,\text{The Troubleshooters}).$

“Films produced by someone who is 1.78m tall.”

$\text{likes}(A):-\text{imdb_producerOf}(B,A),$
 $\text{imdb_height}(B,1.78 \text{ m}).$

Figure 2: Example Rules Learned from IMDb.

of British movies from the 70s and 80s and will set up the Aleph run to learn rules that distinguish them.

For building this application data about the top ranked 250 films and the people involved in making them ($\approx 10,000$ people) has been extracted from the Internet Movie Database (IMDb)³. This gives us an RDF dataset of 150,000 triples. We have conducted preliminary experiments on this data by learning personal preferences, and some rules are presented in Figure 2. These are rules learned from collections of 25 favourite films selected by users. These rules illustrate two things: firstly, it is possible to extract interesting trivia from this data using ILP techniques, and secondly, a goal-directed approach to learning is clearly needed, for example, the height of a director is not relevant to learning the interest profile of a user, but the genre clearly is.

4. CONCLUSION

We have outlined a goal-directed approach to learning from the Semantic Web which has several desirable properties. Learning is seen as an independent process, but is integrated with general inference and information gathering. Although our inference is incomplete and the accuracy of the learned knowledge is not guaranteed, this does allow the agent to handle inconsistent and noisy data, which is very important when working with the SW. Using a goal-directed approach provides methods for reducing the dimensionality of the learning problem and for optimising the search for suitable hypotheses. We believe this is very important as scalability remains the prime concern for the future of the SW.

5. REFERENCES

- [1] G. AA. Grimnes, P. Edwards, and A. Preece. Learning Meta-Descriptions of the FOAF Network. In *Proc. 3rd Int. Semantic Web Conf. (ISWC-04)*, pages 152–165, 2004.
- [2] L. Hunter. Planning To Learn. In *Proc. 12th Annual Conf. of the Cognitive Science Society*, pages 26–34. Erlbaum Associates, 1990.
- [3] R. Michalski. *Machine Learning: A Multistrategy Approach*, volume 4, chapter 1. Morgan Kaufmann, 1994.
- [4] A. Ram and D. Leake. Learning, Goals and Learning Goals: A Perspective on Goal Driven Learning. *Artificial Intelligence Review*, 9(6):387–422, 1995.
- [5] A. Srinivasan. *The Aleph Manual*, 2001. <http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/>.

³<http://www.imdb.com/>