Automated Theory Formation in Bioinformatics

Simon Colton
Department of Computing
Imperial College of Science, Technology and Medicine
180 Queens Gate, London SW7 2BZ, United Kingdom

A theory learned by an inductive logic programming (ILP) system such as Progol [5] usually comprises a set of concepts, expressed as logic programs, which can be employed for a classification task. This classifying ability can, in turn, be used for prediction tasks. A scientific theory, however, comprises much more information: concepts; hypotheses relating concepts; explanations and empirical justifications of the hypotheses; representation schemes; experimental methodologies and so on. Working mainly in mathematics, we have used the HR system [1] to form theories about some objects of interest in a domain. For example, in group theory, where the objects are groups, HR invents concepts, makes conjectures about those concepts, and proves (some of) the conjectures using the Otter theorem prover [4]. Despite it's history in mathematics, we have developed HR as a domain-independent machine learning program. In particular, the format for background information is very similar to that for Progol. Given this, we are currently exploring various possibilities for automated theory formation (ATF) using bioinformatics datasets. We describe here an application of HR to the mutagenesis data set [6] and suggest some advantages of ATF over ILP, some disadvantages, and some possibilities for the fruitful combination of the two techniques.

1 Automated Theory Formation

Given the diverse nature of a scientific theory, our theory of how to automate the production of a theory from some background information is still evolving, and at any one time, we can only take a snapshot of the current state. ATF comprises two main functionalities applicable in bioinformatics, namely concept formation and conjecture making. Automated Theory Formation is more fully described in [1], and we briefly describe these two functionalities below.

HR is able to invent new concepts from old ones, using a set of 10 general production rules. The four we are concerned with here are:

- The exists rule: this adds in existential quantification
- The compose rule: this combines two concepts using conjunction
- The split rule: this performs instantiations
- The negate rule: this introduces negation by finding compliments

HR is also able to estimate the interestingness of the concepts it produces using a set of measures described in [3]. One simple measure is the complexity of a concept, which is calculated as the number of concepts in its construction history.

HR uses empirical evidence to make conjectures about the concepts it invents. In particular, whenever HR invents a concept, it checks whether it has the same examples as a previous concept. If so, it makes the conjecture that the two concepts are equivalent. Similarly, when it invents a concept that is new (there is no concept with the same examples in the theory), HR checks all previous concepts to find any for which the examples are a subset of the examples for the new concept. For each old concept it finds with this property, it makes an appropriate implication conjecture. It similarly checks for concepts where the examples are a superset of the examples for the new concept, and makes the appropriate implication conjecture. HR is also able to make "near" conjectures: given a user-specified lower percentage limit, if HR notices that the percentage of examples shared by the two concepts is above this limit, it makes such a "near" equivalence.

HR is also able to extract simpler conjectures from those it finds empirically. Having made an equivalence conjecture, HR can extract two implication conjectures from that. Given an implication conjecture, HR is able to extract implicates from that. For example, suppose HR made the following equivalence conjecture:

$$p(X,Y) \wedge q(X) \leftrightarrow r(Y,X) \wedge s(X,X).$$

After extracting the two obvious implication conjectures, HR can then extract the following implicates:

$$p(X,Y) \wedge q(X) \rightarrow r(Y,X); p(X,Y) \wedge q(X) \rightarrow s(X,X); r(Y,X) \wedge s(X,X) \rightarrow p(X,Y); r(Y,X) \wedge s(X,X) \rightarrow q(X)$$

HR is able to express its concepts and conjectures in a variety of languages. In particular, it is able to express implicate conjectures as Horn Clauses in a Prolog format. For example, the final implicate above can be expressed as: `q(x) :- r(Y,X), s(X,X).`

2 Mutagenesis Experiment

Predictive toxicology is an important application area in bioinformatics. In its very simplest sense, these problems can be stated as machine learning problems by (i) specifying some examples of chemicals which are known to be toxic, and some examples which are known to be non-toxic (ii) specifying some concepts which describe the chemicals and (iii) asking a learning program to discover a concept (or set of concepts) based on the background information which achieves high accuracy when used to predict the classification of a chemical into toxic or non-toxic.

The mutagenesis data set is one for which ILP has been very successful. Part of this data set comprises a set of 42 chemicals, for which we know 13 are mutagenic and 29 are not. The data for this application involved the concepts of atoms, molecules and bonds. As discussed in [6], Progol discovered the following hypothesis which achieved an 88% accuracy in predicting mutagenicity over the 42 examples supplied:

```
active(A) :- bond(A,B,C,2), bond(A,D,B,1), atom(A,c,21).
```

This states that a molecule is active (mutagenic) if there is a carbon atom of type 21 (on a benzene ring) single bonded to another atom, which is, in turn, double bonded to a third atom. In [6], the authors state that the intended users of this information actually prefer more complicated prediction concepts over less complicated ones. In this context, more complicated concepts describe larger substructures within mutagenic molecules.

To use automated theory formation for this machine learning task, we can simply run HR to form a theory, given the same background information as Progol, then extract any concepts which achieve a similar categorisation of the molecules as the active/inactive one supplied. This is essentially what we did, but enhanced the process with a "reactive" search, which enabled HR to react (in terms of forcing particular concept formation steps) whenever a concept was discovered which had a categorisation close to the one desired. Reactive searches are an extension of the forward look-ahead mechanism described in [2].

Using a leave one out method, for each session, we recorded the concept which scored highest in terms of the predictive accuracy with respect to the active/inactive categorisation. We asked HR to form a theory for 30000 steps using the four production rules previously described, with a complexity depth limit of 8. In each case, the concept HR learned was:

```
active(A) :- bond(A,B,C,1), atom_type(B,F,21), bond(A,C,D,E)
```

This states that a molecule is active if there is an atom of type 21 which is single bonded to another atom, which is bonded (somehow) to a third atom. This concept also achieves an 88% prediction accuracy over the 42 examples. Given that this concept has less information content than the one learned by Progol, this naturally raises the questions of whether (a) HR can learn a more complicated concept for the prediction task, and (b) whether it can learn the same concept as Progol, or even a more complicated one.

Note that HR also discovered many equivalence conjectures, which, while possibly untrue about molecules in general, were certainly true of the examples given as background information. Looking at the theory for a particular run, we asked HR to extract all conjectures that were relevant for its learned prediction concept. To do this, it looked at each conjecture and assessed whether either the left hand or right hand side of the conjecture subsumed, or was subsumed by the concept definition. Using this method, HR extracted around 100 equivalence conjectures.

In particular, it brought to our attention these two alternative definitions (i.e., covering the same set of positive and negative examples) for the prediction concept:

```
active(A) :- \+ (bond(A,B,C,1), atom_type(B,F,22), bond(A,C,D,E))
active(A) :- atom_type(B,E,21), atom_type(C,F,38), bond(A,B,C,D)
```

This first equivalent definition represents a discovery on HR's part: it is a new structural predictor with 88% accuracy. It states that a molecule is mutagenic if it doesn't have an atom of type 22 single bonded to another atom which is itself bonded to another atom.

If we look now at the second alternative definition, then we can use some more equivalence conjectures found by HR to expand the definition. Given that HR made the conjectures that:

```
atom(B,X,21) <-> atom(B,c,21)
atom(B,X,38) <-> atom(B,n,38)
```

Then we can re-write this hypothesis:

```
active(A) :- atom_type(B,E,21), atom_type(C,F,38), bond(A,B,C,D)
```

like this:

```
active(A) :- atom_type(B,c,21), atom_type(C,n,38), bond(A,B,C,D)
```

HR also made this conjecture:

```
bond(A,B,C,X1), atom(C,X2,38) <-> bond(A,B,C,1), atom(C,X3,38)
```

Which enables us to re-write our predictor definition thus:

```
active(A) :- atom_type(B,c,21), atom_type(C,n,38), bond(A,B,C,1)
```

Finally, HR made this conjecture:

```
bond(A,X1,B,X2), atom(B,X3,38) <-> bond(A,B,X4,2), atom(B,X5,38)
```

Which enables us to re-write our predictor definition thus:

```
active(A) :- atom_type(B,c,21), atom_type(C,n,38), bond(A,B,C,1), bond(A,C,D,2)
```

This states that a molecule is active if it has a carbon atom of type 21 single bonded to a nitrogen atom of type 38, and the nitrogen atom is also double bonded to third atom. Note that this subsumes both the prediction concept learned by HR and the prediction concept learned by Progol.

3 Discussion

An advantage of ILP over ATF is the speed in which concepts are learned: the goal based approach of Progol is much more efficient than the explorative approach employed by HR. A potential advantage of ATF of ILP is the construction of theories containing much more information than is required for just doing predictions. It is generally known that scientists prefer short, easy to understand answers to specific problems from machine learning programs. Hence it is not likely that presenting domain scientists with theories learned by HR will be well received. However, theories can be used more constructively, and it is a challenge to determine how much more information scientists could usefully be given.

We are currently investigating the hypothesis that a combination of HR and Progol could be used to improve the quality of the results for bioinformatics tasks such as predictive toxicology studies. One possible combination of the two techniques is the following: Progol is used to find a predictor with high accuracy, and HR is asked to construct a theory about the concepts in the background information. The results from the two programs are combined by using the conjectures discovered by HR to expand the definition(s) of the concepts in the predictor learned by Progol. We have shown that deducing more information about concepts is possible by hand for the mutagenesis example, and we hope to automate this process. Given that HR can output its implicates as Prolog Horn clauses, then another approach to integration of the two techniques is to use HR to pre-process the background information used by Progol in order to increase the efficiency of Progol's search. Within this, there is also a possibility to use HR's near-conjecture making mechanism to produce hypothesis as stochastic logic programs (SLPs) as the learning of SLPs is a new direction in inductive learning.

References

[1] S. Colton. *Automated Theory Formation in Pure Mathematics*. Springer-Verlag, 2002.

[2] S Colton, A Bundy, and T Walsh. Automatic identification of mathematical concepts. In *Machine Learning: Proceedings of the 17th International Conference*, 2000.

[3] S Colton, A Bundy, and T Walsh. On the notion of interestingness in automated mathematical discovery. *International Journal of Human Computer Studies*, 53(3):351–375, 2000.

[4] W McCune. The OTTER user's guide. Technical Report ANL/90/9, Argonne National Laboratories, 1990.

[5] S Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

[6] A. Srinivasan, S. Muggleton, R. King, and M Sternberg. Theories for mutagenicity: a study of first-order and feature based induction. *Artificial Intelligence*, 85(1,2):277–299, 1996.