Module Inducer: Discovering interactions between transcription factor binding sites using inductive logic programming

Oksana Korol Marcel Turcotte

School of Information Technology and Engineering, University of Ottawa, www.uottawa.ca

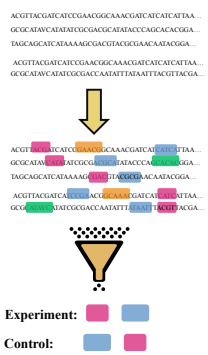
okoro103@uottawa.ca, turcotte@site.uottawa.ca

Objective

Automate scientific theory construction and knowledge discovery in a large data set.

Provide an easy to use tool for discovering higher order relationships between genetic markers in a set of coregulated DNA sequences.

Introduction



The biggest difficulty facing scientists on the forefront of modern biology and medicine is making sense of a huge amount of data. Specifically, noticing patterns in DNA sequences is a very tedious and time consuming task. Recent advancements in the field of bioinformatics have yielded tools to simplify this process. However, the task of discovering complicated, esp. spatial relationships that might constitute a new rule of biological interactions has been mostly done manually.

At the same time, the field of machine learning has been developing new and improving on old methods to analyse and classify data. These methods are now widely used in variety of data mining applications, however researchers in biology are reluctant to adopt most of these methods, since the classification they produce is difficult to verify experimentally for its lack of concise and easy to understand hypothesis.

In this project, we look to develop and easy to use machine learning tool for biologists, that can not only classify, but describe experimental data and discover new knowledge about it using a machine learning approach called inductive logic programming (ILP).

Inductive Logic Programming

Inductive Logic Programming (ILP) was introduced by Stephen Muggleton in 1992.

ILP = Machine Learning Λ Logic Programming = Learning with Logic

Example of ILP classification problem.

Given the following examples and background rules, discover the theory that describes all of the positive and none of the negative examples.

Background rules:

before(X, Y) :- <when position of X is less than position of Y>

adjacent(X, Y) :- <when there is no other object between X and Y>

Resulting theory will be: *positive :- before(circle, triangle), adjacent(triangle, circle).*

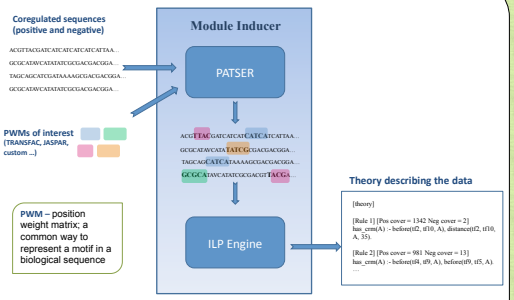
ILP uses first order logic to represent the background knowledge and (positive and negative) examples. It automatically produces as output a theory (set of FOL clauses) that is sound and entails all of the positive and none of the negative examples.

Most currently used machine learning algorithms in bioinformatics (Support Vector Machines, Bayesian Classifiers, Neural Networks) use attribute-based data description, i.e. with no internal structure. ILP allows to represents both the input data and resulting theory using relations, not just the attributes.

Concept learning: given a background knowledge B and experimental observations E (consisting of positive E+ and negative E- examples) find a hypothesis H such that:

$$B \wedge H \models E$$ where *B, E* and *H* are logic programs

Project Description



Module Inducer is a java-based application that utilizes two outside modules:

- PATSER tool: finds PWMs in a sequence; developed by G. Hertz from Stormo Labs
- ILP Engine: written using Aleph; induces theory based on positive and negative examples and background knowledge.

High-level description of the data flow:

User submits a set of coregulated sequences for positive and negative examples and PWMs of interest. The application invokes PATSER tool to locate PWMs in the sequences. The resulting information (which includes sequence names, PWM positions, score of the match, etc.) is submitted to the ILP Engine, which induces and returns the best theory describing the data. The theory is described in terms of the background rules, specified in the engine.

Background rules:

`has_module(Seq, TF)` – describes if a particular sequence has a particular module

`before(TF1, TF2, Seq)` – transcription factor TF1 is located before TF2 in sequence Seq

`distance(TF1, TF2, Seq, Dist)` – calculates the distance between two transcription factors in a sequence

`distance_interval(TF1, TF2, Seq, Dist, Offset)` – same as above, but the distance is described in terms of value +/- offset

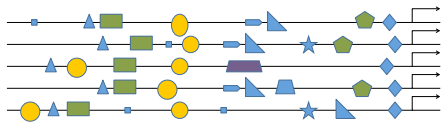
Web interface:

Web interface is written using JSF 2.0. We targeted the use of the interface towards users without a background in computer science or knowledge of ILP. Current demonstration version of the interface severely restricts the amount of data allowed to be submitted for processing. To run Module Inducer on a large dataset, please contact the authors.

http://induce.site.uottawa.ca

Example of a Biological Problem

One example of a problem that Module Inducer can solve is the discovery of *cis*-regulatory modules.



By supplying the sequences, suspected of containing a *cis*-regulatory module and possible transcription factor binding sites (TFBS), Module Inducer will locate the TFBS in the sequences and infer rules describing *cis*-regulatory module in terms of TFBS location in the sequence and relative to each other.

Verification of the Approach

The advantage of ILP over other machine learning methods is that it is not only able to classify the data, but can also induce a theory describing the classified data. Therefore to measure the performance of our method we used a hierarchy of different indicators, which could be divided into 2 classes:

- measures of the results of classification – established ML methods, such as 10-fold cross-validation with stratification and 5-by-2 test with no stratification
- measures of the quality of the theory – comparing the theories of the ILP engine

We have tested our approach on 2 types of data: real *C.elegans* (Zhao et all, 2007) and synthetic. Below are some of the results for synthetic dataset.

	Sensitivity	Sens. Variance	Specificity	Spec. Variance
10-fold	0.8	0.06	0.992	0.000625
5-by-2	0.7592	0.11	0.9835	0.00011

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$Similarity = \frac{Number\ of\ matched\ rules}{Number\ of\ benchmark\ rules}$$

For the quality of the theory, we have used a leave-one-out sampling and achieved the similarity of 0.887 with variance of 0.0213.

Current Application to Human Genome

Currently, Module Inducer is being applied to further analyze a recent study in the difference between expression of cancerous and non-cancerous cells in human tissues[4], which has been jointly performed by University of Ottawa, Ottawa Hospital Research Institute and Fred Hutchinson Cancer Research Center (WA, USA).

As part of the analysis, we look at 2238 positive and 5707 negative sequences, which are extracted from hg18 human genome. The sequences are classified based on the relationship between 4 transcription factor binding sites, found to be present in the sequences. The original study has identified more than 4 transcription factors, however, due to the amount of data and the number of possible combinations of transcription factors and possible relationships between them, the study only looked at the distance between 4 most pronounced matches.

Once we confirm the original findings, we will be able to apply our approach on more than 4 transcription factors as well higher order relationships between them. Our preliminary results already show the ability of our method to discover complex rules like:

```
has_crm(A) :- before('Ebox', 'Gata', A), before('Runx', 'Ebox', A),
distance_interval('Ebox', 'Gata', A, 1, 2).
```

Conclusion

Tests on the real genome data and synthetic dataset have shown that Module Inducer program is able to accept relational data and successfully induce a relational theory. Apart from being able to classify the data, Module Inducer provides a high suggestive value for future research by providing an easy to understand theory, describing the data.

To increase an impact of our work, we plan to implement a number of improvements to the Web-interface of the program, such as allowing of a larger data set to be submitted for processing, providing a natural language translation of the induced theory, exposing more of the system parameters for adjustment by the user, such as noise, Patser cut-off score for accepting a PWM match, several methods of synthetic data generation, etc. (other suggestions are welcome) .

To improve the learning, we plan to use a modified Aleph engine that allows for the use of enrichment analysis with ILP, as well as study the impact of various data selection schemes at the accuracy.

References

[1] Muggleton, S.H. "Inverse Entailment and Progol". In New Generation Computing, 13:245-286, 1995.

[2] Srinivasan A. The Aleph Manual. 2007. http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/

[3] Hertz G.Z., Stormo G.D. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. In Bioinformatics, 15:563-577, 1999.

[4] Palii C.G., Perez-Iratxeta C., Yao Z., Cao Y., Dai F., Davidson J., Atkins H., Allan D., Dilworth1 F.J., Gentleman R., Tapscott S.J., Brand M. Differential genomic targeting of the transcription factor TAL1 in alternate haematopoietic lineages. In EMBO Open. Doi:10.1038, 2010.

[5] Jiline M., Matwin S., Turcotte M. Annotation concept synthesis and enrichment analysis: a logic-based approach to interpretation of high-throughput experiments. In Canadian Conference on Artificial Intelligence 2010, Ottawa, May 31–June 2 2010.

Oksana Korol is funded by an NSERC grant to Marcel Turcotte. The authors would like to thank Marjorie Brand and Jeffrey Dilworth for stimulating discussions. We to thank Marjorie Brand for sharing her data with us.