Pacific Symposium on Biocomputing 5:275-287 (2000)

Visual management of large scale data mining projects

I. Shah
Bioinformatics
American Type Culture Collection
10801 University Boulevard
Manassas, VA 20110
ishah@atcc.org

L. Hunter
Molecular Statistics and Bioinformatics
National Cancer Institute, MS-9105
7550 Wisconsin Ave., Room 3C06
Bethesda, MD 20892-9015
lhunter@nih.gov

This paper describes a unified framework for visualizing the preparations for, and results of, hundreds of machine learning experiments. These experiments were designed to improve the accuracy of enzyme functional predictions from sequence, and in many cases were successful. Our system provides graphical user interfaces for defining and exploring training datasets and various representational alternatives, for inspecting the hypotheses induced by various types of learning algorithms, for visualizing the global results, and for inspecting in detail results for specific training sets (functions) and examples (proteins). The visualization tools serve as a navigational aid through a large amount of sequence data and induced knowledge. They provided significant help in understanding both the significance and the underlying biological explanations of our successes and failures. Using these visualizations it was possible to efficiently identify weaknesses of the modular sequence representations and induction algorithms which suggest better learning strategies. The context in which our data mining visualization toolkit was developed was the problem of accurately predicting enzyme function from protein sequence data. Previous work [9] demonstrated that approximately 6% of enzyme protein sequences are likely to be assigned incorrect functions on the basis of sequence similarity alone. In order to test the hypothesis that more detailed sequence analysis using machine learning techniques and modular domain representations could address many of these failures, we designed a series of more than 250 experiments using information-theoretic decision tree induction and naive Bayesian learning on local sequence domain representations of problematic enzyme function classes. In more than half of these cases, our methods were able to perfectly discriminate among various possible functions of similar sequences [10]. We developed and tested our visualization techniques on this application.

1 Introduction

The application of machine learning techniques (such as neural networks, hidden Markov modeling and information theoretic approaches) to molecular data has been growing in scope and significance. As such projects get more complex,

visualization tools become more important in helping researchers and others manage the data mining process and interpret the results. In this paper, we describe a set of tools for facilitating a data mining project which consisted of a large number of related machine learning problems. Although data mining promises automated inference, much of the work in creating a successful data mining applications involves data set selection, representational choice, and interpretation of the results, which are labor-intensive tasks. Our visualization tools focus on helping people with these aspects of data mining projects.

1.1 Background: Visualization for Large-scale Data Mining

Many of the commercial data mining tools used with most success in molecular biology applications (e.g. SGI's MineSet product) contain extensive toolkits for visualizing data. In many cases, these tools make it possible for human visual pattern recognition abilities to identify important regularities in the data, without any further processing. These tools also make it possible to visualize the results of the application of a particular induction technique to a particular data set, e.g. viewing a complex decision tree. However, it is not easy to use these tools to set up, monitor and globally analyze a large number of related machine learning experiments. We found ourselves confronting just such a situation when attempting to apply machine learning techniques to improve the quality of enzyme function prediction from sequence.

Sequence comparison methods are widely used for predicting protein function; however, a systematic study of these methods shows approximately 6% of sequences are more similar to proteins with different functions [a] than they are to at least some proteins with the same function [9]. These errors are due to a variety of factors, including biologically significant ones such as homology restricted to certain enzymatic subunits and functional differences due to modest sequence variations (e.g. in active sites). In recent work, we were able to improve performance in many cases using a machine learning method for discriminating between functionally different proteins with similar sequences [11]. That work demonstrated the value of automated machine learning methods in a large scale study involving more than 250 functionally defined datasets containing more than 3000 proteins. Designing, managing and understanding the results of such a large-scale effort was greatly aided by the use of a visually-driven user interface for navigating through the many choices that needed to be made and visualizing their results.

[a] We used the Enzyme Commission (EC) classification as a gold standard for protein function. We recognize that this classification is flawed in various ways, but because of its breadth and the dearth of reasonable alternatives, we feel it is appropriate to use it for this study.

1.2 Strategy

The design and development of the user interface and visualization tools were driven by two factors: our top-down understanding of the complete data mining process as including a variety of both manual and automated tasks, and by the particular bottom-up demands of our application problem.

Designing effective machine learning experiments requires a variety of steps other than the automated induction itself [5]. Two of the most important are defining training sets and selecting representations for the data. Theoretical limits on inductive accuracy arise from the interaction of the size of the hypothesis space (a function of representational complexity) and the amount of training data available [4,3]. Therefore, the selection of an appropriate representation depends not only on expert knowledge about the application domain, but also on the amount of training data available for inference. Since the universe of possible representations is difficult to define, and since automating choices among representations increases the effective size of the hypothesis space, human expertise is extremely valuable in this task. However, it can be difficult for people to get a sense of the ramifications of various possible representational choices confronting them. One of the goals in the design of our system was to make representations visually intuitive, so that people could rapidly assess the implications of alternatives, and easily see how a representation interacted with a particular training set.

A second general goal in the design of the system was to make it easy for people analyzing the results of all of these experiments to rapidly draw biologically meaningful conclusions. We designed the visualization system to easily move from global summaries to the detailed results we thought would be most significant for assessing the biological meaning of the results. For example, we wanted to be able to select particularly difficult enzyme functional classes, select from them the particularly difficult examples, and then compare the domain and sequence structures of those proteins.

Finally, we wanted to make design responsive to unexpected analytical and visualization needs that arose during the course of the project. Through the use of object-oriented design, careful definitions of the initial data types and the use of re-usable graphics libraries, we were able to rapidly generate new tools as soon as we discovered the need for them.

2 Methods

In order to explain the details of the visualization techniques, we must first summarize the data mining experiments. A more detailed description of the data mining methods and results can be found in our earlier work [8].

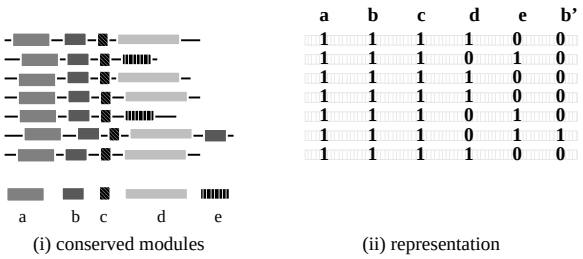

Figure 1: Representing protein sequences on conserved modules. (i) A set of (hypothetical) homologous proteins with conserved modules shown as boxes. There are a total of five modules: a,b,c,d and e. (ii) A representation of the proteins on the basis of the modular attributes. Each protein is shown as a vector of attribute values.

2.1 Defining Training Sets

Our universe of sequences was defined by taking all of the sequences from SwissProt release 33 [1] that were labeled with an Enzyme Commission (EC) classification from Enzyme release 21 [2], resulting in the set of 15,208 SwissProt proteins (out of 52,205 total) which are labeled with one of 1,327 EC classes.

Not all proteins in an EC class must be homologous with each other. Non-homologous subgroups within EC classes may arise due to different evolutionary origins of multiple domain enzymes' subunits, convergent evolution of proteins catalyzing a particular reaction, or by vague or generalized reaction definitions by the EC. Since in this study we were concerned with detecting proteins with similar sequences but divergent functions, we controlled for the presence of non-homologous proteins in EC classes by subdividing the EC classes into putatively homologous subgroups (which we call simgroups) on the basis of sequence similarity. All members of a simgroup have the same (EC) function and similar sequences. Even using simgroups instead of EC classes, sequence similarity is not, in general, enough to establish function. Nearly 40% of simgroups have at least one member that is more similar to an enzyme with a different function than it is to some other member of the simgroup. Setting a lower bound on the size of a simgroup to avoid sampling effects, we found 251 simgroups with at least 10 members that could not be perfectly identified on the basis of sequence similarity alone. These problematic simgroups were each used to create training data for 251 machine learning experiments. The members of each simgroup were defined to be positive examples for a particular experiment, and the union of all proteins in our universe that had significant sequence similarity to one or more members of the group, but were not themselves members were defined to be negative examples.

2.2 Representing Proteins

Homologous proteins with divergent functions may exhibit differences at many levels of description, ranging from point mutations (say, in an active site) to large-scale rearrangements. However, there is an inherent trade-off between the expressive power of a representation (and, therefore, the size of the hypothesis space searched by a learning method using it) and the amount of training data required to induce a particular relationship. Given the modest number of training examples available in most functional classes, we are constrained to select a representation which has modest expressive power. Based on modular theories of protein evolution, and on our observations in the systematic study, we decided to test if the presence and arrangement of conserved subregions of sequence ("domains") could be used to discriminate among functions. Although using a multiple sequence alignment to identify differences at the level of individual amino acids might be desirable, there are far too many differences at this level of description for effective induction.

The ProDom database [13] is one attempt to systematically define protein domains, done on the basis of local sequence alignments within a large set of sequences. We used our visualization tools to explore it and other possible domain definitions (e.g. P-Fam [12]). We judged the larger number of ProDom domains per protein, although possibly more fragmented and less correlated with structure than P-Fam domains, provided an advantage for the machine learning tools, so we selected ProDom for this project.

Since all of our training sets consisted entirely of sequence-similar proteins, we were able to define a simple attribute vector identifying the presence or absence of all domains that are observed anywhere in the training set (see figure 1). We adopted this vector as our representation.

2.3 Induction

Choice of induction method (e.g. neural networks vs. decision tree induction) can make a difference in the outcome of a machine learning experiment. We wanted to evaluate the contribution that selection of induction method might make to our task, but we also wanted to avoid lots of redundant learning experiments on hundreds of datasets. We selected two learning methods, information-theoretic decision tree induction as implemented in C4.5 [7], and our own implementation of naive Bayes, assuming that each element of the representation is statistically independent [6]. Although many other supervised induction methods exist (e.g. artificial neural networks), these two methods represent quite different approaches, so that if there were going to be a significant difference in performance attributable to induction method, it would

likely have appeared among these two methods. Some unpublished preliminary data suggests that neural networks would perform at about the same level as these two methods, and would have taken at least an order of magnitude longer to train.

2.4 Visualization methods

We used perl5 and perl/Tk for rapid prototyping and implementation of our visualization system. These free public tools, used on a Pentium II class machine running the Linux operating system, provided an inexpensive and powerful development environment.

The component objects of the graphical interfaces were based on the underlying data model for the data sets, their modular representations and induced hypotheses. The data model captures the relevant information in the data sets and hypotheses for discriminating between proteins by function. The data sets contain protein instances with similar sequences yet possessing different catalytic functions. We treat protein instances as complex entities which have a number of attributes, including the sequence of IUPAC codes for amino acids and the function or functions, represented as a list of valid EC class numbers. Protein examples also have other attributes like name, comments, references, a biological feature table, a list of homologous proteins and their sequence similarity scores and a list of the ProDom modules which occur in the protein. All proteins are uniquely identified using SwissProt conventions.

More complex structures, such as binary feature vectors, are built by filtering and combining information from these instances. Also included in the data model are representations of probabilistic estimators and decision trees, which are used for visualizing the inductive results.

3 Visualization Tools

The graphical user interfaces (GUIs) represent different views of the underlying data model which allow interactive navigation through the data. The user interfaces conform to a simple and consistent layout of visual components which facilitates ease of learning and use. We describe the visualization tools in the following sections.

3.1 Data sets

The data sets are browsed using the Data Set Viewer. This viewer allows the user to browse a sorted list of available data sets which satisfy a number of criteria. The columns contain the EC number of the data set, the description

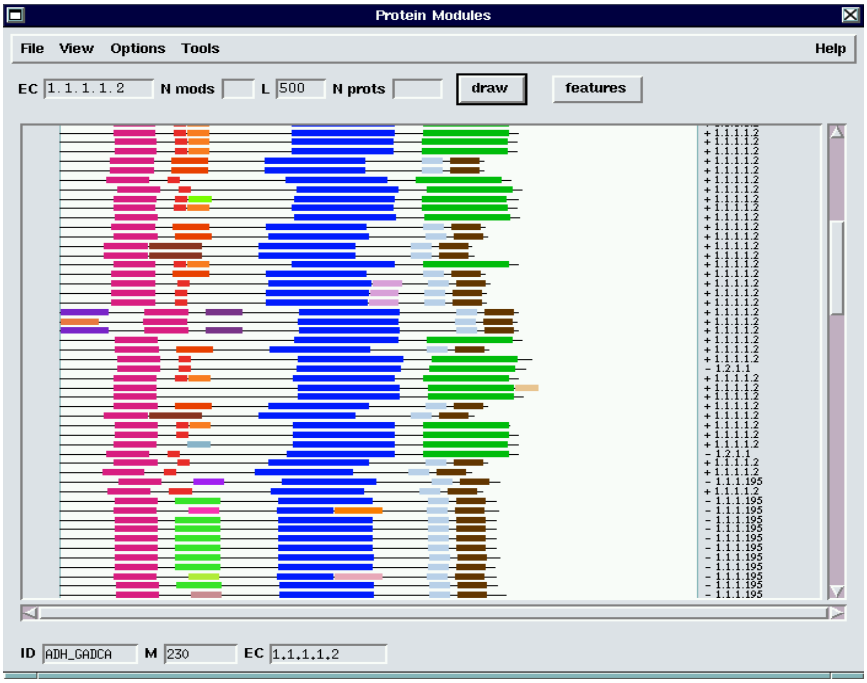

Figure 2: Visualizing protein modules. The figure shows the interface for visualizing modular representations of protein examples in a data set. Protein sequence examples are represented as black lines on which ProDom modules are shown as colored rectangles. The catalytic class of each example is shown on the right hand side: positive characters signify instances with the same function while negative examples are shown with negative marks (-). In this figure the vertical position signifies similarity with instances of the simgroup. See the text for details.

of catalytic activity, the numbers of positive and negative instances, and the predictive accuracy of learning algorithms. Criteria for selecting data sets are specified in the entries in the toolbar. Currently, it is possible to specify the minimum number of positive examples, the induction algorithm, and the output format. The datasets displayed in this view are active. Selecting one takes the user to an interface for visualizing a set of instances, described below.

3.2 Modular representation of proteins

This interface is aimed at emphasizing modular protein sequence representations, protein functional classes and sequence similarity relationships in a data set (see figure 2). In the visualization protein sequence instances are listed

vertically and represented schematically by straight lines whose length is proportional to the number of amino acids. Conserved ProDom modules for each instance are superimposed on the sequence schematic as colored rectangles. Modules with the same ProDom identity are filled with the same color making it easy to visualize conserved regions. All examples are also labeled with EC classes to clearly identify their function; examples with the same function as the selected data set are labeled with a +, and those from a different functional group are labelled –.

It is possible to vertically arrange the sequences in two possible ways. The first arrangement organizes instances by simgroups which emphasizes the conserved sequence modules in each simgroup. In the second arrangement (shown in figure 2) instances are arranged vertically in descending order of similarity with the positive examples. From the second visualization it is possible to see whether sequence similarity alone can accurately distinguish between functionally different homologs. Furthermore, it also possible to see whether modular representations of protein sequences improve predictive accuracy.

The visual representations of a data set support a number of user interactions. First, mouse traversal across any item loads its properties in the statusbar. At present these properties include the SwissProt identifier for the protein sequence, the ProDom identifier for the module under focus, and the EC class label for the protein. The second form of interaction involves the selection of an instance which leads to an interface containing a more detailed description of a protein sequence. The third and final form of interaction advances the user a step along the data mining process to show the feature vector representations of instances.

3.3 Feature Vectors

The feature vector viewer (figure 3 (a)) presents an alternative visualization of the modular descriptions of sequences in a data set. While the modes for organizing sequence instances and interaction are quite similar to the interface for modular arrangements (figure 2), this window presents a visualization for the sequence instances as feature vectors as shown in figure 1. The data set is organized as a table in which the rows correspond to instances (labeled with SwissProt ID and EC class) and the columns stand for individual features. The absence or presence of a ProDom module in a protein sequence is signified with a black or a white square, respectively. The visual regularity in this display facilitates human pattern recognition, making visual analysis of a data set easier. This view is capable of accommodating multiple representations of the sequence data through an "options" menu. We used this capability to quickly and easily compare alternative representational schemes on a variety

of easy and difficult data sets, helping us make an appropriate representational choice.

After mapping instances in data sets to feature vectors the next step is induction of predictive hypotheses. The GUI allows users to retrieve previously computed results, to execute the two available learning techniques on a predefined data set, or to interactively select training and testing instances and then execute either of the learners. The resulting induced hypotheses are presented in interfaces specialized for each learner, described below.

3.4 Naive Bayesian learner

The naive Bayesian learner shows a visualization of the posterior probabilities as computed from the training data (figure 3 (b)). The *a posteriori* probabilities for the feature values and classes, given the training data, are present in a table: columns represent feature values and rows classes. The first row shows the different modules used in the representation as rectangles of distinct colors and the second row shows the allowed values for the modular features. In our simple representation we consider only binary feature values, signifying the absence or presence of modules. These two values are shown as white and black squares below each module. The positive and negative classes are shown in the third and fourth rows respectively. Each cell is a rectangle whose height is proportional to the conditional probability of occurrence of a feature value given for a given class. Marginal probabilities for the occurrence of feature values and classes are given in the last row on the bottom and the last column on the right, respectively. Scrolling below the probabilities one finds a performance summary of the naive Bayesian learner in terms of sensitivity, specificity and accuracy (figure 3(b)).

3.5 Decision tree learner

The decision tree learner interface is quite similar in appearance to the naive Bayesian learner (figure 3(c)). It differs mainly in the representation of the predictive hypothesis which has the form of a tree.

These visualization methods make it possible to rapidly analyze the induction results for biological significance. In our application, we were able to scan the 251 results files for interesting examples spending only a few minutes on each dataset. Several interesting examples are described below, with visualizations.

4 Examples

4.1 Zinc-containing alcohol dehydrogenases

Zinc-containing alcohol dehydrogenases (Zn-ADH, EC 1.1.1.1(2)) are the second largest simgroup in EC 1.1.1.1 and are homologous with a number of

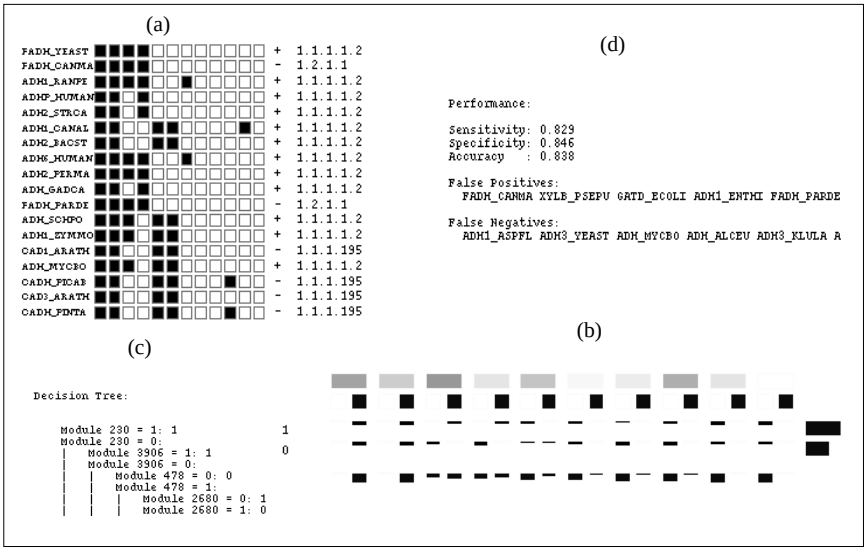

Figure 3: Zinc-containing alcohol dehydrogenases EC 1.1.1.1(2). (a) Feature vectors from modular representation of protein sequence examples. A tabular arrangement is used for representing the data set. Each instance is represented as a row: first column from the left contains the SwissProt identifier of the instance, the second last column contains a positive or negative character signifying function, and the last column shows the EC number. The remaining columns represent the feature vector. Each column stands for a distinct module: the presence of a module is shown by a black square and the absence by a white one. (b) Naive Bayesian (NB) learner visualization shows joint and marginal probabilities computed over a training data set from alcohol dehydrogenase. (c) Visualization of the decision tree induced by using C4.5. (d) Performance summary of NB learner. The learning accuracy of a hypothesis is summarized in terms of the sensitivity, specificity and accuracy. Also given are the instances which are incorrectly classified by the hypothesis.

other enzymes in EC 1.1.1. Visualizing the data set using the modular representation viewer we found a number of homologues with functions which are quite similar: glutathione-dependent fomaldehyde dehydrogenase (EC 1.2.1.1), cinnamyl-alcohol dehydrogenase (EC 1.1.1.195), sorbitol dehydrogenase (EC 1.1.1.14); and the more functionally distant quinone oxidoreductase (EC 1.6.5.5). From the visualization it can be seen that ProDom modules 49 and 43 are conserved in almost all homologues. Scanning through the protein sequence feature visualizations we find that the sequence location of module 49 corresponds to the binding site for the catalytic Zinc ion. The false positive matches from sequence similarity are also found to have the same modular representations which can be seen clearly from the feature vector visualization. In this case

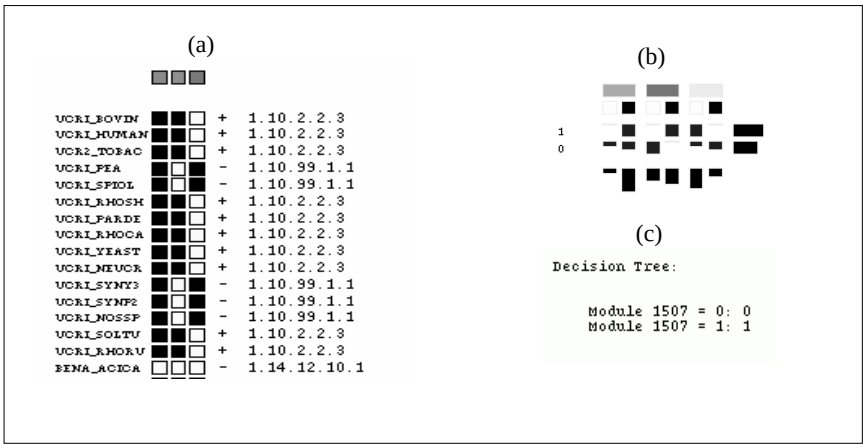

Figure 4: Visualizations for EC 1.10.2.2.3. (a) Feature vectors of modular representations. (b) Naive Bayesian posterior probabilities (c) Decision tree.

modular representations are inadequate for discriminating between functionally distinct homologs. This is confirmed by the naive Bayesian and decision tree learners which have less than ideal performance scores of 0.97 and 0.98, respectively.

4.2 Ubiquinol-cytochrome reductase

Ubiquinol-cytochrome C reductase (EC 1.10.2.2) is an enzyme complex which participates in the mitochondrial electron transport chain. We found seven simgroups in EC 1.10.2.2 which correspond to the different subunits in the complex. Instances of simgroup EC 1.10.2.2(3) consist of Rieske iron-sulfur proteins (RISP) which have homology with subunits of the cytochrome B6-F complex (EC 1.10.99.1). Modular representations of these proteins completely discriminate between all instances of RISP in EC 1.10.2.2 and other homologues using either of the methods (figure 4).

4.3 L-Lactate dehydrogenase

The L-Lactate dehydrogenase (EC 1.1.1.27(1)), malate dehydrogenases (EC 1.1.1.37(1) and EC 1.1.1.82(1)) are homologous and their modular representations exhibit a great deal of similarity.

Through visualization of conserved modules and inspection of protein sequence features we find a common conserved region of around 120 residues

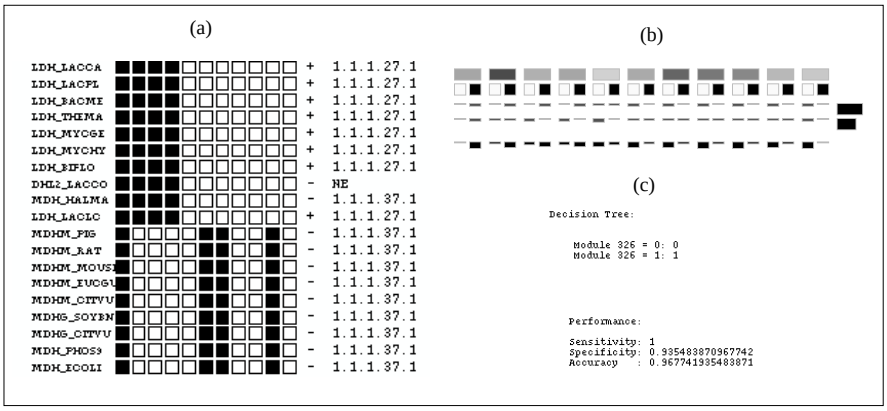

Figure 5: Visualization of the (a) feature vectors (b) decision tree, and (c) naive Bayesian probabilities for EC 1.1.1.27(1)

(ProDom module 139) which contains the proton-relay active site and the carboxyl substrate binding residue. The remainder of the sequence regions do not exhibit a great deal of similarity. There are two false positive matches which cannot be discriminated on the basis of conserved modules. One of these instances has only been assigned a partial EC class of EC 1.1.1 and may be a true positive and the other is an archaebacterial malate dehydrogenase. Since we were not sure about the functional identity of the instance with a partial EC class we recomputed the learning accuracy after interactively removing it from the data set and measured a small increase in performance.

5 Conclusions

This project demonstrated that visualization is useful not only as a way for people to use their pattern recognition abilities on raw data, but also to help guide a large-scale data mining experiment and interpret its results. We were able to rapidly evaluate different representational options, try various approaches to defining our datasets, and to make biological interpretations of the inductive results.

The tools are object-oriented which makes them extensible to allow additional functionality in the form of representation spaces and learning algorithms; the visualizations of the feature vectors and learners are generic and can be easily reused in different machine learning problems. We are also implementing modules that will allow users to import their own data sets from structured text files and database management systems.

We informally estimate that the use of visualization tools saved us hundreds of hours in the design, execution and analysis of these experiments. Given the ease of applying generic visualization tools and of integrating custom tools into generic frameworks, we believe visualization methods such as ours can be used to create and interpret ever larger and more ambitious data mining projects.

1. A. Bairoch. The ENZYME data bank. *Nucleic Acids Res.*, 22:3626–3627, 1994.
2. A. Bairoch and B. Boeckmann. The SWISS-PROT protein sequence data bank. *Nucleic Acids Research*, 20:2019–2022, 1992.
3. A. Blummer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis Dimension. *Journal of the ACM*, 36(4):929–965, 1989.
4. D. Haussler. Quantifying Inductive Bias: AI Learning Algorithms and Valiant's Learning Framework. *Artificial Intelligence*, 36:177–221, 1988.
5. L. Hunter. Planning to learn about protein structure. In L. Hunter, editor, *Artificial Intelligence and Molecular Biology*, pages 259–288. AAAI Press, Menlo Park, California, 1993.
6. T. Mitchell. *Machine Learning.* McGraw Hill, 1997.
7. J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kauffman, 1993.
8. I. Shah. *Predicting Enzyme Function from Sequence.* PhD thesis, George Mason University, 1999.
9. I. Shah and L. Hunter. Predicting enzyme function from sequence: A systematic appraisal. *ISMB*, 5:276–283, 1997.
10. I. Shah and L. Hunter. Identification of divergent functions in homologous proteins by induction over conserved modules. *ISMB*, 6:157–164, 1998.
11. I. Shah and L. Hunter. Visualization based on the enzyme commission nomenclature. *PSB*, 3:142–152, 1998.
12. E. Sonnhammer and R. Durban. Pfam: a comprehensive database of protein domain families based on seed alignments. *Proteins*, 28(3):405–420, 1997.
13. E. Sonnhammer and D. Kahn. The Modular Arrangement of Proteins as Inferred from Analysis of Homology. *Protein Science*, 3:482–489, 1994.