Using Background Knowledge to Improve Inductive Learning

A Case Study in Molecular Biology

Haym Hirsh and Michiel Noordewier, Rutgers University

THE GOAL OF INDUCTIVE LEARNING is to take a collection of labeled "training" data and form a classifier that accurately predicts the labels of future data. If learning is to be successful, the training data must be encoded in a form that lets the learner recognize underlying regularities. This problem is central to all inductive learning efforts, but most work in inductive learning has assumed that the data are provided in a form that is already appropriate for learning.

In the notable exceptions when such an assumption has not been not made, the most common approach has been to carefully engineer a set of features for the particular learning problem. For example, Quinlan spent two person-months defining an appropriate set of features for learning the chess end-game of lost 3-ply with a decision-tree learner.[1] Similarly, Kamm and Singhal explored different methods for sampling a speech signal to find the one that yielded the best encoding of data for a neural-network learner.[2]

One important application of inductive learning has been to learn classifiers for recognizing the functional roles of different regions of DNA.[3–5] (See the first sidebar for a brief introduction to DNA.) Although the genetic information encoded in DNA is given biological meaning by conversion to proteins, not all portions of DNA serve as protein coding regions. Some portions instead serve as regulatory "markers" for the processes that convert coding regions into protein. For example, associated with each coding region in bacterial cells are additional sequences that control transcription. Two of these *regulatory regions* are of particular importance: *Promoters* are sequences that occur before coding regions to signal where the transcription process begins, and *terminators* occur after the coding region to signal where it ends. The difficulty is that the current technology for identifying the functional roles of different regions of DNA is a painstaking laboratory-based process. Our goal is to develop methods for forming classifiers that can identify and distinguish the various functional regions of DNA without the need to resort to laboratory methods.

THIS WORK USES BACKGROUND KNOWLEDGE TO REEXPRESS TRAINING DATA IN A FORM MORE APPROPRIATE FOR INDUCTIVE LEARNING. THE APPROACH DRAMATICALLY IMPROVES THE RESULTS OF DECISION-TREE AND NEURAL-NETWORK LEARNING METHODS.

Our work has focused on the problem of forming classifiers for two very different DNA sequence classes. The first, prokaryotic (bacterial) transcriptional promoters (specifically, for the bacterium *Escherichia coli*) are regions of DNA that label the start of genes. These regions are particularly attractive in that they constitute the most extensively studied class of DNA regulatory sequences. The second sequence class, eukaryotic (nonbacterial) splice junctions, labels those parts of a DNA sequence that, although converted into RNA, are excised before the translation process begins. This second class provides a good test for the generality of our efforts in that it differs tremendously from promoters, both in the nature of the biological role of the region, as well as in the nature of the class of cells in which they occur.

0885-9000/94/$4.00 © 1994 IEEE

An earlier, expanded version of this article appeared as "Using Background Knowledge to Improve Inductive Learning of DNA Sequences" on pp. 351–357 of the *Proceedings of the 10th Conference on Artificial Intelligence for Applications* (CAIA '94), published in 1994 by the IEEE Computer Society Press, Los Alamitos, Calif.

Generalizing from known DNA sequences

Our approach to this problem exploits the fact that past and ongoing laboratory efforts have generated a library of categorized DNA sequences. The basic idea is to generalize from such labeled sequences to form classifiers that can accurately label regions in newly sequenced, uncharacterized DNA. Thus, for example, our approach would take a collection of sequences known to be promoters, and sequences known not to be promoters, and extrapolate from them to form a classifier that takes unlabeled sequences and makes accurate predictions as to whether or not they are promoters. (See the sidebar for a discussion of related work.) Our work uses two of the best-known and most widely used inductive learning methods: a decision-tree learning algorithm called *C4.5rules*,[6] and a neural-network algorithm called *backpropagation*.[7]

C4.5rules uses C4.5 — a noise-tolerant successor to Quinlan's ID3 program — to form a decision tree from a collection of training data. However, C4.5rules further processes the results of C4.5 by converting the decision tree into an equivalent collection of rules. These rules are in turn simplified and returned as the final result of learning. The construction of simple, readable rules is particularly important in this domain, as our intention is to form classifiers that can be passed on to biologists in an intelligible form. Our reported error rates represent the results of 10-fold cross-validation.[8]

Backpropagation, on the other hand, is a neural-network learning algorithm that, for a given network topology, incrementally modifies the weights in an attempt to minimize the network's error on the training data. Our experiments use a single hidden layer, fully connected between input and hidden layers and hidden and output layers, with initial weights set to random values between ±0.01 and weights updated once per epoch. Backpropagation requires specifying an appropriate number of hidden units, as well as values for two parameters: the *learning rate* and *momentum*. We selected the number of hidden units by training networks with different numbers of hidden units for a fixed number of epochs on 25% of the data, and then selecting the number of units corresponding to the network that had the lowest error. On every learning run we set the learning rate and momentum using an automated, hill-climbing search process.[9] Our reported error rates are for 10-fold cross-validation on the remaining 75% of the data, with each run using the number of hidden units and parameters determined on the 25% hold-out set. Finally, backpropagation requires real-valued features, so when learning on raw DNA sequences, each nucleotide position is converted into four input units, one of which is set to 1 to indicate the base present in that example, and the others set to 0. Although backpropagation does not satisfy our desire for interpretable rules, we use it to judge whether our results are specific to C4.5rules.

Reformulating training data

The representation of training data is particularly important for inductive learning when applied to this problem. Although from a biochemical viewpoint DNA is a complex molecule, from a computer-science viewpoint DNA can be considered a very long string over the four-letter alphabet A, C, G, and T (representing the four possible nucleotides in each position of a strand of DNA). All past learning efforts have tried to learn directly from these raw sequence data. However, this representation only expresses low-level information about nucleic acid sequences and does not encode any important features that biologists find useful in describing regions of DNA.

We believed that inductive learning would be a more successful solution to the sequence-identification problem if the training data were expressed using the higher level features that molecular biologists use when discussing nucleic-acid sequences. We therefore surveyed the literature in this domain and encoded various higher level features exhibited in relevant publications. In doing so we were strongly motivated by the goal of generating features applicable to a range of sequence-identification problems, so we tried to include only those features that were applicable to DNA sequences in general, rather than features specific to promoters or splice junctions. We therefore constrained ourselves to features that reflect basic physical and chemical properties of DNA sequences, as opposed to knowledge based on the specific genetic processes acting on a sequence.

Our higher level features fall into two general classes:[10] site-specific and conformational. *Site-specific* features encode simple patterns that are relevant for DNA. Each such pattern results in a single new Boolean feature representing the presence or absence of the feature in the given DNA sequence. For example, the presence of G-T-G/C-A-C pairs is important in DNA/protein interactions, so each reexpressed example has a Boolean feature whose value represents whether the substring G-T-G or C-A-C is present in the given example.

Introduction to DNA

Genes encode heritable traits that define the characteristics of individual organisms, and are the mechanism by which such information is transmitted during cell reproduction. This genetic information is encoded in a cell in the form of DNA (deoxyribonucleic acid). The DNA double helix is composed of two strands, each of which is a large molecule or *polymer* formed from combinations of four smaller units: adenosine, thymidine, cytidine, and guanosine. These units are called *nucleotides*, and are usually abbreviated as A, T, C, and G.

The two strands of the double helix are "complementary," in that when a T resides on one strand, an A occupies the corresponding position on the other strand; when there is a G on one strand, a C occupies the corresponding position on the other.

While DNA encodes an organism's genetic information, it is the conversion of the information into "effector" molecules — usually proteins — that gives this information biological meaning. In brief, particular stretches of the DNA (known as *coding regions*) are copied into an intermediate molecule called RNA (ribonucleic acid, also composed of nucleotides) by a process known as transcription. RNA is then translated into a protein (which is itself a polymer, this time assembled from combinations of one of 20 different amino acids). In the process of transcription and translation, each consecutive triplet of DNA nucleotides is mapped into one amino-acid unit in the protein chain. This many-to-one mapping from DNA triplets to amino acids has become known as the "genetic code."

Conformational features encode a sequence's physical and chemical properties. For example, the number of A's and T's in a sequence affects the temperature at which the DNA helix unwinds, and we thus have an "AT-content" feature that records the percentage of A's and T's in a sequence.

All our higher level features are Boolean or real-valued. Boolean features have value Y if the feature appears in the region of interest, and value N otherwise. The remaining non-Boolean features compute properties of the full training sequence, taking on values between 0 and 1. For instance, if we encoded the sequence A-G-G-A-C-G-T... as an example in the typical fashion, it would be encoded as the list [A, G, G, A, C, G, T, ...]. To reexpress this example, the values for our higher level features would be computed, yielding the list [N, 0.25, ...], where the first element represents the absence of the string G-T-G, the second represents 25% AT-content, and so on. This process is applied to every example, and the resulting reexpressed data are then given to the inductive learner.

For all our experiments, the raw training data consisted of strings of 60 consecutive nucleotides. Our catalog of higher level features contains 19 new features, and thus reexpressed data are represented with 19 features: 14 Boolean and 5 real-valued.

Results

To learn promoters, the inductive learner was given 300 sample promoter sequences and 300 nonpromoter sequences. We obtained the 300 promoters from a compilation produced by Lisser and Margalit.[11] To generate nonpromoters, most studies have generated random sequences. However, since DNA is highly nonrandom, we instead generated negative training data by selecting contiguous substrings from a 1.5 kilobase DNA sequence (provided by T. Record of the University of Wisconsin's Chemistry Department) that does not bind RNA polymerase, and is thus believed not to contain any promoter sites.

Our study differs from past work in that our data are aligned on the biological start of transcription, representing information obtained from laboratory experiments, rather than on human-labeled functional regions, which do not represent true biological knowledge.[12] In particular, past efforts — both consensus methods and inductive approaches — have used data sets in which each example sequence has been edited using a scheme that tries to line up related portions of each training sequence. Our study instead works from raw data sequences; indeed, the elimination of the need for alignments not based on biological principles is a strong feature of our approach. (Unfortunately, this feature also makes it inappropriate to compare our results with two of the better-known approaches to forming promoter recognition rules — consensus methods[11] and Kbann[5] — in that both define features as positions in the sequences after alignment preprocessing has been done.) Finally, our study differs from others in the use of a more recent promoter data set that has had a number of minor errors removed. (Thus our error rates are incomparable to those previously reported in the literature.)

To learn splice junctions, the inductive learner was given 1372 positive and 1372 negative sequences obtained from the Genbank nucleic acid data collection, version 74.0. The

Table 1. Cross-validated error rates for C4.5rules.

Sequence Class	Low-level	High-level	Both
Promoters	20.4%	8.7%	10.6%
Splice Junctions	13.2%	4.2%	5.1%

Table 2. Cross-validated error rates for backpropagation.

Sequence Class	Low-level	High-level	Both
Promoters	25.3%	10.2%	11.3%
Splice Junctions	12.0%	4.8%	7.3%

Other efforts for forming DNA classifiers

Efforts at forming DNA-sequence classifiers from a library of labeled sequences have fallen into two classes: the derivation of consensus sequences, and inductive learning. The first approach takes a collection of examples of a given sequence class and generates a "consensus" sequence that represents a crude sort of "average" of the given sequences.[1] Various methods have been explored for defining such consensus sequences and using them to identify sequences in uncharacterized DNA.[2,3] Such approaches have been used, for example, on the task of promoter recognition.[1] However, the utility of consensus methods is severely limited due to the assumptions underlying the approach, such as treating consecutive nucleotide locations as independent and using data manually aligned to match up well with the consensus sequence.[4]

The second approach uses inductive learning. To generate DNA-sequence classifiers this approach would take as "positive data" a set of sequences known to be members of the class of interest, and as "negative data" examples of sequences known not to be members of this class, and then would form a rule based on these sample sequences that makes accurate predictions as to whether new sequences are members of the class (whether they are "positive" or "negative"). Neural networks are one example of an inductive-learning method, and are the main method reported in the literature for forming such DNA-sequence classifiers.[5–7]

References

1. D. Galas, M. Eggert, and M. Waterman, "Rigorous Pattern-Recognition Methods for DNA Sequences: Analysis of Promoter Sequences from *Escherichia coli*," *J. Molecular Biology*, Vol. 186, No. 1, Nov. 5, 1985, pp. 117–128.
2. M.E. Mulligan and W.R. McClure, "Analysis of the Occurrence of Promoter Sites in DNA," *Nucleic Acids Research*, Vol. 14, No. 1, Jan. 10, 1986, pp. 109–126.
3. R. Staden, "Computer Methods to Locate Signals in Nucleic Acid Sequences," *Nucleic Acids Research*, Vol. 12, No. 1, Pt. 2, Jan. 11, 1984, pp. 505–519.
4. S.W. Norton, "Learning to Recognize Promoter Sequences in *E. coli* by Modeling Uncertainty in the Training Data," *Proc. 12th Nat'l Conf. Artificial Intelligence*, AAAI Press, Menlo Park, Calif., 1994, pp. 657–663.
5. A. Lukashin et al., "Neural Network Models of Promoter Recognition," *J. Biomolecular Structure and Dynamics*, Vol. 6, No. 6, June 1989, pp. 1123–1133.
6. A. Lapedes et al., "Application of Neural Networks and Other Machine Learning Algorithms to DNA Sequence Analysis," *Computers and DNA*, SFI Studies in the Sciences of Complexity, Vol. VII, G. Bell and T. Marr, eds., Addison-Wesley, Reading, Mass., 1989, pp. 157–182.
7. J. Shavlik, G. Towell, and M. Noordewier, "Using Artificial Neural Networks to Refine Existing Biological Knowledge," *Int'l J. Human Genome Research*, Vol. 1, 1992, pp. 81–107.

sequences used are those described as genomic DNA comprising complete genes from the primate and mammal collections. We excluded sequences that do not have a canonical G-T at the splice donor (intron beginning), and an A-G at the acceptor (intron end).

Since our work is partially motivated by the goal of generating interpretable rules for DNA sequences, our initial experiments were done using C4.5rules, which generates such interpretable rules. Table 1 gives the cross-validated error-rate estimates for the results of C4.5rules on both the promoter and splice-junction prediction tasks (we will explain the final column shortly). The use of higher level features caused a dramatic, statistically significant ($p < .01$) decrease in error rates, from 20.4% to 8.7% in the case of promoters, and from 13.2% to 4.2% in the case of splice junctions.

To test whether these improvements were the result of our higher level features, rather than something specific to C4.5rules, we conducted the same learning experiments using backpropagation, yielding the results in Table 2. The improvements were again dramatic, from 25.3% to 10.2% for promoters, and from 12.8% to 4.8% for splice junctions. As with Table 1, these differences are statistically significant, as are the differences between corresponding values in the two tables.

Our next experiments explored whether representing data with the combined sets of features — the higher level features *plus* the raw nucleotide sequence — would improve or harm learning. One intuition is that the higher level features are defined solely from information present in the raw data, so that adding the original raw features would harm or at best leave unchanged the results. On the other hand, a competing intuition is that if some of the sequence-identification information resides at the level of raw nucleotides, then this information would be missed when using only higher level features, and the combination of the two sets of features should improve learning.

To explore this question we conducted the same set of learning runs as was performed above, with data expressed using both sets of features. The resulting error rates are in the last column of Tables 1 and 2. In all cases the combined set of features gave error rates between those of pure raw data and pure higher level data (with statistically significant differences between corresponding values). It appears that adding the raw features hurts learning more than it helps.

OUR RESULTS SUPPORT FOUR claims. The first is simply about the importance of training-data representation in learning: Using our collection of higher level features led to improved error rates. The second claim is that our results are not geared to any particular learning method: Learning was improved across two very different "off-the-shelf" learning methods. The third claim is that our features, while domain-dependent, were not learning-task specific, that is, not geared to the given learning problem: Learning was improved both for prokaryotic (bacterial) promoters as well as for the very different problem of eukaryotic (nonbacterial) splice junctions. Finally, we believe our results support the claim that future work on inductive learning of DNA sequences should use our set of higher level features, rather than learning directly from nucleotide sequences — certainly for promoters and splice junctions, but for other sequences as well.

Future work will investigate the utility of our features for other sequence identification tasks. We also plan to explore the extent to which the results of C4.5rules are comprehensible to molecular biologists, in that declarative specifications are known for few DNA sequence classes. Therefore, an interpretable learned classifier could also serve as a definition of the sequence class in addition to a mere recognizer. These potential sequence-identification definitions could then be tested by laboratory synthesis of DNA sequences that satisfy the definitions.

We also intend to continue our exploration of the importance of background knowledge and training-data representations in molecular biology.[13] This includes the development of training-data representations and associated learning methods better suited to sequence-identification problems. Such methods would be able to exploit the wider range of background knowledge that is available in this domain but that is difficult to integrate into training-data representations.

Acknowledgments

We thank Steve Norton and Lorien Pratt for helpful discussions and comments. This work was partially supported by NSF grant IRI-9209795 and DOE grant DE-FG02-91ER61129.

References

1. J.R. Quinlan, "Learning Efficient Classification Procedures and Their Application to Chess End-Games," in *Machine Learning: An Artificial Intelligence Approach*, R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, eds., Morgan Kaufmann, San Francisco, Calif., 1983, pp. 463–482.
2. C.A. Kamm and S. Singhal, "Effect of Neural Network Input Span on Phoneme Classification," *Proc. Int'l Joint Conf. Neural Networks*, 1990, IEEE Press, Piscataway, N.J., pp. 195–200.
3. A. Lukashin et al., "Neural Network Models of Promoter Recognition," *J. Biomolecular Structure and Dynamics*, Vol. 6, No. 6, June 1989, pp. 1123–1133.
4. A. Lapedes et al., "Application of Neural Networks and Other Machine Learning Algorithms to DNA Sequence Analysis," *Computers and DNA*, SFI Studies in the Sciences of Complexity, Vol. VII, G. Bell and T. Marr, eds., Addison-Wesley, Reading, Mass., 1989, pp. 157–182.
5. J. Shavlik, G. Towell, and M. Noordewier, "Using Artificial Neural Networks to Refine Existing Biological Knowledge," *Int'l J. Human Genome Research*, Vol. 1, No. 1, 1992, pp. 81–107.
6. J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Francisco, Calif., 1993.
7. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing*, D.E. Rumelhart and J.L. McClelland, eds., MIT Press, Cambridge, Mass., 1986, pp. 318–364.
8. S.M. Weiss and C.A. Kulikowski, *Computer Systems That Learn*, Morgan Kaufmann, San Francisco, Calif., 1991.
9. L.Y. Pratt, *Transferring Previously Learned Backpropagation Neural Networks to New Learning Tasks*, doctoral dissertation, Rutgers University, New Brunswick, N.J., 1993.
10. H. Hirsh and M. Noordewier, "Using Background Knowledge to Improve Learning of DNA Sequences," *Proc. Tenth IEEE Conf. Artificial Intelligence for Applications*, IEEE Computer Society Press, Los Alamitos, Calif., 1994, pp. 351–357.
11. S. Lisser and M. Margalit, "Compilation of *E. coli* mRNA Promoter Sequences," *Nucleic Acids Research*, Vol. 21, No. 7, Apr. 1993, pp. 1507–1516.
12. S.W. Norton, "Learning to Recognize Promoter Sequences in *E. coli* by Modeling Uncertainty in the Training Data," *Proc. 12th Nat'l Conf. Artificial Intelligence*, AAAI Press, Menlo Park, Calif., 1994, pp. 657–663.
13. H. Hirsh and N. Japkowicz, "Bootstrapping Training-Data Representations for Inductive Learning: A Case Study in Molecular Biology," *Proc. 12th Nat'l Conf. Artificial Intelligence*, AAAI Press, Menlo Park, Calif., 1994, pp. 639-644.

Haym Hirsh is an assistant professor of Computer Science at Rutgers University. His research interests include applications of machine learning in molecular biology, scientific visualization, and knowledge representation. He received his BS in 1983 from UCLA, and his MS and PhD from Stanford University in 1985 and 1989, respectively.

Michiel Noordewier is an assistant professor of Computer Science at Rutgers University. His research interests include inductive learning in molecular biology, and the role of DNA conformation in genetic regulation. He received his BS and PhD from the University of Wisconsin in 1984 and 1990, respectively.

The authors can be reached at the Department of Computer Science, Rutgers University, Piscataway, NJ 08855; Internet: hirsh@cs.rutgers.edu or noordewi@cs.rutgers.edu.