Modeling DNA Splice Regions by Learning Bayesian Networks

Denver Dash
Decision Systems Laboratory
Intelligent Systems Program

Vanathi Gopalakrishnan
Center for Biomedical Informatics
Department of Medicine
University of Pittsburgh
Pittsburgh, PA 15260
ddash@sis.pitt.edu, vanathi@cbmi.upmc.edu

February 5, 2001

Abstract

This paper describes some new attempts at forming consensus sequences that signal the start of translation and the intron donor and acceptor sites in eukaryotic mRNA. We use probabilistic models that compactly encode complex long-range dependencies in the form of Bayesian networks allowing for more general interactions between nucleotide positions than next-neighbor and tree-like models. We use a greedy multi-layer algorithm to learn Bayesian network from data along with several search heuristics to improve the convergence and to minimize the complexity of the networks. We show that this method for classifying splice sites performs as good or better than previous attempts at classifying start and splice sites. We argue that another key benefit to our technique is that these models can provide qualitative insight about the interaction of nucleotides for these identification problems; a fact which is perhaps even more interesting than improvements in classification accuracy.

1 Introduction

Recent advances in biogenetic technology have produced a plethora of unclassified biological data. The tremendous size of biological sequence databases, as well as the complexity of patterns represented within, present enormous challenges to biology researchers and emphasize the need to develop automated methods for classification and data-mining. We present a system that can aid in the automated classification of eukaryotic DNA splice junctions. Splice junctions are points on a DNA sequence at which 'superfluous DNA' is removed during the process of protein synthesis in higher organisms [5]. Given a sequence of DNA, the problem then is to recognize the boundaries between the exons (parts of the DNA sequence retained after splicing) and the introns (parts of the DNA sequence that are spliced out). These boundaries are called the intron donor (5' region) and acceptor (3' border) sites.

In addition to identifying splice junctions, we seek to learn models that are representative of a consensus sequence for gene start codons that signal the start of translation of nucleotide triplet bases to amino acids. Our system uses recent technology developed in the Artificial Intelligence community for learning Bayesian (probabilistic) networks from data. Bayesian networks allow complex probabilistic relationships to be represented in a relatively compact form and provide efficient methods for answering probabilistic queries, using graph-theoretic methods [9].

Several computational systems for identifying splice junctions have been developed by other researchers, that use artificial intelligence techniques such as neural networks and rule learning to learn models for classifying unseen examples based on combinations of features present in pre-classified sets of training examples. Brunak et al.[1] used a neural network that considered coding measures such as binding energies in addition to positional frequencies of nucleotides. Solovyev et al. [14] used discriminant analysis to identify splice sites by considering features such as triplet counts and octamer counts. More recently, Rampone [11] combined the linear discriminant analysis method with a machine learning algorithm called BRAIN (Batch Relevance-Based Artificial Intelligence) in order to learn disjunctive rules for binary classification of splice sites. This hybrid method showed low error rates and high correlation coefficient measures for donor and acceptor sites in human DNA.

There have been many previous attempts to characterize consensus sequences near the start sites and splice sites in eukaryotes. Kozak [6, 7] has studied extensively the properties of start site sequences and established the well-accepted scanning model of ribosome progression. Other researchers [13, 8] have studied sequence patterns around intron acceptor and donor sites. All of these studies have primarily classified sequences according to the marginal probabilities of nucleotides occuring in positions around the sites. Salzberg [12] analyzed both start sites and splice junctions by assuming that the probability of a nucleotide appearing in a position p depended on the nucleotide in the position immediately upstream from p. He showed that using these first-order conditional probability matrices rather than marginal probabilities can significantly improve the sensitivity and specificity of the classification process. Cai, et al., [3], using a standard learning algorithm for learning Bayesian networks in the form of trees, took this idea even further to allow restricted higher order dependencies to be modeled.

In this paper, we continue this trend by allowing models with more complicated dependencies than tree-like structures. We use algorithms for learning network structure and apply several heuristics to minimize the complexity of the learned models. The models we construct have two advantages over first-order Markov models and tree-models. First, we show that our models classify about as good as, or in the case of start codons, considerably better than, other classifiers. Second, by visually examining the structure of the learned model we can gain qualitative insight into which nucleotides and combinations of nucleotides are most relevant to the identification process. This type of qualitative insight is not available in the Markov model because the structure of the model is determined *a priori*. Similarly, the restrictiveness of the assumptions that underly the tree-based model structure make it difficult to gain insight about which interactions are likely to be more relevant.

In Section 2 we present the details of our methods, in Section 3 we present the final models obtained and compare our classification power to existing techniques, finally in Section 4 we discuss our results.

2 Materials and Methods

2.1 Data Sources

The data for this study was originally collected by Burset and Guigo [2], and subsequently used by Salzberg for splice-site classification [12][1]. A description of the data is reproduced here verbatim from the Salzberg paper:

> The data for this study were originally collected by Burset and Guigo [2], who used it to study gene-finding systems. They collected a large set of genes, and carefully edited the set so as to remove sequences that were likely to contain erroneous annotation or to represent non-standard splicing mechanisms (e.g., alternative splicing or AT-AC splicing). Database entries were discarded if the protein coding region was ambiguous, if the sequences included pseudogenes, if alternatively spliced forms of the gene were listed, or if the gene contained no introns. This produced 1410 sequences, which were then further refined by discarding sequences whose protein coding region did not start with ATG, whose length was not a multiple of three, whose introns did not start with the dinucleotide GT and end with AG, or that had protein coding regions with an in-frame stop

[1] A link to the data was found in this paper, and the data was therefore obtained via ftp from ftp.cs.jhu.edu, directory pub/salzberg/sitedata.

codon. (By using only "standard" GT-AG introns, the statistical methods for characterizing the splice sites should produce a clearer picture of the consensus pattern.) Sequences corresponding to immunoglobulins and histocompatibility antigens were discarded, and finally all sequences entered prior to January 1993 were discarded, leaving only relatively recent entries.

Our study used this data virtually unchanged from the data used in the above study, aside from the preprocessing detailed below. For identifying a target site t (start site or splice junction), the data was parsed into records containing sequences of 50 nucleotides each, labeled

$$\{X_{-25}, X_{-24}, \ldots, X_{-1}, X_1, X_2 \ldots, X_{25}\},$$

where X_{-25} is the most upstream nucleotide and X_{25} is the most downstream nucleotide. The positions of the target site itself (i.e., the nucleotide sequence ATG, AG, or GT) were not included in the data. For the purposes of identifying start codons, 562 positive examples were initially available, 554 of which had sufficiently large window sizes. For negative examples of start sites, we scanned all available sequences for occurrences of the target sequence ATG such that the target found was not one of the start codons in the positive data. All such targets found with appropriate window sizes were included in the database of negative instances, which totaled 45,367 records. All records that contained the wildcard "N" symbol were rejected and not included in this total. Similar processing was applied to donor and acceptor site data. In these cases, 2079 positive records were initially available, 2076 and 2074 of which were used as postive examples of donor and acceptor sites respectively. Negative examples totaling 147,686 donor and 211,277 acceptor site records were generated [2].

2.2 Classification

2.2.1 General Method

The method that we present is designed to balance two competing factors that are encountered when building a classifying model. Namely we are trying to *allow explicit independencies* to be present in our model, without assuming those independencies *a priori*, and thus being able to *model complex interactions* without necessarily creating an overly complex model. To clarify this approach, consider the classification problem in general. When uncertainty in the domain is present, this problem can be reduced to problems involving queries to a joint probability distribution. For example, for determining if a nucleotide sequence $s = (X_1 X_2 X_3 \ldots X_n)$ belongs to the class C of splice sites, we can ask for the probability $P(s \in C \mid s, D)$ that $s \in C$ given the series of nucleotides in s and given our training data D. If P is then above some threshold value that we deem acceptable, we can classify s as being

[2] These data sets can be downloaded via http://www.isp.pitt.edu/ ddash/data/splicejunctions/.

Figure 1: A Bayesian network for recognizing start sites that represents the case that a nucleotide position X_i only depends on the nucleotide that is present in the X_{i-1} position. The A, T, and G positions do not depend on any other positions because their values are always fixed.

in C. This method of classification requires a means of posing queries to a joint probability distribution. A serious drawback is that calculation of the joint distribution for any practical number of variables is completely infeasible. For example, if V is a set of 30 variables, each with 4 possible outcomes, then specification of the full joint distribution $P(V)$ requires $4^{30} - 1 \approx 10^{18}$ independent parameters.

This approach can be made feasible by explicitly modeling independencies thereby decreasing the number of free parameters needed to specify $P(V)$. For example, in the context of splice site recognition, the simplest approximation is to assume that each nucleotide position is independent of all other positions. This model specifies a consensus matrix having only $(4 - 1) \times 30 = 90$ parameters. A better approximation [12] is to assume that the probability of a nucleotide x_i appearing in position X_i depends only on the presence of the nucleotide in position X_{i-1}. Such an assumption can be encoded in the Bayesian network shown in Figure 1, which possesses < 500 parameters. The introduction of independencies into a model can thus dramatically decrease the number of parameters required to specify $P(V)$, and a Bayesian network provides a means to compactly represent, and perform calculations with, complex independence assumptions. Given a Bayesian network, N, the probability $P(S \mid N)$ of any given sequence S occurring can be calculated in time linear in the number of nodes in the network.[3]

In our method, we learned two networks for each classification problem: N_C which was trained on the positive instances and N which was trained on both the positive and negative instances in the data. Let C be the class that we are considering, and let t be the boolean random variable that is *true* if $s \in C$ and *false* otherwise. Given N, N_C, and some nucleotide sequence s, it is possible to calculate both the probability of s occurring given that it is in C, $P(s \mid t, N_C)$, and the marginal probability of s occurring whether or not t =*true*, $P(s \mid N)$. However, in order to perform a probabilistic classification, we need to calculate the probability $P(t \mid s, N, N_C)$ that t is *true* given the sequence s and given our models. According to Bayes' rule, this probability can be re-written as:

$$P(t \mid s, N, N_C) = \frac{P(s \mid t, N, N_C)P(t \mid N, N_C)}{P(s \mid N, N_C)} \propto \frac{P(s \mid t, N_C)}{P(s \mid N)}$$

where the proportionality constant $P(t \mid N, N_T)$ is the marginal probability that t is true, and therefore does not depend on the particular sequence s. We will denote the ratio

[3]See Pearl [9] for example.

$P(s \mid t, N_C)/P(s \mid N)$ as $R_t(s \mid N, N_T)$. In order to use $R_t(s)$ to classify s, it is necessary to first determine an optimal threshold value ρ_t whereby if $R_t(s) > \rho_t$ then s is deemed to be in C.

2.2.2 Window Size Selection

Prior to learning structure, several experiments were performed to obtain an optimal window size $N_{us} + N_{ds}$ for each target site, where N_{us} is the number of upstream nucleotides, and N_{ds} is the number downstream. Our procedure essentially builds the features up, one-by-one, upstream until no improvement, then downstream until no improvement, then upstream again, etc., until we had reached $N_{ds} = N_{us} = 25$. Improvement was defined as an increase in the sum of the percent accuracy, $\mathcal{A}$, of the positive and negative instances: $\mathcal{A} \equiv TP/(TP + FN) + TN/(TN + FP)$, where TP, TN, FP, FN are the true positives, true negatives, false positives and false negatives, respectively. The network structure used for this calculation was a simple, 2nd-order Markov model with edges being directed downstream (i.e., every nucleotide X_i was directed toward X_{i+1} and X_{i+2}). All search on the feature space was performed with the training data by randomly selecting 20% of the training data to use as a validation set for testing the score of the feature set. At each iteration, the MAP value of the parameters was determined (see Equation 3) for each structure using the training set, and the classification performance on the validation set was calculated. The value for ρ_t needed for this calculation was estimated ahead of time by performing classification with a 2nd-order Markov model with *all* features present and selecting a threshold that treated the percent TP roughly equivalent to the percent TN. The value of ρ_t used for features selection for all targets was $\rho_t = 0.6$.

A typical window-size determination is shown in Figure 2. Note in the figure that the algorithm begins adding nucleotides upstream, but after only two iterations the score does not improve, so it begins adding downstream. After it begins adding nucleotides downstream, it essentially continues adding 15 consecutive downstream nucleotides which, it turns out, are essentially all the relevant downstream nucleotides. The path that the window-size algorithm takes in adding nucleotides therefore suggests that downstream nucleotides are more relevant to classifying acceptor sites.

2.2.3 Learning models from the data

The network in Figure 1 dramatically reduced parameters in the joint density calculation by explicating many independencies in the model. However, these independencies were assumed in an *ad hoc* fashion. Our strategy on the other hand, rather than imposing *a priori* assumptions about independencies, was to *learn* from data which independencies are more relevant from a classification standpoint. This section discusses the details of our method for learning parsimonious networks.

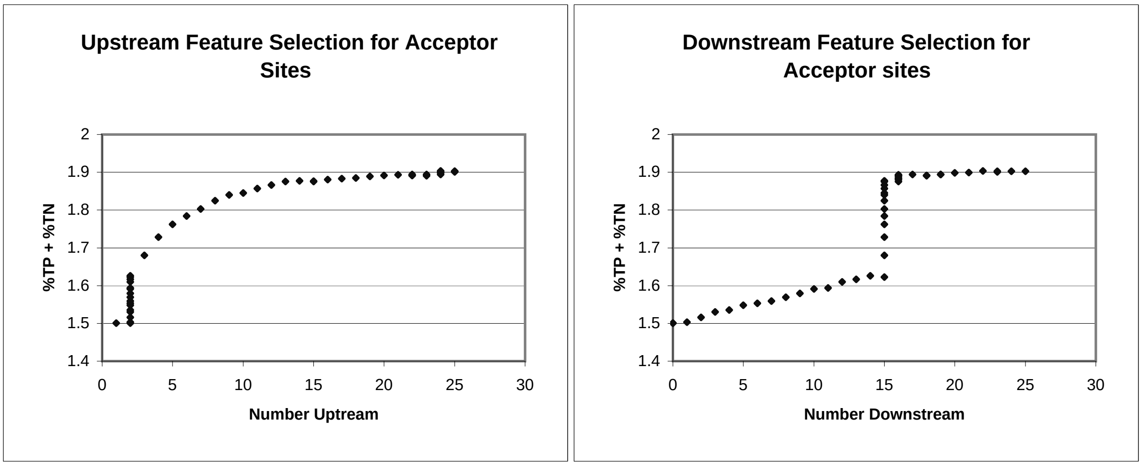

Figure 2: Window size determination for acceptor sites. Windows $>\sim 15$ produced relatively little improvement in classification accuracy.

Our basic approach was to perform a search through the space of network structures, scoring each network by the posterior probability of the structure given the data, and choosing the one with the highest score. Performing an exhaustive search of network structures is impossible for any practical networks due to the large size of the search space, which grows more than exponentially in the number of variables N. Our learning algorithm deals with this limitation by using two-levels of greedy search and by applying two heuristics to limit the search complexity:

```
Thick-Thin(G | V, D)
Input:  data set D and set of nodes V.
Output:  A Bayesian network G with a (locally) maximal score.
Procedure:
   (1) Add most likely arcs to G until no improvement.
   (2) Remove the least likely arcs from G until no improvement.
   (3) Return (G)
```

In addition to the greedy search heuristic listed above, other heuristics were applied to limit the search complexity:

1. The direction of arcs in the networks could only be directed downstream, i.e., an arc $X_i \to X_j$ was forbidden if $j < i$.
2. The number of parents, n_p, of any node was constrained to be no more than two. This condition was enforced *post facto* by repeatedly removing the violating arcs that contributed the least to the score of the network.

Heuristic 1 was used by both Salzberg [12] and Cai, et al. [3]. We also tried variations on heuristic 1, whereby, for example, we directed the ordering upstream. These variations did not appear to change the classification results significantly (see Figure 6). Heuristic 2 is also a more general version of the restrictions made by Salzberg and Cai, et al. where n_p was set to 1.

To measure the quality of the network, we used the marginal likelihood of the data given the structure:

$$p(D \mid S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})} \tag{1}$$

The i, j, k indices define a parameter, θ_{ijk} in the network. The configuration (i, j, k) corresponds to the case when X_i is in the k-th configuration and the parents of X_i in the graph are in the j-th configuration. For example if variable X_i has two parents, each taking values $\{A, T, G, C\}$, then the 16 configurations of X_i's parents are $\{AA, AT, AG, AC, TA, \ldots, CC\}$. In equation 1, N_{ijk} is the number of records in the database in which configuration (i, j, k) appeared, and α_{ijk} are parameters that represent our prior beliefs about configuration (i, j, k) (here we set all α_{ijk} to the small constant value of 5, reflecting that we have little or no prior knowledge about this domain). The terms N_{ij} and α_{ij} are defined as $N_{ij} = \sum_k N_{ijk}$ and $\alpha_{ij} = \sum_k \alpha_{ijk}$. The Γ functions are well defined mathematical functions that can be computed in constant time.[4]

Cooper and Herskovits ([4]) derived Equation 1 for computing the marginal likelihood of the network structure given some basic assumptions. The marginal likelihood scoring metric possesses several nice properties. First, it can be shown using Bayes' theorem that the posterior probability of the network structure S given the data D is proportional to the marginal likelihood:

$$p(S \mid D) = p(S)p(D \mid S)/p(D) \tag{2}$$

The most practical advantage of the marginal likelihood is that it is a *local* criterion. That is, if one arc is changed in the network, the whole score does not need to be recalculated; rather, just the score of the node whose parent set has changed needs be recalculated. This feature is especially useful and contrasts with, for example, a score like the classification accuracy of the network which must be recalculated in full each time the network structure changes.

Once the network structure was established, the maximum *a posteriori* (MAP) parameters of the network, $\hat{\theta}_{ijk}$, can be calculated using the closed-form expression [4]:

$$\hat{\theta}_{ijk} = \frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}} \tag{3}$$

[4]See [10], Section 6.1, for example.

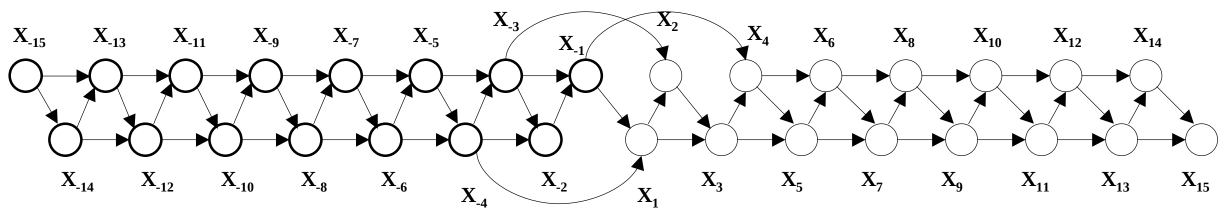

Figure 3: The model of acceptor sites learned from the entire (containing $> 99\%$ negative instances) training dataset. Note that this is essentially a second-order Markov model with some exceptions near the splice junction.

3 Results: Models and Comparisons

One of the key advantages to our approach is that the learned structures, by themselves, can provide important qualitative insight about the interactions between the nucleotides that are responsible for target site identification. Figure 3 shows a typical structure learned over the entire training set (positive plus negative examples). This structure presents the same model that one would expect if the nucleotides are interacting at random, given the heuristics that were applied during the search. Since the search limited the number of parents of each node to ≤ 2 and forced the direction of arcs downstream, one would expect to see a simple second-order Markov model, whereby each node was connected to its two nearest neighbors downstream. Indeed that is essentially what Figure 3 shows, with some exceptions near the junction site. The structure is disrupted near the junction site even though this model was trained on $> 99\%$ negative data because all records have the target sequences (in this case AG) removed from the data. This second order Markov structure is very similar to the networks learned from the entire donor and start site data sets.

Much more interesting were the structures learned from the positive data sets. That network for acceptor sites is shown in Figure 4. Some surprising features are immediately apparent. First of all, note the extremely simple structure among the upstream nucleotides. Even though each node was permitted to have as many as two parents, most upstream nucleotides only chose to have one parent, giving what is essentially a first-order Markov structure among the upstream nucleotides. Second, note that among the downstream nucleotides, not only is the structure considerably more complex than a simple Markov model, but there are surprisingly long-range interactions present. In particular nucleotide X_2 seems to have strong influence on many distant nucleotides. For example, X_2 had a stronger influence on X_{14} than any of X_{14}'s ten nearer neighbors. The fact that the structure is so complex among downstream nucleotides provides evidence that these are more relevant to the identification of acceptor sites. This is consistent with Figure 2 where, during window-size selection, although upstream nucleotides are given the first opportunity to be included in the model, after only the second iteration they cease to improve the score; whereas all 15 downstream nucleotides were added in succession.

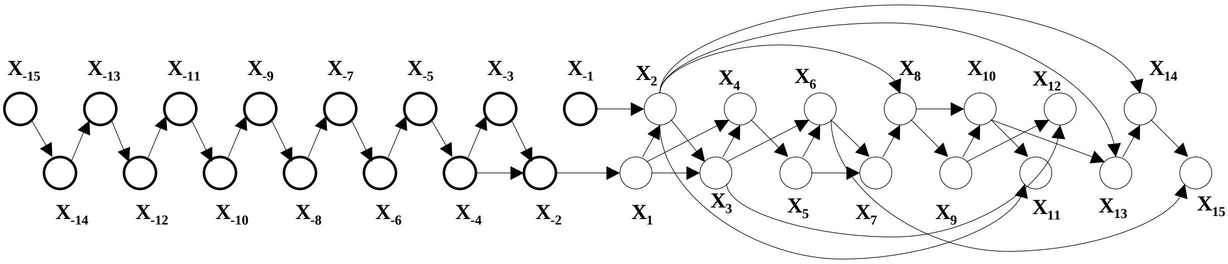

Figure 4: The model of acceptor sites learned from just the positive training data. Note the extremely simple structure (first-order Markov) over the upstream (shown in bold) nucleotides, but the complicated and surprisingly long-range interactions over the downstream nucleotides.

In short, we believe these models themselves can have interesting biological implications. Our analysis at this stage is preliminary, but we hope to develop better biological interpretations of these structures in future work.

The classification accuracy of our models was tested against both first-order Markov models and Independence models. For all comparisons, the Markov and Independent models were subject to the same window-sizes and same arc-directionality restrictions as our Thick-Thin algorithm. The results of this comparison for identifying start sites, shown in Figure 5, indicated that the Thick-Thin algorithm showed considerable improvement over the other methods. Similarly, the results for donor and acceptor junctions showed that our method

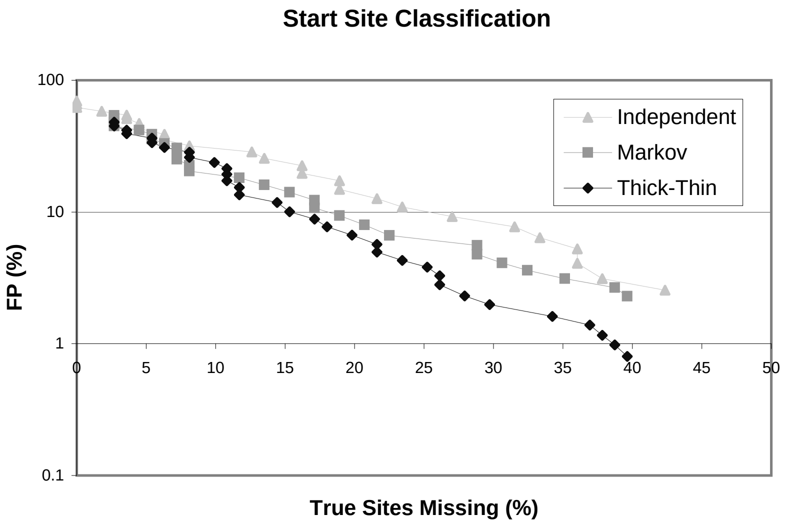

Figure 5: Start site classification results. The Thick-Thin algorithm significantly outperformed the 1st-order Markov model.

was roughly comparable to the Markov model–a little better for donor sites and a little worse for acceptor sites.

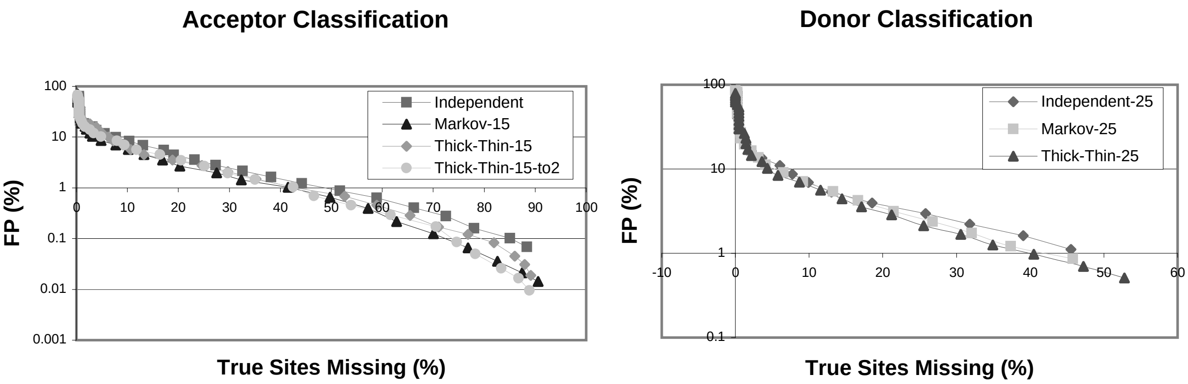

Figure 6: Splice junction classification results. The performance of our Thick-Thin algorithm was about the same as the first-order Markov model.

4 Discussion

It should be emphasized that the results presented in Section 3 are preliminary and should not be used as conclusive evidence that our procedure is a significantly better classifier than the first-order Markov model. In order to come to such a conclusion it would first be necessary to repeat these experiments, perhaps using an n-fold cross-validation scheme in order to get an indication of the statistical significance of the results. It would similarly be desirable to perform systematic comparisons with the tree-learning method discussed in [3]; however their classifier showed similar performance to the first-order Markov model performance.

We have tried to argue that our approach has advantages due to the ability to gain qualitative insight from the learned structures. While it can be argued that the other methods, namely those using tree-based models, are capable of modeling long-range interactions, they are limited in this ability because each node is constrained to have only a single parent. For example, in Figure 4, all nodes that have long-range (> 5) parents also have short-range parents. If forced to choose between one or the other, it would most likely choose the nearer neighbor. This fact is consistent with the results of Cai, et al. [3].

The cost for this insight comes at a computational expense. While our models have identical complexity ($O(N)$) for classifying a given instance once the model has been constructed, the actual act of learning the model involves a search over a highly complex search space, and can therefore take time. For reference, to learn an acceptor model with 30 nodes and > 200000 records takes a little less than an hour on a Pentium 700 Mhz machine using our Thick-Thin algorithm. Despite this drawback, our preliminary experiments reported here seem to suggest that the robust classification ability as well as the qualitative insight gained from this method are likely to make the initial effort worthwhile.

Acknowledgments

This research was supported by the National Aeronautics and Space Administration under the Graduate Students Research Program (GSRP), grant number S99-GSRP-085.

References

[1] S. Brunak, J. Engelbrecht, and S Knudsen. Prediction of the human mRNA donor and acceptor sites from the dna sequence. *J. Mol. Biol.*, 220:49–65, 1991.

[2] M. Burset and R. Guigo. Evaluation of gene structure prediction programs. *Genomics*, 34(3):353–367, 1996.

[3] Deyou Cai, Arthur Delcher, Ben Kao, and Simon Kasif. Modeling splice sites with Bayes networks. *Bioinformatics*, 16(2):152–158, 2000.

[4] Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

[5] M.R. Green. Pre-mRNA splicing. *Annual Review in Genetics*, 20:671–708, 1986.

[6] M. Kozak. An analysis of 5'-noncoding sequences from 699 vertebrate messenger RNAs. *Nucleic Acids Research*, 15(20):8125–8148, 1987.

[7] M. Kozak. A consideration of alternative models for the initiation of translation in eukaryotes. *Critical Reviews in Biochemistry and Molecular Biology*, 27:385–402, 1992.

[8] S. Mount, C. Burks, G. Hertz, G. Stormo, and O. White. Splicing signals in *drosophila*: intron size, information content, and consensus sequences. *Nucleic Acids Research*, 20:4255–4262, 1992.

[9] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1988.

[10] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (Second Edition)*. Cambridge University Press, Cambridge, UK, 1992.

[11] S. Rampone. Recognition of splice junctions on DNA sequences by BRAIN learning algorithm. *Bioinformatics*, 14(8):676–684, 1998.

[12] S. Salzberg. A method for identifying splice sites and translational start sites in eukaryotic mRNA. *Nucleic Acids Research*, 26:365–376, 1997.

[13] P. Senapathy, M.B. Shapiro, and N.L. Harris. Splice junctions, branch points, and exons: sequence statistics, identification, and applications to genome project. *Methods in Enzymology*, 183:252–278, 1990.

[14] V. V. Solovyev, A.A. Salamov, and C. B. Lawrence. Predicting internal exons by oligonucleotide composition and discriminant analysis of spliceable open reading frames. *Nucleic Acids Research*, 22:5156–5163, 1994.