LEGIBILITY NOTICE

A major purpose of the Technical Information Center is to provide the broadest dissemination possible of information contained in DOE's Research and Development Reports to business, industry, the academic community, and federal, state and local governments.

Although a small portion of this report is not reproducible, it is being made available to expedite the availability of information on the research discussed herein.

JUN 07 1989

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

TITLE APPLICATION OF NEURAL NETWORKS AND OTHER MACHINE LEARNING
ALGORITHMS TO DNA SEQUENCE ANALYSIS

LA-UR--89-1788

DE89 013457

AUTHORS Alan G. Lapedes
Christopher Barnes
Christian Burks
Robert M. Farber
Karl M. Sirotkin

Submitted For publication in Santa Fe Institute's
Proceedings: The Interface Between Computational Science
and Nucleic Acid Sequencing, held Dec. 12-16, 1988,
Santa Fe, NM

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DTIC Accession Number

DTIC Accession Number: AD A200 000 (Unannounced) Available from DTIC

DTIC Accession Number

DTIC Accession Number: AD A200 000 (Unannounced) Available from DTIC

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

MASTER

A. Lapedes, C. Barnes, C. Burks, R. Farber, K. Sirotkin

Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545

Application of Neural Networks and Other Machine Learning Algorithms to DNA Sequence Analysis

In this article we report initial, quantitative results on application of simple neural networks, and simple machine learning methods, to two problems in DNA sequence analysis. The two problems we consider are:

(1) Determination of whether procaryotic and eucaryotic DNA sequence segments are translated to protein. An accuracy of 99.4% is reported for procaryotic DNA (*E. coli*) and 98.4% for eucaryotic DNA (*H. Sapiens* genes known to be expressed in liver).

(2) Determination of whether eucaryotic DNA sequence segments containing the dinucleotides "AG" or "GT" are transcribed to RNA splice junctions. Accuracy of 91.2% was achieved on intron/exon splice junctions (acceptor sites) and 92.8% on exon/intron splice junctions (donor sites).

The solution of these two problems, by use of information processing algorithms operating on unannotated base sequences and without recourse to biological laboratory work, is relevant to the Human Genome Project. A variety of neural network, machine learning, and information theoretic algorithms are used. (For the purposes of this article, we view neural networks solely as an information processing procedure and do not consider the possible relation of these formal models to biological networks of neurons.) The accuracies obtained exceed those of previous investigations for which quantitative results are available in the literature. They result from an ongoing program of research that applies machine learning algorithms to the problem of determining biological function of DNA sequences. Some predictions of possible new genes using these methods are listed, although a complete survey of the *H. sapiens* and *E. coli* sections of GenBank will be given elsewhere.

INTRODUCTION

Annotated DNA sequence data presently available in databases such as GenBank¹ provide an important source of data for evaluating machine learning algorithms (statistical techniques and information theoretic methods) that may be useful in the Human Genome Project. When larger amounts of data are available from the Human Genome Project², it will be essential to have automated, accurate, and verified biological information processing techniques that help sift the raw sequence data for

biochemically important features and to reduce the amount of laboratory work required to answer questions of biological interest⁸. The value of such techniques depends on their accuracy, which may be determined only by careful quantitative analysis.

In this article we use relatively new ideas from the field of neural network research (in combination with other machine learning algorithms, statistical analysis, and information theory) to address two problems:

- (1) Whether procaryotic and eucaryotic DNA sequence segments are translated to protein, and
- (2) Whether eucaryotic DNA sequence segments containing the dinucleotides "AG" or "GT" are transcribed to RNA splice junctions.

There is a wide literature¹⁷ on applications of statistics to genetic analysis, and some of the neural net techniques we use are related to certain types of statistical approaches. In particular, we note that simple statistical weighting schemes commonly used in DNA analysis^{32,33} are a subset of general neural net approaches, and therefore neural nets may be more general and more useful. Although we are able to decipher the way the neural net arrived at the results for the first problem we consider here, this is not always possible; and application of "automatic rule-based" systems provide an alternative method by which the explicit rules for arriving at an answer are inferred. Finally, we describe information theoretic techniques to produce graphs illustrating where in the DNA sequence the information relevant to solving the two problems resides.

The approach advocated here, i.e., quantitative analysis of various new (and older) information processing methodologies, can be used to judge which algorithms might best play a practical role in the analysis of genetic data. We describe algorithms that we found to have significant success and present quantitative verification of accuracy. Due to the sparsity of rigorous quantitative statements concerning other approaches in the literature, only rarely were we able to compare the new algorithms to other methods. We found significantly improved accuracies for the two problems considered here. Extensive surveys of GenBank will be presented elsewhere.

GENERAL METHODOLOGY AND SPECIFIC DNA SEQUENCES USED:

Problem 1

The first problem deals with isolated sequence segments of DNA (ranging in length from 15 to 270 bases). We address the question of whether the segment contains sufficient local information to determine if it is translated to protein. We consider DNA from *E. coli* and also from *H. sapiens*. The problem is virtually identical to one considered by J. Fickett¹¹ in a careful investigation using different methods. Accuracies reported here are significantly higher — although we caution that due to the variability of genetic data, comparisons of accuracies are best performed on identical data sets. We did not have Fickett's data available.

Data from GenBank release number 57.0 were used in the following manner: A "training set" of fragments was constructed from two subsets of data. The true subset consisted of in reading frame segments of DNA of fixed length taken from known coding regions. The "false" subset consisted of segments of identical length annotated in GenBank as not coding for protein — selected at random. (For *H. sapiens* the "false" set was selected solely from segments contained in annotated introns.) A DNA segment in the false set can contain an "in frame" stop codon (relative to a "frame" defined by the start of the segment), and hence this could provide a trivial distinction between segments from the "true" set (with no in-frame stop codons) and segments from the "false" set. Thus the "false" segments had any stop codons in them removed to avoid the obvious bias of classifying any segment with an "in frame" stop codon as a member of the false set. We emphasize that we therefore deal only with classifying Open Reading Frames (ORF's). To eliminate stop codons in the "false" set we investigated two alternative methods. The first method eliminated an entire segment of DNA if a stop codon appeared in a segment of the false set. That is, if in the process of selecting segments of a specified length from introns, a stop codon appeared "in frame" in that segment (relative to the start of the segment), then that segment was not used in forming the "false" set. The second method merely excised the stop codon and replaced it with the next succeeding codon. Interestingly, accuracy seemed to differ significantly depending on what method of dealing with stop codons was used (not shown). These true and false subsets comprise the "training" set.

The first approximately 50 non-duplicated *E. coli* GenBank entries from an older release of the database were processed as follows. Regions annotated as coding

for protein were taken for the “true” set. Regions neither coding for protein nor coding for structural RNAs were taken for the “false” set. The “true” and “false” sets for the training procedure for *E. coli* were constructed by extracting the first 400 bases of every other sequence from the above sequences. These data were then segmented into lengths of 15, 45, 90, 120, 150 base long fragments in order to test the effects of segment length. The prediction set was the rest of *E. coli*. After training is complete one may “window” a new sequence into appropriate segments, and attempt identification of protein coding regions.

For *H. sapiens* the “true” set came from regions annotated as coding (“pept”) and being expressed in liver (“liver cDNA” in SOURCE field). Pseudo genes and duplicate members from multigenes were removed. The “false” set was prepared similarly, except from introns (“IVS”), and tissue type was ignored.

Data from the training set were used to fix any adjustable parameters in the various algorithms. A disjoint “prediction set” of true/false segments was formed in a similar manner to the training set from sequences not used in constructing the training set. After the training set was used to adjust parameters appearing in the algorithm, the algorithm was evaluated on the prediction set. Evaluation of the algorithm on the prediction set, which consists of data not used in construction of the algorithm, is indicative of the accuracy of the algorithm on new data produced by experiment¹⁰.

[If new, experimentally produced data is not representative of the known selection effects in current GenBank data (e.g., over representation of genes coding for abundant proteins), then accuracy on the prediction set may not be indicative of the accuracy on new data. This is an unavoidable consequence of current experimental techniques. In a more detailed analysis, one may wish to train networks on subclasses of GenBank data that may then generalize to special situations.]

We point out that the methodology of dividing the data into a “training set” and “prediction set” is critical for obtaining meaningful results. Prediction on the training set (and not on new data) can lead to gross overestimates of accuracy, particularly when data sets for training are limited (see Tables 1b, 2b, 3b, 4b). Numerous papers in the literature use training data in attempts to evaluate the accuracy of algorithms. The method of eliminating “in frame” stop codons from the false set can also affect results, as well as the size, of the training set. (Therefore, care must be taken in comparing the results of algorithms used here to results present in the literature.) Our results for Problem 10 were reproduced to within 0.1% upon different random partitions between training and prediction sets, when the size of the partitions is the same between experiments.

Problem 2

The second problem deals with the determination of functional splice sites in eucaryotic DNA. The analysis used a similar methodology to the previous problem. A training set and a separate prediction set were constructed. The training set and prediction set each separately contained true and false examples. The training and prediction sets were constructed from Release 57 of GenBank. First the known (true) splice junctions were extracted along with a surrounding window of bases of specified length from both sides of the highly conserved dinucleotide. This was done separately for the boundary between an intron and an exon (acceptor sites, conserved dinucleotide = "AG") and for the boundary between an exon and intron (donor sites, conserved dinucleotide = "GT"). We did not consider the rare cases of deviation from these dinucleotides. This procedure was followed three separate times in order to test the effects of three lengths of the surrounding window of bases: a total of 11 bases, 21 bases, and 41 bases.

The "true" training sets and "true" prediction sets were constructed by first randomly shuffling the order of these true examples and then selecting all but 50 of the examples to be a true training set for each length category, while the remaining 50 were the true prediction set for that category. The reason for using a majority of data in the training set is that the total amount of presently available splice junction data is limited. Accuracy was increased by using more of the data during training to accurately determine parameters found in the algorithm. The resulting small size of the prediction set was then compensated for by repeating the random division into training sets and prediction sets a number of times and averaging the accuracies of the resulting networks. The results we report are averaged over five trials. This method is a variant of the classic "leave one out" procedure commonly used in applications of machine learning algorithms to small data sets¹⁰.

The false sets were constructed by similarly extracting 11, 21, and 41 base windows surrounding the dinucleotides, "AG" (false acceptor splice sites) and "GT" (false donor sites), occurring inside exons. For each length category, the number of false examples in the training set far exceeded the number of true examples. This overemphasis on the false category examples can be compensated for in the neural net training algorithm. In a separate investigation we studied the effect of choosing examples of the false set from windows surrounding AG or GT in both exons and introns (as opposed to selecting the false set from windows surrounding AG or GT that are only in exons). Here the number of false examples was chosen to equal the number of true examples. Little variation in results was found between choosing the false set from both introns and exons, or solely from exons.

[Even when care is taken to divide the data into separate predict and train sets, one finds for Problem (2) that identical patterns can occur between the train and predict sets. This is due to the highly restricted nature of the base composition surrounding the splice sites. A small, but not insignificant increase in accuracy may therefore be obtained by mere table lookup. In the future, when one knows whether the duplicated patterns are representative of a true prototypical class, one may wish to capitalize on this increase in accuracy. In this paper we present results on non-overlapping train and predict sets in order to obtain insight into the ability of the algorithms we use to generalize to new situations. The results we present are therefore presumably a conservative, lower bound on predictive accuracy. Results on incorporating the increased table lookup accuracy (best done by "rule-based" systems) will be reported elsewhere.]

We note that "clean" determination of false splice sites is complicated by the possibility of alternative splicing. In this preliminary study, to avoid complications due to alternative splicing, we removed genes that are prone to alternative splicing (hemoglobin, oncogenes, etc.). Certainly other constructions of a training set are possible. The procedure used, however, does seem to be a reasonable "first cut" at the problem.

ALGORITHMS

A variety of methods were tested, all of which have their genesis in various approaches to machine learning. A great deal of knowledge concerning these algorithms is available in the literature, and their behavior on numerous problems involving the processing of symbolic informations is a focus of continuing research^{9, 29, 30}.

Neural Networks:

Important work of Hopfield and others^{9, 11, 29, 30} has caused a resurgence of interest in neural net formalisms in the past five years. A classic neural net algorithm, the Perceptron Method¹⁰, is the forerunner of many neural net algorithms in use today. This method has previously been used by Nakata, Kaneisha, and DeLisi²⁴ in an investigation of the splice site recognition (our Problem 2) and by Stormo, Schneider, Gold, and Ehrenfeucht²⁴ in attempts to determine ribosome binding sites. Accuracies were not as high as could be desired. We first describe the general idea of the Perceptron Method and then discuss recent extensions^{21, 29} that have been shown to yield improved accuracies in many other interesting problems.

In the context of DNA analysis one starts by providing a suitable encoding of the symbols "A", "T", "G", and "C" into a string of bits, i.e., 0's and 1's. For example, each symbol can be represented by four bits, and these four bits = symbol

are concatenated to provide a bit string representing a given length of DNA (see Fig. 1b). Other representations may be chosen, such as using 64 bits to represent each of the 64 possible codons.

A “neural network” can be constructed by assigning a “neuron” to each bit of the input string (with a state “on” or “off”), by assigning one more additional neuron to be an output neuron, and by viewing the arrangement as a network where the input neurons influence the state of the output neuron. After training the network, the output neuron should take on a value near one if the input string comes from the true subset of the training set, and a value near zero otherwise (see Fig. 1b). The state of the output neuron is given by

$$\mathbf{O} = g\left(\sum_j T_j I_j + \theta\right), \quad (1)$$

where I_j represents the input values (0 or 1), $\mathbf{O}$ represents the state of the output neuron, T_j are numerical “weights” to be determined by a learning algorithm, and θ is an additional constant also determined by the learning algorithm. $g(x)$ is an arbitrary smooth function, monotonically rising from the value 0.0 for negative values of x , to the value 1.0 for positive values of x . We chose

$$g(x) = \frac{1}{2} (1 + \tanh(x)), \quad (2)$$

where *tanh* is “hyperbolic tangent,” although other forms may be used without affecting results.

The weights are initialized to small random values. Training the network (i.e., finding the optimal T_j and θ) is accomplished as follows. Let p index the various examples in the training set, and let $t^{(p)}$ represent the target output value (0 or 1) for the p^{th} input example. If the input example belongs to the true subset of the training data, the target output for that example should be a 1, and should be a 0 otherwise. Similarly, let $\mathbf{O}^{(p)}$ represent the actual output of the net for the p^{th} input example. The mean square error made by the network on the training set is given by

$$E = \sum_p \left(t^{(p)} - \mathbf{O}^{(p)}\right)^2 \quad (3)$$

One clearly wishes to minimize the error, E . One way to do this is to make successive changes, ΔT_j and $\Delta\theta$, in the weights, T_j , and in θ by

$$\Delta T_j = -\frac{\partial E}{\partial T_j} \cdot \epsilon \quad (4a)$$

$$\Delta\theta = -\frac{\partial E}{\partial \theta} \cdot \epsilon \quad (4b)$$

where ϵ is a small parameter. Simple algebra shows that the error will decrease to a minimum. We actually use an efficient conjugate gradient method²⁷ to minimize E , but Eq. (4) is often used in the literature, and it approaches the same result (albeit slowly).

After minimizing the error on the training set, the weights, T_j and also θ , will be known. One may then test the network's accuracy on the "predict set."

An alternative approach^{3,15} is to view $\mathbf{O} = g\left(\sum_j T_j I_j + \theta\right)$, which ranges between zero and one, as a probability of a discrete state output being a 1.0 given the input I . One can then compute the joint probability over the training set that all the examples from the true subset yield an output of 1.0, and that all examples from the false set yield an output of 0.0. This joint probability may be written as

$$P = \exp \left(\sum_p t^{(p)} \log(\mathbf{O}^{(p)}) + \sum_p (1 - t^{(p)}) \log(1 - \mathbf{O}^{(p)}) \right) \quad (4c)$$

One may then define E to be the negative log of this probability

$$E = -\log(P) \quad (4d)$$

and minimize this quantity in a manner similar to Eq. (4b). This is equivalent to maximizing the log of the probability P and therefore to maximizing the probability P . We refer to this method of training as the "maximum likelihood" method.

Rumelhart and McClelland¹⁴ have recently extended this formalism by including additional layers of neurons between input and output that may be viewed as a kind of intermediate processing layer. These neurons are conventionally called "hidden neurons" (i.e., neither accessible as outputs or as inputs) and considerably

extend the power of the network in Fig. 1. The outputs of the hidden neurons, given by Eq. (1) are now fed as inputs to the final output neuron. The equations become more complicated, but the form of Eq. (4) that is used to find the weights remains the same. Details may be found in Ref. 29. This algorithm is commonly called the back propagation neural net method and is quite popular.

Another modification of the network has been suggested by Y. C. Lee et al.²¹ (see also Rumelhart's sigma-pi units²⁹). Equation (1) is modified so that the output is no longer a function of the linearly weighted sum of inputs. Instead a linearly weighted sum of products of inputs is used. One may use products of any desired order; however, one is often limited by the availability of data used to determine the additional weights or limited by computer resources. For the latter reason, we have not yet gone above third order. In this case Eq. (1) becomes

$$O = g \left(\sum_{ijk} T_{ijk} I_i I_j I_k + \theta \right) . \quad (5)$$

The analogue of Eq. (4) (with the appropriate extra indices) is used to find the T_{ijk} . This method is called the high order correlation neural net method.

This method is closely related to classic statistical techniques of multivariate analysis. Also, all of the techniques above are a form of discriminant analysis, and relations may therefore be found to classic statistical and discriminant techniques¹⁹. Note that if I_j represents bases, then Eq. (5) includes the effects of base correlations up to third order. Therefore, Eq. (5) can subsume the correlations among bases due to codons, and to other base correlations as well. One advantage of the more powerful network formalisms is that they are capable of automatically forming appropriate "internal representations" of the data. For example, if the R, Y representation (i.e., the purine, pyrimidine representation of data from the IUB system) is more useful for solution of a particular problem, then a base representation may be used on the input layer, and an internal R, Y representation (or other appropriate representation) can be formed automatically by the network on the "hidden" layers. Similar "internal" representations may be found automatically by high order neural networks. Decoding what internal representation was chosen by a network is in general a difficult problem, but is one of obvious biological importance. We used Eq. (5) for the analysis of Problem 1 and were able to distinguish the effects of codon usage from other base correlations in a controlled fashion.

As we have seen, there are a variety of neural net formalisms, the simplest of which are reminiscent of the simple statistical weighting schemes previously used in DNA analysis. The power of the new neural net formalisms may also be understood as follows. It is possible to show that the simplest weighting scheme, Eq. (1), (with no "hidden" neurons) attempts to discriminate the true class from the false class by passing a separating hyperplane between the two classes¹⁰. This plane exists in a high dimensional space, the dimensionality of which is related to the number of input neurons used to represent the data. The learning algorithm, Eq. (4), attempts to move the plane so that true patterns in the training set fall on one side of the plane and the false patterns on the other. A similar interpretation exists for attempts to adjust the weights in simple statistical schemes used in the biological literature (except that optimal procedures are not used to adjust the plane). However, it is easy to construct situations in which the two classes (true and false) do NOT fall on opposite sides of a plane, and one suspects that this situation is generic (see Fig. 2). The extension of Eqs. (1) and (4) using hidden neurons, or Eq. (5) using high order networks, allows curved separating surfaces to be used^{20,22}, and one can see from Fig. 2 that this is exactly what is required in the generic situation. This is one way in which the new neural net approaches achieve additional power over the older Perceptron approaches and related simple statistical weighting schemes.

However, it is important to understand that one can use less powerful networks (e.g., Perceptrons) with heavily pre-processed input patterns, to achieve any result the newer algorithms produce. That is, there is a trade-off between putting processing power into the network versus putting it into a pre-processing stage that changes the representation of the input data. In terms of Fig. 2, these two situations correspond to (1) using a powerful network to derive an appropriate curved surface separating the two classes versus (2) using a minimal network capable of producing only planar separating surfaces, but pre-processing the input data so that the separate class regions are deformed into convex shapes that can be separated by a plane. When one has some idea of the appropriate pre-processing (e.g., using an input representation for codons instead of single bases for Problem 1) then the latter procedure is useful. However, when one has little idea of the correct pre-processing (which is the more usual situation), then turning the job of classification over to a powerful net capable of automatically forming curved separating surfaces is useful. Clearly, the best approach is to encode as much relevant information as possible about the problem into the input representation and then to use a sufficiently powerful network to perform accurate classification. In fact, we concluded that a representation using codons (as opposed to bases) is useful for Problem 1 by first using a powerful network (a third order net) on a base representation and then deducing that the network was responding most strongly to third order base

correlations over adjacent bases (i.e., codons). Thus in certain situations, one may deduce an optimal input representation by analyzing more powerful networks and then return to using the less powerful networks that operate on the optimal input representation. Appropriate input representations can greatly reduce training times.

Automatic Rule Based Methods, Metric Methods, and Information Theory

Automatic rule-based methods^{13,25,28} are a class of methods that attempt to infer an algorithm from a set of examples (as do neural nets) but operate in a much different fashion from neural networks. Typically, at the end of a training procedure for rule-based systems one obtains a collection of rules such as

if(condition on symbols) $\rightarrow$ (outcome)

Chaining these rules together provides a logical route to go from the symbolic input data (e.g., a string of "A, T, G, or C's" for DNA) to a conclusion, e.g., "true splice junction" or "false splice junction." One advantage of rule-based systems is that the logic involved in arriving at a conclusion is transparent; whereas for neural networks, the logic is encoded in the network architecture and the network weights. The latter can sometimes make biological interpretation difficult. The set of rules obtained from a rule-based system are similar to "expert systems"^{6,36} used in Artificial Intelligence; however, the rules are derived automatically without recourse to an outside "expert," thereby avoiding the "knowledge acquisition bottleneck" problem of expert systems. Rule-based systems have enjoyed considerable success in various problems^{13,25,28} and in certain interesting cases have been shown to exceed neural nets in accuracy¹.

The particular rule-based system we employed is due to Quinlan²⁸. The idea is to grow a decision tree from the training set samples and use this tree to classify the prediction set. The decision tree is grown by finding those base positions that contribute the most information towards the classification of an example as "true" or "false" (e.g., "true" or "false" splice junctions) and then finding the most informative bases in those bases positions. "Information" is used in a technical sense and refers to the "mutual information" of the base position and the category "true" or "false"²⁸. The result of the algorithm is a binary tree that descends from a root node and becomes increasingly specific as one descends down the levels of the tree. Each node contains a template that displays which bases are the most important in each base position and also contains a classification category of "true" or "false" for examples matching each template. To use the tree for prediction, one takes the

DNA sequence segment to be classified and checks if one of the templates on the bottom level of the tree matches it. If a match occurs, the sample is classified with the class category associated with the matching template. If no match occurs on one level, then one goes up a level in the tree (hence using less specific templates) and checks for a match, repeating the procedure until a match is found (which is guaranteed). Because the accuracy of the rule based system, in the cases we considered did not exceed the accuracy of the neural networks, we will not give full details of the rule based system here. Explicit details of all algorithms used, as well as a complete survey of the *H. sapiens* and *E. coli* sections of GenBank using the various methods outlined here, will be available in Ref. 2.

As explained above, one advantage of the rule based system is that the logic leading to the classification of true or false is transparent. When the rule based algorithm outlined above was used on the splice site recognition problem, it was possible to find a relatively small set of rules classifying the splice junctions with almost the same accuracy as the neural net. An example of some of these rules is given in Fig. 5.

Further work along these lines, combined with efforts to reduce rule systems to the minimal number of rules, could result in a relatively small set of biologically interpretable rules that can correctly allocate putative splice junctions to the true or false class with high accuracies.

Other methods exist for determining where information resides in a base sequence that contributes to the sequence being designated as one class or another (e.g., true or false splice sites). Considerable attention has been paid to consensus sequences associated with splice sites²³. Consensus sequences describe the observation that true splice sites have highly conserved dinucleotides ("AG" and "GT" for acceptor and donor splice sites, respectively) and in addition, in nearby positions, tend to have particular bases with significant frequencies. However, consensus sequences alone do not convey information that allows discrimination of true splice sites from false splice sites since they are constructed solely from base frequencies in the true set. If similar frequencies occur in the false set, then consensus sequences have little informative value. What is needed is a measure quantifying the differences in frequencies of bases surrounding true splice sites from those surrounding false splice sites. A suitable measure for comparing two such probability distributions is the Kullback divergence measure¹⁸, which in this context is a formula for the average number of bits of information in each base position contributing toward the decision true or false. The Kullback divergence measure is defined by $K(p, q)$ where

$$K_s(b) = (P_s^T(b) - P_s^F(b)) \log_2 \left(\frac{P_s^T(b)}{P_s^F(b)} \right) \quad (6a)$$

$$K(b) = \sum_{s \in \{A, T, G, C\}} K_s(b) \quad (6b)$$

Here, b is an integer labeling base position, and $P_s^T(b)$ represents the probability of symbol s ($s \in \{A, T, G, C\}$) occurring at base position b in the true set, and $P_s^F(b)$ is similarly defined for the false set.

The utility of such a measure is evident in Figs. 3 and 4. In Fig. 3 the true and false sets were those for the splice junction problem. Figure 3 clearly shows that the information due to single base positions contributing to the discrimination between true and false acceptor splice sites predominantly reside in roughly 15 bases on the intron side of an acceptor splice site. A similar plot (not shown) showed that the information contributing to the discrimination between true and false donor sites predominantly resides in four bases on the intron side of an donor site, but with additional information coming from approximately three bases on the exon side. When the window of bases surrounding the dinucleotides “AG” and “GT” were truncated to the widths mentioned above, we noticed that the multilayer neural net method of prediction suffered only a small loss in accuracy, thereby confirming that information in the regions described above is a key contributor to discrimination. Considerable discriminatory information residing in the intron side of splice junctions is in accord with the intron “lariat” mechanism proposed for SnRNA’s during the intron excision¹². The information residing on the exon side of donor splice junctions suggests that there may well be a biological correlate at work here as well.

Consensus sequences have also been used in attempts to represent the importance of base positions in splice sites. Typically such attempts display the most common base in a base position along with its frequency of occurrence in the “true” set. However, it is the *relative* probabilities in the “true” and “false” sets that determine the efficacy of the frequently occurring bases as a signal for a splice junction. This point is quite clear if one considers consensus sequences for the start of translation. Obviously “ATG” would appear with high probability, yet “ATG” also appears with high probability at other base positions not involved in the start of translation. Thus the value of the “ATG” as a signal for the start of translation is reduced due to the the “false” “ATG”’s appearing elsewhere. The Kullback divergence quantifies the discriminatory power of a putative signal sequence.

The occurrence of significant information in a relatively few number of bases also suggests the use of other methods. A classic, and simple, machine learning method is the k nearest neighbor method¹⁰. One takes an element from the predict set, finds the k nearest neighbors in the training set using a suitable metric, and predicts the unknown pattern as being a true splice site if the majority of these nearest neighbors in the training set are true splice sites, and predicts false for the opposite case. The success of the method hinges on the choice of a "suitable" metric or distance measure. We investigated various metrics.

A simple Hamming metric for the distance between two base sequences may be defined as the number of base positions in which the two sequences have different bases. This measures the effect of mutations but can indicate a large distance if insertions or deletions of even a single base are involved. (The two sequences are prealigned so that the conserved dinucleotides (either "AG" or "GT") are in the same base position in both sequences.) A variant of this procedure is to weight each base position with the Kullback measure, so that if a discrepancy between the two sequences occurs in an unimportant base position [low K for base position (b)], then the contribution to the distance is minimal. We call this a "weighted" Hamming measure.

Effects of insertions and deletions (which can have radical effects on Hamming measures and also network methods) may be accounted for by using a k nearest neighbor where the metric is that commonly used in sequence alignment¹¹. We call this a "similarity metric" method. Note that if similarity to small nuclear RNA's (snRNA's)¹² is important in discriminating true splice sites from false, then the k nearest neighbor similarity method should work well. We did not find this to be the case.

RESULTS

Neural net methods achieved the highest accuracies for Problems 1 and 2. Automatic rule based methods achieved slightly lower accuracies for Problem 2 and were not applied to Problem 1. Hamming and weighted Hamming metric methods were consistently poor performers (70-80% accuracy on Problem 2), were not applied to Problem 1, and will not be reported in detail. These methods were included because they are relatively simple algorithms that have worked well in other investigations. Which method works best is generally problem dependent, thus, it is prudent to survey a number of methods when a problem is of particular interest. The similarity metric method (less sensitive to inserts and deletes) also performed roughly the same as the other metric methods and so will not be reported in detail.

either. Use of the Kullback divergence measure provides more useful information than consensus sequences, and the results are reported in Figs. 3 and 4.

Tables of percentage accuracy for Problems 1 and 2 versus DNA segment length. Percentage accuracy is defined as the average of two numbers. The first number is the ratio of the number of elements in the "true" prediction set, correctly labeled as true, to the total number of true elements in the prediction set. The second number is similarly defined for the false prediction set. The total percentage accuracy is the average of these two numbers, i.e., the average accuracy on both the true and false sets. The two percentage accuracies (percentage of true correctly called as true and the percentage of the false set correctly called as false) did not differ significantly when using neural net methods, and therefore the average is a useful expression of overall accuracy. The following tables result from using neural net methods. For *H. sapiens* the coding regions were limited to those known to be expressed in liver.

PROBLEM (1)

Determination of whether or not DNA sequence segments are translated to protein.

Table 1a: *E. coli*, accuracy on Predict Set

<i>DNA Segment Length</i>	<i>Percentage Accuracy</i>
15 bases	76.1
45 bases	87.2
90 bases	93.8
135 bases	97.6
180 bases	99.5
270 bases	99.5

Table 2a: *H. sapiens*, accuracy on Predict Set

<i>DNA Segment Length</i>	<i>Percentage Accuracy</i>
15 bases	79.0
135 bases	91.9
270 bases	98.4

PROBLEM (2)

Determination of whether or not eucaryotic DNA sequence segments are transcribed to RNA splice junctions.

Table 3a: *H. sapiens*. Acceptor sites, accuracy on Predict Set

<i>DNA Segment Length</i>	<i>Percentage Accuracy</i>
11 bases	85.3
21 bases	89.7
41 bases	91.2

Table 3b: *H. sapiens*. Acceptor sites, accuracy on Train Set

<i>DNA Segment Length</i>	<i>Percentage Accuracy</i>
11 bases	85.0
21 bases	91.3
41 bases	96.0

Table 4a: *H. sapiens*. Donor sites, accuracy on Predict Set

<i>DNA Segment Length</i>	<i>Percentage Accuracy</i>
11 bases	94.5
21 bases	94.5
41 bases	91.3

Table 4b: *H. sapiens*. Donor sites, accuracy on Train Set

<i>DNA Segment Length</i>	<i>Percentage Accuracy</i>
11 bases	94.5
21 bases	95.7
41 bases	99.0

Accuracies obtained on Problem 1 compare favorably to Fickett in previous work on a related problem¹¹. A minimum of 200 bases was required in this prior work, and an accuracy of 85% was achieved (previously quoted accuracies of 95% for Fickett's algorithm assume a 20% "no opinion" rate). Eighty five per cent accuracy should be used to compare with our results because we do not use a "no opinion" category. We caution that the data sets used in the two algorithms were not identical and that this can affect a comparison of results.

Accuracies on Problem 2 exceed those for algorithms previously described in the literature such as the statistical method of Shapiro and Senapathy³². The Perceptron accuracies were significantly higher, which indicates the advantage of a procedure for optimally determining the weights. Nakata, Kaneisi³³ and DeLisi²⁴ have previously implemented a Perceptron algorithm on the splice junction problem. Again, our results are significantly higher, which we attribute to somewhat different methodologies, and also to the fact that Nakata et al. were limited by the small data sets available at the time of their investigation. A rough rule of thumb is that the number of training examples should be at least a few times the number of network weights in order for accurate predictions to be possible. Also note that accuracies on the training set are significantly higher than those achieved on the predict set and indicate that such an attempt to evaluate the accuracy of an algorithm yields significant overestimates.

A complete survey of the *E. coli* and *H. sapiens* sections of the GenBank database⁵ using our methods will be presented elsewhere². One may perform this survey by presenting the output of the networks in a graphical form that indicate coding regions and splice sites. An example of such a graphical description incorporating network predictions is in Fig. (6). In the course of this investigation we did predict some potential protein-coding regions. One example * was the prediction of a previously unannotated protein-coding region in the GenBank entry (accession number M12486) including the 5' UTR of *E. coli* flnA gene⁷; this prediction was subsequently determined to have been confirmed experimentally as corresponding to the *pon B* gene⁴. Other predictions were also made**. In view of the high accuracies obtained for correct identification and for rejection of false positives (see above),

* Initially reported in an oral presentation by A. Lapedes at the Los Alamos Conference on Nonlinearity in Biology and Medicine (1987).

** PREDICTED *E. coli* GENES (GENBANK LOCUS BY LOCUS POSITIONS) (ECOWRHL from 148 to 515), (ECOWSPAW from 1666 to 2764), (ECOWCA from less than 1 to 486), (ECOWDHL from less than 1 to 292), (ECOWPNMPC from 484 to 1500), (ECOWHCA from less than 1 to 420), (ECOWOLA from less than 1 to 466).

we expect that these predictions can be verified by the appropriate experimental work.

CONCLUSIONS

In applying various neural net, machine learning, and information theoretic techniques to the problem of detecting protein coding regions in *E. coli* and in Human DNA and to the problem of determining true splice sites from false in Human DNA, neural net methods provided the highest accuracies. These accuracies significantly exceeded those previously reported in the literature. In particular, an accuracy of 99.5% was achieved in determining whether fragments of DNA 270 bases long from *E. coli* were from known coding regions. For DNA from genes known to be expressed in human liver tissue, an accuracy of 98.5% was achieved on 270 base long fragments. All fragments were ORFS. Discussion of the importance of codon usage and other base correlations was presented. For the problem of distinguishing true splice junctions from false, an accuracy of 92% was achieved. The methodology was always to "train" a network on a set of sequences that was distinct from the set used to evaluate the accuracy of the algorithm.

Although these accuracies are cause for optimism about the techniques used in this paper, we caution that much more work needs to be done (in progress) to properly assess their relevance. A single number, the percentage accuracy, is only relevant in the context of the data in which it was obtained, and the proper data to use to derive an index of accuracy is a subject open for discussion. Therefore, we feel that at this preliminary stage equal importance should be attached to general methodology, as opposed to particular percentage accuracies arising from special techniques. In this context various points are made. Simple statistical weighting schemes commonly used in previous methods are related to the simplest neural net approaches. More powerful neural net approaches, therefore, yield additional algorithms that can be used for various problems, and increased accuracies can result. Whether enhanced accuracies are obtained is problem dependent. The practice of testing an algorithm on training data and not on a distinct prediction set (which is not uncommon in the literature) is susceptible to overestimates of accuracy. The notion of a "consensus sequence," constructed entirely from examples of true splice junctions, may be extended to a measure (the Kullback divergence measure) that identifies which base positions contain significant information that allows discrimination of true from false splice junctions.

In summary, initial results were presented from an extensive survey of procaryotic and eucaryotic DNA using various neural net, machine learning algorithms, and information theoretic techniques. High accuracies were obtained, and in so far as comparison to previous techniques are possible at this early stage, we conclude that there is cause for optimism concerning the usefulness of the methods proposed.

ACKNOWLEDGMENTS

This work was done under the auspices of the U.S. Department of Energy and was largely funded by a grant (1-R01-GM40789) from the National Institutes of Health. C.B. was also supported by a grant from the National Institutes of Health (GM-37812). K.S. was additionally supported by a grant from the U.S. Department of Energy (KP-04-04-00-0). R.M.F. and A.S.L. would like to express gratitude for the hospitality of both the Santa Fe Institute (Santa Fe, New Mexico) and the Institute for Theoretical Physics (Santa Barbara, California), where part of this work was performed. We would also like to thank Dr. Dan Davison for a careful reading of the manuscript and for helpful suggestions, and Dr. Patricia Reitemeier for manuscript preparation.

REFERENCES

1. An, Z. (1987). "HierTalker," Center for Nonlinear Studies (LANL) Technical Report: LALP 87-27.
2. Barnes, C., C. Burks, R. Farber, A. Lapedes, K. Sirevkin (work in progress).
3. Baum, E. and F. Wilezek (1988). "Supervised Learning of Probability Distributions by Neural Networks," D. Z. Anderson, ed., *Neural Information Processing Systems* (New York: American Institute of Physics Press).
4. Broome-Smith, J. K., A. Edelman, S. Yousif, and B. G. Spratt (1985). "The Nucleotide Sequences of the ponA and ponB Genes Encoding Penicillin Binding Proteins 1A and 1B of *Escherichia coli* K12," *Eur. J. Biochem.* **147**, 437-446.
5. Burks, C., J. W. Fickett, W. B. Goad, M. Kanehisa, F. I. Lewitter, W. P. Rindone, C. D. Swindell, C. S. Tung, and H. S. Bilofsky (1985), "The GenBank Nucleic Acid Sequence Database," *Comp. Applic. Biosci.* **1**, 225-233.
6. Cohen, F., R. Arbanek, I. Kuntz, R. Fletterick (1983), "Secondary Structure Assignment for Alpha/Beta Proteins by a Combinatorial Approach," *Biochem.* **22**, 4894-4904.
7. Coulton, J. W., P. Mason, D. R. Cameron, G. Carmel, R. Jean, and H. N. Rode (1986). "Protein Fusions of Beta-Galactosidase to the Ferrichrome Iron Receptor of *Escherichia coli* K 12," *J. Bacteriol.* **165**, 181-192.
8. DeLisi, C. (1988), "Computers in Molecular Biology: Current Applications and Emerging Trends," *Science* **240**, 47-52.
9. Denker, J., ed. (1986), *Proceedings of the Conference: "Neural Networks for Computing, Snowbird, UT"* (New York: American Institute of Physics Press).
10. Duda, R. O. and P. E. Hart (1973), *Pattern Classification and Scene Analysis* (New York: Wiley-Interscience).

11. Fickett, J. (1982). "Recognition of Protein Coding Regions in DNA Sequences," *Nucl. Acids Res.* **10**, 5303-5318.
12. Green, M. R. (1986). "Pre-mRNA Splicing," *Ann. Rev. Genet.* **20**, 671-708.
13. Holland, J. K. Holyoak, R. Nisbett, and P. Thogard (1986), *Induction: Processes of Inference, Learning and Discovery* (Cambridge, MA: MIT Press).
14. Hopfield, J. J. (1982). "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *PNAS* **79**, 2554-2558.
15. Hopfield, J. (1987), "Learning Algorithms and Probability Distribution in Feed-forward and Feedback Networks," *P.N.A.S.* **84**, 8429-8433.
16. *The Human Genome Project*, (1987), "Health and Environmental Safety Research Committee Advisory Report", D.O.E., Wash. D.C.
17. Jungck, J. R. and R. M. Friedman (1984), "Mathematical Tools for Molecular Genetics Data: An Annotated Bibliography," *Bull. Math. Biology* **46**, no.4, pp. 699-744. (An annotated bibliography.)
18. Kullback, S. (1959), *Statistics and Information Theory* (New York: J. Wiley and Sons).
19. Lachenbruch, P. A. and M. Goldstein (1979), "Discriminant Analysis," *Biometrics* **35**, 69-85.
20. Lapédes, A. and R. Farber (1987), "Genetic Database Analysis with Neural Networks" (abstract), *Proceedings of IEEE Conference. Neural Information Processing Systems: Natural and Synthetic* (New York: American Institute of Physics Press), p. 28.
21. Lee, Y. C. et. al. (1986), "Machine Learning Using a Higher Order Correlation Network," *Physica* **22D**, 276-306.
22. Lippmann, R. (1987), "Acoustics, Speech, and Signal Processing: An Introduction to Computing with Neural Nets," *IEEE ASSP Magazine* **4**, 4-22.

23. Mount, S. (1982), "A Catalogue of Splice Junction Sequences," *Nuc. Acids Res.* **10**, 459-472.
24. Nakata, K., M. Kaneisha, and C. DeLisi (1985), "Prediction of Splice Junctions in mRNA Sequences," *Nuc. Acids Res.* **13**, 5327-5340.
25. Omohundro, S. (1987), "Efficient Algorithms with Neural Network Behavior," *Complex Systems* **1**, 273-347.
26. Polack, S., H. Hicks, and W. Harrison (1971), *Decision Tables* (New York: Wiley-Interscience).
27. Press, W., B. Flannery, S. Teukolsky, and W. Vetterling (1986), *Numerical Recipes: The Art of Scientific Computing*, New York: Cambridge University Press).
28. Quinlan, J. (1986), "Induction of Decision Trees," *Machine Learning* **1**, 81-106.
29. Rumelhart, D. and J. McClelland (1986), *Parallel Distributed Processing*, Vol. 1, (Cambridge, MA: M.I.T. Press).
30. Sejnowski, T. J. and C. R. Rosenberg (1987), "Parallel Networks That Learn to Pronounce English Text," *Complex Systems* **1**, No. 1, 145-168.
31. Sellers, P. H. (1979), "Pattern Recognition in Genetic Sequences," *PNAS* **76**, 3041.
32. Shapiro, M. B. and P. Senapathy (1987), "RNA Splice Junctions of Different Classes of Eukaryotes: Sequence Statistics and Functional Implications in Gene Expression," *Nuc. Acids Res.* **15**, 7155-7174.
33. Staden, R. (1984), "Computer Methods to Locate Signals in Nucleic Acid Sequences," *Nucl. Acids Res.* **12**, 505-519.
34. Stormo, G. D., T. D. Schneider, L. Gold, and A. Ehrenfeucht (1982), "Use of the 'Perceptron' Algorithm to Distinguish Translational Initiation Sites in *E. coli*," *Nuc. Acids Res.* **10**, 2997-3011.

35. Trifinov, E. (1987). "Translation Framing Code and Frame-Monitoring Mechanism As Suggested by the Analysis of mRNA and 16 S rRNA Nucleotide Sequences," *J. Molec. Biol.* **194**, 643-652.
36. Waterman, D. (1978). *A Guide to Expert Systems* (New York: Addison-Wesley).

FIGURE CAPTIONS

Fig. 1. A first order neural network, with one of the input representations discussed in the text. Each symbol is represented by four bits. A string of symbols is represented by concatenating the four bits for each symbol in the string. If a codon representation, instead of a base representation were used, each symbol (codon) would be represented by sixty four bits. The line of circles represents input neurons, which take on values of 0 or 1. The output neuron state is determined by the input neuron's states according to a set of connection weights and after training is complete should attain a value near 1.0 if the input example comes from the "true" class; near 0.0 otherwise.

Fig. 2. A two-dimensional example of "true/false" classes that can not be separated by a line. In higher dimensions this would correspond to classes that could not be separated by a hyperplane.

Fig. 3. The Kullback information measure plotted as a function of base position for splice junction acceptor sites. This is the average number of bits of information in each base position that aid in discrimination of "true" from "false" sites.

Fig. 4. The Kullback information measure obtained for *E. coli* coding regions and no-coding regions using Eq. (6). The left most bar shows the information for each base position. The other bars show the contribution due to individual bases.

Fig. 5. Decision tree. The figure represents the first two levels of a five-level decision tree constructed to solve the problem of recognizing acceptor splice junctions. The tree is constructed so that each deeper level reflects an increasing number of relevant base positions. The learning algorithm tests all base positions at a node to decide which is most relevant at that level. (For example, to go from the pattern shown at the top right-hand of the figure, ...A...A..., to the next lower level, the remaining 19 base positions would be tested, and the position with the highest information content would be selected.) The 21 boxes each represent base locations in the sequence upon which decisions may be based. The set of 21 taken together represents one window, or sequence segment. The vertical line denotes the position of the potential intron/exon boundary with the potential intron to the left. The AG consensus dinucleotide is shown in lower case. Since this dinucleotide is present in both "True" and "False" data sets, its base positions are not considered relevant by the learning algorithm. The upper case letters shown in the figure represent the bases at the most meaningful positions chosen by the algorithm. The numbers of examples from the True and False sets

chosen by each decision rule are shown below the representation of each rule. At the first level the algorithm decided to partition on the eighth base. For example, when that base is an "A", the rule selected 9 true acceptor splice sites and 158 windows taken from non-splice-sites. Once the data are partitioned with an "A" in the eighth position, as shown at the top of the figure, the algorithm decides that the next most important position is the fourth position. When that is an "A," there are no actual examples from acceptor sites, but there are 28 examples taken from non-splice sites. In practice, the algorithms probe to deeper levels until either the true or false populations are either zero or acceptably small.

Fig. 6. Graphical display of results for locus HUMFBRG. The entire figure is divided into different sections. The top section displays scale information and an iconic display of the relevant GenBank FEATURES. Six rectangular boxes follow that contain information both about the output of the neural net and about the sequences themselves. Working from the top down, we first have the title, which reads "Max Like CodeNoCodeNoStops" because maximum likelihood minimization [Eq. (4c)] was used during training on 135 base long windows within ORFS. Directly under the title is the scale line, which contains ticks at 1 kilobase units, as documented at the left of the scale line itself. Under the ends of the scale line are numbers that represent the limits of the GenBank sequence that was analyzed (the entire sequence need not be analyzed). In this case the left limit was 0 and the right 10,564 since those are the boundaries for this GenBank entry. Directly under the scale line is a line that represents the sequence; above this line see HUMFBRG, which is the GenBank LOCUS name for the region that contains the sequence for the region containing the human fibrinogen gene. Under the line that represents the sequence there are three other types of icons. One type is a thick solid line, which represents the regions that code for proteins and are annotated as "pept" in GenBank. Additionally, there are right pointing and left pointing small triangles that represent splice donor and acceptor sites, respectively. Finally, between these icons and the line representing the sequence, there is a long thin line representing the extent of the RNA transcript. Under these icons, the beginning of the first word(s) from the feature description field in GenBank is displayed—in this case "fibrino" from "fibrinogen gamma chain" and "g-p g" from "g, g-p signal pept."

Each of the six boxes represents the output from the neural networks for different frames. The top three for the same frame as is in GenBank, the bottom three for the complementary strand. As the complementary strand is the reverse complement, 5' to 3' for transcribed RNA would be right to left instead of left to right.

In the left margin of each box are statistics comparing the output of the net for the Problem 1 coding discrimination to the annotation in GenBank. This is only relevant for ORFs that are bigger than the window size = 135 in this case. Additionally, windows are only compared to GenBank's calls if they are wholly within or outside of a known coding region; overlapping windows are not counted. The statistics are interpreted as follows: If in that frame there are any coding regions containing windows large enough, the percent of those correctly identified as coding by the net are scored as NN% +. Those windows in ORFS that do not code for protein and are correctly identified as not coding, are scored as NN% -.

To the left of each box are four numbers. The top number, 0.5, sets the threshold used to determine the minimum signal strength to identify an ORF as coding. The lower three define the positions for lines identifying the maximum signal strength from the net (1.0), the threshold mentioned above (0.5), and the minimum signal strength from the net (0.0).

The most obvious features in each box are shaded regions. The height of this region represents the strength of the signal from the net. Generally, signals above the 0.5 line are identified as coding, and those below are identified as not coding. The window size being finite and not a point source (135 in this case) causes a problem with representation. It is dealt with as follows. At the start of an ORF, the beginning of the window is marked (left on top three boxes, right on bottom three). Subsequently, the end of the window is marked, displaying the entire region presented to the net.

The triangles mark splice sites identified by the net; however, they are only displayed if they are above 0.5 signal strength and in an ORF that has at least one window above 0.4 coding strength. The highest donor and acceptor triangle representations in an ORF are solid; the others are only outlined.

At the top of each box there is additional information. The thick solid lines on the top line represent coding regions, **as annotated in GenBank**, and they are only displayed for the frame in which they would be coding. Directly under them are lines (with ends marked) that denote the extent of the ORFS that were analyzed. For example, in the left part of the figure in frame 0 there are three of these lines without any apparent signal from the net. This occurs because the net signal was so weak that the shaded areas are not visible.

Sometimes boxes appear under the line that shows the extent of the ORFS. These represent regions identified as coding by the net — the signal from the net was above the threshold.

In this example, most of the windows 135 bases long from coding ORFs were recognized - all of those in frame 0, and most from frame 1. The exons in frame 2 are less than 135 bases, so no positive statistic is generated. There are many windows that are not known to be coding that the net indicates have coding character. For example, see the extreme right-hand end of frame 2 where there is a region that looks like a coding region with a potential acceptor splice site that the net identifies as being usable. It is tempting to speculate that there may be a cell type in which alternative splicing causes this region to be expressed.

Decision Tree: Acceptor Splice Junctions

First Level

Second Level

Intron	A a g	Exon	True/False	9/158	A	A a g	True/False	0/28
					C	A a g	True/False	2/51
					G	A a g	True/False	0/52
					T	A a g	True/False	7/27
	C a g		True/False	479/215	A	C a g	True/False	27/44
					C	C a g	True/False	216/67
					G	C a g	True/False	23/62
					T	C a g	True/False	213/42
	G a g		True/False	7/210	A	G a g	True/False	0/64
					C	G a g	True/False	4/65
					G	G a g	True/False	0/42
					T	G a g	True/False	3/39
	T a g		True/False	110/22	A	T a g	True/False	13/8
					C	T a g	True/False	29/4
					G	T a g	True/False	11/7
					T	T a g	True/False	57/3

Table 0.10.007.000

No comment

0.000

1.000

HUMPHREY

0.000

0.000

0.000

0.000

0.000

0.000

0.000

Frame 0

1.0%+
3%-

Frame 1

8%+
6%-

Frame 2

4%-

Frame 0 Complement

7%-

Frame 1 Complement

8%-

Frame 2 Complement

9%-

96.4%+

86.0%+