

GeneScout: A Data Mining System for Predicting Vertebrate Genes in Genomic DNA Sequences

Michael M. Yin and Jason T. L. Wang

Department of Computer Science
New Jersey Institute of Technology
University Heights, Newark, NJ 07102, USA

ABSTRACT

Automated detection or prediction of coding sequences from within genomic DNA has been a major rate-limiting step in the pursuit of vertebrate genes. Programs currently available are far from being powerful enough to elucidate a gene structure completely. In this paper, we present a new system, called GeneScout, for predicting gene structures in vertebrate genomic DNA. The system contains specially designed hidden Markov models (HMMs) for detecting functional sites including protein-translation start sites, mRNA splicing junction donor and acceptor sites, etc. An HMM model is also proposed for exon coding potential computation. Our main hypothesis is that, given a vertebrate genomic DNA sequence S , it is always possible to construct a directed acyclic graph G such that the path for the actual coding region of S is in the set of all paths on G . Thus, the gene detection problem is reduced to that of analyzing the paths in the graph G . A dynamic programming algorithm is used to find the optimal path in G . The proposed system is trained using an expectation-maximization (EM) algorithm and its performance on vertebrate gene prediction is evaluated using the 10-way cross-validation method. Experimental results show that the proposed system performs well and is comparable to existing gene discovery tools.

Keywords: Bioinformatics, Gene finding, Hidden Markov models, Knowledge discovery, Soft computing

1. INTRODUCTION

Data mining, or knowledge discovery from data, refers to the process of extracting interesting, non-trivial, implicit, previously unknown and potentially useful information or patterns from data.¹⁴ In life sciences, this process could refer to finding clustering rules for gene expressions, discovering classification rules for proteins, detecting associations

Corresponding author: J.T.L. Wang. Email: wangj@njit.edu, Tel: (973) 596-3396, Fax: (973) 596-5777.

between metabolic pathways, predicting genes in genomic DNA sequences, etc.³³⁻³⁶ This paper presents a data mining system for automated gene discovery.

A genomic DNA sequence is comprised of four types of nucleotides, or bases, represented by English letters A, C, G, and T. Identification or prediction of coding regions from within a genomic DNA sequence has been a major rate-limiting step in the pursuit of genes. The Human Genome Project has produced millions of nucleotides of sequences, and it becomes increasingly important to rapidly identify genes in these sequences.²⁶ The basic structure for a vertebrate gene includes a promoter, a start codon, introns, exons, donors, acceptors, and a stop codon; cf. Figure 1(A). The exon sequences of a gene are also called the *coding sequences* of this gene, and the whole exon sequences of a gene are called the *coding region* of the gene (which is the region for making proteins); cf. Figure 1(B). Intron sequences range in size from about 80 nucleotides to 10,000 nucleotides or more. Introns in genes have no function at all and are actually the genetic “junk”.^{1,4} They differ dramatically from exons in that their exact nucleotide sequences seem to be unimportant. The only highly conserved sequences in introns are those required for intron removal from DNA.

Biologists study gene structures based on lab experiments such as PCR on cDNA libraries, Northern blot, sequencing, etc. However, characterizing the 60,000 to 100,000 genes thought to be hidden in the human genome by means of merely lab experiments is not feasible. A current trend is to complement the lab study with bioinformatics approaches. The bioinformatics approach for gene detection means using computer programs to elucidate a gene structure from DNA sequence signals, including start codons, splicing junction donor sites and acceptor sites, stop codons, etc.

A number of programs have been developed for locating gene coding regions (exons), with varying degrees of success.^{7,26} However, the vertebrate DNA sequence signals involved in gene determination are usually ill defined, degenerate and highly unspecific. Given the current detection methods, it is usually impossible to distinguish the signals truly processed by the cellular machinery from those that are apparently non-functional.¹² Furthermore, the inherent conservatism of the currently popular methods such as similarity search, GRAIL, etc. greatly limits our ability for making unexpected biological discoveries from increasingly abundant genomic data. Except for a very limited subset of trivial cases, the automated interpretation without experimental validation of genomic data is still a myth.⁸ Unlike the situations in bacteria and yeast organisms, in which computer systems have substantially contributed to the automatic analysis of genomes, automatic sequence analysis and structure elucidation for the genomes of vertebrate organisms are far from being a reality.¹²

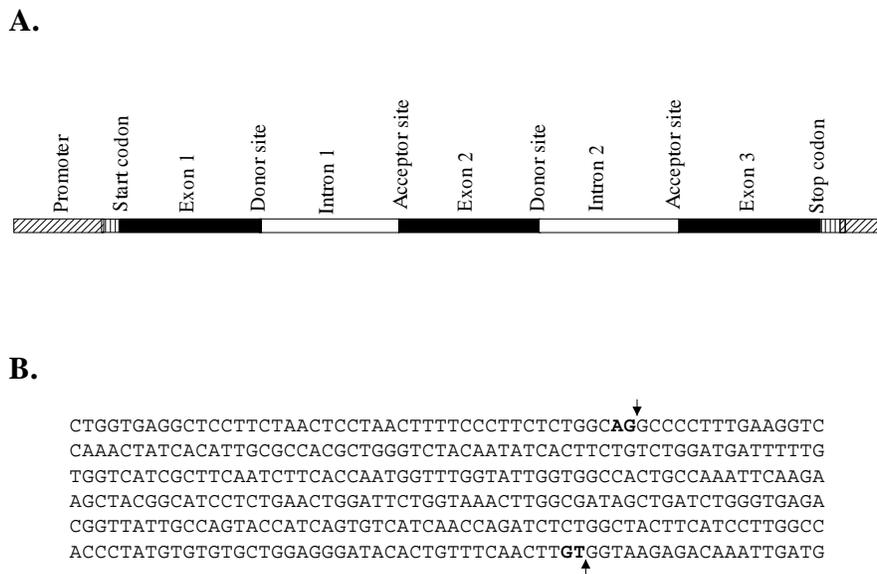


Figure 1. (A) A vertebrate gene structure. (B) A sequence fragment containing an exon of 296 nucleotides. The AG bases preceding the first arrow are the conserved nucleotides in a splicing junction acceptor site. The GT bases preceding the second arrow are the conserved nucleotides in a splicing junction donor site. The nucleotides between the two arrows constitute the exon.

Our research is targeted toward developing effective and accurate methods for automatically detecting gene structures in the genomes of high eukaryotic organisms. We have developed effective hidden Markov models for detecting splicing junction sites in vertebrate genomic DNA sequences.³⁹⁻⁴³ In this paper, we combine the gene structure signal information with global gene structure information and present GeneScout, a full-scale gene structure detection system.

The rest of the paper is organized as follows. Section 2 surveys related work. Section 3 presents our approach to gene discovery. Section 4 reports experimental results. Section 5 concludes the paper.

2. RELATED WORK

Although methods for predicting coding regions in genomic DNA sequences had existed since the 1980s, the programs for assembling coding sequences into translatable mRNA sequences were not available until the early 1990s.⁷ Recently there have been several programs available for biologists, such as GenViewer,²³ GeneID,¹³ GenLang,⁹ GeneParser,²⁷ FGENEH,²⁸ SORFIND,¹⁷ Xpound,²⁹ GRAIL,³⁷ VEIL,¹⁶ GenScan,⁶ etc. Among the tools, GRAIL and GenScan are widely used in academia and industry. GRAIL is available on the BLAST Web site (<http://www.ncbi.nlm.nih.gov>) and GenScan is available at MIT's Web site (<http://genes.mit.edu/license.html>). Algorithms employed by these various tools are based on *consensus search*,¹¹ *weight matrices*,¹⁹ *pattern recognition*,¹¹ etc., though the most popular approaches are based on *neural networks* (NNs)³⁷ and, more recently, *hidden Markov models* (HMMs).^{20,25,43} We describe below basic algorithms using NN and HMM techniques.

2.1. NN-Based Techniques

A learning sample of NN-based techniques consists of two classes: sites and non-sites. The non-sites class is usually formed by randomly choosing fragments from the natural DNA. The basic steps of the NN-based techniques are as follows^{21,22,31,32}:

1. Creation of a learning sample;
2. Choice and encoding of signal features;
3. Iterative correction of recognition rules according to results of discrimination between the two classes obtained in the previous round;
4. Testing on an independent sample.

Often, an NN consists of a layer of input neurons, several layers of hidden neurons, and an output neuron. When an unlabeled candidate site is presented to the NN, the input neurons check whether the site possesses the corresponding features and send binary signals to the neurons of the first hidden layer. Each hidden neuron sums the weighted signals coming, by connection, from lower-level neurons, compares the result with some thresholds, and sends a binary signal to upper-level neurons. The output neuron provides the user with a final site/non-site decision. Several research groups implemented NNs for gene structure prediction, and they adopted two different techniques in combination with NNs¹¹: (i) the use of *ad hoc* heuristic procedures and (ii) the use of combinatorial algorithms.

The first technique was adopted in GRAIL.^{30,37} The exon recognition module of GRAIL employs an NN that integrates values of various coding potentials and other statistics. The first step is to perform splicing site prediction by similar multisensor networks. The second step is to perform exon prediction by Bayesian analysis of frame-specific coding potentials. Finally, all legitimate combinations of predicted exons are used for gene assembly. At this step, GRAIL employs a heuristic procedure that scores candidate exons using some combination of the site scores and the coding potentials obtained in the previous steps. Low-scoring exons are eliminated from further consideration. The software then performs an exhaustive search over the set of structures generated by the remaining high-scoring exons.

GeneParser²⁷ is an example of using the second technique in which dynamic programming is applied to get rid of the exhaustive search over exon-intron structures. GeneParser combines the classical dynamic programming with an NN, and uses both coding potentials and weight matrix scores of splicing sites for exon prediction. The learning stage consists of a series of iterations. At each iteration, a dynamic programming procedure is applied to find the highest-scoring structures for all sequences in the learning sample. Then another dynamic programming module is used to find the optimal structures for new sequences.¹¹

2.2. HMM-Based Techniques

HMMs have been used extensively to describe sequential data or processes such as speech recognition. An HMM model is a process in which some of the details of the sequential data are unknown, or *hidden*. A general description of a Markov model is that it models a stochastic process using a number of states and probabilistic state transitions. An HMM is usually represented by a graph where the states correspond to vertices and the transitions correspond to edges in the graph. Each state s is associated with a discrete output probability distribution, $P(s)$. Similarly, each transition has a probability, which represents the probability that a generating process makes that transition.

Thus the sum of the probabilities of all the transitions from a given state s to all other states must be 1.

HMMs have been remarkably successful in the field of speech recognition,¹⁸ where they are used in most state-of-the-art systems. Researchers in computational biology have recently started to use HMMs for biological sequence analysis. Lukashin, Borodovsky²⁰ and their colleagues⁵ successfully applied HMMs to the detection of coding regions in prokaryotes. Audic and Claverie² reported their use of Markov transition matrices to detect eukaryotic promoters. Salzberg²⁵ has used HMMs for identifying splice sites and translational start sites in eukaryotic genes, and those splice site models have been adopted in the VEIL system¹⁶ for finding vertebrate genes. Salzberg's group also developed an interpolated Markov model to find genes in malaria parasite *Plasmodium falciparum*, a small eukaryotic species.²⁶

3. OUR APPROACH

Our proposed GeneScout system contains several specially designed HMM models for predicting functional sites as well as an HMM model for calculating coding potentials. The functional sites are some sequence signals common for all sites of a given type, which are recognized by corresponding DNA- and RNA-binding proteins.¹¹ Basic units of such functional sites in a vertebrate gene sequence include translational start sites, splicing junction donor and acceptor sites, etc. Effectively predicting these functional sites is the first and crucial step for gene finding.

3.1. HMM Models for Predicting Functional Sites

Often, the functional sites include (almost) invariant (consensus) nucleotides and other degenerate features. Thus, the invariant nucleotides themselves do not completely characterize a functional site. For example, a start codon is always a sequence of **ATG** and it is the start position in mRNA for protein translation, so the start codon is the first 3 bases of the coding region of a gene. **ATG** is also the codon for Methionine,* a regular amino acid occurring at many positions in all of the known proteins. This means one is unable to detect the start codon by simply searching for **ATG** in a genomic DNA sequence.

It is reported that there are some statistic relations between a start codon **ATG** and the 13 nucleotides immediately preceding it and the 3 bases immediately following it.²⁵ We call these 19 bases containing a start codon a *start site*. We build an HMM model, called the Start Site Model, to model the start site. Figure 2 illustrates the model. As shown in Figure 2, there are 19 states for the Start Site Model. Except for states 14, 15 and 16, there are four

*A codon is a triplet of contiguous bases in mRNA that code for specific amino acids, which in turn are used for building proteins.³⁵

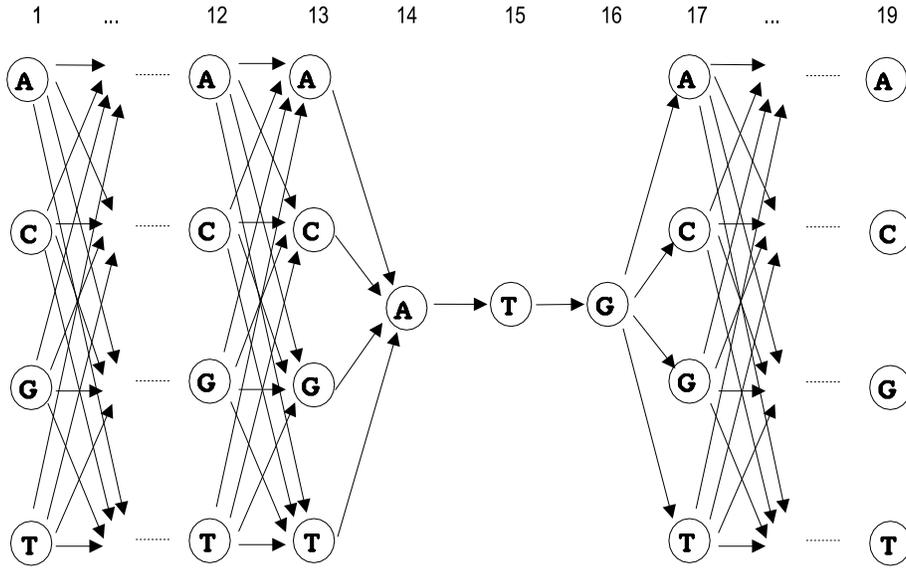


Figure 2. The Start Site Model.

possible bases at each state, and a base at one state may have four possible ways to transit to the next state. States 14, 15 and 16 are constant states (representing a start codon), and the transitions from state 14 to 15 and from state 15 to 16 are also constant with a probability of 1. With the Start Site Model, we can use the HMM algorithms described in our previously published paper⁴³ to detect a start site.[†] The topology of our HMM models is different from those of other models^{6,16,25} designed for functional site detection in gene prediction systems. To capture the consensus and degenerate features of the functional sites, we introduced constant states and constant state transitions into the hidden Markov models. This innovative approach conceptually simplifies the functional site models and the computational process of using the models.^{38,43}

[†]In the previously published paper,⁴³ we presented HMM models and algorithms for detecting splicing junction donor and acceptor sites. The HMM model for a donor site contains 9 states whereas the HMM model for an acceptor site contains 16 states. The algorithms used for training the HMM models for start sites, donor sites and acceptor sites and for detecting these functional sites are similar. Please see related publications^{38,43} for details.

3.2. Graph Representation of the Gene Detection Problem

The goal of GeneScout is to find coding regions as illustrated in Figure 1. Like all other major gene-finding systems surveyed in Section 2, GeneScout does not find the promoter at the very beginning of a gene structure as well as the beginning or the end of transcription. A more accurate term for this process might be “coding region detection”, though traditionally the term “gene detection” has been used when referring to these systems.²⁵

The main hypothesis used in our work is that, given a vertebrate genomic DNA sequence S , it is always possible to construct a directed acyclic graph G such that the path for the actual coding region of S is in the set of all paths on G . Thus, the gene detection problem is reduced to the analysis of paths in the graph G . We use a dynamic programming algorithm to find the optimal path in G .

We define a *candidate exon* to be a sequence fragment whose left boundary is an acceptor site or a start codon, and whose right boundary is a donor site or a stop codon. A *candidate intron* is a sequence fragment with a donor site at its left boundary and an acceptor site at its right boundary. We define a *candidate gene* as a chain of non-intersecting alternating exons and introns that satisfy the following biological consistency conditions:²⁴

1. the total length of exons is divisible by 3;
2. there are no in-frame stop codons in exons;
3. the first intron-exon boundary is a start codon, and the last exon-intron boundary is a stop codon.

However, like most of the existing gene detection programs, our algorithm can be easily generalized for incomplete genes violating condition 3 and possibly condition 1.

Referring to Figure 1 again, if one could detect the start codon and all the splicing junction donor and acceptor sites correctly, the coding region would be found immediately. Unfortunately, there is no program that can correctly and precisely detect the start codon and all the splicing junction sites without any error. In our case, even though we have developed effective hidden Markov models for detecting splicing junction sites, there are still false positives mistaken as donor or acceptor sites.⁴³ Suppose we start with a vertebrate genomic DNA sequence on which we mark the positions for the start codons, donor sites and acceptor sites that are detected by our HMM algorithms.⁴³ Then these sites generate a set of candidate exons and candidate introns, and their combinations form a set of candidate genes. If we assume all the true functional sites have been detected, one (and only one) subset of the candidate exons must constitute the real coding region.

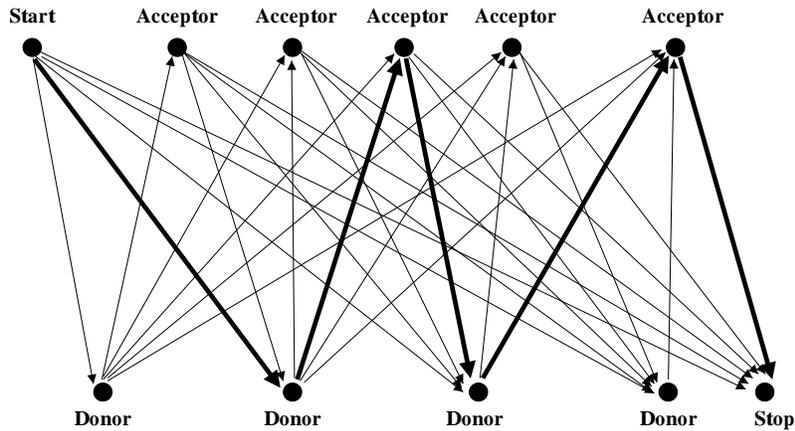


Figure 3. A site graph for gene detection with the boldface edges representing real exons-introns.

Consider a directed acyclic graph G where vertices are functional sites, and edges are exons and introns (Figure 3). All the edges from the top vertices to the bottom vertices in the graph G are candidate exons, and the edges from the bottom vertices to the top vertices are candidate introns. There must be a path on G representing real exons-introns as shown by the boldface edges in Figure 3. So, given a vertebrate genomic DNA sequence with detected sites, it is always possible to construct a directed acyclic graph G such that the path for real exons-introns is in the set of all paths on G . Thus the gene detection problem is reduced to the analysis of paths in the graph G .

3.3. A Dynamic Programming Algorithm

Consider again the graph G in Figure 3. A candidate gene is represented by a path in G . Let S_G denote the set of all paths in G . We assign a score to each functional site based on the HMM models and algorithms described in Section 3.1.^{38,43} The score is used as the weight of the corresponding vertex v in S_G , and we denote that weight as $W(v)$. We associate each edge (v_1, v_2) in S_G with a weight $W(v_1, v_2)$. The weight $W(v_1, v_2)$ equals to the coding

potential of the candidate exon or intron corresponding to the edge (v_1, v_2) (the calculation of the coding potential will be described in Section 3.4). Let p be a path in S_G ; the weight of p is denoted as $W(p)$.

Now, let v be a vertex in S_G and let $(v_1, v), \dots, (v_k, v)$ be all edges entering v . Let $S(v)$ be the set of all paths entering the vertex v . Let \odot represent the concatenating operation, and \oplus represent the simple addition. We can calculate the weight of the optimal path in $S(v)$, denoted $\Theta(S(v))$, as follows:

$$\Theta(S(v)) = \max_{p \in S(v)} W(p) = \max_{i=1}^k \left(\max_{p \in S(v_i)} W(p \odot (v_i, v)) \right) \quad (1)$$

$$= \max_{i=1}^k \left(\left(\max_{p \in S(v_i)} W(p) \right) \oplus W(v_i) \oplus W(v_i, v) \right) \quad (2)$$

$$= \max_{i=1}^k (\Theta(S(v_i)) \oplus W(v_i) \oplus W(v_i, v)) \quad (3)$$

This recurrence formula can be used for computing $\theta(S(v))$ given the set of weights $\theta(S(v_i))$, $i = 1, \dots, k$. Thus a dynamic programming algorithm can be used to find the weight of the optimal path and locate the path itself in the graph G . This path indicates the real exons (coding region) in the given genomic DNA sequence.

3.4. An HMM Model for Computing Coding Potentials

Because vertebrate genes have coding regions and noncoding regions, *coding potential* is used here to measure the difference in statistical characteristics between coding and noncoding regions. Our approach for computing coding potentials is based on the analysis of codon usage, which reflects the following phenomena¹¹: The universal structure of the genetic code, the average amino acid composition of proteins, the genome-specific patterns of the usage of synonymous codons, and genes intend to use preferred codons in the coding regions. We develop a Codon Model as the basic unit for calculating coding potentials.

Figure 4 shows an HMM with 3 states and a set of transitions used for modeling a codon in a vertebrate gene. The HMM is represented as a digraph where states correspond to vertices and transitions to edges. At each state, the HMM generates a base b in $\{A, C, G, T\}$ according to the state and transition probabilities. Notice that, if the HMM generates base $b = T$ in the first state, and generates base $b = A$ or $b = G$ in the second state, the third state can only be $b = C$ or $b = T$. In other words, if the first state is base T , the transitions (from state 2 to state 3) $A \rightarrow A$, $A \rightarrow G$, $G \rightarrow A$ and $G \rightarrow G$ are not defined. This is because TAA , TAG , TGA and TGG are stop codons.

Let $ftr(b_i, b_{i+1})$ be the state transition probability from state i to state $i + 1$ (the calculation of $ftr(b_i, b_{i+1})$ will be described in Section 3.5). Let U_{codon}^j be the codon usage probability for a codon starting at position j in a coding

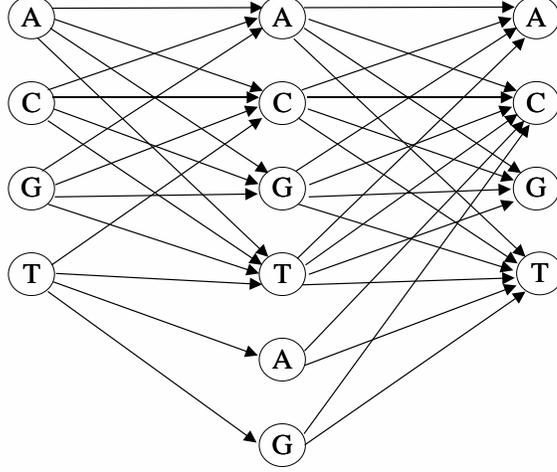


Figure 4. The Codon Model.

region. We can write:

$$U_{codon}^j = \prod_{i=j}^{j+1} ftr(b_i, b_{i+1}) \quad (4)$$

Let P_{coding}^m be the coding potential for a sequence with m bases. Then the coding potential for the sequence can be calculated as follows:

$$P_{coding}^m = \sum_{j=1,4,7,\dots}^{m-2} U_{codon}^j \quad (5)$$

In this study, we use Equation (5) to calculate the coding potentials for candidate exons, which are represented by the edges starting at the top vertices and ending at the bottom vertices in the site graph G shown in Figure 3. Notice that if the transition from state i to $i + 1$ does not exist, $ftr(b_i, b_{i+1})$ is not defined. If $ftr(b_i, b_{i+1})$ is not defined, U_{codon}^j is not defined, neither is P_{coding}^m . This means that, given a candidate exon represented as an

edge in the graph G shown in Figure 3, if there are any undefined state transitions, that candidate exon has no coding potential and it is not a real exon. A coding potential value of 0 is assigned to all candidate introns that are represented by the edges starting at the bottom vertices and ending at the top vertices in the site graph G shown in Figure 3.

3.5. Training and Predicting Algorithms

Let M represent the set of sequences that are randomly picked from a training data set and let E be the set containing the remaining sequences in the training data set. (In the study presented here, M contains about 10% of the sequences in our training data set.) The Codon Model described in Section 3.4 is trained using an expectation maximization (EM)³ algorithm with the exons in the set E . The topology for the Codon Model is fixed, and all the transition probabilities and state probabilities are initialized to random values. We pick one tenth of the sequences from E and input them into the Codon Model. We record the number of the individual bases at each state and the number of individual transitions from one state to another state. We then compute the post probabilities for all the states and transitions in the Codon Model.

Let $Ttr(b_i, b_{i+1})$ be the total number of transitions from a base b_i at state i to a base b_{i+1} at state $i + 1$ in the Codon Model. Let T_{in} be the total number of codons that have been input into the model. The state transition probability $ftr(b_i, b_{i+1})$ in the model can be calculated as follows:

$$ftr(b_i, b_{i+1}) = \frac{Ttr(b_i, b_{i+1})}{T_{in}} \quad (6)$$

The re-estimation procedure then adjusts all of the probabilities hidden in the Codon Model. Newly picked sequences in E are then run through the model and the probabilities are further refined. This process is iterated until $ftr(b_i, b_{i+1})$ is maximized or until E becomes empty. This algorithm is guaranteed to converge to a locally optimal estimate of all the probabilities in the HMM.¹⁶

Next, we treat all the sequences in the set M as unlabeled sequences and input each of the sequences into the HMMs described in Section 3.1 to detect start sites, donor sites and acceptor sites. Then the training algorithm constructs the site graph G shown in Figure 3 for an input sequence with the detected functional sites on it. The coding region for the input sequence is detected by dynamic programming as specified in Equation (3). Comparing the detected coding region with the known gene structures for each of the sequences in M , we obtain the approximation correlation (AC) value introduced by Burset and Guigo⁷ (the calculation of the AC value will be described in detail

in Section 4.2). In a nutshell, the AC value is the measure that summarizes the prediction accuracy at the nucleotide level. AC ranges from -1 to 1. A value of 1 corresponds to a perfect prediction, while -1 corresponds to a prediction in which each coding nucleotide is predicted as a non-coding nucleotide, and vice versa. A value of 0 is expected for a random prediction.¹² The training process is iterated until the largest possible AC value is obtained.

In the testing (prediction, respectively) phase where an unlabeled test (new, respectively) sequence S is given, GeneScout first detects the functional sites on S and then builds a directed acyclic graph G using the detected functional sites as vertices. Next, GeneScout finds the optimal path on G and outputs the vertices (functional sites) and edges on the optimal path, which displays the coding region on S .

4. EXPERIMENTS AND RESULTS

4.1. Data

In evaluating the accuracy of the proposed GeneScout system for detecting vertebrate genes, we adopted the database of human DNA sequences originally collected by Burset and Guigo.⁷ The authors used this database to compare a number of gene-finding programs. The sequences in this database were obtained from the vertebrate divisions of GenBank release 85.0 (October, 1994). There are 570 vertebrate sequences in the database and they all have simple and standard gene structures. Each entry contains a complete coding sequence with no in-frame stop codons. There are 28,992,149 nucleotides in these 570 sequences, and there are 2,649 exons, corresponding to 444,498 coding nucleotides. There are at least one exon and one intron in each entry in the database. All the functional sites mentioned in the paper that appear in these sequences are standard sites. This means that all the start sites have ATG as the start codon, and all the donor sites have GT and all the acceptor sites have AG at appropriate positions.^{38,43} This database now becomes the standard data set for evaluating gene-finding programs.

We applied the 10-way cross-validation method^{40,43} to evaluating how well GeneScout performs when tested on sequences that are not in the training data set. Cross validation is a standard experimental technique for determining how well a classifier performs on unseen data.¹⁶ Specifically, we randomly partition the 570 sequences at hand into 10 sets. These sets have roughly the same number of exons and introns. For each run in the 10-way cross-validation experiment, we use nine out of the ten sets as the training data set, and use the remaining one set as the test data set. The GeneScout system is trained using the training data set (i.e., all sequences excluding those in the test data set are used as the training data) and then is tested on the sequences in the test data set. Thus, for each run, the

training data set contains 90% of the total exons and the test data set contains 10% of the exons. Notice that each of the 570 sequences is used exactly once in the test data set.

4.2. Results

Table 1 shows the results obtained in each run of cross-validation, and the average over all the ten runs. We estimate the prediction accuracy at both the nucleotide level and the exon level. At the nucleotide level, let TP_c be the number of true positives, FP_c be the number of false positives, TN_c be the number of true negatives, and FN_c be the number of false negatives. A *true positive* is a coding nucleotide that is correctly predicted as a coding nucleotide. A *false positive* is a non-coding nucleotide that is incorrectly predicted as a coding nucleotide. A *true negative* is a non-coding nucleotide that is correctly predicted as a non-coding nucleotide. A *false negative* is a coding nucleotide that is incorrectly predicted as a non-coding nucleotide. The sensitivity (S_c^n) and specificity (S_c^p) at the nucleotide level described in Table 1 are defined as follows:

$$S_c^n = \frac{TP_c}{TP_c + FN_c} \quad (7)$$

$$S_c^p = \frac{TP_c}{TP_c + FP_c} \quad (8)$$

As mentioned before, the approximation correlation (AC) is the measure that summarizes the prediction accuracy at the nucleotide level. AC ranges from -1 to 1. A value of 1 corresponds to a perfect prediction, while -1 corresponds to a prediction in which each coding nucleotide is predicted as a non-coding nucleotide, and vice versa. Formally, AC is defined as follows⁷:

$$AC = \left[\frac{1}{4} \left(\frac{TP_c}{TP_c + FN_c} + \frac{TP_c}{TP_c + FP_c} + \frac{TN_c}{TN_c + FP_c} + \frac{TN_c}{TN_c + FN_c} \right) - 0.5 \right] \times 2 \quad (9)$$

$$= \frac{1}{2} \left(\frac{TP_c}{TP_c + FN_c} + \frac{TP_c}{TP_c + FP_c} + \frac{TN_c}{TN_c + FP_c} + \frac{TN_c}{TN_c + FN_c} \right) - 1 \quad (10)$$

At the exon level, let TP_e be the number of true positives, FP_e be the number of false positives, TN_e be the number of true negatives, and FN_e be the number of false negatives. A *true positive* is an exon that is correctly predicted as an exon. A *false positive* is a non-exon that is incorrectly predicted as an exon. A *true negative* is a non-exon that is correctly predicted as a non-exon. A *false negative* is an exon that is incorrectly predicted as a non-exon. The sensitivity (S_e^n) and specificity (S_e^p) at the exon level described in Table 1 are defined as follows:

$$S_e^n = \frac{TP_e}{TP_e + FN_e} \quad (11)$$

<i>Run</i>	<i>Nucleotide</i>			<i>Exon</i>	
	S_c^n	S_c^p	AC	S_e^n	S_e^p
1	0.86	0.78	0.77	0.51	0.49
2	0.85	0.79	0.77	0.50	0.48
3	0.86	0.80	0.78	0.52	0.50
4	0.85	0.78	0.75	0.49	0.51
5	0.87	0.78	0.78	0.53	0.48
6	0.85	0.79	0.77	0.53	0.49
7	0.84	0.80	0.77	0.52	0.49
8	0.87	0.77	0.76	0.49	0.47
9	0.86	0.78	0.77	0.51	0.48
10	0.86	0.80	0.77	0.52	0.50
Average	0.86	0.79	0.77	0.51	0.49

Table 1. Performance evaluation of the proposed GeneScout system for gene detection.

$$S_e^p = \frac{TP_e}{TP_e + FP_e} \quad (12)$$

The result in Table 1 shows that, on average, GeneScout can correctly detect 86 percent of the coding nucleotides in the test data set. Among the predicted coding nucleotides, 79 percent are real coding nucleotides. At the exon level, GeneScout achieved a sensitivity of 51 percent and a specificity of 49 percent. This means GeneScout can detect 51 percent of exons in the test data set with both of their 5' and 3' ends being exactly correct.

Table 2 compares GeneScout with other gene finding tools on the same 570 vertebrate genomic DNA sequences. The performance data for the other tools shown in the table are taken from the paper authored by Buset and Guigo⁷ except for the VEIL system and GenScan. For VEIL, the data is taken from the paper authored by Henderson *et al.*¹⁶ For GenScan, the data is published by Burge and Karlin.⁶

It can be seen from Table 2 that GeneScout beats or is comparable to these other programs except GenScan. GenScan is more accurate than GeneScout at both the nucleotide level and the exon level. However, as indicated by GenScan's inventors Burge and Karlin,⁶ many of the 570 sequences collected by Buset and Guigo⁷ were used to

<i>System</i>	<i>Nucleotide</i>			<i>Exon</i>	
	S_e^n	S_e^p	AC	S_e^n	S_e^p
GeneScout	0.86	0.79	0.77	0.51	0.49
VEIL	0.83	0.72	0.73	0.53	0.49
FGENEH	0.77	0.88	0.78	0.61	0.64
GeneID	0.63	0.81	0.67	0.44	0.46
GeneParser 2	0.66	0.79	0.67	0.35	0.40
GenLang	0.72	0.79	0.69	0.51	0.52
GRAIL 2	0.72	0.87	0.75	0.36	0.43
SORFIND	0.71	0.85	0.73	0.42	0.47
Xpound	0.61	0.87	0.68	0.15	0.18
GenScan	0.93	0.90	0.91	0.78	0.81

Table 2. Performance comparison between GeneScout and other systems for gene detection.

train the GenScan system. This means that a portion of the test sequences were used in GenScan’s training process. In contrast, GeneScout is tested on the sequences that are completely unseen in the training phase. When examining the experimental results in more detail, we found that GenScan did not correctly predict any coding nucleotide on eight sequences of the 570 sequences. In contrast, for these eight sequences, GeneScout correctly detected about 85% of the coding nucleotides on them.

Furthermore, GeneScout runs much faster than GenScan. For example, it takes 0.8 seconds for GeneScout to predict the gene structure on a 5 kb sequence, while GenScan needs several seconds for the same sequence. The time complexity for the GeneScout algorithm is $O(NV^2)$ where N is the length of the input sequence and V is the number of vertices on the site graph constructed during the gene predicting process. In practice, the running time grows approximately linearly with the sequence length for sequences of several kb or more. The wall clock time for the GeneScout program when run on an X kb sequence on a Sun Sparc10 workstation is about $0.1 \times (X + 3)$ seconds.

5. CONCLUSIONS

In this paper, we have presented the GeneScout data mining system for detecting gene structures in vertebrate genomic DNA. GeneScout uses hidden Markov models to detect functional sites, including start codon sites, splicing junction donor sites and acceptor sites. Our main hypothesis is that, given a vertebrate genomic DNA sequence S , it is always possible to construct a directed acyclic graph G such that the path for the actual coding region of S is in the set of all paths on G . Thus, the gene detection problem is reduced to the analysis of paths in the graph G . A dynamic programming algorithm is employed by GeneScout to find the optimal path in G . The system is trained using an expectation maximization algorithm, and its performance on vertebrate gene detection is evaluated using the 10-way cross-validation method on the data set collected by Buset and Guigo.⁷

Our experiment results show that, GeneScout can correctly detect 86% of the coding nucleotides in the data set with 79% of detected coding nucleotides being correct. The approximation correlation value is 0.77 in predicting coding nucleotides. At the exon level, GeneScout achieves 51% sensitivity and 49% specificity. This means that GeneScout can detect 51% of exons in the data set with both 5' and 3' ends being exactly correct. It was also shown experimentally that GeneScout is comparable to existing gene discovery tools.

Future work includes the incorporation of more parameters or criteria into GeneScout. One source of possible new parameters could be obtained from the analysis of potential coding regions, such as preferred exon and intron lengths,¹⁵ and positions of exon-intron junctions relative to the reading frame.¹⁰ We may also model more functional sites such as those in the upstream or downstream of a coding region. These efforts will further improve GeneScout's performance to make it more accurate for vertebrate gene detection.

Acknowledgment

We thank the anonymous reviewers for their critical comments, which helped to improve the paper. This work was supported in part by the U.S. National Science Foundation under Grant No. IIS-9988636.

REFERENCES

1. B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J. D. Watson, *Molecular Biology of the Cell*, 3rd ed. Garland Publishing, Inc., New York and London, 1989.
2. S. Audic and J. Claverie, "Detection of eukaryotic promoters using Markov transition matrices," *Computers and Chemistry*, 21:223-227, 1997.

3. T. L. Bailey, M. E. Baker, and C. Elkan, "An artificial intelligence approach to motif discovery in protein sequences: Application to steroid dehydrogenases," *J. Steroid Biochemistry*, 62(1):29–44, 1997.
4. J. J. W. Baker and G. E. Allen, *The Study of Biology*, 4th ed. Addison-Wesley Publishing Company, Inc., 1982.
5. M. Borodovsky and J. McIninch, "GENMARK: Parallel gene recognition for both DNA strands," *Computers and Chemistry*, 17:123–133, 1993.
6. C. Burge and S. Karlin, "Prediction of complete gene structures in human genomic DNA," *J. Mol. Biol.*, 268:78–94, 1997.
7. M. Burset and R. Guigo, "Evaluation of gene structure prediction programs," *Genomics*, 34(3):353–367, 1996.
8. J. Claverie, O. Poirot, and F. Lopez, "The difficulty of identifying genes in anonymous vertebrate sequences," *Computers and Chemistry*, 21(4):203–214, 1997.
9. S. Dong and G. D. Stormo, "Gene structure prediction by linguistic methods," *Genomics*, 23:540–551, 1994.
10. A. Fedorov, G. Suboch, and L. Fedorova, "Analysis of nonuniformity in intron phase distribution," *Nucleic Acids Research*, 20:2553–2557, 1992.
11. M. S. Gelfand, "Prediction of function in DNA sequence analysis," *Journal of Computational Biology*, 2(1):87–115, 1995.
12. R. Guigo, "Computational gene identification: An open problem," *Computers and Chemistry*, 21(4):215–222, 1997.
13. R. Guigo, S. Knudsen, N. Drake, and T. F. Smith, "Prediction of gene structure," *Journal of Molecular Biology*, 226:141–157, 1995.
14. J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, San Francisco, California, 2000.
15. J. D. Hawkins, "A survey on intron and exon lengths," *Nucleic Acids Research*, 11:9893–9905, 1988.
16. J. Henderson, S. Salzberg, and K. H. Fasman, "Finding genes in DNA with a hidden Markov model," *Journal of Computational Biology*, 4(2):127–141, 1997.
17. G. B. Hutchinson and M. R. Hayden, "The prediction of exons through an analysis of spliceable open reading frames," *Nucleic Acids Research*, 20:3453–3462, 1992.
18. K. F. Lee, *Automatic Speech Recognition: The Development of the SPHINX System*. Kluwer Academic, Boston, Massachusetts, 1989.

19. Y. Lida, "DNA sequences and multivariate statistical analysis . Categorical discrimination approach to 5' splice site signals of mRNA precursors in higher eukaryotes' genes," *Comput. Appl. Biosci*, 3:93–98, 1987.
20. A. V. Lukashin and M. Borodovsky, "GeneMark.hmm: New solutions for gene finding," *Nucleic Acids Research*, 26(4):1107–1115, 1998.
21. Q. Ma and J. T. L. Wang, "Biological data mining using Bayesian neural networks: A case study," *International Journal on Artificial Intelligence Tools*, 8(4):433–451, 1999.
22. Q. Ma, J. T. L. Wang, D. Shasha, and C. H. Wu, "DNA sequence classification via an expectation maximization algorithm and neural networks: A case study," *IEEE Transactions on Systems, Man, and Cybernetics*, Part C: Applications and Reviews, 31(4):468–475, 2001.
23. L. Milanesi, N. A. Kolchanov, I. B. Rogozin, I. V. Ischenlo, A. E. Kel, Y. L. Orlov, M. P. Ponomarenko, and P. Vezzoni, "GenViewer: A computing tool for protein coding regions prediction in nucleotide sequences," *Proceedings of the 2nd International Congress on Bioinformatics, Supercomputing and Complex Genome Analysis*, World Scientific, Singapore, pages 573–587, 1993.
24. M. A. Roytberg, T. V. Astakhova, and M. S. Gelfand, "Combinatorial approaches to gene recognition," *Computers Chem.*, 21(4):229–235, 1997.
25. S. L. Salzberg, "A method for identifying splice sites and translational start sites in eukaryotic mRNA," *Computer Applications in the Biosciences*, 13(4):365–376, 1997.
26. S. L. Salzberg, M. Pertea, A. Delcher, M. J. Gardner, and H. Tetterlin, "Interpolated Markov models for eukaryotic gene finding," *Genomics*, 59:24–31, 1999.
27. E. E. Snyder and G. D. Stormo, "Identification of coding regions in genomic DNA sequences: An application of dynamic programming and neural networks," *Nucleic Acids Research*, 21:607–613, 1993.
28. V. V. Solovyev, A. A. Salamov, and C. B. Lawrence, "Prediction of internal exons by oligonucleotide composition and discriminant analysis of spliceable open reading frames," *Nucleic Acids Research*, 22:5156–5163, 1994.
29. A. Thomas and M. H. Skolnick, "A probabilistic model for detecting coding regions in DNA sequences," *IMA J. Math. Appl. Med. Biol.*, 11:149–160, 1992.
30. E. C. Uberbacher, J. R. Einstein, X. Fuan, and R. J. Mural, "Gene recognition and assembly in the GRAIL system: Progress and challenges," *Proceedings of the 2nd International Congress on Bioinformatics, Supercomputing and Complex Genome Analysis*, pages 465–476, World Scientific, Singapore, 1993.

31. J. T. L. Wang, Q. Ma, D. Shasha, and C. H. Wu, "Application of neural networks to biological data mining: A case study in protein sequence classification," *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 305–309, Boston, Massachusetts, August 2000.
32. J. T. L. Wang, Q. Ma, D. Shasha, and C. H. Wu, "New techniques for extracting features from protein sequences," *IBM Systems Journal*, 40(2):426–441, 2001.
33. J. T. L. Wang, T. G. Marr, D. Shasha, B. A. Shapiro, G. W. Chirn, and T. Y. Lee, "Complementary classification approaches for protein sequences," *Protein Engineering*, 9(5):381–386, 1996.
34. J. T. L. Wang, S. Rozen, B. A. Shapiro, D. Shasha, Z. Wang, and M. Yin, "New techniques for DNA sequence classification," *Journal of Computational Biology*, 6(2):209–218, 1999.
35. J. T. L. Wang, B. A. Shapiro, and D. Shasha, editors, *Pattern Discovery in Biomolecular Data: Tools, Techniques and Applications*. Oxford University Press, New York, New York, 1999.
36. J. T. L. Wang, C. H. Wu, and P. P. Wang, editors, *Computational Biology and Genome Informatics*. World Scientific Publishers, Singapore, 2003.
37. Y. Xu, R. J. Mural, and E. C. Uberbacher, "Constructing gene models from accurately predicted exons: An application of dynamic programming," *Comput. Appl. Biosci.*, 10:613–623, 1994.
38. M. M. Yin, *Knowledge Discovery and Modeling in Genomic Databases*. Ph.D. Dissertation, Department of Computer Science, New Jersey Institute of Technology, 2002.
39. M. Yin and J. T. L. Wang, "Algorithms for splicing junction donor recognition in genomic DNA sequences," *Proceedings of the IEEE International Joint Symposia on Intelligence and Systems*, pages 169–176, Rockville, Maryland, May 1998.
40. M. M. Yin and J. T. L. Wang, "Application of hidden Markov models to gene prediction in DNA," *Proceedings of the IEEE International Conference on Information, Intelligence and Systems*, pages 40–47, Bethesda, Maryland, November 1999.
41. M. M. Yin and J. T. L. Wang, "Recognizing splicing junction acceptors in eukaryotic genes using hidden Markov models and machine learning methods," *Proceedings of the 5th Joint Conference on Information Sciences*, pages 786–789, Atlantic City, New Jersey, February 2000.
42. M. M. Yin and J. T. L. Wang, "Application of hidden Markov models to biological data mining: A case study," *Data Mining and Knowledge Discovery: Theory, Tools, and Technology II. Proceedings of SPIE*,

B. V. Dasarathy (ed.), Vol. 4057, pages 352–358, SPIE-The International Society for Optical Engineering, USA, 2000.

43. M. M. Yin and J. T. L. Wang, “Effective hidden Markov models for detecting splicing junction sites in DNA sequences,” *Information Sciences*, 139(1-2):139–163, 2001.