BIOINFORMATICS

Vol. 18 no. 1 2002
Pages 19–27

A Bayesian framework for combining gene predictions*

Vladimir Pavlović [1], Ashutosh Garg [2] and Simon Kasif [1]

[1]Bioinformatics Program, Department of Bioengineering, Boston University, Boston, MA 02215, USA and [2]Beckman Institute, University of Illinois, Urbana, IL 61801, USA

Received on April 2, 2001; revised on August 24, 2001; accepted on September 7, 2001

ABSTRACT

Motivation: Gene identification and gene discovery in new genomic sequences is one of the most timely computational questions addressed by bioinformatics scientists. This computational research has resulted in several systems that have been used successfully in many whole-genome analysis projects. As the number of such systems grows the need for a rigorous way to combine the predictions becomes more essential.

Results: In this paper we provide a Bayesian network framework for combining gene predictions from multiple systems. The framework allows us to treat the problem as combining the advice of multiple experts. Previous work in the area used relatively simple ideas such as majority voting. We introduce, for the first time, the use of hidden input/output Markov models for combining gene predictions. We apply the framework to the analysis of the Adh region in *Drosophila* that has been carefully studied in the context of gene finding and used as a basis for the GASP competition. The main challenge in combination of gene prediction programs is the fact that the systems are relying on similar features such as cod on usage and as a result the predictions are often correlated. We show that our approach is promising to improve the prediction accuracy and provides a systematic and flexible framework for incorporating multiple sources of evidence into gene prediction systems.

Availability: Software can be made available on request from the authors.

Contact: vladimir@bu.edu

1 INTRODUCTION

Biology and biotechnology are undergoing a technological revolution which is transforming research into an information-rich enterprise. Novel technologies such as high-throughput DNA sequencing and DNA microarrays are generating unprecedented amounts of data. A typical bacterial genome sequence is comprised of several million bases of DNA and contains several thousand genes. Many microbial genomes have been sequenced by the major genome centers, and the total number of such 'small' genomes is expected to reach 100 shortly. Substantial progress is being made on sequencing the genomes of higher organisms as well. The genomes of eukaryotes are typically much larger; e.g. the human genome is approximately 3 billion bases long and it contains approximately 30 000 putative genes identified thus far.

Gene identification and gene discovery in newly sequenced genomic sequences is one of the most timely computational questions addressed by bioinformatics scientists. This research resulted in several successful systems that have been successfully deployed in several highly visible genome analysis projects. Popular gene finding systems include Glimmer, Genmark, Genscan, Genie, Genewise, and Grail (Burge and Karlin, 1997; Salzberg *et al.*, 1998a; Xu *et al.*, 1996; Kulp *et al.*, 1996; Borodovsky and Mcininch, 1993). The annotations produced by gene finding systems have been made available to the public. Such projects include the genomes of over thirty microbial organisms, as well as Malaria, *Drosophila*, *Caenorhabditis elegans*, mouse, Human chromosome 22 and others. For instance, Glimmer (Salzberg *et al.*, 1998a) has been widely used in the analysis of many microbial genomes and has reported over 98% accuracy in prediction accuracy (e.g. Fraser *et al.*, 1997). Genie (Kulp *et al.*, 1996) has been deployed in the analysis of the *Drosophila* genome and Genscan (Burge and Karlin, 1997) was used for analysis of human chromosome 22.

In addition to these central projects, a large number of proprietary genome analysis projects using gene-finding systems are in progress at the major bioinformatics centers in drug companies, bioinformatics companies, and other industrial organizations. As a result, a large number of research projects are underway with the goal of improving the performance of such systems, primarily targeting improvements in accuracy of reported genes.

*Part of this research was presented at Computational Genomics 2000, Baltimore, MD, November 2000. Portions of this research were conducted at Compaq Computer Corporation, Cambridge Research Laboratory, Cambridge, MA.

In fact, one of the current controversies involves producing an accurate estimate on the number of genes in the human genome. The current number of genes actually found by the gene finding programs are substantially lower than previous estimates.

1.1 Gene identification

As mentioned in the introduction computational gene identification is one of the main successes of bioinformatics research. Early gene identification efforts have started almost 20 years ago (Nakata *et al.*, 1985) and produced a number of reasonably effective systems. For a more detailed description and further references the reader is referred to Burge and Karlin (1997); Salzberg *et al.* (1998a,b).

On a very high level, genes in human DNA and many other organisms have a relatively regular structure. All eukaryotic genes, including human genes, are thought to share a similar layout. This layout adheres to the following 'grammar': start codon, exon, $(intron–exon)^n$, stop codon. The start codon is a specific 3-base sequence (e.g. ATG) which signals the beginning of the gene. Exons are regions in a split-gene sequence that are expressed in either the final protein product or the RNA product. Introns are spacer segments of DNA whose function is not clearly understood. And finally stop codons (e.g. TAA) which signal the end of the gene. The notation $(intron–exon)_n$ simply means that there are n alternating intron–exon segments. At the intron–exon boundaries we find splice junctions (or acceptor/donor sites) that aid the process of RNA-splicing. The main problem facing automated methods for gene discovery is the fact that our current understanding of the genomic transcription process is not sufficient to produce a perfect predictive model of gene recognition in whole genomes. For instance, the 'signals' for start coding (e.g. ATG) and end coding (e.g. TAA) are relatively short DNA sequences that appear very frequently in both coding and non-coding DNA. Similarly, the regions where splicing occurs (splice sites) have relatively weak consensus (based on current data), and most consensus bases automated detection methods for splice detection have relatively high False Positive (FP) rates (see Mount *et al.*, 1995; Burge, 1998; Cai *et al.*, 2000).

Computational gene prediction methods typically rely on dynamic programming formalisms that integrate a variety of probabilistic evidence such as coding potential of exons, splice site detection, duration modeling for introns, branch points and others (Burge and Karlin, 1997).

2 SYSTEM

An implementation of the Bayesian framework for combining gene predictions is written in C and MATLAB (Mathworks Inc.). The code runs on any platform that supports MATLAB and has an ANSI C compiler.

3 METHODS

3.1 Combination of experts

The proliferation of gene prediction systems, especially systems that focus on exon prediction raises the question whether a careful combination of the predictions made by these systems would produce a significantly improved gene detection system. We propose a systematic way to build such a system based on the framework for *combination of experts*.

Combination of experts has drawn significant interest in the machine learning community. Theory and practice of combining experts have been extensively studied in the literature (Wolpert, 1992; Jordan and Jacobs, 1994; Zhang *et al.*, 1992; Heath *et al.*, 1993). These methods are often referred to as ensemble methods, committee methods or mixture of experts. The main goal of these techniques is to reduce the variance and/or the bias of the individual predictors. The choice of a particular way of combining expert predictions depends on the properties of individual experts and the demands posed by the problem at hand. In the problem of gene annotation we expect an expert combination system to have the following two properties:

(1) capture correlation between predictions of individual experts;

(2) model sequential dependencies between combined predictions in a nucleotide sequence.

Most techniques for combination of gene predictions proposed in the past have been rather simple or have relied on ad-hoc combinations of experts using essentially logical rules. For example, Murakami and Takagi (1998) proposed a system for gene recognition which combines several gene-finding programs. They implemented AND and OR combination, HIGHEST-method (best individual expert), RULE-method (decisions using sets of expert rules), and an ad-hoc BOUNDARY-method. The best of these methods achieved an improvement in general accuracy of 3–5% over the individual gene finders. Similar expert combination scheme based on majority voting was recently used at The Institute for Genomic Research (TIGR) and reported in the 12th International Genome Sequencing Conference, September 2000. However, it only achieved moderate improvements in prediction.

We are interested in a system for combination of individual experts which is *learned from data*. Such a system should exploit learned dependencies between experts and form a prediction maximally consistent with known gene data. Statistically, predictions of the system will then have the potential to carry over to genes undiscovered by any of the individual experts.

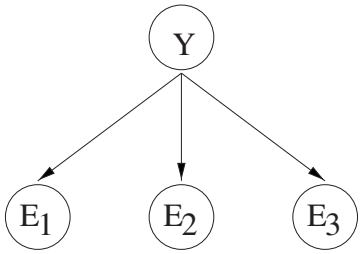

Fig. 1. SNB.

An attractive way of combining experts which exploits their joint statistical behavior and can thus satisfy requirements of our task is based on Bayesian networks. We next propose a Bayesian network framework for the task of combining different gene prediction systems.

3.2 Bayesian networks

Bayesian networks are probabilistic models that graphically encode probabilistic dependencies between random variables (Pearl, 1998; Salzberg *et al.*, 1998b). The graphical structure of the model imposes qualitative dependence constraints. An example of a Bayesian network is shown in Figure 1. For example, a directed arc between variables Y and E_1 denotes conditional dependency of E_1 on Y, as determined by the direction of the arc. In addition to this graphical representation, Bayesian networks include a quantitative measure of dependencies. For each variable and its parents this measure is defined using a conditional probability function or a table. In the example of Figure 1, one of such measures is the probability $\Pr(E_1|Y)$. Together, the graphical structure and the conditional probability functions/tables completely specify a Bayesian network probabilistic model. This model, in turn, specifies a particular factorization of the joint probability distribution function over the variables in the network. Hence, Figure 1 defines $\Pr(Y, E_1, E_2, E_3)$ to be

$$\Pr(Y, E_1, E_2, E_3) = \Pr(E_1|Y)\Pr(E_2|Y)\Pr(E_3|Y)\Pr(Y).$$

Bayesian network probabilistic models provide flexible and powerful framework for statistical inference as well as learning of model parameters from data. The goal of inference is to find a distribution of a random variable in the network conditioned on evidence (values of other variables in the network), e.g. $\Pr(Y|E_1, E_2, E_3)$ Bayes nets encompass efficient inference algorithms, such as Jensen's junction tree (Jensen, 1995) or Pearl's message passing (Pearl, 1998). Inside a learning loop, such algorithms can be used to efficiently estimate optimal values of model's parameters from data (cf. Jordan, 1998), such as the table $\Pr(E_1|Y)$ in the previous example. Furthermore, techniques exist that can optimally determine the topology of a Bayesian network together with its parameters directly from data.

As probabilistic models, Bayesian networks provide a convenient framework for combination of experts. Weights and influences of individual experts can be optimally learned from data rather than being *ad-hoc* or user-specified. We next propose a number of Bayesian network architectures of increasing complexity for the problem of combined gene prediction.

4 ALGORITHM

4.1 Notation

We denote the decisions given by an individual expert i, at base t in the sequence, by E_i^t. E_i^t can take on values from some finite set of decisions provided by the expert. For instance, $E_i^t \in \{startcodon, stopcodon, exon, non\text{-}exon\}$. Combined prediction is denoted by Y^t. Again, $Y^t \in \{startcodon, stopcodon, exon, non\text{-}exon\}$. A probability distribution function associated with E_i^t, for instance, is denoted by $\Pr(E_i^k)$. Absence of the base index t in any of the variables, such as E_i, indicates that there is no sequential dependency in the model involving that variable. Parameters of Bayesian network models (probability tables) are denoted by capital letters, e.g. $A_1(i, j) = \Pr(Y^t = i|E_1^t = j)$. Estimates of parameters are obtained using empirical frequencies of data, C. In this case, $C(E_1^t = i, Y^t = j) = 1$ if $E_1^t = i$ and $Y^t = j$, otherwise it is zero.

4.2 Static naive Bayes

The simplest Bayesian network that one can use for combining of multiple gene predictors is a naive Bayesian classifier. An example of a naive Bayes gene prediction combiner is shown in Figure 1. In this figure, three gene predictors are represented as nodes (E_1, E_2, E_3) and the combined prediction is denoted by Y. In other words, the nodes in the figure are random variables and the edges describe dependencies. The network models joint probability distribution on the experts and the 'true prediction' as:

$$\Pr(Y, E_1, E_2, E_3) = \Pr(E_1|Y)\Pr(E_2|Y)\Pr(E_3|Y)\Pr(Y).$$

Parameters of the network are $A_k(i, j) = \Pr(E_k = i|Y = j)$, $k = 1, \ldots, 3$ and $B(i) = \Pr(Y = i)$, and can be estimated in a number of ways. A common way is to select the parameters such that they maximize the likelihood of data on a training set. In that case, optimal parameters estimates are proportional to frequencies of events in the set. For instance,

$$A_1(i, j) \sim \sum_t C(E_1^t = i, Y^t = j).$$

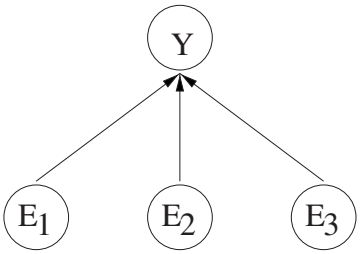

Fig. 2. SFB.

Additional regularization constraints (priors) can also be imposed. They appear as additive constants in the count statistics C.

It is now easy to show that the predictions of individual experts (E_1, E_2, E_3) an optimal combined prediction is found using a simple Bayesian inference,

$$\arg\max_y \Pr(y|E_1, E_2, E_3) = \arg\max_y A_1(E_1, y)A_2(E_2, y)A_3(E_3, y)B(y).$$

Naive Bayes modeling scheme assumes independence of individual experts, given a known combined prediction. In the context of genome annotation, this would imply that the annotation of the experts is independent given the true annotation. Although a successful technique in a wide range of machine learning task, naive Bayes combiner loses its charm as it neither models the correlation of individual experts nor the dependence between the adjacent nucleotides in the sequence.

4.3 Static full Bayes

Correlation between individual experts can be easily modeled using a full Bayes model. This is shown in Figure 2.

Distribution defined by the network is $\Pr(Y|E_1, E_2, E_3)\Pr(E_1)\Pr(E_2)\Pr(E_3)$. Parameters of the complete network are $A(i, j, k, l) = \Pr(E_1 = i, E_2 = j, E_3 = k, Y = l)$, $B_k(i) = \Pr(E_k = i)$, $k = 1, 2, 3$. Using maximum likelihood estimation without priors, one gets

$$A(i, j, k, l) \sim \sum_t C(E_1^t = i, E_2^t = j, E_3^t = k, Y^t = l).$$

Values of the other parameters can be estimated in a similar manner.

Given this model, the optimal combined prediction of predictions from the individual experts (E_1, E_2, E_3) is now

$$\arg\max_y \Pr(y|E_1, E_2, E_3) = \arg\max_y A(E_1, E_2, E_3, y).$$

Rather than a product of probabilities associated with individual experts, as is the case in the naive Bayes combiner, the full Bayes associates one probability with each combination of those experts.

It can be easily shown that the performance of the full Bayes model is *at least* as good as that of the best individual expert. Furthermore, the often used AND, OR and majority models are special cases of the full Bayes combiner. Nevertheless, this model still assumes that the annotation of a particular nucleotide is independent of the annotation of any other nucleotide in the sequence.

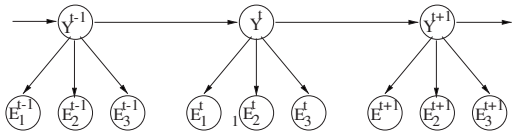

Fig. 3. OHMM.

4.4 Output hidden Markov model

Hidden Markov Models (HMMs) are a special case of Bayesian network architectures that have gained wide popularity in analysis of sequences, including genome annotation. HMMs model probabilistic dependence between adjacent samples in a sequence. In fact, the most popular HMM architectures used in many gene finding systems such as Genscan and protein family modeling system such as PFAM is the Output Hidden Markov Model (OHMM). That is, the observed evidence (e.g. DNA sequence in a typical gene finding system) is assumed to be emitted in the hidden states as output. The output generated in a particular state of the HMM only depends on the state. The OHMM architecture can also be deployed for combination of individual gene predictors, as shown in Figure 3.

HMM model probabilistic dependence between the samples at adjacent positions, t and $t - 1$. Namely, the OHMM proposed here is a sequential extension of the Static Naive Bayes (SNB) model of Section 4.2. The hidden variables in the OHMM network we use correspond to the true predictions (e.g. exon, intron), and the evidence nodes corresponds to the different prediction by the experts. The OHMM assumes that the predictions generated in position t are independent given the 'true prediction' and that the 'true' prediction in position t only depends on the 'true prediction' in position $t - 1$. Parameters of the prediction HMM are $A(i, j) = \Pr(Y^t = i|Y^{t-1} = j)$, $B_k(i, j) = \Pr(E_k^t = i|Y^t = j)$ and $D(i) = \Pr(Y^1 = i)$. These parameters can be estimated using one iteration of the Baum–Welch algorithm.

Optimal gene prediction using this model and given predictions of individual experts can be easily obtained using classic inference/Viterbi decoding in HMMs, (cf. Rabiner and Juang, 1993).

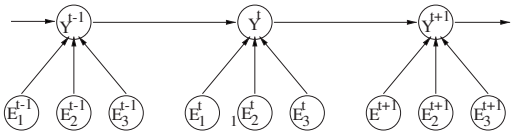

Fig. 4. IHMM.

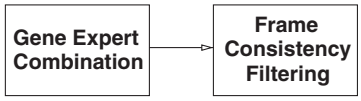

Fig. 5. Frame consistent expert combination. Frame consistency is imposed by filtering the soft decisions of exon predictor with a frame consistent statistical filter.

4.5 Input hidden Markov model

The OHMM-inspired architecture in Section 4.4 addresses the problem of sequential correlation of experts, however it does not model the correlation of individual experts at the same position in a sequence (much like the SNB). We propose a modified network shown in Figure 4 as the Bayesian network that combines the predictions of individual experts without assuming independence. This Input Hidden Markov Model (IHMM) captures the dependencies between individual experts as well as the dependencies between adjacent true predictions. In particular, the 'true prediction' variable in position t depends on all the expert-predictions in this position as well as the 'true prediction' in position $t-1$.

The IHMM defines the following distribution on the sequence of expert and final predictions:

$$\begin{aligned}&\Pr(E_1^1, E_2^1, E_3^1, Y^1, \ldots, E_T^1, E_T^1, E_T^1, Y^T)\\&\quad = \Pr(Y^1|E_1^1, E_2^1, E_3^1)\Pr(E_1^1)\Pr(E_2^1)\Pr(E_3^1)\\&\quad\prod_{t=2}^{T}\Pr(Y^t|E_1^t, E_2^t, E_3^t, Y^{t-1})\Pr(E_1^t)\Pr(E_2^t)\Pr(E_3^t).\end{aligned}$$

This distribution is parameterized using a set of parameters, $A(i,j,k,l,m) = \Pr(Y^t = i|E_1^t = j, E_2^t = k, E_3^t = l, Y^{t-1} = m)$, $A_1(i,j,k,l) = \Pr(Y^t = i|E_1^t = j, E_2^t = k, E_3^t = l)$, and $B_k(i) = \Pr(E_k^t = i)$, $k = 1,2,3$. Much like everywhere else in this method, the parameters can be easily estimated using the count statistics, with or without priors.

Probabilistic analysis for optimal prediction in the IHMM is different from an ordinary HMM and the OHMM of the previous section. Nevertheless, the inference can be accomplished using a standard Bayes net probability propagation technique adapted to IHMMs. Applying a forward probability propagation to the model yields

$$\begin{aligned}&\Pr(Y^t|E_1^1, E_2^1, E_3^1, \ldots, E_1^T, E_2^T, E_3^T)\\&\quad = \prod_{k=2}^{t} A(:, E_1^\tau, E_2^\tau, E_3^\tau, :)A_0(:, E_1^1, E_2^1, E_3^1),\end{aligned}$$

where $A(:, E_1^\tau, E_2^\tau, E_3^\tau, :)$ denotes the two-dimensional table (matrix) obtained from A by evaluating it at $E_1^\tau, E_2^\tau, E_3^\tau$, and products are taken in the matrix domain.

Optimal decision in the IHMM case is

$$\arg\max_{Y_t} \Pr(Y_t|E_1^1, E_2^1, E_3^1, \ldots, E_1^T, E_2^T, E_3^T).$$

This is, in some sense, equivalent to forward–backward inference in OHMMs. Alternative decision can be obtained using a winner-takes-all inference over the whole sequence. In this case the decision rule is

$$\arg\max_{Y_1,\ldots,Y_T} \Pr(Y_1, \ldots, Y_T|E_1^1, E_2^1, E_3^1, \ldots, E_1^T, E_2^T, E_3^T).$$

The solution can be found using dynamic programming, in a fashion analogous to Viterbi inference in OHMMs.

Learning of IHMMs is often not feasible in domains with large state spaces and sparse data points. However, the choice of the state space (as described in the section to follow) and abundance of data in genomic sequences make these models appealing in this domain.

4.6 Frame consistency filtering and gene prediction

Exon prediction results obtained using one of the proposed combination techniques (static, IHMM, OHMM) do not guarantee frame consistency of predicted 'genes'. We impose frame consistency in a post-processing stage using a *frame consistency filter*, as shown in Figure 5. A portion of this filter is depicted in Figure 6. The role of the filter is to select the most likely frame consistent solution proposed by the chosen combiner across the whole sequence. Combined experts, as described in the previous few sections, propose different local decisions at each base in the sequence. The frame consistency filters strings together the most plausible explanations across the whole sequence, based on the combined expert decisions, that also satisfies the frame consistency constraint.

For that to happen the combiner such as an IHMM needs to output *soft decisions*, $\Pr(Y^t|\text{experts})$: each possible outcome (exon, intron, start codon, stop codon, or intergenic region) at every base in the sequence is assigned a probability score (see Figure 6b). The filter selects the most likely path through the trellis of probability scores such that the path is also frame consistent. This is depicted in the portion of the trellis shown in Figure 6c.

Frame consistency filter can be easily implemented using dynamic programming. For example, the total cost

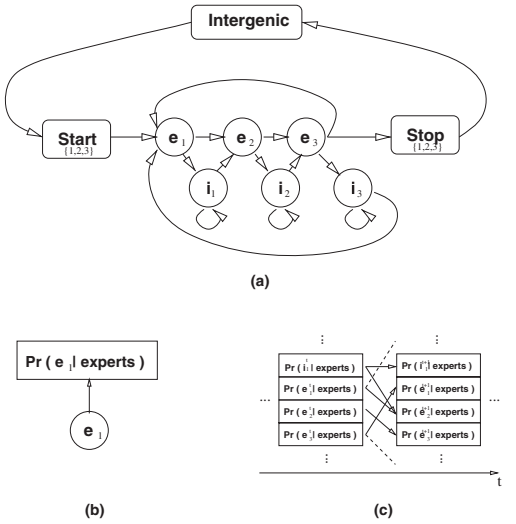

Fig. 6. Frame consistency filter. (a) The state transition diagram of the consistency filter. 'e' denotes the 'true' exon prediction, while 'i' denotes the true intron prediction per each base in the sequence. (b) Each filtered symbol has a score associated with it, Pr (symbol–experts), as determined by the combination of experts. (c) Portion of the trellis of filtered symbols together with their expert-predicted scores and transitions imposed by the state transition diagram.

of predicting e_1 state at position $t+1$ in the sequence, $J(e_1^{t+1})$, can be computed recursively as

$$J(e_1^{t+1}) = -\log(\Pr(e_1^{t+1}|\text{experts})) + \min_{k\in\{\text{start}^t, e_1^t, i_1^t \ldots\}} [-\log(\Pr(e_1^{t+1}|k^t)) + J(k^t)],$$

where $\Pr(e_1^{t+1}|\text{experts})$ is given by the combined expert prediction model and $\Pr(e_1^{t+1}|k^t)$ is determined by the state transition diagram. Most likely frame consistent solutions can be traced back from the lowest cost terminal decision.

Equivalently, the filter can be seen as a HMM whose state transition diagram is shown in Figure 6 and whose emission probabilities are determined by the gene expert combiner score, $\Pr(Y^t|\text{experts})$. Parameters in the state transition diagram can all be estimated using, for instance, maximum likelihood estimation on already sequenced genes.

5 RESULTS

An annotated *Drosophila* sequence was used to conduct the experiments and to obtain the measure of the systems performance. The data is a 2.9 Mb long sequence of nucleotides. We used the same set of experts as the one presented in GASP (Reese *et al.*, 2000a). Our goal was to annotate the sequence into exon (coding region) and intron (non-coding region) using a combination of GASP experts.

For that purpose, we assumed that each individual experts provides the following binary decision. An expert produces a single labeling for every nucleotide in a sequence: E if the nucleotide is a part of an exon and I if it belongs to an intron. Using the notation of our models, $E_i \in \{E, I\}$ for an expert i. Similarly, a combined decision Y is either E or I. Parameters of each of the four combination-of-expert Bayesian network models were learned using a standard maximum likelihood estimation in the BN framework. All prediction results were then obtained using a 5-fold cross-validation.

To compare the performance of the combined system with that of individual GASP experts we used the following performance measures:

5.0.1 Sensitivity and specificity. The results are presented at both the base level and the exon level. Sensitivity and specificity are the two measures that are used at the base level. These are defined as

$$\text{Sn} = \frac{\text{TP}}{\text{TP}+\text{FN}} \quad \text{Sp} = \frac{\text{TP}}{\text{TP}+\text{FP}}$$

where Sn is the sensitivity and Sp is the specificity. TP, FP, FN refers to True Positive, False Positive and False Negative respectively. TP refers to those nucleotides that were correctly labeled as exons. FP refers to nucleotides that were labeled as exons even though they were actually part of introns. Finally FN are nucleotides that were labeled as introns while the actual annotation claimed them to be a part of exons.

5.0.2 Overpredicted and missed exons. Two more measures of error were used only at the exon level: overpredicted exons and missed exons. Figure 7 provides some insight into the performance measures at the exon level. An exon is said to be exactly predicted only if both its ending and beginning points coincides with that of a true exon. An exon is said to be missed if there is no overlap with any of the predicted exons. ME gives the percentage of missed exons whereas WE gives the percentage of wrongly or overpredicted exons. To compute these two numbers, we look for any overlap between a true and a predicted exon.

For our experiments we used 'Fgenes CGG1' (Salamov and Solovyev, 2000), 'Genie EST' (Reese *et al.*, 2000b) and 'HMM Gene' (Krogh, 2000). The performance results for these along with that of 'Genie' (Reese *et al.*, 2000b) are presented in Tables 1 and 2. Base level results are presented in Table 1. Table 2 gives the performance of the experts at the exon level. We also present an entry in the table (WE + ME) which provides a measure of the overall performance of the experts at the exon level.

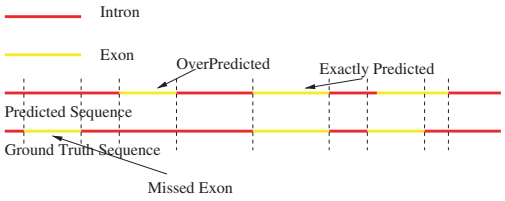

Fig. 7. Exon level performance measure.

Table 1. Base level performance of some experts

	Fgenes CGG1	Genie	Genie EST	HMM gene
Sn	0.89	0.96	0.97	0.97
Sp	0.77	0.92	0.91	0.91

Tables 3 and 4 show results for the mixture of experts framework. We show results for the SNB classifier, Static Full Bayes (SFB), OHMM and IHMM. We also provide results for frame consistent versions of these classifiers, indicated by an 'f' in front of the classifier's name. Finally, we list benchmark performance measures for standard AND and OR experts. Other than being two of the simplest combination techniques the AND and OR combiners also provide sensitivity and specificity bounds. The Specificity for the AND case is 94% and this is the bound on what can be achieved using a (static) mixture of experts framework. Similarly the sensitivity of the OR sequence (98%) bounds the achievable sensitivity.

The base level results indicate that an improvement in prediction can be obtained using the mixture of experts framework. However, a look at exon level performance in Table 4 reveals a substantially more significant improvement. Among the unfiltered classifiers, we see that IHMM performs significantly better than any of the individual experts or any other expert combination technique. The overall performance (ME + WE) shows an improvement of 10% over the best individual expert. However, we observe that the sensitivity and specificity of the IHMM are worse than that of the individual experts. This is because the IHMM, in general, forward-shifts the predicted exon regions. The problem could be alleviated by introducing additional exon boundary detectors (e.g. $\{exon\text{–}start, exon\text{–}stop\}$) to the framework.

SNBs and SFBs performed very well, significantly better than any individual expert, both in exon (Sn = 94%, Sn = 84%, MS + WE = 17%) and in base-level measures (Sn = 97%, Sp = 93%). This reflects both models' ability to capture proper joint decisions inferred from individual expert predictions. Performance of SNB and SFB was comparable, pointing to weak dependence of individual expert decisions.

Table 2. Exon level performance of some experts

	Fgenes CGG1	Genie	Genie EST	HMM gene
Sn	0.65	0.70	0.77	0.68
Sp	0.49	0.57	0.55	0.53
ME	10.5	8.1	4.8	4.8
WE	31.6	17.4	20.1	20.2
ME + WE	42.1	25.5	24.9	25.0

Frame-consistent combined predictions are also significantly better than predictions of individual experts. They are, on the other hand, only slightly worse than those of unfiltered combiners. (e.g. ME + WE = 19.17% versus 18.02% in the SFB case, with comparable sensitivities and specificities). Degraded performance can be expected because the filtered models impose more stringent constraints on predictions and some of the performance measures (e.g. ME + WE) are not sensitive to frame inconsistency. Nevertheless, this is a price worth paying for having full-gene predictions rather than potentially frame-inconsistent predicted exons.

In all of the above cases, probabilistic decisions inferred by the combination of experts framework always outperformed deterministic rules. AND and OR rules define, respectively, specificity and sensitivity bounds of combined expert performance. Probabilistic learned decisions, on the other hand, attempt to *simultaneously* approach both bounds. This study considered only the simplest of deterministic rules (AND and OR). While more complex rules may indeed perform better, they may be tedious to design. In our framework, probabilistic decisions are learned from data and their performance is only constrained by the structure of the combination model (which can, in principle, also be learned.)

Overall, experiments suggest that fSFB and fSNB significantly improve over the best single expert while producing frame-consistent decisions. In particular, fSFB produces 20% ME + WE error and maintains very high exon level sensitivity and specificity, Sn = 90% and Sp = 83%. Similar results are obtained using fSNB. SNB slightly outperforms fSFB, at the cost of allowing frame-inconsistent predictions. IHMM and fIHMM, on the other hand suffer from low specificity and sensitivity that can be contributed to the forward-shifting property of these models.

6 DISCUSSION

In this paper we proposed a systematic framework for learning to combine gene prediction systems. The main advantage of our approach is its ability to model the statistical dependencies of the experts. We recently heard

Table 3. Base level performance of mixture of experts framework

	OR	AND	SNB	SFB	OHMM	IHMM	fSNB	fSFB	fIHMM
Sn	0.98	0.83	0.97	0.97	0.98	0.97	0.97	0.94	0.89
Sp	0.75	0.96	0.93	0.93	0.75	0.84	0.93	0.93	0.91

Table 4. Exon level performance of mixture of experts framework

	OR	AND	SNB	SFB	OHMM	IHMM	fSNB	fSFB	fIHMM
Sn	94.90	70.71	94.40	93.35	95.17	33.33	94.27	90.26	11.67
Sp	50.13	89.49	83.66	83.16	50.86	67.38	79.16	83.15	19.75
ME	4.00	22.55	4.46	5.21	4.01	7.88	4.46	7.12	15.16
WE	39.83	6.97	12.85	12.82	39.65	7.03	16.10	12.05	9.68
ME + WE	43.84	29.52	17.31	18.02	43.66	14.92	20.56	19.17	24.84

that a combiner approach was used with moderate success for gene prediction at TIGR for plant genomes. This result was reported in the 12th International Genome Sequencing Conference, September 2000. The approach there relied on a majority voting algorithm.

We described the application of a family of combiners of increasing statistical complexity starting from a simple naive Bayes to IHMMs. Our preliminary results suggest that the probabilistic network appears promising for exon prediction, producing a reasonable exon-level improvement in prediction accuracy.

We proposed a system for combining predictions of individual experts in a frame-consistent manner. The system relies on the stochastic frame consistency filter, implemented as a Bayesian network, in the post-combination stage. As such, the system enables the use of expert combiners for general gene prediction. Our experiments suggest that the system significantly improves over the best single expert while producing a frame-consistent decision.

We also observe that the approach we described is in principle applicable to other predictive tasks such as promoter or transcription elements recognition.

REFERENCES

Borodovsky,M. and Mcininch,J. (1993) Genemark: parallel gene recognition for both DNA strands. *Comput. Chem.*, **17**, 123–133.

Burge,C. (1998) Modeling dependencies in pre-mRNA splicing signals. In Salzberg,S., Searls,D. and Kasif,S. (eds), *Computational Methods in Molecular Biology*, New Comprehensive Biochemistry, Elsevier Science, Amsterdam, pp. 129–164.

Burge,C. and Karlin,S. (1997) Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.*, **268**, 78–94.

Cai,D., Delcher,A., Kao,B. and Kasif,S. (2000) Modeling splice sites with Bayes networks. *Bioinformatics*, **2**, 152–158.

Fraser,C., Casjens,S., Huang,W., Sutton,G., Clayton,R., Lathigra,R., White,O., Ketchum,K., Dodson,R., Hickey,E., Gwinn,M., Dougherty,B., Tomb,J.-F., Fleischmann,R., Richardson,D., Peterson,J., Kerlavage,A., Quackenbush,J., Salzberg,S., Hanson,M., van Vugt,R., Palmer,N., Adams,M., Gocayne,J., Weidman,J., Utterback,T., Watthey,L., McDonald,L., Artiach,P., Bowman,C., Garland,S., Fujii,C., Cotton,M., Horst,K., Roberts,K., Hatch,B., Smith,H. and Venter,J. (1997) Genomic sequence of a Lyme disease spirochaete, *Borrelia burgdorferi*. *Nature*, **390**, 580–586.

Heath,D., Kasif,S. and Salzberg,S. (1993) Learning oblique decision trees. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, Chambery, France, pp. 1002–1007.

Jensen,V.F. (1995) *An Introduction to Bayesian Networks*. Springer, Berlin.

Jordan,M.I. (ed.) (1998) *Learning in Graphical Models*. Kluwer Academic, Dordrecht.

Jordan,M.I. and Jacobs,R.A. (1994) Hierarchial mixture of experts and the em algorithm. *Neural Comput.*, **6**, 181–214.

Krogh,A. (2000) Using database matches with HMMgene for automated gene detection in *Drosophila*. In *Genome Research*, Vol. 10, pp. 523–528.

Kulp,D., Haussler,D., Reese,M.G. and Eeckman,F.H. (1996) A generalized hidden Markov model for the recognition of human genes in DNA. In *ISMB-96: Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, Menlo Park, CA, pp. 134–141.

Mount,S., Peng,X. and Meier,E. (1995) Some nasty little facts to bear in mind when predicting splice sites. In *Gene-finding and Gene Structure Prediction Workshop*. Philadelphia, PA.

Murakami,K. and Takagi,T. (1998) Gene recognition by combination of several gene-finding programs. *Bioinformatics*, **14**, 665–675.

Nakata,K., Kanehisa,M. and DeLisi,C. (1985) Prediction of splice junctions in mRNA sequences. *Nucleic Acids Res.*, **14**, 5327–5340.

Pearl,J. (1998) *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA.

Rabiner,L.R. and Juang,B. (1993) *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ.

Reese,M., Harris,N., Hartzell,G., Ohler,U., Abril,J.F. and S.,L. (2000a) Genome annotation assessment in *Drosophila melanogaster*. In *Genome Research*, Vol. 10, pp. 483–501.

Reese,M., Kulp,D., Tammana,H. and Haussler,D. (2000b) Genie—gene finding in *Drosophila melanogaster*. In *Genome Research*, Vol. 10, pp. 529–538.

Salamov,A.A. and Solovyev,V.V. (2000) Ab initio gene finding in *Drosophila* genomic DNA. In *Genome Research*, Vol. 10, pp. 516–522.

Salzberg,S., Delcher,A., Kasif,S. and White,O. (1998a) Microbial gene identification using interpolated Markov models. *Nucleic Acids Res.*, **26**, 544–548.

Salzberg,S., Searls,D. and Kasif,S. (eds) (1998b) *Computational Methods in Molecular Biology*, Vol. 32 of *New Comprehensive Biochemistry*, Elsevier Science, Amsterdam.

Wolpert,D.H. (1992) Stacked generalization. *Neural Netw.*, **5**, 241–259.

Xu,Y., Mural,R., Einstein,J., Shah,M. and Uberbacher,E. (1996) Grail: a multi-agent neural network system for gene identification. *Proc. IEEE*, **84**, 1544–1552.

Zhang,X., Mesirov,J.P. and Waltz,D.L. (1992) A hybrid system for protein secondary structure prediction. *J. Mol. Biol.*, **225**, 1049–1063.