Chapter 5

Application of GAZE training to the development of a vertebrate gene finder

5.1 Introduction

In this chapter, I demonstrate how the parameter estimation methods described in chapter 4 can be used to tune the performance of a gene prediction system created with GAZE. By way of a contrast to chapter 3, the methods are applied to the problem of predicting gene structures in the sequences of higher vertebrates such as human [112] and mouse [28]. Gene prediction is more difficult in vertebrate sequences than in the sequences of primitive animals such as *C.elegans* due primarily to a lower signal-to-noise ratio and also other factors discussed in section 1.5 and briefly in chapter 3.

The chapter follows a similar format to that of chapter 3. After summarising the materials for vertebrate gene finding that are to be used, an initial GAZE configuration is outlined. I then describe how the methods of the previous chapter can be used to optimise the parameters of the model, and the effectiveness of the Maximum

Likelihood and Maximal Feature Discrimination methods is compared. I finally go on to produce three variant models, each incorporating a different, new type of gene prediction evidence, and investigate the effectiveness of Maximal Feature Discrimination in weighting the scores of the new data appropriately.

5.2 Materials for gene prediction in vertebrate sequences

5.2.1 Datasets for training and testing

Programs to predict gene structures in vertebrate genomic DNA have historically been trained and tested on sequences that each contain a single, complete gene structure. In chapter 3, I discussed the disadvantages of using single gene sequences for testing. Large, contiguous genomic sequences which are completely understood in terms of their gene structures are even harder to come by for vertebrate sequences than for *C.elegans*. Even for the human chromosomes that have been declared "finished" at time of writing [35][24][30], the gene structures are under continual review, and it is likely that these sequences contain unannotated genes.

In the work on *C.elegans* gene predictions described earlier, the problem was addressed by constructing an artificial genomic sequence of cDNA-confirmed gene structures. This was done by extracting the genomic DNA underlying each gene structure, along with some upstream and and downstream intergenic sequences. Another approach was taken by Guigo and co-workers [53] in the generation of their SAGS (semi-artificial genomic sequences) dataset. They also started with a set of single-gene sequences and their annotated gene structures, but not having any genomic context for the sequences, they created an artificial one. In essence, the sequences were separated by artificial intergenic regions, the lengths of which were normally distributed, and the content of which were based on a fifth order Markov chain based on a C+G content of 38% [54].

The problem with both of these methods is that they do not take account of the variation in mammalian gene structural properties with C+G content (see chapter

1). For programs like GENSCAN that have distinct sets of parameters for different C+G% strata, a sequence constructed with either method would provide an unfair test. By way of illustration, each of the 42 sequences comprising the Guigo SAGS dataset has a C+G content of less than 40%. GENSCAN therefore uses the same, low-C+G% set of parameters for these sequences, which could be one of the main reasons for its relatively poor performance [53].

For this reason, and also for consistency with other literature assessing the accuracy of gene prediction programs in vertebrate sequences, the results presented in this chapter are based upon two distinct sets of single-gene sequences. It is important to note however that all of the GAZE gene structure models used are sufficiently general to allow for multiple genes on both strands of the input sequence, even allowing partial genes at the ends of the sequence.

Training: the H176 dataset

This set was constructed by Guigo and colleagues [53] for an analysis of the accuracy of various gene prediction programs available at the time. It was made in a similar manner to the Burset and Guigo dataset benchmark set vertebrate gene sequences [23], the principle difference being that this set comprises human gene sequences only, and that entries were extracted from the EMBL nucleotide database version 50 (March 1997). There are 178 sequences in the set, from which I removed 2 sequences which had annotated gene structures that were clearly incorrect (HSADH6, with a gene structure containing seven introns each 25 base-pairs in length; and HSPVALB, gene structure with three 24-base introns). The resulting 176-sequence set is referred to as H176 or *the training set*.

Testing: the HMR195 dataset

This set was constructed by Rogic and colleagues for another survey of the accuracy of gene prediction methods [94], using filtering rules ostensibly the same as those used by Guigo in the construction of H176, the primary differences being (i) mouse and

	Human Genome	H176	HMR195	WormSeq
Number of genes	-	176	195	325
Coding density	-	0.13	0.14	0.24
Single exon genes	-	40	43	16
Exons per (multi-exon) gene	8.8	6.2	6.0	7.2
Mean internal exon length	145	146	138	241
Mean CDS length	1340	970	1021	1542
Mean intron length	3365	672	854	308

Table 5.1: Some properties of the training and test sets of vertebrate sequence in comparison with the WormSeq dataset of chapter 3, and estimates for the human genome published in [112].

rat sequences were retained, as well as human; (ii) the Genbank nucleotide database was used (version 111.0, April 1999), and sequences deposited before August 1997 were discarded, to ensure as far as possible that none of the sequences could have been used to train the gene prediction programs being evaluated. The resulting 195-sequence set is referred to as HMR195 or *the test set*. There are no entries in the test set that are also in the training set. The degree of overlap between this set and the training set is discussed below.

By retaining mouse and rat genes in this set, I am assuming that any properties of the H176 (human only) dataset learned by the training process also generalise to the sequences of these organisms. The architects of the dataset partitioned it into human (103 sequences) and mouse/rat (92) subsets, and examined the accuracy in each of a number of programs that were trained on human sequences only. They found the difference to be insignificant, with the many of the programs having marginally better accuracy in the mouse/rat sequences.

5.2.2 Properties of the gene sets

Some properties of HMR195 in comparison with H176 and the WormSeq dataset of chapter 3 are summarised in table 5.1.

The table shows that two vertebrate datasets have properties that are fairly consistent with each other. They both support the observation that vertebrate genes have longer introns and shorter exons than worm genes, although the larger survey summarised in the first column [112] suggests that both sets are still atypical, most noticeably having smaller introns and fewer exons.

The two vertebrate datasets provide a far easier problem for gene prediction programs than they would usually face in practice; a protein coding density of 13-14 percent is about 5 times higher than in the human genome as a whole (for example). Furthermore, the complexity of the genes, in terms of their numbers of exons and introns, and total CDS length, is lower than that of the WormSeq dataset of *C. elegans* sequences. Also, 23 percent of the vertebrate genes contain coding regions that are confined to a single exon, compared to 5 percent in the *C. elegans* dataset). This is not indicative of any characteristic difference in complexity of vertebrate and worm genes, but due to simple sample bias; short genes with small numbers of exons were easier to sequence genomically before the human genome project, hence their disproportionate frequency as separate entries in the nucleotide databases. The *C. elegans* gene structures in WormSeq were confirmed by full-length cDNAs however, so would be affected less by such database bias.

It is technically necessary to correct for such biases in the dataset before training. When calculating the transition probabilities of the GENSCAN HMM for example, Burge took the "true" proportion of single-exon genes to be one half of that observed in the training set. He comments that although this approach is rather *ad hoc*, it is better than no correction at all, although it is more difficult to correct for other biases in databases. I have chosen to ignore all biases because the primary aim of this chapter is to assess the effectiveness of the training methods, and there is no reason to believe that that any biases present in the training set should not also be present in the test set.

There are no entries that occur in both datasets (ensured by construction; see above), but the approach taken by many would be to remove entries in the test set

that show a significant degree of similarity to an entry in the training set. Burge for example in the training and testing of GENSCAN removed sequences from his training set for which the translation of the annotated gene showed more than 25% identity to the translation of a gene in his intended test set [20]. I take the approach of not attempting to make the training and test sets non-redundant with respect to each other, also taken by others [102] [94]; it is reasonable to expect a gene prediction program to often be presented with a sequence containing a gene which shows some degree of similarity to a member of its training set.

One final point to note is that all analysis presented in this chapter was performed on data derived from the raw sequence, with no masking of repeats. There is no mechanism in GAZE itself to account for the possibility that the input feature list may have been derived from repeat-masked sequence, and will therefore contain regions with no stop-codons that should not be considered as candidate coding exons. Repetitive regions therefore have to be explicitly accounted for in the configuration file. One approach is to make segments for the repeats, giving their scores high negative weights, and making them contribute towards the score for candidate protein-coding regions.

Accounting for repeats in this way would be necessary for the analysis of large, unannotated stretches of genomic DNA, but is less important here. Although the repeat-content of the human genome (for example) has been estimated to be in excess of 50% [112], RepeatMasker [A.F.A. Smit and P. Green, unpublished] identifies 21% and 15% of the training and test sets (respectively) as repetitive[1]. Masking these repeats had no significant impact on the accuracy of the standard gene prediction programs (results not shown).

[1]The -nolow option was used which does not mask out low-complexity regions, as these can sometimes be protein-coding, for example in proteins with coiled-coil regions.

5.2.3 A source of gene prediction features: GENEID

As in chapter 3, I draw upon the work of others for a source of gene prediction data. In this case, I used the most recent version of the GENEID program [84], primarily because it is designed for gene finding in vertebrate sequences (specifically human) and optionally outputs candidate gene features as well as predictions of complete gene structures.

The signal and content sensing models used by GENEID are fixed and described below. The parameters for these models however are external to the system and supplied by the user in a file. The program comes with two parameter files for gene finding in human sequences. The first provides distinct parameter-sets for sequences in three C+G% strata. The second file contains a single set of parameters obtained without considering C+G content. Except where otherwise stated, I have chosen to work with features generated from this latter, homogeneous set of parameters.

Signal sensing models in GENEID

GENEID detects signals by calculating weight matrices from frequency tables compiled from real and pseudo examples of the feature of interest. The signals provided are: (i) translation start sites, representing positions -8 through +5 (where 0 is the position of the A in the conserved ATG); (ii) translation stop sites, representing positions -5 through +3 (where 0 is the position of the first nucleotide of one of the three stop codons); (iii) donor splice sites, representing positions -3 through +5 (where 0 is the position of the G in the conserved GT); and (iv) acceptor splice sites, representing positions -22 through +4 (where 0 is the position of the A in the conserved AG). The first three of these signals are modelled using the weight matrix method. The acceptor splice site model is more sophisticated being a first order weight array model. Predicted features using these models were generated using GENEID in signal output mode with default cutoffs (command line option '-bdal').

Content sensing models in GENEID

Unlike GENEFINDER, GENEID does not explicitly provide prediction of likely protein-coding regions, but the supplied parameter files contain details of the model used to score candidate exons for coding potential. As described in [84], log-likelihood ratios $C^j(b_1b_2b_3b_4b_5b_6)$ for hexamer $b_1b_2b_3b_4b_5b_6$ beginning in codon position j, are calculated according to the relative frequency of the hexamer in a set of real exons compared with introns. A "coding" score for a candidate exon $S_1 \ldots S_n$ with pre-assigned phase j (i.e. the codon position of the first base of the exon is known) can then be calculated by summing the scores for the hexamers along the length of the exon:

$$L_C(S_1 \ldots S_n, j) = \sum_{i=1}^{n-5} C^{(j+i-1)\%3}(S_i \ldots S_{i+5})$$

where % is the modulus operator. Rather than using C^j values given in the GENEID parameter file to obtain a set of segments corresponding to likely coding regions in each frame (as was done in chapter 3), six GFF segments were made for each 6 base-pair region in the training and test sequences, three for each strand, by assuming that the region starts in each of the three codon positions. A coding score for a candidate exon can be calculated by summing the segments lying in the region, and I show later the configuration file directives that make this possible. Although requiring a large amount of storage for the GFF files of these segments (largely due to the un-necessary redundancy in this case of GFF), this technique will allow for the calculation of an exact coding likelihood for a candidate exon. The reason that the GAZE models for *C.elegans* gene prediction were not able to duplicate the performance of GENEFINDER was due to the inexactness of the coding score calculation; see chapter 3.

Length penalty functions in GENEID

The GENEID scoring function, explained in more detail shortly, does not include a length penalty component. However, exon scores are subject to a constant, length-

independent penalty, the value of which is supplied as a parameter. These were used in arriving at initial configuration for human gene finding, as explained next.

5.3 A GAZE configuration for human gene finding

A natural approach towards developing a GAZE configuration for the integration of signal and content sensors from GENEID is to start by implementing a system that integrates the data in a similar manner to that program.

5.3.1 A GAZE configuration based on GENEID

The model of gene structure used by GENEID for human gene finding has the same basic components and connectivity as the GAZE_std model for *C.elegans* (figure 3.4). The are minor differences in the maximum and minimum distance constraints as well as the way that partial exons are disallowed (with incomplete introns at the end of the sequence still permissible).

There are however more significant differences in the way in which gene structures are scored. In GENEID, the score of a candidate gene structure is a sum of scores for each of the exons it comprises:

$$L_G(e_1 e2 \ldots e_n) = L_E(e_1) + L_E(e_2) + \ldots + L_E(e_n)$$

The exon scores themselves are calculated as sums of the scores of (i) the upstream defining feature L_U (translation start site or acceptor splice site), weighted by a constant W_U; (ii) the downstream defining feature L_D (translation stop site or donor splice site), weighted by a constant W_D; (iii) a coding score L_C as as described above, weighted by a constant W_C and (iv) a constant W_E:

$$L_E(e) = W_U L_U(e) + W_C L_C(e) + W_D L_D(e) + W_E$$

Restrictions on the form of the parameters W reduces their number from 4 to 2, namely $W_U = W_D$, and $W_U + W_D + W_C = 1$. Values for the two parameters

were chosen to maximise the correlation coefficient (CC) between annotated and predicted protein-coding nucleotides in a set of training examples [84].

It is straightforward to mimic this scoring function in a GAZE configuration. The additive exon score constant W_E can by achieved by the creation of a length-penalty function for which the penalty is the same for all distances. The other parameters W_U, W_C and W_D can be accommodated as the "evidence weights" of the modified GAZE scoring function of the last chapter. The only slight difficulty is the calculation of the coding likelihood $L_E(e)$ for a candidate exon e. Figure 5.1 shows how segments computed from the hexamer log-likelihoods are incorporated in the model. I refer to the complete configuration as GAZE_GeneID.

```
<declarations>
    ⋮
    <segment id="hex_0" scoring="standard_sum" partial="FALSE"/>
    <segment id="hex_1" scoring="standard_sum" partial="FALSE"/>
    <segment id="hex_2" scoring="standard_sum" partial="FALSE"/>
    ⋮
<declarations>

<gff2gaze>
    ⋮
    <gffline feature="cod_hex" source="GENEID" strand="+" frame="0">
        <seg id="hex_0"/>
    </gffline>

    <gffline feature="cod_hex" source="GENEID" strand="+" frame="1">
        <seg id="hex_1"/>
    </gffline>

    <gffline feature="cod_hex" source="GENEID" strand="+" frame="2">
        <seg id="hex_2"/>
    </gffline>
    ⋮
<gff2gaze>
```

```
⋮
<model>
    ⋮
    <target id="stop">
        <useseg id="hex_0" phase="0" />
        <useseg id="hex_2" phase="1" />
        <useseg id="hex_1" phase="2" />
        <killfeat id="stop" phase="0"/>

        <source id="start" mindis="60" len_fun="sngl_ex_pen" phase="0">
            <output feature="CDS_term" strand="+" frame="0"/>
        </source>

        <source id="3ss_0" mindis="0" len_fun="term_ex_pen" phase="0">
            <output feature="CDS_term" strand="+" frame="0"/>
        </source>

        <source id="3ss_1" mindis="0" len_fun="term_ex_pen" phase="2">
            <output feature="CDS_term" strand="+" frame="1"/>
        </source>

        <source id="3ss_2" mindis="0" len_fun="term_ex_pen" phase="1">
            <output feature="CDS_term" strand="+" frame="2"/>
        </source>
    </target>
    ⋮
</model>
```

Figure 5.1: Fragment of a GAZE-XML configuration derived from GENEID showing how hexamer coding segments contribute towards the score. Each 6-tuple in the query sequence has three segments, one for each possible codon position that the hexamer might begin at. A segment type for each codon position is therefore defined, and the GFF "frame" attribute is used to construct segments of the corresponding types. The segment score for regions ending with for example the "stop" target feature, comprises a sum of separate scores for each segment type, and the "phase" attribute is used to ensure that only the single correct segment at each sequence position (i.e. that starting at the appropriate codon position) contributes towards the score. All reverse-strand elements of the model are omitted for clarity.

	Base	Exon			Gene		
	CC	Av	ME	WE	Av	MG	WG
GAZE_GeneID	0.818	0.642	0.179	0.133	0.104	0.04	0.09
GENEID	0.821	0.644	0.181	0.129	0.112	0.04	0.09

Table 5.2: Accuracy of GAZE_GeneID compared with GENEID on a combination of the H176 and HMR195 datasets. The accuracy measures are explained in section 1.4.1.

5.3.2 Accuracy of the model

Table 5.2 shows the accuracy of GAZE_GeneID compared with GENEID on both datasets of vertebrate sequences. Reassuringly, the results are extremely similar but it is interesting to ask why they are not identical.

There are two reasons for the discrepancy. The first reason is that GENEID does not allow for the fact that under its own model of gene structure, initial and terminal exons can be less than 6 base-pairs in length. Its calculation of a coding score for such small exons is therefore incorrect. The same small exons are possible in GAZE_GeneID, but since no hexamer-derived segment can fit completely within a region smaller than 6 base pairs in length, they are given a segment score of zero.

The second reason is that when forming a pool of candidate exons, GENEID considers only the 5 highest-scoring upstream candidate acceptor splice sites for each candidate donor. This heuristic was probably implemented for reasons of space or time efficiency but means that GENEID is not guaranteed to identify the globally highest-scoring gene structure in the sequence. This does not seem to have any impact on the accuracy however, as the tables above show. Although GENEID missed more exons than GAZE_GeneID, fewer of its exon predictions are wrong.

5.4 Optimising the parameters of the model

As explained above, the GENEID scoring function has effectively four free parameters which are chosen to maximise the correlation coefficient between annotated and

predicted coding nucleotides in a training set. To make this optimisation procedure tractable with straightforward methods, a specific relationship between the parameters is imposed, reducing the effective number of free variables in the function to 2. The GAZE training method described in the last chapter however is specifically designed for the simultaneous optimisation of a function with several free variables. In addition, both the Maximum Likelihood and Maximal Feature Discrimination estimation methods are quite different from a technique based on maximising the correlation coefficient. In this section, I investigate the effectiveness of the two GAZE training methods in obtaining values for the parameters that give rise to accurate gene predictions.

5.4.1 Defining the parameters of the model

As a starting point, I expand the effective two free parameters of the GENEID scoring function to eight in GAZE_GeneID in the following way. Firstly, the scaling factor for the scores of the defining features of a candidate exon ($W_U = W_D$) is replaced by three untied weights in GAZE_GeneID for each of translation start sites, translation stop sites, and splice sites. There are in fact 16 distinct exon-defining features in GAZE_GeneID but sensible tying reduces the number of weights first to six (for example, the scores of donor and acceptor splice sites in each of three phases are all subject to the same weight) and then to three (the model is made strand neutral by tying together the weights for the corresponding forward and reverse strand version of a feature).

Secondly, the scaling factor for the coding score for a candidate exon in GENEID (W_C) is represented by a single weight for the same quantity in GAZE_GeneID.

Thirdly and finally, the additive constant for candidate exons in GENEID (W_E) is replaced by four separate but identical length-independent penalty functions for initial, internal, terminal and single exons (i.e. single exon genes). Each function has its own weight, effectively allowing different additive constants for each exon type.

5.4.2 Accuracy of the trained model

Optimal values for these 8 parameters on the H176 training set of sequences were obtained by both the Maximum Likelihood and Maximal Feature Discrimination methods described in the last chapter. The resulting accuracy in the prediction of gene structures in both the H176 training set and the HMR195 test set are is shown in table 5.3. The table shows that the MFD method seems to perform better than the ML method, at all levels of accuracy. Indeed, at the base-pair and exon levels ML training leads to an increase in specificity and decrease in sensitivity, with little difference in average performance. Both ML and MFD training give significantly improved accuracy at the gene level however, with again MFD having the edge.

It is apparent from the results that both the ML-trained and the MFD-trained models perform slightly worse on the test set than on the training set. This could be because the test set is by chance a more challenging data set for gene prediction, either because it has more "difficult" genes (for example 5.1 shows the test set to have an observably larger mean intron length than the training set), or because the signal and content sensors used by GENEID were derived from examples that share more in common with the training set than the test set. Inspection of the gene level results however reveals GAZE_GeneID to perform better on the test set than on the training set, whilst the reverse is true for the trained versions of the model.

A natural explanation for this is a problem often encountered in machine-learning, namely that the GAZE_GeneID$^{\mathrm{ML}}$ and GAZE_GeneID$^{\mathrm{MFD}}$ have been "over-fitted" and the derived parameters represent specific aspects of the training set that do not generalise to the test set. A related problem is that observed by Henderson *et. al.* in the training of the VEIL gene prediction program [56]. They used the Baum-Welch algorithm [5] to obtain the set of parameters for their Class hidden Markov model that maximised the likelihood of a set of training sequences. Aware of a possible non-correspondence between the likelihood of the model and its accuracy of gene prediction (in terms of the classical measures), they they took a series of snap-shots of the parameter values after each iteration of their optimisation procedure,

(a)

Base level	H176 (training)			HMR195 (test)		
	Sn	Sp	CC	Sn	Sp	CC
GAZE_GeneID	0.86	0.83	0.82	0.81	0.87	0.82
GAZE_GeneID^{ML}	0.83	0.86	0.82	0.76	0.89	0.79
GAZE_GeneID^{MFD}	0.86	0.85	0.83	0.81	0.88	0.82
GENSCAN	0.97	0.86	0.90	0.95	0.86	0.89
FGENESH	0.81	0.79	0.76	0.83	0.82	0.80

(b)

Exon level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	ME	WE	Sn	Sp	Av	ME	WE
GAZE_GeneID	0.64	0.67	0.66	0.17	0.14	0.61	0.65	0.63	0.19	0.13
GAZE_GeneID^{ML}	0.63	0.70	0.66	0.21	0.13	0.56	0.68	0.62	0.28	0.14
GAZE_GeneID^{MFD}	0.69	0.70	0.69	0.17	0.16	0.64	0.69	0.66	0.20	0.14
GENSCAN	0.82	0.75	0.79	0.06	0.15	0.77	0.73	0.75	0.08	0.14
FGENESH	0.64	0.60	0.62	0.14	0.19	0.64	0.63	0.63	0.19	0.19

(c)

Gene level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	MG	WG	Sn	Sp	Av	MG	WG
GAZE_GeneID	0.09	0.08	0.08	0.03	0.08	0.13	0.12	0.12	0.04	0.10
GAZE_GeneID^{ML}	0.26	0.22	0.24	0.03	0.18	0.23	0.19	0.21	0.05	0.16
GAZE_GeneID^{MFD}	0.29	0.23	0.26	0.02	0.22	0.27	0.22	0.24	0.04	0.17
GENSCAN	0.40	0.35	0.37	0.01	0.13	0.37	0.33	0.35	0.02	0.12
FGENESH	0.30	0.25	0.28	0.04	0.15	0.32	0.29	0.31	0.04	0.13

Table 5.3: Accuracy at (a) base-pair level, (b) exon level and (c) gene level on the H176 and HMR195 datasets of GAZE_GeneID after training with the Maximum Likelihood (ML) and Maximal Feature Discrimination (MFD) methods on H176. Results for GENSCAN and FGENESH are shown for comparison. The accuracy measures are explained in section 1.4.1.

and evaluated the accuracy of prediction in the *training* sequences obtained with each (as measured by the average of exon sensitivity and specificity). The result of this analysis was that even though the likelihood of the model was increased with each iteration, after a certain number of iterations the accuracy of prediction in the training sequences started to decrease. The authors refer to this phenomenon as "over-training" [56].

The situation is similar here in that neither optimisation function is the same as gene prediction accuracy when measured in the standard ways, although of course we would hope for a good correlation. It is therefore insightful to plot how the accuracy of the model on the *test* set (for over-fitting) and *training* set (for over-training) evolves during both optimisation procedures. This is made simple by the conjugate gradient descent method described in the last chapter, whereby the end of each line minimisation provides a natural point at which to "freeze" the process and assess the accuracy.

Figures 5.2 and 5.3 show how the accuracy of prediction in the training sequences, and respectively the test sequences, varies during the conjugate gradient descent algorithm, for both the ML and MFD objective functions. The main conclusion to draw from these plots is that gene level accuracy in both the training and test sets increases steadily during the optimisation. Although at the exon level the increase in accuracy is less smooth, it is certainly not the case that there is a point during the optimisation after which further line minimisations reduce the accuracy. With these datasets and this configuration at least, neither over-fitting nor over-training seems to be a problem.

The pattern of increase of gene level accuracy during the optimisation is not dissimilar to that shown by the objective functions themselves, suggesting that it is this accuracy measure that these functions most closely reflect. At the exon level, noticeable improvement is only observed in the MFD-trained model; ML-training results in only marginal increase in exon accuracy, in both training and test sets.

Figures 5.2 and 5.3 reveal much about the way in which overall performance is

improved by the two methods. Both at the gene-level and exon level, the untrained model with parameters taken from GENEID has similar sensitivity and specificity. As the training progresses, the sensitivity and specificity diverge, producing models that are more sensitive than specific at the gene level, and more specific than sensitive at the exon level. This is true of both ML and MFD training, but the divergence is most dramatic in the exon-level accuracy of the ML-trained model. Another reason to favour the MFD-training therefore is that it seems to produce parameters which achieve a better balance of sensitivity and specificity.

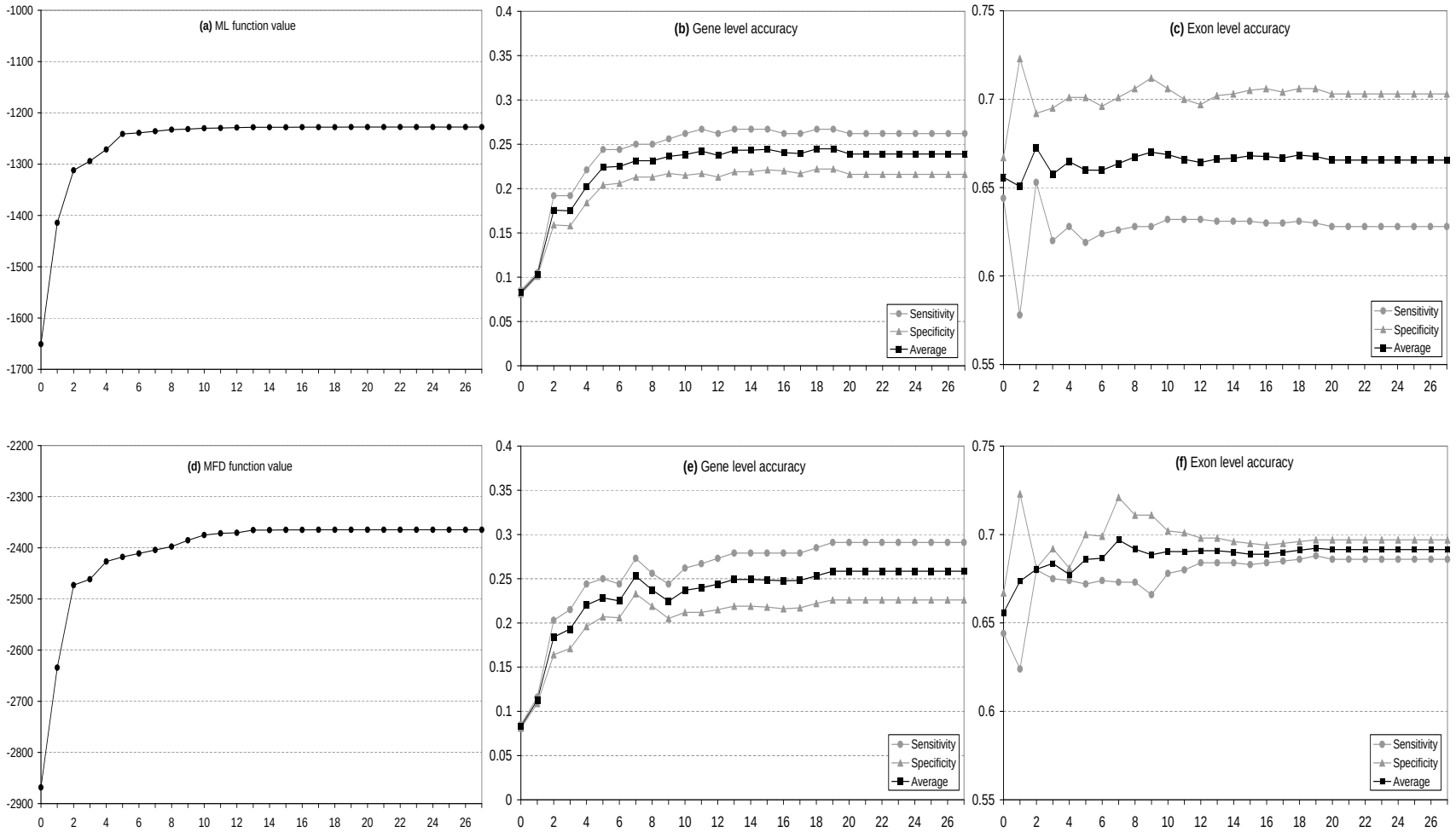

Figure 5.2: Prediction accuracy in the *training* set (H176) after each line minimisation iteration of the conjugate gradient descent algorithm. Plotted are (a) the value of the maximum likelihood function and resulting (b) gene-level and (c) exon-level accuracies. (d,e,f) Similar plots for the maximum feature discrimination training procedure.

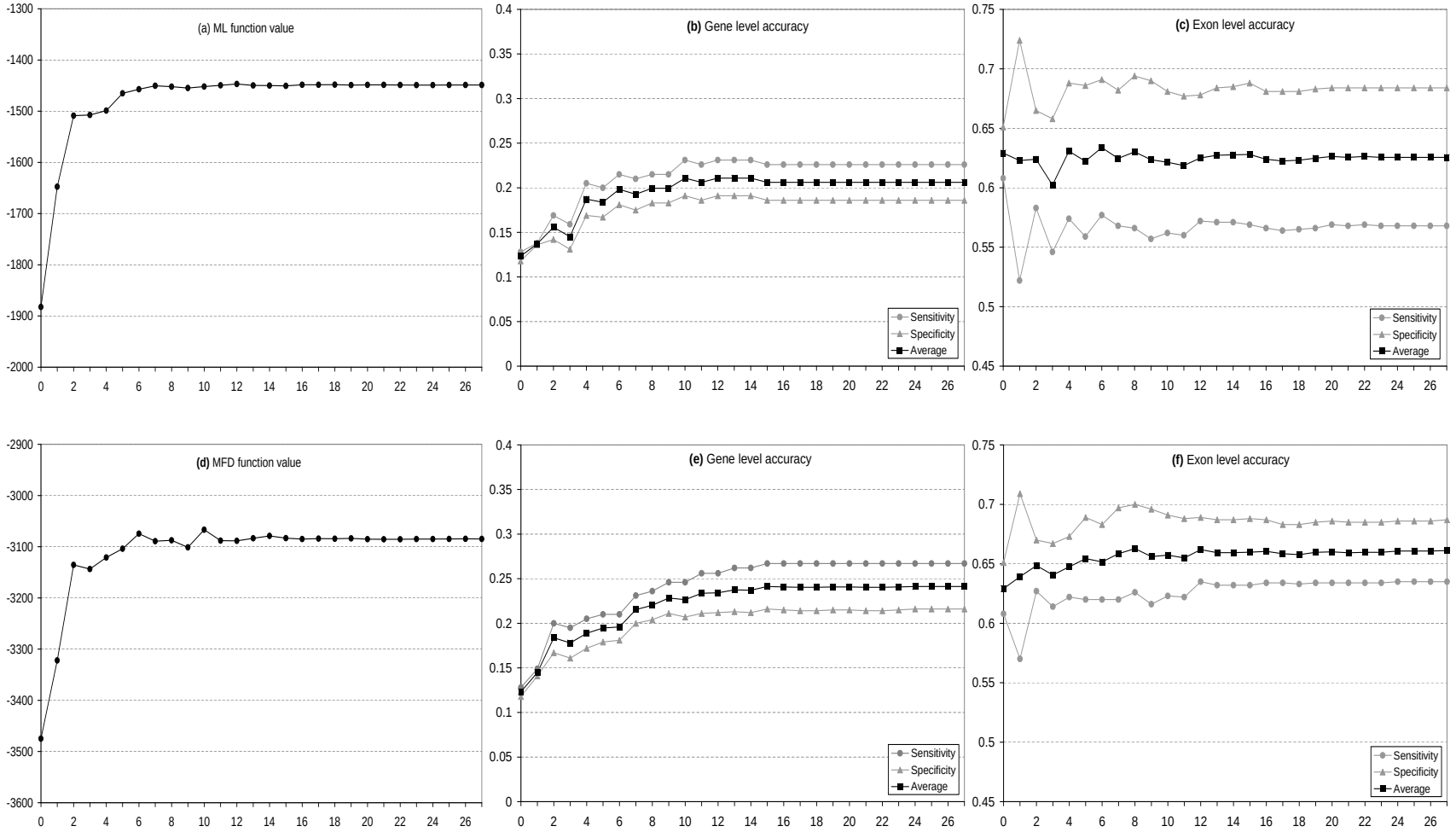

Figure 5.3: Prediction accuracy in the *test* set (HMR195) after each line minimisation iteration of the conjugate gradient descent algorithm applied to the *training* examples (H176). See legend for figure 5.2.

5.5 Investigating three ways to improve accuracy

One of the most striking aspects of the results of the previous section is the degree to which GENSCAN out-performs all GENEID-derived GAZE configurations, untrained, ML-trained and MFD-trained. Given the sophistication of the GENSCAN model of gene structure and the signal and content sensing models that it employs, this is perhaps unsurprising. In particular, the program contains at least three specific innovations that might contribute to its impressive accuracy. Firstly, its model of gene structure includes sub-models for non-protein-coding elements of the transcript, in particular the sites of transcription initiation. This effectively provides an additional source of evidence for genes. Secondly, rather than using standard statistical distributions for the lengths of initial, internal, terminal and single exons (or not having any length distribution at all) the probability of observing each exon length (given the type) was estimated directly from the training set, using a smoothing procedure to avoid over-fitting. Thirdly, to take account of variations in human gene structural properties with C+G content, the model has distinct parameters for each of four C+G% strata. When presented with a query sequence, its C+G% is computed and the appropriate set of parameters used for prediction.

In this section, I examine the effectiveness of the GAZE framework, supplemented by Maximal Feature Discrimination training, in accommodating each of these particular kinds of innovation in turn. I start by showing how promoter prediction data produced by the EPONINE_SCAN program [34] can be put to effective use by a simple refinement to the gene structure model. I then go on to examine the effectiveness with which external region-length distribution data can be incorporated, by supplementing the standard configuration with the smoothed exon-length data derived by Burge [20]. Thirdly, I show how GAZE can be used with datasets stratified by C+G content. In all three cases, the focus of the presentation is the role of the Maximal Feature Discrimination training method and its effectiveness in weighting the scores of the relevant model elements.

5.5.1 Incorporating promoter prediction data

The most well-known and widely used model for eukaryotic promoter recognition (used by GENSCAN and FGENESH for example) was defined by Bucher [19]. It consists of a pair weight matrices for the TATA-box (15 base pairs) and the downstream transcription initiation site (8 base pairs), separated by a distance of 14-20 nucleotides. Each of the distances in this range is assumed to be equally probable, and the DNA itself in the interval is generated according to a "background" intergenic model.

A source or promoter prediction data: EPONINE_SCAN

Rather than re-implementing this model for eukaryotic promoters, I have chosen to make use of a more recent development in the field of promoter detection. The EPONINE_SCAN program for transcription start site detection [34] consists of four weight matrices, representing (1) a CpG island downstream of the start site; (2) a TATAAA motif upstream of the start site (corresponding to the TATA-box mentioned above); (3) a short region of high C+G content just upstream of (2); (4) likewise a short region of high C+G content just downstream of (2). Representing the promoter signal as a series of weight matrices is not new idea, but unlike the simple Bucher model described above, each weight matrix is associated with a probability distribution describing its position relative to the transcription start site. This allows for a more accurate representation of the way in which the distances between different components of the promoter signal vary in real examples. In comparisons with other published promoter detection methods, EPONINE_SCAN performs favourably, being as sensitive as the next best method (PROMOTERINSPECTOR; [98]), but more specific.

Using EPONINE_SCAN with GAZE

EPONINE_SCAN is freely available on the world-wide web (http://www.sanger.ac.uk/-Software/analysis/eponine), and outputs transcription start site predictions with associated scores in GFF, making it ideal to be used with GAZE.

Threshold	0.999	0.99	0.5	0.1
Total predictions	2491	24887	377087	923369
Predictions per sequence	21	117	1030	2489
Seqs with no prediction	252	159	5	0

Table 5.4: Number of predictions reported by EPONINE_SCAN on the H176 and HMR195 datasets at four different score thresholds. In each case, sequences with no prediction were excluded from the calculation of the mean number of predictions per sequence.

The scores reported by the program are intended to be interpreted as the probability of the correctness of the prediction, so range between 0 and 1. Since the scores of the other features used in the system are log probability-ratios, it is likely that the best results would be obtained if the EPONINE_SCAN scores were also in this form. The reported scores x were therefore transformed into log probability-ratio form x' in the following way, inverting the logistic link function used as the output stage of EPONINE_SCAN (T. Down, pers. comm.):

$$x' = \log \frac{x}{1-x}$$

The scoring threshold recommended by the authors to give the highest average of sensitivity and specificity is 0.999. However, the sensitivity reported for this cutoff for a dataset based on the confirmed transcripts in human chromosome 22 is 0.54 [34]. For GAZE to be effective in using these predictions, it is therefore necessary to lower the threshold to increase the sensitivity. Table 5.4 shows the number of predictions made by EPONINE_SCAN on the H176 and HMR195 datasets for a variety of thresholds. Although a threshold of 0.1 leads to a very high false positive rate (only at most one prediction on each sequence can be correct), this should not be a problem for GAZE; both the scores of the predictions of themselves and the surrounding genomic context (particularly in the protein-coding part of the gene) should provide enough information to disregard most (ideally all) of the false positives.

It is interesting to note that even at the recommended threshold of 0.999, the mean number of predictions per sequence when those sequences for which there is no predictions are excluded is 21. This gives an upper bound on the specificity for these sequences (assuming one prediction on each sequence is correct) of 0.05, far lower that the figure of 0.74 previously reported [34]. The authors state that it was necessary to group predictions into clusters to achieve the results reported, with prediction within 1000 nucleotides of each other being considered a single prediction. Again, it is not be necessary to perform this step with GAZE, because the gene structure rules will ensure that only the highest scoring site in a cluster will be included in the prediction of a downstream gene.

Only a slight modification to the standard gene structure model is required to make use of the transcription start site predictions reported by EPONINE_SCAN. The resulting model architecture is a simpler version of that shown pictorially in figure 3.9 for modelling the untranslated regions at the end of *C. elegans* genes, omitting the *trans*-splice and transcription termination features. Figure 5.4 shows how the new EPONINE_SCAN-produced "transcript_start" feature slots into the standard model of gene structure. The resulting model is referred to as GAZE_GeneID$_{\text{tss}}$.

Obtaining optimal weights for EPONINE_SCAN scores

After modifying the model to incorporate the new feature, the next step is to obtain an optimal weight for the converted scores reported by EPONINE_SCAN. In addition to a parameter for the feature itself (with weights for the forward and reverse versions of the feature tied to be the same), I have defined a constant length penalty function for untranslated regions (with associated weight); the reason for this is to maintain consistency with the rest of the elements of the GAZE_GeneID configuration, the coding exons of which are also subject to a weighted, constant penalty (see earlier). There are therefore two free parameters associated with the scores reported by EPONINE_SCAN: a multiplication factor (corresponding to the weight of the "transcript_start" feature score) and an additive constant (corresponding to the

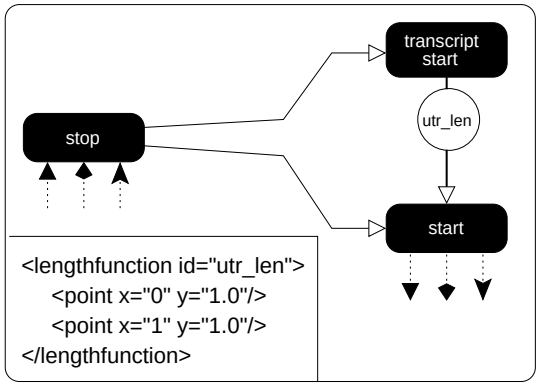

Figure 5.4: Changes to the standard GAZE_GeneID model of gene structure to allow for the possibility of a transcription start site upstream of the start of the protein-coding region. The forward-strand half of the model only is shown for clarity. A length-penalty function for the untranslated region between the transcription start and translation start ("utr_len") is defined to be an arbitrary constant (inset), which will be optimised by training.

weight of the "utr_len" length penalty function).

Considering also the parameters of the basic GAZE_GeneID configuration, the GAZE_GeneID$_{\text{tss}}$ model has 10 parameters to optimise. It is natural to think that only 2 of these 10 parameters need to be optimised because optimal values for the other 8 have already been obtained (GAZE_GeneID$^{\text{MFD}}$ in table 5.3). However it is not necessarily true that these values are optimal with respect to GAZE_GeneID$_{\text{tss}}$ model. The overall score of a gene (which effectively determines whether it appears in the output) will now have two additional components if it includes a putative transcription start. It may be therefore that the other elements of the score (for the features, segments and length penalty functions defining the coding part of the gene) have to be down-weighted to compensate.

Table 5.5 shows the result of optimising the parameters of GAZE_GeneID$_{\text{tss}}$ on the H176 training set, for both the training set itself and the HMR176 test set. In this case, it is interesting to note that only the MFD method is applicable, because it was not known in advance which (if any) of the EPONINE_SCAN-predicted transcription start sites were correct; the ML method requires every candidate feature to be

(a)

Base level	H176 (training)			HMR195 (test)		
	Sn	Sp	CC	Sn	Sp	CC
GAZE_GeneID$_{\text{tss}}^{\text{MFD}-2}$	0.87	0.89	0.86	0.81	0.90	0.83
GAZE_GeneID$_{\text{tss}}^{\text{MFD}-10}$	0.88	0.89	0.86	0.82	0.91	0.84
GAZE_GeneID$^{\text{MFD}}$	0.86	0.85	0.83	0.81	0.88	0.82

(b)

Exon level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	ME	WE	Sn	Sp	Av	ME	WE
GAZE_GeneID$_{\text{tss}}^{\text{MFD}-2}$	0.71	0.73	0.72	0.15	0.13	0.64	0.71	0.67	0.21	0.12
GAZE_GeneID$_{\text{tss}}^{\text{MFD}-10}$	0.72	0.73	0.73	0.15	0.13	0.65	0.71	0.68	0.20	0.12
GAZE_GeneID$^{\text{MFD}}$	0.69	0.70	0.69	0.17	0.16	0.64	0.69	0.66	0.20	0.14

(c)

Gene level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	MG	WG	Sn	Sp	Av	MG	WG
GAZE_GeneID$_{\text{tss}}^{\text{MFD}-2}$	0.34	0.27	0.30	0.02	0.19	0.28	0.23	0.25	0.05	0.15
GAZE_GeneID$_{\text{tss}}^{\text{MFD}-10}$	0.32	0.27	0.30	0.02	0.18	0.27	0.23	0.25	0.06	0.15
GAZE_GeneID$^{\text{MFD}}$	0.29	0.23	0.26	0.02	0.22	0.27	0.22	0.24	0.04	0.17

Table 5.5: The resulting accuracy of the GAZE model incorporating EPONINE_SCAN predictions (GAZE_GeneID$_{\text{tss}}$) trained using the Maximal Feature Discrimination method in two different ways as explained in the text. The results of using the same method to train the standard model (GAZE_GeneID), described earlier, are repeated here for comparison. (a) base-level accuracy; (b) exon-level accuracy; (c) gene-level accuracy. The accuracy measures are explained in section 1.4.1.

declared "correct" or "incorrect" whilst the MFD method requires this only for a subset of feature types (see previous chapter).

The model was trained in two ways: firstly with the optimal values obtained for GAZE_GeneID$^{\mathrm{MFD}}$ fixed and only the two parameters associated with the new feature subject to optimisation (GAZE_GeneID$^{\mathrm{MFD}-2}_{\mathrm{tss}}$) and secondly with all 10 parameters optimised simultaneously (GAZE_GeneID$^{\mathrm{MFD}-10}_{\mathrm{tss}}$). The table shows that there is little (if any) significant difference between the accuracies of the models resulting from the two optimisations. This can be explained by the fact that the difference between the two optimisations in terms of the final value of the objective function and resulting weights for the model elements was marginal.

Given this result, it is tempting therefore to assume that when introducing a new feature type into a gene structure model, only the weights for the scores of the new features (and related length penalty functions) need to be re-estimated. This would be of practical value to the system as a whole, due to the fact that models with smaller numbers of free parameters can be optimised far more quickly with the conjugate gradient descent algorithm. However, there is no obvious reason why the assumption should hold in general.

We would expect promoter information to provide more accurate identification of the 5' end of the gene, particularly the translation start site. Looking at the results more carefully, this seems only marginally to be the case; the GAZE_GeneID$^{\mathrm{MFD}-2}_{\mathrm{tss}}$ model identified 109 of the translation starts in the HMR195 data set, compared with 105 for the GAZE_GeneID$^{\mathrm{MFD}}$ model. When looking at the results for the H176 (training) set however, it is evident that a degree of over-fitting has occurred, the increase being from 90 to 111 in the training set.

With only a marginal increase in accuracy in the test-set, it is natural to ask whether the transcription start site predictions are being used at all in prediction. The optimal values for the weights of the feature scores and the associated 5' UTR length-independent penalty function (with constant value 1.0) obtained by the MFD method were 5.1 and 24.7 respectively. This means that only EPONINE_SCAN predic-

tions with a converted score before weighting of above 4.8 will contribute positively towards the prediction. For future applications, predictions with score less than this can be filtered out before running GAZE without affecting the highest scoring gene structure.

As well as improving prediction accuracy, the inclusion of promoter prediction information in the model also gives it the ability to predict 5' untranslated regions in the genome (although real UTRs are defined in the processed mRNA). It is difficult to measure the accuracy of these UTR predictions without knowledge of the correct site of transcription initiation for the test sequences. However, some insight can be gained by isolating those predicted genes for which the *translation* start site has been identified correctly. The assumption is that the UTRs implied by the subset of these gene structures that include an EPONINE_SCAN-predicted site of transcription initiation carry a higher than otherwise likelihood of being correct. For the HMR195 (test) dataset: of the 235 genes predicted by the GAZE_GeneID$^{\mathrm{MFD}-2}_{\mathrm{tss}}$ model, 109 include a correctly-identified translation start site, 75 include an EPONINE_SCAN-predicted site of transcription initiation (and therefore UTR), and 43 contain both.

5.5.2 Using exon length distributions

The length penalty component of the GAZE scoring function has been used in a very simplistic way for the models presented so far in this chapter. Although penalty functions for each of initial, internal, terminal and single exons have been defined and weighted independently, the functions themselves are defined to be constant, i.e. the same penalty is applied to exons of a particular type, whatever their length. It is certainly not the case however that exon lengths are uniformly distributed in the genomes of eukaryotes. A plot of the lengths of internal exons (for example) in the human genome [112] revealed a mean length of 145 bp (median 122), with the majority falling between 50 bp and 300 bp and a sharp peak at just over 100 bp. Another analysis has shown this distribution to be close to log-normal [121]. It would seem therefore that there is value in more accurately reflecting the likely

lengths of exons in the penalty functions employed in GAZE_GeneID for exonic regions (i.e. by associating unlikely lengths with high penalties).

A source of exon-length data: GENSCAN

Many gene prediction methods (particularly those based around standard hidden Markov models) by their nature assume that the likelihood of a region of a particular type (e.g. exon) decays exponentially with length. Burge showed that this geometric model was an inaccurate representation for the lengths of each of the four different types of coding exon [21]. The Generalised Hidden Markov model framework employed by GENSCAN naturally accommodates length probability distributions based upon direct observation rather than an assumed (e.g. geometric) model (see chapter 1), but it was still non-trivial to obtain these distributions in the first place.

The main problem in obtaining realistic probabilities for the lengths of each of the four types of exons was small size of the training set (238, 1016, 238, 142 for each of initial, internal, terminal and single exons). This gave rise to spikey distributions for which the probability of many lengths was zero simply because they were not observed in the training set. A smoothing procedure was therefore employed, based on a simple underlying model of exon evolution [20].

Using the GENSCAN exon length distributions with GAZE

The GENSCAN parameter file for gene-finding in human sequences includes a probability for each length (up to a maximum) for each of initial, internal, terminal and single exons. The first task is to convert the given exon length probabilities into a form that can be presented to GAZE as a length penalty function. Taking the negative logarithm of each probability has the desired effect, converting low probabilities to high penalties[2].

[2]The base of the logarithm is unimportant, as this will effectively be determined by training a weight for the function.

In addition, a small change to the configuration of GAZE_GeneID is required, swapping out the constant length penalty functions for exonic regions and swapping in the new ones. No other changes to the model are required, and I refer to the revised configuration as GAZE_GeneID$_{\text{exo}}$.

Obtaining optimal weights for the exon length penalties

In optimising the parameters for GAZE_GeneID earlier, separate weighting factors were attached to the constant length-penalty functions for each of the four exon types. The same is done here, with the added qualification that what is meant by weighting a non-constant length penalty function is that the penalties for all lengths are subject to this same scaling factor. The assumption here therefore is that the penalty functions derived from the given probability distributions are of the correct *shape*, but need to be weighted appropriately in relation to the other elements involved in the scoring function (see previous chapter).

Table 5.6 shows the result of optimising the parameters of the GAZE_GeneID$_{\text{exo}}$ model on the training set. The model has the same 8 parameters as the standard configuration, but as with the GAZE_GeneID$_{\text{tss}}$ model, a choice must be made whether to re-estimate all 8 parameters from scratch, or to optimise only the 4 parameters for the exon length penalties, anchoring the others to their values obtained by the earlier optimisation (GAZE_GeneID$^{\text{MFD}}$). As before then, the model is trained using the MFD method in both ways, and the table shows the results gained to be almost indistinguishable. Again though, there is no justifiable reason for why this might be true in general.

Examining the relative difference in exon-level prediction accuracy between trained versions of the standard and explicit-exon-length configurations, it is evident that the degree by which the latter out-performs the former is more marginal in the test set than in the training set. This implies a degree of over-fitting has taken occurred, supported by the base-level results. Nevertheless, improvements on the test set at all levels are however still apparent.

(a)

Base level	H176 (training)			HMR195 (test)		
	Sn	Sp	CC	Sn	Sp	CC
GAZE_GeneID$_{\text{exo}}^{\text{MFD}-4}$	0.86	0.88	0.85	0.81	0.90	0.83
GAZE_GeneID$_{\text{exo}}^{\text{MFD}-10}$	0.86	0.88	0.85	0.80	0.90	0.82
GAZE_GeneID$^{\text{MFD}}$	0.86	0.85	0.83	0.81	0.88	0.82

(b)

Exon level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	ME	WE	Sn	Sp	Av	ME	WE
GAZE_GeneID$_{\text{exo}}^{\text{MFD}-4}$	0.73	0.74	0.73	0.15	0.14	0.67	0.72	0.69	0.19	0.13
GAZE_GeneID$_{\text{exo}}^{\text{MFD}-8}$	0.73	0.74	0.73	0.15	0.14	0.66	0.71	0.69	0.20	0.13
GAZE_GeneID$^{\text{MFD}}$	0.69	0.70	0.69	0.17	0.16	0.64	0.69	0.66	0.20	0.14

(c)

Gene level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	MG	WG	Sn	Sp	Av	MG	WG
GAZE_GeneID$_{\text{exo}}^{\text{MFD}-4}$	0.34	0.28	0.31	0.03	0.20	0.29	0.24	0.27	0.05	0.16
GAZE_GeneID$_{\text{exo}}^{\text{MFD}-8}$	0.34	0.28	0.31	0.03	0.19	0.28	0.23	0.26	0.05	0.16
GAZE_GeneID$^{\text{MFD}}$	0.29	0.23	0.26	0.02	0.22	0.27	0.22	0.24	0.04	0.17

Table 5.6: The resulting accuracy of the GAZE model incorporating GENSCAN-derived exon length penalties (GAZE_GeneID$_{\text{exo}}$) trained using the Maximal Feature Discrimination method in two different ways as explained in the text. The results of using the same method to train the standard model (GAZE_GeneID), described earlier, are repeated here for comparison. (a) base-level accuracy; (b) exon-level accuracy; (c) gene-level accuracy. The accuracy measures are explained in section 1.4.1.

The ability to define arbitrary length penalty functions for any region type is perhaps the main advantages that GAZE has over other similar systems. It is therefore unfortunate that the treatment of penalty functions by the training procedure might be considered to be its weakest aspect, being reliant upon a pre-defined function with the correct shape. Ideally, the method would estimate an optimal shape for each penalty function directly from the training sequence. A possible way to address this would involve associating several weights to a single function, each being applied to a range of lengths (e.g. 1-50, 51-100 etc.), or at the extreme having a separate weight for each specific length.

An alternative approach would be to retain the binding of a single length function to a single weight, but to allow several functions to be associated to a particular $src \rightarrow tgt$ rule. This would require a simple extension to the scoring function, whereby the length-penalty component would be calculated as a sum of penalties taken from each penalty function specified in the rule. This approach is strictly more general than that above; a distinct weight for each distance can be represented by having a separate function for each distance which takes a positive number for the distance of interest and zero for all others[3]. Furthermore, it would allow the definition of composite functions, although it is not clear how such an approach would impact the search-space pruning strategy described in chapter 2, which assumes that it is possible to identify a point at which the penalty function for a rule becomes monotonically increasing (see section 2.6.3).

The main problem with both of these methods however is the large increase in the number of free variables in the optimisation. This will impact both the time taken to reach the function maximum, but also more importantly the number of training sequences that are required to obtain sensible estimates for this large number of parameters.

[3]although the definition of thousands or more penalty functions in a single $src \rightarrow tgt$ rule in the XML configuration file would be tedious at the very least!

5.5.3 Introducing C+G%-dependent model parameters

The natural way to address this problem is to divide the training sequences into pools (or *strata*) according to their C+G content, and then perform a separate training run for each, arriving at a distinct set of parameter values for each C+G% stratum. An early example of this approach is GENEPARSER [102] whereby separate neural networks were trained on each of three training sets containing respectively sequences with "low", "medium" and "high" C+G content. The boundaries for the categories were determined by calculating the mean C+G content of all full-length genes in Genbank, and classifying entries in the training set with C+G% more than one standard deviation away from this mean as "low" or "high" as appropriate.

By way of contrast, the training set of GENSCAN was initially divided into three according to the L1, L2, H1, H2 and H3 categories defined by Bernardi [7] and others: L1+L2 (less than 43% C+G), H1+H2 (43-51% C+G) and H3 (more than 51% C+G). Since the H3 group turned out to be excessively populated compared with the others, it was itself split into H3-1 (51-57% C+G) and H3-2 (more than 57% C+G). The parameters for the GHMM (e.g. the mean and variance of the geometric distributions for introns and intergenic regions, and certain transition and emission probabilities) were then derived separately from the sequences in each of the four datasets. The emission distribution for coding regions on the other hand was defined to have two parameterisations only, for low C+G% (L1+L2), and high (H1+H2 + H3-1 + H3-2).

The GENEID model for scoring coding regions differently according to C+G%

For all of the models so far, GENEID has been used as a source for the majority of the features and segments. It is fortunate therefore that GENEID is supplied with an additional parameter file for human gene finding that contains separate parameters for each of three C+G% strata: 0-45%, 45-55%, and 55-100% C+G. On inspection it was evident that the majority of parameters are the same between the

	I (0-45%)	II (45-55%)	III (55-100%)
H176	37	78	61
HMR195	38	107	50

Table 5.7: Breakdown of the training and test sequence sets by C+G content

three strata; only the parameters for the hexamer-based model for scoring coding regions display differences between strata. Just as the coding model defined in the original parameter file was used to make GAZE coding segments (see section 5.2.3), this time the C+G content of each sequence in the training and test sets was computed and coding segments were made by using the parameters corresponding to the appropriate C+G% stratum.

Table 5.7 shows the result of partitioning the training and test datasets according to the C+G% groupings of the three coding model used by GENEID. Interestingly, it shows the "medium" C+G% band to the most highly populated. Although apparently contrary to the GENSCAN findings outlined above, the difference can be explained by the higher boundary here between "medium" and "high" and strata.

Using GAZE with C+G%-stratified datasets

Unlike many other programs, there is no functionality in GAZE itself to account for varying parameters in sequences with different C+G content. However, one of the design aims of GAZE was to be sufficiently flexibility to allow different signal and content models, and even different models of gene structure, to be used in different situations. As a result of this design, it turns out to be straightforward to use GAZE with C+G%-specific parameters, and there are variety of ways in which this can be done.

The most natural approach is to compute the C+G content of the sequences in advance, and then construct GFF files of features and segments accordingly. This was done as explained above in obtaining C+G%-dependent coding segments for the

sequences. Then GAZE can be used exactly as before, with the same configuration file used for all the sequences. This method highlights one of the advantages of GAZE over many other existing systems in that it allows for variation across C+G% strata of not only the *parameters* of a set assumed underlying sequence-generating models for signal and content, but of the models themselves.

Another approach would be to pre-compute features and segments for *all* C+G% strata in advance and assess the C+G content at run-time. A simple Perl wrapper is then run in place of GAZE, which (i) computes the C+G content of the input sequence, and (ii) invokes GAZE with the appropriate GFF files. The method is more general in that it allows for the use of different GAZE configurations for the C+G% strata, allowing (among other things) model-element weightings, length penalty functions, and even gene structure models themselves that vary with C+G content.

I have used GAZE with the C+G%-stratified data in two ways. The first method was to use the default GAZE_GeneID configuration (my GAZE re-implementation of GENEID) on the H176 and H195 datasets, using for each sequence the set of coding segments that are appropriate for its C+G content. The results for this experiment are referred to as GAZE_GeneID$_{\mathrm{GC}}$ in table 5.8. An improvement in performance over the default model (i.e. that using the coding segments obtained using a global, C+G%-independent parameterisation of the GENEID coding model), at all levels of accuracy, is evident, most strikingly the correlation co-efficient.

Obtaining optimal weights for the C+G%-dependent models

The second way I have used GAZE with the C+G% stratified data is to use MFD training to optimise three specific sets of values for the model-element weights, one for each of the subsets of the sequences in the H176 (training) set having C+G content falling respectively into the ranges 0-45%, 45-55% and 55-100%. The result is three GAZE configurations, and the Perl wrapper mentioned above was used to obtain the results referred to in table 5.8 as GAZE_GeneID$_{\mathrm{GC}}^{\mathrm{MFD}}$.

(a)

Base level	H176 (training)			HMR195 (test)		
	Sn	Sp	CC	Sn	Sp	CC
GAZE_GeneID$_{\mathrm{GC}}$	0.92	0.86	0.88	0.92	0.87	0.87
GAZE_GeneID$^{\mathrm{MFD}}_{\mathrm{GC}}$	0.91	0.86	0.87	0.91	0.88	0.88
GAZE_GeneID	0.86	0.83	0.82	0.81	0.87	0.82
GAZE_GeneID$^{\mathrm{MFD}}$	0.86	0.85	0.83	0.81	0.88	0.82

(b)

Exon level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	ME	WE	Sn	Sp	Av	ME	WE
GAZE_GeneID$_{\mathrm{GC}}$	0.68	0.69	0.69	0.13	0.13	0.69	0.70	0.69	0.14	0.13
GAZE_GeneID$^{\mathrm{MFD}}_{\mathrm{GC}}$	0.72	0.73	0.73	0.14	0.13	0.70	0.74	0.72	0.17	0.13
GAZE_GeneID	0.64	0.67	0.66	0.17	0.14	0.61	0.65	0.63	0.19	0.13
GAZE_GeneID$^{\mathrm{MFD}}$	0.69	0.70	0.69	0.17	0.16	0.64	0.69	0.66	0.20	0.14

(c)

Gene level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	MG	WG	Sn	Sp	Av	MG	WG
GAZE_GeneID$_{\mathrm{GC}}$	0.11	0.11	0.11	0.01	0.09	0.16	0.14	0.15	0.01	0.09
GAZE_GeneID$^{\mathrm{MFD}}_{\mathrm{GC}}$	0.32	0.26	0.29	0.01	0.19	0.27	0.22	0.24	0.01	0.16
GAZE_GeneID	0.09	0.08	0.08	0.03	0.08	0.13	0.12	0.12	0.04	0.10
GAZE_GeneID$^{\mathrm{MFD}}$	0.29	0.23	0.26	0.02	0.22	0.27	0.22	0.24	0.04	0.17

Table 5.8: Accuracy of GAZE_GeneID using three C+G content dependent models for scoring coding regions; (a) base-level accuracy; (b) exon-level accuracy; (c) gene-level accuracy.

The same high correlation co-efficient is observed for the trained system as was seen in the untrained C+G%-stratified system. Marked improvements over the untrained system however are observed at the exon-level and the gene level, although these improvements are less striking when compared against the MFD-trained default, C+G%-independent model. These results suggest that the relatively high accuracy of the GAZE_GeneID^{MFD} system at all three levels is achieved at the base-pair level by the use of C+G%-stratified coding segments, and at the exon and gene levels by the MFD training of the model element weights.

The higher accuracy of gene prediction programs that take the C+G content of the underlying sequence into account has previously been explained by the observation that gene structural properties, such as average intron length and codon usage, vary according to this property (see chapter 1). However, the different sets of values for the weights obtained by training on sequences from each of the three C+G% strata are not representative of these structural differences, because the model itself is the same in each case (with the exception of the coding segments, explained above). Instead, different weight values for different C+G% strata implies that the relative importance of the model components varies according to C+G content. Although there is no biological reason to explain this result, it is consistent with previous observations that the difficulty with which localised signals for gene features can be detected also varies with C+G content. Burge has shown that the accuracy of discrimination between localised coding regions and non-coding regions is positively correlated with C+G content, and proposes this as a possible reason for the poor performance of gene prediction programs on A+T-rich sequences [20]. Levine on the other hand constructed a model for splice site detection and showed that the accuracy of discrimination between true and pseudo splice sites is poorest in sequences with high C+G content [74]. Based on these two observations, we might expect firstly the value for the splice site score weight obtained in the low C+G% optimisation to be higher than that obtained in the high C+G% optimisation, and vice-versa for the weights for the coding segments. This is indeed the case, with the

(a)

Base level	H176 (training)			HMR195 (test)		
	Sn	Sp	CC	Sn	Sp	CC
GAZE_GeneID$^{\text{MFD}}_{\text{all}}$	0.91	0.89	0.88	0.88	0.90	0.87
GENSCAN	0.97	0.86	0.90	0.95	0.86	0.89

(b)

Exon level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	ME	WE	Sn	Sp	Av	ME	WE
GAZE_GeneID$^{\text{MFD}}_{\text{all}}$	0.76	0.76	0.76	0.13	0.13	0.70	0.74	0.72	0.17	0.12
GENSCAN	0.82	0.75	0.79	0.06	0.15	0.77	0.73	0.75	0.08	0.14

(c)

Gene level	H176 (training)					HMR195 (test)				
	Sn	Sp	Av	MG	WG	Sn	Sp	Av	MG	WG
GAZE_GeneID$^{\text{MFD}}_{\text{all}}$	0.37	0.31	0.34	0.02	0.18	0.33	0.27	0.30	0.02	0.17
GENSCAN	0.40	0.35	0.37	0.01	0.13	0.37	0.33	0.35	0.02	0.12

Table 5.9: Accuracy of GAZE_GeneID$_{\text{all}}$ (which includes transcription start sites, exon length penalties and C+G% dependency) when trained with MFD. The results for GENSCAN are repeated for ease of comparison. The accuracy measures are explained in section 1.4.1.

splice site weights for the low and high C+G% strata being 0.96 and 0.92, and the coding segment weights for these strata being 0.54 and 0.61.

5.5.4 Combining all three types of evidence

By way of postscript, it is interesting to consider the effect of including the innovations of the past three sections in one system, training the weights using Maximal Feature Discrimination. The trained system is referred to as GAZE_GeneID$^{\text{MFD}}_{\text{all}}$ in table 5.9.

Although the accuracy of the model as a whole is the best achieved by any GAZE model so far, it is still marginally out-performed by GENSCAN. One reason

for this could be the relative sophistication of the signal models used for donor and acceptor splice sites compared with those employed by GENEID. The splice acceptor model used by GENSCAN includes the branch-point region between 21 and 38 nucleotides upstream of the conserved AG. This specific region is modelled with a "windowed" weight array, allowing the capture of second-order dependencies. The GENEID splice acceptor model on the other hand does not extend upstream as far as the branch-point (see section 5.2.3). In addition, GENSCAN uses Maximal Dependence Decomposition for the modelling of the donor splice signal, allowing the capture of long-range dependencies, whereas GENEID uses a simple weight matrix (see section 5.2.3 and [20]).

It has been shown that these sophisticated models can be more accurate than simpler models (such as those used by GENEID) when judged by an "isolated" test of the discrimination between true and pseudo splice sites [20]. A natural extension to the work described here would involve implementing these models in a program that outputs scored splice site predictions in GFF. GAZE could then be used to investigate their accuracy when judged in an "integrated" test of their influence on the prediction of complete gene structures.