Journal of Bioinformatics and Computational Biology

Data Mining Tools for Biological Sequences

Huiqing Liu
Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
huiqing@i2r.a-star.edu.sg

Limsoon Wong
Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
limsoon@i2r.a-star.edu.sg

Received 4 April 2003
Revised 7 April 2003
Accepted 7 April 2003

We describe a methodology, as well as some related data mining tools, for analyzing sequence data. The methodology comprises three steps: (a) generating candidate features from the sequences, (b) selecting relevant features from the candidates, and (c) integrating the selected features to build a system to recognize specific properties in sequence data. We also give relevant techniques for each of these three steps. For generating candidate features, we present various types of features based on the idea of k-grams. For selecting relevant features, we discuss signal-to-noise, t-statistics, and entropy measures, as well as a correlation-based feature selection method. For integrating selected features, we use machine learning methods, including C4.5, SVM, and Naive Bayes. We illustrate this methodology on the problem of recognizing translation initiation sites. We discuss how to generate and select features that are useful for understanding the distinction between ATG sites that are translation initiation sites and those that are not. We also discuss how to use such features to build reliable systems for recognizing translation initiation sites in DNA sequences.

Keywords: Translation initiation sites; data mining; machine learning; feature generation; feature selection; feature integration; k-grams; stop codons.

1. Overview

Powerful data mining tools developed in the Computer Science community can be fruitfully applied to Bioinformatics. However, how such techniques can be applied is nontrivial. This tutorial is intended to introduce to the biologists some of these powerful data mining techniques, and to illustrate how they can be applied using one example.

Let us begin by noting that biomedical data are quite different from business transaction data that most of the data mining tools developed in the Computer Science community are designed for. In particular, not all biomedical data contain

explicit signals or features.

DNA sequences and protein sequences represent the spectrum of biomedical data that possess no explicit features. For example, a genomic sequence is typically just a string consisting of the letters "A", "C", "G", and "T" in an apparently random order. Yet a genomic sequence possesses biologically meaningful functional sites such as transcription start site, translation initiation site, and so on that are associated with the primary structure of genes. Recognition of these biological functional sites in a genomic sequence is an important bioinformatics application[4].

Traditional machine learning and data mining techniques cannot be straightforwardly applied to this type of recognition problems to obtain good results. So there is a need to adapt these existing techniques to this type of problems and a need for good techniques for generating explicit features underlying such functional sites. In this tutorial, we describe a methodology and related techniques for this type of recognition problems. To make our presentation more concrete, we use the recognition of Translation Initiation Sites (TIS) in vertebrate sequences as our working example.

In one of the earlier works on the TIS recognition problem, Pedersen and Nielsen[78] directly feed DNA sequences into an Artificial Neural Network (ANN) to train the system to recognize TIS. The performance of the system is promising, as its accuracy is 85.0%. In Zien *et al.*[111] and Hatzigeorgiou[42], improved accuracy of 88.6% and 94.0% is obtained by more complex methods: a careful engineering of kernel functions for a Support Vector Machine (SVM) and a multi-step integrated ANN. More recent works of Wong *et al.*[108] and Li *et al.*[63] show that good performance can be obtained using a methodology comprising the following three straightforward steps: (a) generate candidate features from the sequences using the idea of k-grams, (b) select relevant features from these candidates using entropy or other statistical measures, and (c) integrate the selected features for decision making using a standard machine learning method. These selected features are highly informative on the distinction of TIS from non-TIS. Moreover, these two recent works indicate that so long as the features are selected well, any machine learning method can be used in step (c) to obtain good accuracy for recognizing TIS and other functional sites in sequences.

We provide in Section 2 some biological background on protein translation initiation sites. The approaches of Pedersen and Nielsen[78], Zien *et al.*[110,111], Hatzigeorgiou[42], Wong *et al.*[108], and Li *et al.*[63] are then briefly reviewed. The dataset of Pedersen and Nielsen is then presented in Section 3.

After that, we devote our attention to a detailed exposition of the methodology of Wong *et al.*[108] and Li *et al.*[63] applied to recognition of important functional sites from sequence data, using TIS as our working example. We deal with the three steps of this methodology as follows. Section 4 introduces a few good techniques for generating candidate features from mRNA and DNA sequences. The techniques are centered around the basic ideas of k-grams[109,3]. Section 5 explains several methods for ranking the usefulness of candidate features for classification

problems, including the entropy measure[29] and the Correlation-based Feature Selection method (CFS)[39]. The CFS is then applied to select 9 features that are relevant for TIS recognition. Most of these 9 features turn out to have some underlying biological reasons. Section 6 gives brief overviews of three machine learning methods, *viz.* C4.5[82], SVM[103], and Naive Bayes (NB)[59]. Each of these methods are then trained on the features selected in Section 5 to recognize TIS.

Then, we introduce in Section 7 some additional techniques for generating features, via a translation of the original mRNA and DNA sequences to amino acid sequences. The entropy measure discussed in Section 5 is then used to select relevant features. The C4.5, NB, and SVM algorithms are then trained using these selected features and used to recognize TIS.

Finally, we provide some concluding remarks in Section 8 together with a survey of additional data mining tools that have been used in biomedicine.

2. Background on Recognition of Translation Initiation Sites

Proteins are synthesized from mRNAs by a process called translation. The process can be divided into three distinct stages: initiation, elongation of the polypeptide chain, and termination[15]. The region at which the process initiates is called the Translation Initiation Site (TIS). The coding sequence is flanked by non-coding regions which are the 5' and 3' untranslated regions respectively. The translation initiation site prediction problem is to correctly identify TIS in a mRNA, cDNA, or genomic sequence. This forms an important step in genomic analysis to determine protein coding from nucleotide sequences.

In eukaryotes, the scanning model postulates that the ribosome attaches first to the 5' end of the mRNA and scans along the 5'-to-3' direction until it encounters the first AUG[54]. While this simple rule of first AUG holds in many cases, there are also exceptions. Some mechanisms proposed to explain the exceptions are: leaky scanning where the first AUG is bypassed for reasons such as poor context; reinitiation where a short up-stream open reading frame causes a second initiation to occur; and also other alternative proposed mechanisms[54,38]. Translation can also occur with non-AUG codons, however this is reported to be rare in eukaryotes[54] and is not considered here.

The problem of recognizing TIS is compounded in real-life sequence analysis by the difficulty of obtaining full-length and error-free mRNA sequences. Pedersen and Nielsen found that almost 40% of the mRNAs extracted from GenBank contain up-stream AUGs[78]. The problem becomes more complex when using unannotated genome data or analyzing expressed sequence tags (ESTs), which usually contain more errors, and are not guaranteed to give the correct 5' end[10,11]. Thus, the prediction of the correct TIS is a non-trivial task since the biological mechanisms are not fully understood and sequences may have errors and may not be complete.

In Pedersen and Nielsen[78], an artificial neural network (ANN) was trained on a 203 nucleotide window centered on the AUG. It is a feed-forward ANN with three

layers of neurons. Inputs are presented to this ANN by encoding each of nucleotide by four binary digits, *viz.* 0001 for "A", 0010 for "C", 0100 for "G", and 1000 for "T". The output layer has two neurons. The first neuron predicts if the input is a TIS. The second neuron predicts if the the input is a non-TIS. Whichever of these two neurons gives the bigger score wins. According to Pedersen and Nielsen[78], the number of neurons in the hidden layer of the ANN does not significantly affect the performance of the ANN. They obtain results of 78% sensitivity on start AUGs and 87% specificity on non-start AUGs on their vertebrate dataset, giving an overall accuracy of 85%. Their system is available on the Internet as the NetStart 1.0 prediction server accessible at *http://www.cbs.dtu.dk/services/NetStart*.

Pedersen and Nielsen[78] also carry out additional analysis to try to uncover features in their sequences that are important for distinguishing TIS from non-TIS. In one of the analysis, they supply their neural network with input windows which covered the aforementioned 203 nucleotides, except for for one position—a " hole"—from which input was disregarded. The hole is shifted along the input window in a series of runs of the neural network and the impact of the hole in each position is noted. This experiment reveals that position –3 is crucial to TIS recognition, as the error rate of the neural network drops precipitously when a hole is present in this position. Pedersen and Nielsen[78] also analyse the positions of non-translation initiating ATGs that are misclassified by their neural network as TIS. In this analysis, they discover that ATGs that are in-frame to the TIS are more likely to be misclassified as TIS regardless of whether they are up-stream or down-stream of the TIS.

Zien *et al.*[110,111] work on the same vertebrate dataset from Pedersen and Nielsen by using SVM instead. The same 203 nucleotide window is used as the underlying features to be learnt. Each nucleotide is encoded using the same sparse binary encoding as Pedersen and Nielsen. Homogeneous polynomial kernels[92] of degree d, $k(X,Y) = (X \cdot Y)^d = \sum_{i_1} \ldots \sum_{i_d} [X]_{i_1} * \ldots * [X]_{i_d} * [Y]_{i_1} * \ldots * [Y]_{i_d}$, are commonly used in SVM. Due to the encoding used for nucleotides, the position of each bit that is set indicates it is A, C, G, or T. Consequently, the dot product $(X \cdot Y)$ is equivalent to a count of the number of nucleotides that coincide in the two sequences represented by vectors X and Y. Similarly, $(X \cdot Y)^d$ is equivalent to a correlation of the nucleotide frequencies at any d sequence positions. Zien *et al.*[110] report that SVM achieves TIS recognition performance comparable to Pedersen and Nielsen's ANN using this standard type of kernels.

Zien *et al.*[110] also show how to obtain improvements by appropriate engineering of the kernel function—using a locality-improved kernel with a small window on each position. The locality-improved kernel emphasizes correlations between sequence positions that are close together, and a span of 3 nucleotides up- and down-stream is empirically determined as optimal. The locality-improved kernel is thus defined as $k(X,Y) = \sum_{p=1}^{l} win_p(X,Y)$, where $win_p(X,Y) = (\sum_{j=-3}^{3} w_j * match_{p+j}(X,Y))^4 = \sum_{j_1=-3}^{3} \cdots \sum_{j_4=-3}^{3} w_{j_1} * match_{p+j_1}(X,Y) * \ldots *$

$w_{j_4} * match_{p+j_4}(X, Y)$, w_j is an appropriate weight, and $match_{p+j}(X, Y) = 1$ if the nucleotides at position $p + j$ of X and Y are the same and $= 0$ otherwise. With the locality-improved kernel[110], they obtain an accuracy of 69.9% and 94.1% on start and non-start AUGs respectively, giving an overall accuracy of 88.1%.

Zien *et al.*[111] further improve their previous results by engineering a more sophisticated kernel—a so-called Salzberg kernel—using conditional positional probabilities. The Salzberg kernel gives an overall accuracy of 88.6%.

Hatzigeorgiou[42] reports a highly accurate TIS prediction program, DIANA-TIS, using artificial neural networks trained on human sequences. Their dataset contains full-length cDNA sequences which has been filtered for errors. An overall accuracy of 94% is obtained using an integrated method which combines a consensus ANN with a coding ANN together with the ribosome scanning model. The consensus ANN assesses the candidate TIS and its immediate surrounding comprising a window from positions −7 to +5 relative to the candidate TIS. The consensus ANN is a feed-forward ANN with short cut connections and two hidden units. The coding ANN assesses the coding potential of the regions up-stream and down-stream of TIS. The coding ANN works on a window of 54 nucleotides. As every three nucleotides form a codon that translates into an amino acid, there are 64 possible codons. To assess the coding potential of the window of 54 nucleotides, this window is transformed into a vector of 64 units before feeding into the coding ANN. Each unit represents one of the 64 codons and gives the normalized frequency of the corresponding codon appearing in the window. The coding ANN is a feed-forward ANN and has two hidden units, but no short cut connection.

Note that in the ribosome scanning model[1,54], an mRNA sequence is scanned from left to right, testing each ATG in turn until one of them is classified as TIS; all the ATGs to the right of this ATG are skipped and classified as non-TIS. In short, exactly one prediction is made per mRNA under the ribosome scanning model. Hence, accuracy figures based on the ribosome scanning model should not be compared with models that tests every ATG. In addition, Hatzigeorgiou uses a dataset that is different from Pedersen and Nielsen. So her results cannot be directly compared to results obtained on the Pedersen and Nielsen dataset or obtained without using the ribosome scanning model.

Wong *et al.*[108] show that good performance comparable to the best results can be obtained by a methodology based on the following three steps: (a) generate candidate features from the sequences, (b) select relevant features from the candidates, and (c) integrate the selected features using a machine learning method to build a system to recognize specific properties—in this case, TIS—in sequence data. The highest overall accuracy they obtained on the Pedersen and Nielsen dataset is 89.4% using C4.5[82] trained on 9 highly informative features. When this classifier is used together with the ribosome scanning model[54], they obtain an overall accuracy of 94.4%. Their approach and results will be presented and discussed in greater detail in Sections 4, 5, and 6.

Li *et al.*[63] refines the approach of Wong *et al.*[108] by generating features based on a translation to amino acids instead of the original DNA sequences. They then use a data mining method, PCL[64], based on prediction by likelihood of emerging patterns derived from amino acid sequence translated from DNA sequences. They describe results on the same vertebrate dataset from Pedersen and Nielsen where PCL predicts 84.7% and 88.7% of start AUGs and non-start AUGs respectively, and an overall accuracy of 87.7%. This refinement will be discussed in Section 7, together with some new results that we have obtained while writing this tutorial.

Although the accuracy of 87.7% of PCL in the study of Li *et al.*[63] appears to be weaker than the 88.1% of Zien *et al.*[110] and 89.4% of Wong *et al.*[108], one may want to take sensitivity into consideration. The sensitivity of Zien *et al.*[110] is 69.9% and that of Wong *et al.*[108] is 74.0%, while that of PCL[63] is 84.7%. The non-start AUGs out-number the start AUGs in a ratio of 3 to 1 in the Pedersen and Nielsen dataset. The non-start AUGs out-number the start AUGs in even greater ratios if we are doing a genome-wide scan for TIS. Therefore, the superior sensitivity of PCL may be more desirable in such a situation.

These performance results are summarized in Part I of Figure 3.

The approach of Pedersen and Nielsen[78] is interesting in that their inputs are extremely low level—just a string of nucleotides—and relies entirely on their ANN to learn high-level correlations to make prediction. Unfortunately, it is not easy to extract these correlations out of their ANN to gain more insight into sequence features that distinguish TIS from non-TIS. Nevertheless, by more extensive experiments and analysis, Pedersen and Nielsen are able to suggest that position –3 is crucial to distinguishing TIS from non-TIS.

The approach of Zien *et al.*[110,111] and Hatzigeorgiou[42] are also very interesting in that they show us how to perform sophisticated engineerings of SVM kernel functions and ANNs. Unfortunately, it is also not easy to extract more insight into sequence features that distinguish TIS from non-TIS from their systems. Nevertheless, the improved results of the locality-improved kernel in Zien *et al.*[110] over that of standard polynomial kernels suggest that local correlations are more important than long-ranged correlations in distinguishing TIS from non-TIS.

The approach of Wong *et al.*[108] and Li *et al.*[63] is interesting in that they focus on deriving high-level understandable features first, and then use these features to distinguish TIS from non-TIS. In the remainder of this tutorial, we use the dataset of Pedersen and Nielsen to discuss this approach in greater depth.

3. Dataset

We use the vertebrate dataset provided by Pedersen and Nielsen[78]. This dataset was also used by Zien *et al.*[110,111] These sequences are processed by removing possible introns and joining the exons[78] to obtain the corresponding mRNA sequences. From these sequences, only those with an annotated TIS, and with at least 10 upstream nucleotides as well as 150 downstream nucleotides are selected. The sequences are

```
299 HSU27655.1 CAT U27655 Homo sapiens
CGTGTGTGCAGCAGCCTGCAGCTGCCCCAAGCCATGGCTGAACACTGACTCCCAGCTGTG 80
CCCAGGGCTTCAAAGACTTCTCAGCTTCGAGCATGGCTTTTGGCTGTCAGGGCAGCTGTA 160
GGAGGCAGATGAGAAGAGGGAGATGGCCTTGGAGGAAGGGAAGGGGCCTGGTGCCGAGGA 240
CCTCTCCTGGCCAGGAGCTTCCTCCAGGACAAGACCTTCCACCCAACAAGGACTCCCCT
.............................................................. 80
.............................iEEEEEEEEEEEEEEEEEEEEEEEEE 160
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE 240
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
```

Fig. 1. An example annotated sequence from the dataset of Pedersen and Nielsen. The 4 occurrences of ATG are underlined. The second ATG is the TIS. The other 3 ATGs are non-TIS. The 100 nucleotides up-stream of the TIS are marked by an overline. The 100 nucleotides down-stream of the TIS are marked by a double overline. The ".", "i", and "E" are annotations indicating whether the corresponding nucleotide is up-stream (.), TIS (i), or down-stream (E).

then filtered to remove those belonging to same gene families, homologous genes from different organisms and sequences added multiple times to the database. Since the dataset is processed DNA, the TIS site is ATG—that is, a place in the sequence where "A", "T", and "G" occur in consecutive positions in that order.

An example entry from this dataset is given in Figure 1. There are 4 ATGs in the example sequence shown. The second ATG is the TIS. The other 3 ATGS are non-TIS. ATGs to the left of the TIS are termed up-stream ATGs. So the first ATG in the figure is an up-stream ATG. ATGs to the right of the TIS are termed down-stream ATGs. So the third and fourth ATGs in the figure are down-stream ATGs.

There are a total of 13375 ATG sites in the Pedersen and Nielsen dataset. Of these ATG sites, 3312 (24.76%) are the true TIS, while the other 10063 (75.23%) are non-TIS. Of the non-TIS, 2077 (15.5%) are up-stream of a true TIS. At each of these ATG sites, we extract a sequence segment consisting of 100 nucleotides up-stream and 100 nucleotides down-stream. These 13375 sequence segments are the inputs upon which we perform the recognition of TIS. If an ATG does not have enough up-stream or down-stream context—that is, there are less than 100 nucleotides to its left or to its right—we pad the missing context with the appropriate number of dont-care symbols.

In the rest of this tutorial, we discuss techniques for identifying features in these sequence segments that are relevent for recognizing TIS. After that, we apply one of these techniques to select good features. Then we show that a number of different machine learning methods trained on these good features all have good accuracy. The selection of features and the training and testing of the machine learning methods are all performed with a 3-fold cross validation process. That is, the dataset is divided into three equal parts, and each part is in turn reserved for testing the classification trained on the features selected from the other two parts

of the data[41].

4. Feature Generation

The sequence segments prepared in the previous section are not suitable for direct application of most machine learning techniques, as these techniques rely on explicit signals of high quality. It is necessary to devise various signals and sensors for these signals, so that given a sequence segment, a score is produced to indicate the possible presence of such a signal in that sequence segment. The obvious strategy to devising these signals and sensors is to generate a large number of candidate features, and to evaluate them against an annotated dataset to decide which ones are signals and which ones are noise.

A general technique for producing these candidate features is the idea of k-grams frequency[109,3]. A k-gram is simply a pattern of k consecutive letters, which could be amino acid symbols or nucleic acid symbols. K-grams can also be restricted those in-frame ones. Each k-gram and its frequency in the said sequence fragment becomes a candidate feature. Another general technique for producing these candidate features is the idea of position-specific k-gram. The sensor for such a feature simply reports what k-gram is seen in a particular position in the sequence fragment.

For ease of discussion, given a sequence segment, we refer to each position in the sequence segment relative to the target ATG of that segment. The "A" in the target ATG is numbered as +1 and consecutive down-stream positions—that is, to the right—from the target ATG are numbered from +4 onwards. The first up-stream position—that is, to the left—adjacent to the target ATG is –1 and decreases for consecutive positions towards the 5' end—that is, the left end of the sequence fragment.

Let us use the sequence segment centered at the second ATG in Figure 1 and comprising 100 nucleotides up-stream and 100 nucleotides down-stream of this ATG for illustration of the various k-gram features described above. These up-stream and down-stream nucleotides are marked using overline and double overline in the figure.

For the basic k-grams, k is the length of a nucleotide pattern to be generated. Some typical values for k are 1, 2, 3, 4, and 5. Since there are 4 possible letters for each position, there are 4^k possible basic k-grams for each value of k. For example, for k = 3, one of the k-grams is ATG and the frequency of this k-gram is 4 in our example sequence segment. The candidate feature (and value assignment) corresponding to this is "ATG=4".

The up-stream region of a TIS is non-coding and the down-stream region of a TIS is coding. We can therefore expect they have some different underlying features. So it is wise to introduce additional classes of k-grams to attempt to capture these differences. We call these the up-stream and down-stream k-grams.

For the up-stream k-grams, we count only occurrences of the corresponding patterns up-stream of the target ATG. Again, for each value of k, there are 4^k up-stream k-grams. For example, for k = 3, some k-grams are: ATG, which has

frequency 1 in this context; GCT, which has frequency 5 in this context; and TTT, which has frequency 0 in this context. The candidate features and value assignments corresponding to these k-grams are "up-stream ATG=1", "up-stream GCT=5", and "up-stream TTT=0".

For the down-stream k-grams, we count only occurences of the corresponding patterns down-stream of the target ATG. Again, for each value of k, there are 4^k down-stream k-grams. For example, for k = 3, some k-grams are: ATG, which has frequency 2 in this context; GCT, which has frequency 3 in this context; and TTT, which has frequency 2 in this context. The candidate features and value assignments corresponding to these k-grams are "down-stream ATG=2", "down-stream GCT=3", and "down-stream TTT=2".

The biological process of translating from nucleotides to amino acids is to have 3 nucleotides—the so-called codons—codes for one amino acid, starting from the TIS. Therefore, 3-grams in positions ..., -9, -6, -3, +4, +7, +10, ... are aligned to the TIS. We call those 3-grams in positions ..., -9, -6, and -3, the in-frame up-stream 3-grams, and those 3-grams in positions +4, +7, +10, ..., the in-frame down-stream 3-grams. As these 3-grams are in positions that have biological meaning, they are also good candidate features. There are $2*4^3$ such 3-grams. In our example sequence fragment, some in-frame down-stream 3-grams are: GCT, which has frequency 1 in this context; TTT, which has frequency 1 in this context; ATG, which has frequency 1 in this context. The corresponding candidate features and value assignments are "in-frame down-stream GCT=1", "in-frame down-stream TTT=1", and "in-frame down-stream ATG=1". Some in-frame up-stream 3-grams are: GCT, which has frequency 2 in this context; TTT, which has frequency 0 in this context; ATG, which has frequency 0 in this context. The corresponding candidate features and value assignments are "in-frame up-stream GCT=2", "in-frame up-stream TTT=0", and "in-frame up-stream ATG=0".

Another type of features are what we call the position-specific k-grams. For this type of k-grams, we simply record which k-gram appears in a particular position in the sequence segment. We consider only 1-grams, that is, k-grams for k = 1. Since our sequence segment has 100 nucleotides flanking each side of the target ATG, there are 200 position-specific 1-grams. In our example sequence segment, some position-specific 1-grams are: at position +4 is G and at position –3 is G. The corresponding candidate features and value assignments are "position+4=G" and "position–3=G".

Combining all the features discussed above, each sequence segment is coded into a record having $(\sum_{k=1}^{5} 4^k + 4^k + 4^k) + 2 * 4^3 + 200 = 4436$ features. For illustration, our example sequence segment is coded into this record: {..., "ATG=4", ..., "up-stream ATG=1", "up-stream GCT=5", "up-stream TTT=0", ..., "down-stream ATG=2", "down-stream GCT=3", down-stream TTT=2", ..., "in-frame down-stream GCT=1", "in-frame down-stream TTT=1", "in-frame down-stream ATG=1", ..., "in-frame up-stream GCT=2", "in-frame up-stream TTT=0", "in-frame up-stream ATG=0", ..., ..., position–3=G, ..., position+4=G, ...}. Such a

record is often called a feature vector.

These 4436 features as described above are generated for each of the 13375 sequence segments corresponding to the 13375 ATG sites in the Pedersen and Nielsen dataset[78]. We note that other techniques for generating candidate features are possible. For example, we can compute a specific statistic on the sequence segment, such as its GC ratio. Specific biological knowledge can also be used to devise specialized sensors and features, such as CpG island[34], Kozak consensus pattern[55], etc. For certain problems, such as exon recognition, a "shifting window" technique is also used to generate signals. A window is taken on the sequence fragment and the coding potential[30] of this window is computed to give a signal, and the window is then repeatedly shifted to compute more such signals. An exhaustive exposition is outside the scope of our tutorial. In the next two sections, we show how to reliably recognize TIS on the basis of a small subset of our 4436 candidate features.

5. Feature Selection

Recall that the total number of candidate features generated as described in the previous section is 4436. This high dimensionality causes three problems. The first problem is that of efficiency because most data mining and machine learning methods have time complexity that are high with respect to the number of dimensions[40]. The second problem is that of noise because most data mining and machine learning methods suffer from the "curse of dimensionality"—these methods typically require an exponential increase in the number of training samples with respect to an increase in the dimensionality of the samples in order to uncover and learn the relationship of the various dimensions to the nature of the samples[41]. The third problem is that it would be difficult for anyone to attain a useful understanding of the underlying problem out of 4436 features.

Let us assume that we have two classes $\mathcal{A}$ and $\mathcal{B}$ of samples. For example, $\mathcal{A}$ could be the feature vectors corresponding to the TIS sequence segments and $\mathcal{B}$ could be the feature vectors corresponding to the non-TIS sequence segments. Then a feature is relevant if it contributes to separating samples in $\mathcal{A}$ from those in $\mathcal{B}$. Conversely, a feature is irrelevant if it does not contribute much to separating samples in $\mathcal{A}$ from those in $\mathcal{B}$. In order to alleviate the impact of the three problems caused by high dimensionality mentioned above, it is desirable to first discard as many features that are irrelevant as possible.

In this section, we present several techniques for deciding whether a feature is relevant, *viz.* t-statistics[19], signal-to-noise[37], and entropy measures[29], as well as the correlation-based feature selection method known as CFS[39]. Then we apply the CFS method to the Pedersen and Nielsen dataset and discuss the relevant features selected.

A basic concept for distinguishing a relevant feature from an irrelevant one is the following: if the values of a feature in samples in $\mathcal{A}$ are significantly different from the values of the same feature in samples in $\mathcal{B}$, then the feature is likely to be

more relevant than a feature that has similar values in $\mathcal{A}$ and $\mathcal{B}$. More specifically, in order for a feature f to be relevant, its mean value $\mu_f^{\mathcal{A}}$ in $\mathcal{A}$ should be significantly different from its mean value $\mu_f^{\mathcal{B}}$ in $\mathcal{B}$. However, if the values of a feature f varies greatly within the same class of samples, even if $\mu_f^{\mathcal{A}}$ differs greatly from $\mu_f^{\mathcal{B}}$, the feature f is not a reliable one. This defficiency gives us a second basic concept: the standard deviation $\sigma_f^{\mathcal{A}}$ and variance $(\sigma_f^{\mathcal{A}})^2$ of f in $\mathcal{A}$ and the standard deviation $\sigma_f^{\mathcal{B}}$ and variance $(\sigma_f^{\mathcal{B}})^2$ of f in $\mathcal{B}$ should be small.

One obvious way to combine these two concepts is the **signal-to-noise measure** proposed in the first paper[37] that applied gene expression profiling for disease diagnosis:

$$s(f, \mathcal{A}, \mathcal{B}) = \frac{|\mu_f^{\mathcal{A}} - \mu_f^{\mathcal{B}}|}{\sigma_f^{\mathcal{A}} + \sigma_f^{\mathcal{B}}}$$

However, the statistical property of $s(f, \mathcal{A}, \mathcal{B})$ is not fully understood. Subsequently, a second and older way—the t-test—to combine these two concepts was rediscovered. The classical t-test statistical measure[8,19] is known to follow a Student distribution with $((\sigma_f^{\mathcal{A}})^2/n^{\mathcal{A}} + (\sigma_f^{\mathcal{B}})^2/n^{\mathcal{B}})^2/((((\sigma_f^{\mathcal{A}})^2/n^{\mathcal{A}})^2/(n^{\mathcal{A}} - 1)) + (((\sigma_f^{\mathcal{B}})^2/n^{\mathcal{B}})^2/(n^{\mathcal{B}} - 1)))$ degrees of freedom, where $n^{\mathcal{A}}$ and $n^{\mathcal{B}}$ are the number of samples in $\mathcal{A}$ and $\mathcal{B}$. The **t-test statistical measure** is given below:

$$t(f, \mathcal{A}, \mathcal{B}) = \frac{|\mu_f^{\mathcal{A}} - \mu_f^{\mathcal{B}}|}{\sqrt{(\sigma_f^{\mathcal{A}})^2/n^{\mathcal{A}} + (\sigma_f^{\mathcal{B}})^2/n^{\mathcal{B}}}}$$

Both of these measures are easy to compute and thus straightforward to use. However, these measures have two important defficiencies in the situations described below. The first situation is where the population sizes $n^{\mathcal{A}}$ and $n^{\mathcal{B}}$ are small. Small population sizes can lead to significant underestimates of the standard deviations and variances. The second situation is more subtle and we explain using the following example. Let f_1 and f_2 be two features. Suppose f_1 has values ranging from 0 to 99 in class $\mathcal{A}$ with $\mu_{f_1}^{\mathcal{A}} = 75$ and has values ranging from 100 to 199 in class $\mathcal{B}$ with $\mu_{f_1}^{\mathcal{B}} = 125$. Suppose f_2 has values ranging from 25 to 125 in class $\mathcal{A}$ with $\mu_{f_2}^{\mathcal{A}} = 50$ and has values ranging from 100 to 175 in class $\mathcal{B}$ with $\mu_{f_2}^{\mathcal{B}} = 150$. We see that $\mu_{f_2}^{\mathcal{B}} - \mu_{f_2}^{\mathcal{A}} = 100 > 50 = \mu_{f_1}^{\mathcal{B}} - \mu_{f_1}^{\mathcal{A}}$. Suppose the variances of f_1 and f_2 in $\mathcal{A}$ and $\mathcal{B}$ are comparable. Then according to the signal-to-noise and t-statistics measures, f_2 is better than f_1. However, we note that the values of f_1 are distributed so that all those in $\mathcal{A}$ are below 100 and all those in $\mathcal{B}$ are at least 100. In contract, the values of f_2 in $\mathcal{A}$ and $\mathcal{B}$ overlap in the range 100 to 125. Then clearly f_1 should be preferred. The effect is caused by the fact that $t(f, \mathcal{A}, \mathcal{B})$ and $s(f, \mathcal{A}, \mathcal{B})$ are sensitive to all changes in the values of f, including those changes that may not be important.

So, one can consider alternative statistical measures that are less sensitive to certain types of unimportant changes in the value of f. What types of changes in values of f are not important? One obvious type is those that do not shift the

values of f from the range of $\mathcal{A}$ into the range of $\mathcal{B}$. An alternative that takes this into consideration is the entropy measure[29].

Let $P(f,\mathcal{C},S)$ be the proportion of samples whose feature f has value in the range S and are in class $\mathcal{C}$. The *class entropy* of a range S with respect to feature f and a collection of classes $\mathcal{U}$ is defined as $Ent(f,\mathcal{U},S) = -\sum_{\mathcal{C}\in\mathcal{U}} P(f,\mathcal{C},S)log(P(f,\mathcal{C},S))$. It follows from this definition that the purer the range S is, so that samples in that range are more dominated by one particular class, the closer $Ent(f,\mathcal{U},S)$ is to the ideal entropy value of 0. Let T partitions the values of f into two ranges S_1 (of values less than T) and S_2 (of values at least T). We sometimes refer to T as the **cutting point** of the values of f. The **entropy measure** $e(f,\mathcal{U})$ of a feature f is then defined as $\min\{E(f,\mathcal{U},S_1,S_2) \mid (S_1,S_2)$ is a partitioning of the values of f in $\bigcup\mathcal{U}$ by some point $T\}$. Here, $E(f,\mathcal{U},S_1,S_2)$ is the *class information entropy* of partition (S_1,S_2). The definition is given below, where $n(f,\mathcal{U},S)$ means the number of samples in the classes in $\mathcal{U}$ whose feature f has value in the range S,

$$E(f,\mathcal{U},S_1,S_2) = \frac{n(f,\mathcal{U},S_1)}{n(f,\mathcal{U},S_1 \cup S_2)}Ent(f,\mathcal{U},S_1) + \frac{n(f,\mathcal{U},S_2)}{n(f,\mathcal{U},S_1 \cup S_2)}Ent(f,\mathcal{U},S_2)$$

A refinement of the entropy measure is to recursively partition the ranges S_1 and S_2 until some stopping criteria is reached[29]. A commonly used stopping criteria is the so-called minimal description length principle. Another refinement is the χ^2 measure[65]. Here, instead of the entropy measure $e(f,\{\mathcal{A},\mathcal{B}\})$ itself, we use the χ^2 correlation of the partitions S_1 and S_2 induced by the entropy measure to the classes $\mathcal{A}$ and $\mathcal{B}$. Some other refinements include the information gain measure and the information gain ratio that are used respectively in ID3[81] and C4.5[82] to induce decision trees. These two measures are presented later in Section 6.

All of the preceeding measures provide a rank ordering of the features in terms of their individual relevance to separating $\mathcal{A}$ and $\mathcal{B}$. One would rank the features using one of these measures and select the top n features. However, one must appreciate that there may be a variety of independent "reasons" why a sample is in $\mathcal{A}$ or is in $\mathcal{B}$. For example, there can be a number of different contexts and mechanisms that enable protein translation. If a primary context or mechanism involves n signals, the procedure above may select only these n features and may ignore signals in other secondary contexts and mechanisms. Consequently, concentrating on such top n features may cause us to lose sight of the secondary contexts and mechanisms underlying protein translation.

This issue above calls for a third concept in feature selection: select a group of features that are correlated with separating $\mathcal{A}$ and $\mathcal{B}$ but are not correlated with each other. The cardinality in a such a group may suggest the number of independent factors that cause the separation of $\mathcal{A}$ and $\mathcal{B}$. A well-known technique that implements this feature selection strategy is the Correlation-based Feature Selection (CFS) method[39].

Rather than scoring and ranking individual features, **the CFS method** scores

and ranks the worth of subsets of features. As the feature subset space is usually huge, CFS uses a best-first-search heuristic. This heuristic algorithm embodies our third concept that takes into account the usefulness of individual features for predicting the class along with the level of intercorrelation among them. CFS first calculates a matrix of feature-class and feature-feature correlations from the training data. Then a score of a subset of features is assigned using the following heuristics:

$$Merit_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}},$$

where $Merit_S$ is the heuristic merit of a feature subset S containing k features, $\overline{r_{cf}}$ is the average feature-class correlation, and $\overline{r_{ff}}$ is the average feature-feature intercorrelation. **Symmetrical uncertainties** are used in CFS to estimate the degree of association between discrete features or between features and classes[39]. The formula below measures the intercorrelation between two features or the correlation between a feature X and a class Y which is in the range $[0, 1]$.

$$r_{xy} = 2.0 * \left(\frac{gain}{H(X) + H(Y)}\right)$$

where $gain = H(X) + H(Y) - H(X, Y)$ is the information gain between features and classes, $H(X)$ is the entropy of the feature. CFS starts from the empty set of features and uses the best-first-search heuristic with a stopping criterion of 5 consecutive fully expanded non-improving subsets. The subset with the highest merit found during the search will be selected.

Wong *et al.*[108] apply CFS to the feature vectors derived from the Pedersen and Nielsen dataset as described in Section 4 in a 3-fold cross validation setting. It turns out that in each fold, exactly the same 9 features are selected by CFS:

(1) "position–3",
(2) "in-frame up-stream ATG",
(3) "in-frame down-stream TAA",
(4) "in-frame down-stream TAG",
(5) "in-frame down-stream TGA",
(6) "in-frame down-stream CTG",
(7) "in-frame down-stream GAC",
(8) "in-frame down-stream GAG", and
(9) "in-frame down-stream GCC".

These 9 features are thus very robust differentiators of TIS from non-TIS. Furthermore, there are good biological reasons for most of them.

"Position–3" can be explained by the known correspondence to the well-known Kozak consensus sequence, GCC[AG]CCAUGG, for vertebrate translation initiation sites[53,50]. Hence having a "A" or "G" in this position indicates that the target ATG is more likely to be a TIS. This is the same feature deduced as important by Pedersen and Nielsen[78] in their "hole-shifting" experiment discussed in Section 2.

"In-frame up-stream ATG" can be explained by the ribosome scanning model[1,54]. The ribosome scans the mRNA in a 5'-to-3' (that is, left-to-right) manner until it encounters the first ATG with the right context for translation initiation. Thus a ATG that is closer to the 5' end have a higher probability to be a TIS. Consequently, the presence of an in-frame ATG up-stream of the target ATG indicates that the target ATG is less likely to be a TIS. This is also consistent with the observation by Rogozin *et al.*[87] that a negative correlation exists between the strength of the start context and the number of up-stream ATGs. This is also a feature deduced by Pedersen and Nielsen[78] in a detailed analysis on the erroneous predictions made by their neural network.

"In-frame down-stream TAA", "in-frame down-stream TAG", and "in-frame down-stream TGA" can be explained as they correspond to in-frame stop codons down-stream from the target ATG. These 3 nucleotide triplets—TAA, TAG, TGA—do not code for amino acids. They are called the stop codons. The biological process of translating in-frame codons into amino acids stops upon encountering an in-frame stop codons. Thus the presence of any of these three features means there is an in-frame stop codon within 100 nucleotides down-stream of the target ATG. Consequently, the protein product corresponding to the sequence is no more than 33 amino acids. This is smaller than most proteins. Hence the target ATG is not likely to be a TIS. This group of stop codon features are not reported by Pedersen and Nielsen[78], Zien *et al.*[110,111], nor Hatzigeorgiou[42], presumably the complex and/or low-level nature of their systems prevented them from noticing this important group.

We do not have a clear biological explanation for the remaining 4 selected features, other than that of a possibility of codon bias in the context of TIS. We look forward to be enlightened on the biological role of these 4 remaining features in the protein translation process by the reader.

In the next section, we use the 9 relevant features identified above to construct systems for recognizing translation initiation sites in vertebrate sequences.

6. Feature Integration

In order to show that the 9 features identified in the previous section are indeed relevant and reliable differentiators of TIS from non-TIS, we need to show that reliable systems for recognizing TIS can be built from them. In fact, Wong *et al.*[108] demonstrate in a 3-fold cross validation setting that almost any machine learning methods can be trained on these 9 features to produce TIS recognizers of extremely competitive accuracy. In this section, we introduce three different machine learning methods—Naive Bayes (NB)[59], Support Vector Machine (SVM)[103], and C4.5[82]—and present their accuracy trained on these 9 features.

The results of cross validation are evaluated using standard performance measures given in Figure 2 and defined as follows. **Sensitivity** measures the proportion of TIS that are correctly recognized as TIS. **Specificity** measures the proportion of

The sensitivity, specificity, precision, and accuracy of a classifier are defined as follows:

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

$$\text{Specificity} = \frac{TN}{TN+FP}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Accuracy} = \frac{TP+TN}{TP+FN+TN+FP}$$

where TP, TN, FP, and FN are as given in the contingency table below of results from testing a prediction system:

	Classified as Yes	Classified as No
Actual Yes	No. of True Positives (TP)	No. of False Negatives (FN)
Actual No	No. of False Positives (FP)	No. of True Negative (TN)

Fig. 2. Definitions of sensitivity, specificity, precision, and accuracy of a prediction system.

non-TIS that are correctly recognized as non-TIS. **Precision** measures the proportion of the claimed TIS that are indeed TIS. **Accuracy** measures the proportion of predictions, both for TIS and non-TIS, that are correct.

The three machine learning methods that we use in this section, as well as many of the feature selection measures used in Section 5, are all available without cost in a Java-based software package called WEKA. The default settings of WEKA are used throughout our discussion here. WEKA can be obtained at *http://www.cs.waikato.ac.nz/∼ml/weka.*

We start with **NB**, a probabilistic learner based on the Bayes theorem[59]. The theorem states that $P(h|d) = P(d|h) * P(h)/P(d)$, where $P(h)$ is the prior probability that a hypothesis h holds, $P(d|h)$ is the probability of observing data d given some world that h holds, and $P(h|d)$ is the posterior probability that h holds given the observed data d.

Let H be all the possible classes. Then given a test instance with feature vector $(f_1, \ldots, f_n)$, the most probable classification is $\text{argmax}_{h_j \in H} P(h_j|f_1, \ldots, f_n)$. Using the Bayes theorem, this is rewritten to $\text{argmax}_{h_j \in H} P(f_1, \ldots, f_n|h_j) * P(h_j)/P(f_1, \ldots, f_n)$. Since the denominator is independent of h_j, this can be simplified to $\text{argmax}_{h_j \in H} P(f_1, \ldots, f_n|h_j) * P(h_j)$. However, estimating $P(f_1, \ldots, f_n|h_j)$ accurately may not be feasible unless the training set is sufficiently large.

So, NB assumes that the effect of a feature value on a given class is independent of the values of other features. This assumption is called class conditional independence. It is made to simplify computation and it is in this sense that NB is considered to be "naive". Under this class conditional independence assumption, $\text{argmax}_{h_j \in H} P(f_1, \ldots, f_n | h_j) * P(h_j) = \text{argmax}_{h_j \in H} \prod_i P(f_i | h_j) * P(h_j)$. $P(h_j)$ and $P(f_i | h_j)$ can often be estimated reliably from typical training sets.

The class conditional independence assumption is a strong assumption and is sometimes invalid. However, the bias in estimating probabilities often may not make a big difference in practice. The reason is that it is often the order of the probabilities, not their exact values, that determine the classifications. According to Wong *et al.*[108], NB trained on the 9 features selected in Section 5 yields an effective TIS recognizer with sensitivity = 84.3%, specificity = 86.1%, precision = 66.3%, and accuracy = 85.7%.

Next we discuss **SVM**, which is based on a very well-developed statistical theory and has been successfully applied to a wide range of pattern recognition problems[18,103]. A SVM selects a small number of critical boundary samples from each class and builds a linear discriminant function that separates them as widely as possible. In the case that no linear separation is possible, the technique of "kernel" is used to automatically inject the training samples into a higher-dimensional space, and to learn a separator in that space.

The training of a SVM is a quadratic programming problem on maximizing the Lagrangian dual objective function

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{l} [\alpha]_i - \frac{1}{2} * \sum_{i=1}^{l} \sum_{j=1}^{l} [\alpha]_i * [\alpha]_j * [Y]_i * [Y]_j * k([X]_i, [X]_j)$$

subject to the constraint that $\forall i. 0 \leq [\alpha]_i \leq C$ and $\sum_{i=1}^{l} [\alpha]_i * [Y]_i = 0$. Here, C is a bound on errors, $[X]_i$ and $[X]_j$ are the ith and jth training samples, $[Y]_i$ and $[Y]_j$ are the corresponding class labels (which are mapped to 1 and –1), and $k(\cdot, \cdot)$ is the kernel function. The kernel function used here is a polynomial kernel $k(X, X') = (X \cdot X')^d = \sum_{i_1} \ldots \sum_{i_d} [X]_{i_1} * \ldots * [X]_{i_d} * [X']_{i_1} * \ldots * [X']_{i_d}$. In the WEKA package, this quadratic programming problem above is solved by sequential minimal optimization[79].

Once the optimal solution α is obtained from solving the quadratic programming problem above, the SVM decision function $G(T)$ on a test sample T can be constructed as $G(T) = sign(\sum_i [\alpha]_i * [Y]_i * k(T, [X]_i) + b)$, for a threshold b. To estimate b, we use the fact that the optimal solution α must satisfy the so-called Karush-Kuhn-Tucker conditions[79,92,41]. In particular, for each $[\alpha]_j > 0$, $[Y]_j * G([X]_j) = 1$. Hence $[Y]_j = sign(\sum_i [\alpha]_i * [Y]_i * k([X]_j, [X]_i) + b)$. So we estimate b by averaging $[Y]_i - \sum_i [\alpha]_i * [Y]_i * k([X]_j, [X]_i)$ for all $[\alpha]_j > 0$.

According to Wong *et al.*[108], SVM trained on our 9 features yields an accurate TIS recognizer with sensitivity = 73.9%, specificity = 93.2%, precision = 77.9%, and accuracy = 88.5%.

Our third machine learning method is **C4.5**, a widely used decision tree based classifier[82]. C4.5 has an important advantage over machine learning methods such as ANN, SVM, and NB in a qualitative dimension: rules produced by C4.5 are easier to understand than kernel functions. C4.5 constructs a decision tree in a recursive process. The process starts by determining the feature that is most discriminatory with regard to the entire training data. Then it uses this feature to split the data into non-overlapping groups. Each group contains multi-class or single class samples, as categorized by this feature. Next, a significant feature of each of the groups is used to recursively partition them until a stopping criteria is satisfied. Let us next describe how C4.5 selects the most discriminatory feature in at each node of the decision tree construction process.

Recall that from Section 5 that $e(f,\mathcal{U}) = \min\{E(f,\mathcal{U},S_1,S_2) \mid (S_1,S_2)$ is a partitioning of the values of f in $\bigcup\mathcal{U}$ by some point $T\}$. We can interpret this number as the amount of information needed to identify the class of an element of $\mathcal{U}$. Let $\mathcal{U} = \{\mathcal{A},\mathcal{B}\}$, where $\mathcal{A}$ and $\mathcal{B}$ are the two classes of samples. Let f be a feature and S be the range of values that f can take in the samples. Let S be partitioned into two subranges S_1 and S_2. Then the difference between the information needed to identify the class of a sample in $\mathcal{U}$ before and after the value of the feature f is revealed is $Gain(f,\mathcal{U},S_1,S_2) = Ent(f,\mathcal{U},S_1 \cup S_2) - E(f,\mathcal{U},S_1,S_2)$. Then the **information gain** is the amount of information that is gained by looking at the value of the feature f, and is defined as $g(f,\mathcal{U}) = \max\{Gain(f,\mathcal{U},S_1,S_2) \mid (S_1,S_2)$ is a partitioning of the values of f in $\bigcup\mathcal{U}$ by some point $T\}$. This information gain measure $g(f,\mathcal{U})$ can also be used for selecting features that are relevant. In fact, the ID3 decision tree induction algorithm uses it as the measure for picking discriminatory features for tree nodes[81].

However, $g(f,\mathcal{U})$ tends to favour features that have a large number of values. The **information gain ratio** is a refinement to compensate for this disadvantage. Let $GainRatio(f,\mathcal{U},S_1,S_2) = Gain(f,\mathcal{U},S_1,S_2)/SplitInfo(f,\mathcal{U},S_1,S_2)$, where $SplitInfo(f,\mathcal{U},S_1,S_2) = Ent(f,\{\mathcal{U}_f^{S_1},\mathcal{U}_f^{S_2}\},S_1 \cup S_2)$, and $\mathcal{U}_f^S = \bigcup_{\mathcal{C}\in\mathcal{U}}\{d \in \mathcal{C} \mid$ the feature f in sample d has value in range $S\}$. Then the information gain ratio is defined as $gr(f,\mathcal{U}) = \max\{GainRatio(f,\mathcal{U},S_1,S_2) \mid (S_1,S_2)$ is a partitioning of the values of f in $\bigcup\mathcal{U}$ by some point $T\}$.

C4.5 uses the information gain ratio to determine which feature is most discriminatory at each step of its decision tree induction process. According to Wong *et al.*[108], C4.5 trained on our 9 features also yields an accurate TIS recognizer with sensitivity = 74.0%, specificity = 94.4%, precision = 81.1%, and accuracy = 89.4%.

These results are summarised in Part II of Figure 3. Since NB, SVM, and C4.5 are very different machine learning methods, and they are all performing well when trained on the 9 features selected in Section 5, we can be fairly certain that these 9 features correspond to key signals for separating TIS from non-TIS.

I. Results reported by some existing systems for TIS recognition. A "-" means that particular figure is not available. The results by Pedersen and Nielsen[78], Zien *et al.*[110,111], and Li *et al.*[63] are directly comparable to results in Parts II and III below as they are all on the Pedersen and Nielsen dataset. The result by Hatzigeorgiou[42] is not directly comparable to any of the other systems because she uses a different dataset and also because she uses the ribosome scanning model.

Classifier	Sensitivity	Specificity	Accuracy
Pedersen and Nielsen[78]	78.0%	87.0%	85.0%
Zien *et al.*[110]	69.9%	94.1%	88.1%
Zien *et al.*[111]	-	-	88.6%
Hatzigeorgiou[42]	-	-	94.0%
Li *et al.*[63]	84.7%	88.7%	87.7%

II. 3-fold cross-validation accuracy of NB, SVM, and C4.5 on the Pedersen and Nielsen dataset based on the 9 features selected using CFS in Section 5. These are results reported by Wong *et al.*[108].

Classifier	Sensitivity	Specificity	Precision	Accuracy
NB	84.3%	86.1%	66.3%	85.7%
SVM	73.9%	93.2%	77.9%	88.5%
C4.5	74.0%	94.4%	81.1%	89.4%

III. 3-fold cross-validation accuracy of NB, SVM, and C4.5 on the Pedersen and Nielsen dataset based on the 100 features selected using the entropy measure in Section 7.

Classifier	Sensitivity	Specificity	Precision	Accuracy
NB	70.53%	87.76%	65.47%	83.49%
SVM	80.19%	96.48%	88.24%	92.45%
C4.5	74.88%	93.65%	79.51%	89.00%

Fig. 3. Accuracy results of various feature selection and machine learning methods for TIS recognition.

7. Refinements

As every 3 nucleotides code for an amino acid, it is legitimate to investigate if an alternative approach to generating features based on amino acids can produce effective TIS recognizers. Also, in the previous sections, we use features selected by CFS, hence it is legitimate to investigate if features selected by other methods can produce effective TIS recognizers. Li *et al.*[63] pursue these two alternatives. In this section, we discuss their results and add some of our own.

For generating features, we take the sequence segments of 100 nucleotides up-stream and 100 nucleotides down-stream of the target ATG as before. Then we consider 3-grams that are in-frame—that is, those 3-grams that are aligned to the ATG at positions ..., -6, -3, +4, +7, ... Those in-frame 3-grams that code for amino acids are converted into the corresponding amino acid letters. Those in-frame 3-grams that are stop codons are converted into a special letter symbolizing a stop codon. From the conversion above, Li *et al.*[63] generate the following types of k-grams:

(1) up-X, which counts the number of times the amino acid letter X appears in the up-stream part, for X ranging over the standard 20 amino acid letters and the special stop symbol.
(2) down-X, which counts the number of times the amino acid letter X appears in the down-stream part, for X ranging over the standard 20 amino acid letters and the special stop symbols.
(3) up-XY, which counts the number of times the two amino acid letters XY appear as a substring in the up-stream part, for X and Y ranging over the standard 20 amino acid letters and the special stop symbol.
(4) down-XY, which counts the number of times the two amino acid letters XY appear as a substring in the up-stream part, for X and Y ranging over the standard 20 amino acid letters and the special stop symbol.

Li *et al.*[63] also generate the follow Boolean features from the original sequence fragments: UP-ATG, which indicates that an in-frame ATG occurs in the up-stream part; up3-AorG, which indicates that an "A" or a "G" appears in position –3 ; down4-G, which indicates that a "G" appears in position +4. These last two features are inspired by the Kozak consensus sequence, GCC[AG]CCAUGG, for vertebrate translation initiation sites[53,50]. A total of $2 * 21 + 2 * 21^2 + 3 = 927$ features are thus generated as described above.

For selecting features, we use the entropy measure to rank the relevance of each of these 927 candidate features in a 3-fold cross validation setting. In each fold, the top 100 features are selected. The following features are consistently among the top 10 features in each of the 3 folds: up-ATG, down-STOP, down-L, down-D, down-E, down-A, up3-AorG, up-A, down-V. Up-M also appears among the top features in each fold, but we exclude it as it is redundant given that up-ATG is true if and only if up-M > 0. The detailed ranking of these features in each fold is given in

Fold	up ATG	down STOP	up3 AorG	down A	down V	up A	down L	down D	down E
1	1	2	4	3	6	5	8	9	7
2	1	2	3	4	5	6	7	8	9
3	1	2	3	4	5	6	8	9	7

Fig. 4. Ranking of the top 9 features selected by the entropy measure method as relevant in each of the 3 folds.

Figure 4.

Interestingly, most of these features, except up-A and down-V, correspond to those selected by CFS on the original nucleotide sequence fragments in Section 5. Specifically, up-ATG corresponds to "in-frame up-stream ATG"; down-stop corresponds to "in-frame down-stream TAA", "in-frame down-stream TAG", and "in-frame down-stream TGA"; up3-AorG corresponds to "position–3"; down-L corresponds to "in-frame down-stream CTG"; down-D corresponds to "in-frame down-stream GAC"; down-E corresponds to "in-frame down-stream GAG"; and down-A corresponds to "in-frame down-stream GCC".

For validating whether accurate systems for recognizing TIS can be developed using features based on amino acids, we test the C4.5, SVM, and NB machine learning methods in 3-fold cross validations. The top 100 features selected by the entropy measure are used in each fold.

For C4.5, we obtain sensivity = 74.88%, specificity = 93.65%, precision = 79.51%, and accuracy = 89.00%. This is comparable to the performance of C4.5 using the 9 features selected by CFS in Section 5.

For SVM, we obtain sensitivity = 80.19%, specificity = 96.48%, precision = 88.24%, and accuracy = 92.45%. This is significantly better than the performance of SVM using the 9 features selected by CFS in Section 5. It is the best reported results on the Pedersen and Nielsen dataset[78] that we know of.

For NB, we obtain sensitivity = 70.53%, specificity = 87.76%, precision = 65.47%, and accuracy = 83.49%. This is considerably worse than the other results. The increase from 9 features in Section 6 to 100 features here has apparently confused NB.

Li *et al.*[63] use just the top 10 features selected by the entropy measure in their study. As mentioned in Section 2, they use PCL (Prediction by Collective Likelihood of emerging patterns)[64] to integrate these top 10 entropy features. PCL focuses on (a) fast techniques for identifying patterns—in the case of Li *et al.*, made up of combinations of the top 10 features—whose frequencies in two classes differ by a large ratio[23], which are the so-called **emerging patterns**; and on (b) combining these patterns to make decision.

As PCL is a relatively new data mining method, we describe it in more detail here. **PCL** has two phases. Given two training datasets $D^{\mathcal{A}}$ (instances of class $\mathcal{A}$) and $D^{\mathcal{B}}$ (instances of class $\mathcal{B}$) and a test sample T, PCL first discovers two groups of most general emerging patterns from $D^{\mathcal{A}}$ and $D^{\mathcal{B}}$. Denote the most general emerging patterns of $D^{\mathcal{A}}$ as, $EP_1^{\mathcal{A}}$, $EP_2^{\mathcal{A}}$, $\cdots$, $EP_i^{\mathcal{A}}$, in descending order of frequency. Denote the most general emerging patterns of $D^{\mathcal{B}}$ as $EP_1^{\mathcal{B}}$, $EP_2^{\mathcal{B}}$, $\cdots$, $EP_j^{\mathcal{B}}$, in descending order of frequency. Suppose the test sample T contains these most general emerging patterns of $D^{\mathcal{A}}$: $EP_{i_1}^{\mathcal{A}}$, $EP_{i_2}^{\mathcal{A}}$, $\cdots$, $EP_{i_x}^{\mathcal{A}}$, $i_1 < i_2 < \cdots < i_x \leq i$, and these most general emerging patterns of $D^{\mathcal{B}}$: $EP_{j_1}^{\mathcal{B}}$, $EP_{j_2}^{\mathcal{B}}$, $\cdots$, $EP_{j_y}^{\mathcal{B}}$, $j_1 < j_2 < \cdots < j_y \leq j$. The next step is to calculate two scores for predicting the class label of T. Suppose we use k ($k \ll i$ and $k \ll j$) top-ranked most general emerging patterns of $D^{\mathcal{A}}$ and $D^{\mathcal{B}}$. Then we define the score of T in the $D^{\mathcal{A}}$ class as

$$score(T, D^{\mathcal{A}}) = \sum_{m=1}^{k} \frac{frequency(EP_{i_m}^{\mathcal{A}})}{frequency(EP_m^{\mathcal{A}})},$$

and the score in the $D^{\mathcal{B}}$ class is similarly defined in terms of $EP_{j_m}^{\mathcal{B}}$ and $EP_m^{\mathcal{B}}$. If $score(T, D^{\mathcal{A}}) > score(T, D^{\mathcal{B}})$, then T is predicted as the class of $D^{\mathcal{A}}$. Otherwise it is predicted as the class of $D^{\mathcal{B}}$. We use the size of $D^{\mathcal{A}}$ and $D^{\mathcal{B}}$ to break tie.

The emerging patterns in Li *et al.*[63] are built by PCL from the top 10 features selected by the entropy method shown in Figure 4. They report that PCL achieves 84.7% sensitivity, 88.7% specificity and 87.7% overall accuracy.

These results are summarized in Part III of Figure 3. We believe further improvements are possible in a more careful selection of features than simply using the top 100 features of the lowest entropy. But that pursuit is beyond the scope of this tutorial.

8. Remarks

We have described a methodology for analyzing sequence data. The methodology comprises the three steps of (a) generate candidate features from the sequences, (b) select relevant features from the candidates, and (c) integrate the selected features using a machine learning method to build a system to recognize specific properties in sequence data. While this simple methodology is applicable to a variety of recognition problems for functional sites in biological sequences, we have illustrated it in this tutorial on the problem of recognizing translation initiation sites, and used it to identify features that are useful for understanding the distinction between ATG sites that are translation initiation sites and those that are not. In connection with this, we have surveyed some techniques for recognizing translation initiation sites[78,110,111,42,108,63].

For generating candidate features, we have presented various types of features based on the idea of k-grams, including basic k-grams, in-frame k-grams, and position-specific features. These types of features are generally good candidates to try in problems pertaining to recognition of functional sites in biological sequences.

The basic k-grams are also useful in other problems, particularly those pertaining to the analysis of biomedical terms in scientific texts[105].

For selecting relevant features, we have discussed signal-to-noise[37], t-statistics[19], entropy measure[29], information gain measure[81], and information gain ratio[82], and CFS[39]. For integrating selected features, we have described C4.5[82], SVM[103,79], Naive Bayes[59], and PCL[64]. In connection with the description of SVM, we have also given a brief exposition on the engineering of a locality-improved kernel function[110]. These techniques are all very general feature selection and data mining methods. They can be applied, as we have done here, on recognizing functional sites in biological sequences, as well as in non-sequence problems. For example, the analysis of gene expression profiles and proteomic profiles typically comprises the two steps of feature selection and feature integration for decision making, using these same techniques[37,61,107,62,25,33].

Let us end this tutorial on data mining tools for analyzing sequence data with a brief mention of a number of data mining tools and their applications in the biomedical arena in the context of classification and prediction. Any of these tools surveyed below can be used in step (c) of the three-step methodology discussed in the main text of this tutorial.

The most popular classification technique is the idea of decision tree induction. We have already seen the C4.5 method in Section 6. Other algorithms for decision tree induction include CART[14], ID3[81], SLIQ[74], FACT[67], QUEST[66], PUBLIC[84], CHAID[51], ID5[102], SPRINT[95], and BOAT[35]. This group of algorithms are most successful for analysis of clinical data and for diagnosis from clinical data. Some examples are diagnosis of central nervous system involvement in hematooncologic patients[69], prediction of post-traumatic acute lung injury[83], identification of acute cardiac ischemia[94], prediction of neurobehavioral outcome in head-injury survivors[99], and diagnosis of myoinvasion[68].

Another important group of techniques[7,24,75,49,43,48,90,60] are based on the Bayes theorem. The Naive Bayes method presented in Section 6 is perhaps the most popular example in this group. Some example applications of Bayesian classifiers in the biomedical context are mapping of locus controlling a genetic trait[36], screening for macromolecular crystallization[44], classification of cNMP-binding proteins[73], prediction of carboplatin exposure[46], prediction of prostate cancer recurrence[22], prognosis of femoral neck fracture recovery[58], and prediction of protein secondary structure[52,98,2].

Related to the Bayesian classifiers are the hidden Markov models or HMMs[7,56,26,27]. A HMM is a stochastic generative model for sequences defined by a finite set S of states, a finite alphabet A of symbols, a transition probability matrix T, and an emission probability matrix E. The system moves from state to state according to T while emitting symbols according to E. In an n-th order HMM, the matrices T and E depend on all n previous states. HMMs have been applied to a variety of problems in sequence analysis, including protein family classification and prediction[9,6,57], tRNA detection in genomic sequences[70], methy-

lation guide snoRNA screening[71], gene finding and gene structure prediction in DNA sequences[13,12,5,56,91], protein secondary structure modeling[31], and promoter recognition [106,77].

Artificial neural networks[89,7,20] are another important approach to classification that have high tolerance to noisy data. We have seen two examples in Section 2. Successful applications of artificial neural networks in the biomedical context include protein secondary structure prediction[86,88,80], signal peptide prediction[21,76,28], gene finding and gene structure prediction[101,96], T-cell epitope prediction[45], RNA secondary structure prediction[97], toxicity prediction[17], disease diagnosis and outcome prediction[104,93,100], as well as protein translation initiation site recognition[78,42].

Last but not least, support vector machines are another approach to the classification problem that has clear connections to statistical learning theory. We have seen in Section 6 a detailed exposition on their construction. A SVM is largely characterized by the choice of its kernel function. Thus SVMs connect the problem they are designed for to a large body of existing research on kernel-based methods[85,103,18]. Some recent applications of SVM in the biomedical context include protein homology detection[47], microarray gene expression data classification[16], breast cancer diagnosis[72,32], as well as protein translation initiation site recognition[110,111].

References

1. P. Agarwal and V. Bafna. The ribosome scanning model for translation initiation: Implications for gene prediction and full-length cDNA detection. *Intelligent Systems for Molecular Biology*, 6:2–7, 1998.
2. G.E. Arnold, A.K. Dunker, S. L. Johns, and R.J. Douthart. Use of conditional probabilities for determining relationships between amino acid sequence and protein secondary structure. *Proteins*, 12:382–399, April 1992.
3. V. B. Bajic, S. H. Seah, A. Chong, G. Zhang, J. Koh, and V. Brusic. Dragon Promoter Finder: Recognition of vertebrate RNA polymerase II promoters. *Bioinformatics*, 18:198–199, 2002.
4. V. B. Bajic, S. Tang, H. Han, V. Brusic, and A. G. Hatzigeorgiou. Artificial neural networks based systems for recognition of genomic signals and regions: A review. *Informatica*, 26:389–400, 2002.
5. P. Baldi, S. Brunak, Y. Chauvin, and A. Krogh. Hidden Markov models for human genes: Periodic patterns in exon sequences. In *Theoretical and Computational Methods in Genome Research*, pages 15–32, 1997.
6. P. Baldi and Y. Chauvin. Hidden Markov models of G-protein-coupled receptor family. *Journal of Computational Biology*, 1:311–335, 1994.
7. Pierre Baldi and Soren Brunak. *Bioinformatics: The Machine Learning Approach*. MIT Press, Cambridge, MA, 1999.
8. P. Baldi and A. D. Long. A Bayesian framework for the analysis of microarray expression data: Regularized t-test and statistical inferences of gene changes. *Bioinformatics*, 17:509–519, 2001.
9. A. Bateman, E. Birney, R. Durbin, S. R. Eddy, R. D. Finn, and E. L. L. Sonnhammer. Pfam 3.1: 1313 multiple alignments and profile HMMs match the majority of proteins. *Nucleic Acids Research*, 27:260–262, 1999.

10. M. S. Boguski, T. M. J. Lowe, and C. M. Tolstoshev. dbEST—database for "expressed sequence tags". *Nature Genetics*, 4:332–333, 1993.
11. M.S. Boguski and C.M. Tolstoshev. Gene discovery in dbEST. *Science*, 265:1993–1994, 1994.
12. M. Borodovsky and J. D. McIninch. GENEMARK: Parallel gene recognition for both DNA strands. *Computers and Chemistry*, 17(2):123–133, 1993.
13. M. Borodovsky, J. D. McIninch, E.V. Koonin, K.E. Rudd, C. Medigue, and A. Danchin. Detection of new genes in a bacterial genome using Markov models for three gene classes. *Nucleic Acids Research*, 23:3554–3562, 1995.
14. L. Breiman, L. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, 1984.
15. R.J. Brooker. *Genetics: Analysis and Principles*. Addison-Wesley, Reading, MA, 1999.
16. M.P. Brown, W.N. Grundy, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares Jr, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proceedings of the National Academy of Sciences of the United States of America*, 97(1):262–267, 2000.
17. F. R. Burden and D. A. Winkler. A quantitative structure-activity relationships model for the acute toxicity of substituted benzenes to *tetrahymena pyriformis* using Bayesian-regularised neural networks. *Chemical Research in Toxicology*, 13:436–440, 2000.
18. C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.
19. M. Caria. *Measurement Analysis: An Introduction to the Statistical Analysis of Laboratory Data in Physics, Chemistry, and the Life Sciences*. Imperial College Press, London, 2000.
20. Y. Chauvin and D. Rumelhart. *Backpropagation: Theory, Architectures, and Applications*. Lawrence Erlbaum, Hillsdale, NJ, 1995.
21. M.G. Claros, S. Brunak, and G. von Heijne. Prediction of n-terminal protein sorting signals. *Current Opinion in Structural Biology*, 7:394–398, 1997.
22. J. Demsar, B. Zupan, M.W. Kattan, J.R. Beck, and I. Bratko. Naive Bayesian-based nomogram for prediction of prostate cancer recurrence. *Studies Health Technology and Informatics*, 68:436–441, 1999.
23. G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *Proceedings of 5th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 15–18, 1999.
24. R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
25. S. Dudoit, J. Fridlyand, and T.P. Speed. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of American Statistical Association*, 97:77–87, 2002.
26. R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, editors. *Biological Sequence Analysis: Probabilistics Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.
27. S.R. Eddy. Hidden Markov models. *Current Opinion in Structural Biology*, 6:361–365, 1996.
28. O. Emanuelsson, H. Nielsen, and G. von Heijne. ChloroP, a neural network-based method for predicting chloroplast transit peptides and their cleavage sites. *Protein Science*, 8:978–984, May 1999.
29. U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of 13th International Joint Conference on*

Artificial Intelligence, pages 1022–1029, 1993.
30. J.W. Fickett and C.S. Tung. Assessment of protein coding measures. *Nucleic Acids Research*, 20:6441–6450, 1992.
31. V. Di Francesco, J. Granier, and P.J. Munson. Protein topology recognition from secondary structure sequences—applications of the hidden Markov models to the alpha class proteins. *Journal of Molecular Biology*, 267:446–463, 1997.
32. T.-T. Friess, N. Cristianini, and C. Campbell. The kernel adatron algorithm: A fast and simple learning procedure for support vector machines. In *Proceedings of 15th International Conference on Machine Learning*, pages 188–196, 1998.
33. T.S. Furey, N. Cristianini, N. Duffy, D.W. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.
34. M. Gardiner-Garden and M. Frommer. CpG islands in vertebrate genomes. *Journal of Molecular Biology*, 196:261–282, 1987.
35. J. Gehrke, V. Ganti, R. Ramakrishnan, and W. Y. Loh. BOAT—optimistic decision tree construction. In *Proceedings of ACM-SIGMOD International Conference on Management of Data*, pages 169–180, 1999.
36. S. Ghosh and P.P. Majumder. Mapping a quantitative trait locus via the EM algorithm and Bayesian classification. *Genetic Epidemiology*, 19(2):97–126, September 2000.
37. T.R. Golub and D.K. Slonim and P. Tamayo and C. Huard and M. Gaasenbeek and J.P. Misirov and H. Coller and M.L. Loh and J.R. Downing and M.A. Caligiuri and C.D. Bloomfield and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
38. N.K. Gray and M. Wickens. Control of translation initiation in animals. *Annual Review of Cells & Developmental Biology*, 14:399–458, 1998.
39. M.A. Hall. *Correlation-based feature selection machine learning*. PhD thesis, Department of Computer Science, University of Waikato, New Zealand, 1998.
40. J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, CA, 2000.
41. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, Berlin, 2001.
42. A. G. Hatzigeorgiou. Translation initiation start prediction in human cDNAs with high accuracy. *Bioinformatics*, 18:343–350, 2002.
43. D. Heckerman. Bayesian networks for knowledge discovery. In *Advances in Knowledge Discovery and Data Mining*, pages 273–305, Cambridge, MA, 1996. MIT Press.
44. D. Hennessy, B. Buchanan, D. Subramanian, P.A. Wilkosz, and J.M. Rosenberg. Statistical methods for the objective design of screening procedures for macromolecular crystallization. *Acta Crystallogr. D Biol. Crystallogr.*, 56:817–827, 2000.
45. M. C. Honeyman, V. Brusic, N. Stone, and L. C. Harrison. Neural network-based prediction of candidate T-cell epitopes. *Nature Biotechnology*, 16:966–969, 1998.
46. A.D. Huitema, R.A. Mathot, M.M. Tibben, J.H. Schellens, S. Rodenhuis, and J.H. Beijnen. Validation of techniques for the prediction of carboplatin exposure: Application of Bayesian methods. *Clinical Pharmacology & Therapeutics*, 67:621–630, 2000.
47. T. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 7:95–114, 2000.
48. F. V. Jensen. *An Introduction to Bayesian Networks*. Springer-Verlag, New York, 1996.
49. G. H. John. *Enhancements to the Data Mining Process*. PhD thesis, Stanford University, 1997.

50. C. P. Joshi, H. Zhou, X. Huang, and V. L. Chiang. Context sequences of translation initiation codon in plants. *Plant Molecular Biology*, 35:993–1001, 1997.
51. G. V. Kaas. An exploratory technique for investigating large quantities of categorical data. *Applied Statistics*, 29:119–127, 1980.
52. S. Kasif and A. L. Delcher. Modeling biological data and structure with probabilistic networks. In *Computational Methods in Molecular Biology*, pages 335–352, 1998.
53. M. Kozak. An analysis of vertebrate mRNA sequences: Intimations of translational control. *The Journal of Cell Biology*, 115:887–903, 1991.
54. M. Kozak. Initiation of translation in prokaryotes and eukaryotes. *Gene*, 234:187–208, 1999.
55. M. Kozak. An analysis of 5'-noncoding sequences from 699 vertebrate messenger RNAs. *Nucleic Acids Research*, 15:8125–8148, 1987.
56. A. Krogh. An introduction to hidden Markov models for biological sequences. In *Computational Methods in Molecular Biology*, pages 45–62, 1998.
57. A. Krogh, M. Brown, I. S. Mian, K. Sjolander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, 1994.
58. M. Kukar, I. Kononenko, and T. Silvester. Machine learning in prognosis of the femoral neck fracture recovery. *Artificial Intelligence in Medicine*, 8:431–451, 1996.
59. P. Langley, W. Iba, and K. Thompson. An analysis of Bayesian classifier. In *Proceedings of 10th National Conference on Artificial Intelligence*, pages 223–228, 1992.
60. S. L. Lauritzen. The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19:191–201, 1995.
61. J. Li, H. Liu, J. R. Downing, A. E.-J. Yeoh, and L. Wong. Simple rules underlying gene expression profiles of more than six subtypes of acute lymphoblastic leukemia (ALL) patients. *Bioinformatics*, 19:71–78, 2003.
62. J. Li, H. Liu, and L. Wong. A comparative study on feature selection and classification methods using a large set of gene expression profiles. In *Proceedings of 13th International Conference on Genome Informatics*, pages 51–60, Tokyo, Japan, December 2002.
63. J. Li, S.-K. Ng, and L. Wong. Bioinformatics adventures in database research. In *LNCS 2572: Proceedings of 9th International Conference on Database Theory*, pages 31–46, 2003.
64. J. Li and L. Wong. Geography of differences between two classes of data. In *Proceedings 6th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 325–337, 2002.
65. H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of IEEE 7th International Conference on Tools with Artificial Intelligence*, pages 338–391, 1995.
66. W. Y. Loh and Y. S. Shih. Split selection methods for classification trees. *Statistica Sinica*, 7:815–840, 1997.
67. W. Y. Loh and N. Vanichsetakul. Tree-structured classification via generalized discriminant analysis. *Journal of American Statistical Association*, 83:715–728, 1988.
68. T.A. Longacre, M.H. Chung, D.N. Jensen, and M.R. Hendrickson. Proposed criteria for the diagnosis of well-differentiated endometrial carcinoma. A diagnostic test for myoinvasion. *American Journal of Surgical Pathology*, 19(4):371–406, 1995.
69. I.S. Lossos, R. Breuer, O. Intrator, and A. Lossos. Cerebrospinal fluid lactate dehydrogenase isoenzyme analysis for the diagnosis of central nervous system involvement in hematooncologic patients. *Cancer*, 88(7):1599–1604, 2000.
70. T. M. Lowe and S. R. Eddy. tRNAscan-SE: A program for improved detection of

transfer RNA genes in genomic sequence. *Nucleic Acids Research*, 25:955–964, 1997.
71. T. M. Lowe and S. R. Eddy. A computational screen for methylation guide snoRNAs in yeast. *Science*, 283:1168–1171, 1999.
72. O. L. Mangasarian, W. Nick Street, and W. H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4):570–577, 1995.
73. L.A. McCue, K.A. McDonough, and C.E. Lawrence. Functional classification of cnmp-binding proteins and nucleotide cyclases with implications for novel regulatory pathways in mycobacterium tuberculosis. *Genome Research*, 10(2):204–219, 2000.
74. M. Mehta, R. Agrawal, and J. Rissanen. SLIQ: A fast scalable classifier for data mining. In *Proceedings of International Conference on Extending Database Technology*, pages 18–32, Avignon, France, 1996.
75. T.M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.
76. H. Nielsen, J. Engelbrecht, S. Brunak, and G. von Heijne. Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Protein Sci.*, 3:3–14, 1994.
77. A.G. Pedersen, P. Baldi, S. Brunak, and Y. Chauvin. Characterization of prokaryotic and eukaryotic promoters using hidden Markov models. *Intelligent Systems for Molecular Biology*, 4:182–191, 1996.
78. A. G. Pedersen and H. Nielsen. Neural network prediction of translation initiation sites in eukaryotes: Perspectives for EST and genome analysis. *Intelligent Systems for Molecular Biology*, 5:226–233, 1997.
79. J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods—Support Vector Learning*. MIT Press, Cambridge, MA, 1998.
80. N. Qian and T.J. Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology*, 202:865–884, 1988.
81. J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.
82. J. R. Quinlan. *C4.5: Program for Machine Learning*. Morgan Kaufmann, San Francisco, CA, 1993.
83. T.H. Rainer, P.K. Lam, E.M. wong, and R.A. Cocks. Derivation of a prediction rule for post-traumatic acute lung injury. *Resuscitation*, 42(3):187–196, 1999.
84. R. Rastogi and K. Shim. Public: A decision tree classifier that integrates building and pruning. In *Proceedings of 24th International Conference on Very Large Data Bases*, pages 404–415, 1998.
85. S. Raudys. How good are support vector machines? *Neural Network*, 13:17–19, 2000.
86. S.K. Riis and A. Krogh. Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments. *Journal of Computational Biology*, 3:163–183, 1996.
87. I. B. Rogozin, A. V. Kochetov, F. A. Kondrashov, E. V. Koonin, and L. Milanesi. Presence of ATG triplets in 5' untranslated regions of eukaryotic cDNAs correlates with a 'weak' context of the start codon. *Bioinformatics*, 17:890–900, 2001.
88. B. Rost and C. Sander. Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins*, 19:55–72, 1994.
89. D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.
90. S. Russell, J. Binder, D. Koller, and K. Kanazawa. Local learning in probabilistic networks with hidden variables. In *Proceedings of 14th Joint International Conference on Artificial Intelligence, volume 2*, pages 1146–1152, 1995.
91. S. L. Salzberg, A. L. Delcher, S. Kasif, and O. White. Microbial gene identification using interpolated Markov models. *Nucleic Acids Research*, 26:544–548, 1998.

92. B. Scholkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
93. J.A. Scott, E. L. Palmer, and A.J. Fischman. How well can radiologists using neural network software diagnose pulmonary embolism? *AJR Am. J. Roentgenol.*, 175(2):399–405, August 2000.
94. H.P. Selker, J.L. Griffith, S. Patil, W.J. Long, and R.B. D'Agostino. A comparison of performance of mathematical predictive methods for medical diagnosis: identifying acute cardiac ischemia among emergency department patients. *J. Investig. Med.*, 43(5):468–476, October 1995.
95. J. Shafer, R. Agrawal, and M. Mehta. SPRINT: A scalable parallel classifier for data mining. In *Proceedings of 22nd International Conference on Very Large Data Bases*, pages 544–555, Bombay, India, September 1996.
96. E. E. Snyder and G. D. Stormo. Identification of protein coding regions in genomic DNA. *Journal of Molecular Biology*, 248:1–18, 1995.
97. Evan W. Steeg. Neural networks, adaptive optimization, and RNA secondary structure prediction. In *Artificial Intelligence and Molecular Biology*, pages 121–160, 1993.
98. C.M. Stultz, R. Nambudripad, R.H. Lathrop, and J.V. White. Predicting protein structure with probabilistic models. In *Protein Structural Biology in Biomedical Research*, pages 447–506, 1997.
99. N.R. Temkin, R. Holubkov, J.E. Machamer, H.R. Winn, and S.S. Dikmen. Classification and regression trees (CART) for prediction of function at 1 year following head trauma. *Journal of Neurosurgery*, 82:764–771, 1995.
100. E.P. Turton, D.J. Scott, M. Delbridge, S. Snowden, and R.c. Kester. Ruptured abdominal aortic aneurysm: A novel method of outcome prediction using neural network technology. *European Journal of Vascular and Endovascular Surgery*, 19:184–189, 2000.
101. E. C. Uberbacher and R. J. Mural. Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach. *Proceedings of the National Academy of Sciences of the United States of America*, 88:11261–11265, 1991.
102. P. E. Utgoff. An incremental ID3. In *Proceedings of 5th International Conference on Machine Learning*, pages 107–120, 1988.
103. V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, 1995.
104. J.L. Vriesema, H.G. van der Poel, F.M. Debruyne, J.A. Schalken, L.P. Kok, and M.E. Boon. Neural network-based digitized cell image diagnosis of bladder wash cytology. *Diagnostic Cytopathology*, 23:171–179, 2000.
105. W.J. Wilbur, G.F. Hazard Jr., G. Divita, J.G. Mork, A.R. Aronson, and A.C. Browne. Analysis of biomedical text for biochemical names: A comparison of three methods. In *Proc. AMIA Symposium'99*, pages 176–180, 1999.
106. T. Yada, M. Ishikawa, H. Tanaka, and K. Asai. Extraction of hidden Markov model representations of signal patterns in DNA sequences. In *Proceedings of Pacific Symposium on Biocomputing*, pages 686–696, 1996.
107. A. E.-J. Yeoh, M. E. Ross, S. A. Shurtleff, W. K. William, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Reilling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, and J. R. Downing. Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell*, 1:133–143, 2002.
108. F. Zeng, R. Yap, and L. Wong. Using feature generation and feature selection for accurate prediction of translation initiation sites. In *Proceedings of 13th International*

Conference on Genome Informatics, pages 192–200, 2002.
109. M.Q. Zhang. Identification of human gene core promoter in silico. *Genome Research*, 8:319–326, 1998.
110. A. Zien, G. Raatsch, S. Mika, B. Schoelkopf, C. Lemmem, A. Smola, T. Lengauer, and K.R. Mueller. Engineering support vector machine kernels that recognize translation initiation sites. In *Proceedings of German Conference on Bioinformatics*, pages 37–43, 1999.
111. A. Zien, G. Raatsch, S. Mika, B. Schoelkopf, T. Lengauer, and K.R. Mueller. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16:799–807, 2000.

Photo and Bibliography

Huiqing Liu is Senior Engineer at the Institute for Infocomm Research, Singapore. She is currently working on knowlege discovery technologies and their application to biomedical data analysis problems. Before joining the Institute for Infocomm Research, she was research fellow and project leader in National University of Singapore (NUS) Bioinformatics Center. She received her Bachelor degree in Quantitative Economics in 1991 from Huazhong University of Science and Technology (China) and her Master degree in Computer Science in 1997 from NUS. She is now persuing her PhD with NUS.

Limsoon Wong is Deputy Executive Director (Research) at the Institute for Infocomm Research, Singapore. He is currently working mostly on knowledge discovery technologies and is especially interested in their application to biomedicine. Prior to that, he has done significant research in database query language theory and finite model theory, as well as significant development work in broad-scale data integration systems. He is a managing editor of the *Journal of Bioinformatics and Computational Biology* and an advisory editor of *BioSilico: Innovations in Computational Drug Discovery*. He received his BSc(Eng) in 1988 from Imperial College London and his PhD in 1994 from University of Pennsylvania, both in Computing.