Using Amino Acid Patterns to Accurately Predict Translation Initiation Sites

Huiqing Liu* Hao Han Jinyan Li Limsoon Wong

Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore, 119613
{huiqing, hanhao, jinyan, limsoon}@i2r.a-star.edu.sg

Summary

The translation initiation site (TIS) prediction problem is about how to correctly identify TIS in mRNA, cDNA, or other types of genomic sequences. High prediction accuracy can be helpful in a better understanding of protein coding from nucleotide sequences. This is an important step in genomic analysis to determine protein coding from nucleotide sequences. In this paper, we present an in-silico method to predict translation initiation sites in cDNA or mRNA sequences. This method consists of three sequential steps as follows. In the first step, candidate features are generated using k-gram amino acid patterns. In the second step, a small number of top-ranked features are selected by an entropy-based algorithm. In the third step, a classification model is built to recognize true TISs by applying support vector machines or ensembles of decision trees to the selected features. We have tested our method on several independent data sets, including two public ones and our own extracted sequences. The experimental results achieved are better than those reported previously using the same data sets. Our high accuracy not only demonstrates the feasibility of our method, but also indicates that there are "amino acid" motifs around TIS in cDNA and mRNA sequences.

Key words: translation initiation site, feature generation, k-gram amino acid patterns, feature selection, classification.

Introduction

The selection of the start site for translation is an important step in the initial phase of protein synthesis. In eukaryotic mRNA, the context of the start codon (normally "AUG") and the sequences around it are crucial for recruitment of the small ribosome subunit. Thus, the characterization of the features around translation start site will be helpful in a better understanding of translation regulation and accurate gene predication of coding region in genomic and mRNA/cDNA sequences. However, since DNA sequences and protein sequences represent the spectrum of biomedical data, they do not possess explicit signals or features. For example, a genomic sequence is just a string consisting of the letters "A", "C", "G", and "T" in an apparently random order. Therefore, when applying traditional machine learning techniques to this recognition problem, there is a need for good methodologies for generating explicit features underlying translation initiation site.

Since 1987, the recognition of TIS has been extensively studied using biological approaches, data mining techniques, and statistical models.[8, 10–13, 17–19, 21, 24, 26] In one of works, Pedersen and Nielsen[18] directly fed DNA sequences into an artificial neural network (ANN) for training the system to recognize true TIS. They achieved a result of 78% sensitivity on start ATGs (true TISs) and 87% specificity on non-start ATGs (false TISs) on a vertebrate data set, giving an overall accuracy of 85%. In Zien et al.,[26] they studied the same vertebrate data set, but replaced ANN with support vector machines (SVMs) using different kinds of kernel functions. They believe that carefully designed kernel functions are useful for achieving higher TIS prediction accuracy. One of their kernels is called "locality-improved" kernel, which emphasizes correlations between any two sequence positions that are close together, and a span of 3 nucleotides up- and down-stream is empirically determined as optimal. Recently, Hatzigeorgiou[8] built a multi-step ANN system named "DIANA-TIS" to study the recognition problem. This ANN system combines a consensus ANN and a coding ANN with the ribosome scanning model. They obtained an overall accuracy of 94% on a data set containing full-length human cDNA sequences. All of these methods use nucleotide sequence data directly; they do not generate any new and explicit features for the differentiation between true and false TISs.

There are some related works that use statistical features. The program ATGpr[21] uses a linear discriminant function that combines some statistical features derived from the sequence. Each of those features is proposed to distinguish true TIS from false TIS. In a more recent work,[17] an improved version of ATGpr called ATGpr_sim was de-

*To whom correspondence should be addressed.

veloped, which uses both statistical information and similarities with other known proteins to obtain higher accuracy of fullness prediction for fragment sequences of cDNA clones. In our previous study,[16,24] features were generated from nucleotide acid patterns and feature selection was conducted to choose significant features for building classification models using machine learning techniques.

Our proposed method consists of the following three steps: (1) generating new features using k-gram amino acid patterns and generating the values of the new features using the frequency of the patterns in the amino acid sequences coded from the original cDNA or mRNA sequences; (2) ranking the newly generated features via their entropy value and selecting important ones; (3) integrating the selected features by machine learning techniques — support vector machines (SVMs) or an ensembles of decision trees method — to recognize true TIS. Our idea is different from traditional methodologies because it generates new features and also transforms the original nucleotide sequence data to amino acid sequence data, and finally to k-gram frequency vectors.

We apply our method on three independent data sets. The first set (data set I) contains 3312 vertebrate sequences,[18] while the second one (data set II) contains 480 completely sequenced and annotated human cDNA sequences.[8] Our cross validation accuracy on the first data set is 92.45%, which is better than 89.4%, the best reported result on this data set. Our cross validation accuracy on the second data set is 98.16%; we did not find literature cross validation results on this data set for comparison. (Note that the results of Hatzigeorgiou[8] is not directly comparable due to the use of the ribosome scanning model and different way of data splitting.) We also conduct experiment to test the accuracy on one data set when using our model trained on another data set. Besides, this kind of cross data sets validation is further applied to genomic data. We formed our data set III by extracting a number of well-characterized and annotated human genes of Chromosome X and Chromosome 21 from Human Genome Build30. Such a validation is highlighted because all the previously reported results are based on only one data set, the performance of those methods on other independent data are not reported. Our good accuracy in different experiments not only demonstrates the feasibility of our method, but also indicates that there are "amino acid" motifs around TIS in cDNA and mRNA sequences.

Results

To verify the effectiveness of our method, we designed a series of experiments on three data sets:

a. Conducting computational cross validations in data set I and data set II separately. In k-fold cross validation, data set is divided randomly into k disjoint subsets of approximately equal size, in each of which the class is represented in approximately the same proportions as in the full data set.[23] We train the model k times, each time one of the subsets is held out in turn from training while feature selection and classification model building are conducted on the remaining $k-1$ subsets and evaluated on the holdout set. After all subsets being tested, an overall performance is produced.

b. Selecting features and building classification model using data set I. Applying the well-trained model to data set II to obtain a blind testing accuracy.

c. Incorporating the idea of ribosome scanning into the classification model.

d. Applying the model built in experiment-b to genomic sequences.

Validation in different data sets

To strictly compare with our previous study results presented in,[16,24] we conduct the same 3-fold cross validation. Table 1 shows our results on the data set I and data set II when the top 100 features are selected by the entropy-based algorithm. Using the simple linear kernel function, SVMs achieves accuracy of 92.45% at 80.19% sensitivity and 96.48% specificity on data set I. This is better than the accuracy of 89.4% at 74.0% sensitivity and 94.4% specificity, which is the previous best result reported on the same data set.[24] On data set II, SVMs (with linear kernel) achieves an accuracy of 98.16% at 63.75% sensitivity and 99.67% specificity. Note that we can not find previously reported results on this data set under similar across validation.

Validation across two data sets

The good cross validation results within the individual data set encouraged us to extend our study to span the two data sets. In this experiment, we use the whole data set I as training data to select features and build the classification model, then we test the well-trained model on data set II to get a test accuracy.

Before doing this validation test, we removed from data set II 292 sequences that are similar to the training data. Using the classification model learnt from 100 top-ranked features of data set I, we got a test accuracy of 89.42% at 96.28% sensitivity and 89.15% specificity on data set II using SVMs built on the linear kernel function. The training accuracy is 92.77% at 80.68% sensitivity and 96.75% specificity (on data set I). We note that the testing accuracy

Table 1: The results by 3-fold cross validation on the two data sets when 100 top features are considered (experiment-a). SVM(linear/quad) means the classification model is built by linear/quadratic polynomial kernel function.

Data	Algorithm	Sensitivity	Specificity	Precision	Accuracy
I	SVMs(linear)	80.19%	96.48%	88.24%	92.45%
	SVMs(quad)	80.19%	96.17%	87.34%	92.22%
	Ensemble Trees	76.18%	96.14%	86.67%	91.20 %
II	SVMs(linear)	63.75%	99.67%	87.18%	98.16%
	SVMs(quad)	71.25%	99.42%	81.24%	98.46%
	Ensemble Trees	83.54%	97.67%	55.93%	97.19%

Table 2: Classification accuracy when using data set I as training and data set II as testing (experiment-b). The row of II** is the testing accuracy on data set II before similar sequences being removed.

Data	Algorithm	Sensitivity	Specificity	Precision	Accuracy
I (train)	SVMs(linear)	80.68%	96.75%	89.10%	92.77%
	SVMs(quad)	86.05%	98.14%	93.84%	95.15%
	Ensemble Trees	85.54%	97.91%	93.10%	94.85%
II (test)	SVMs(linear)	96.28%	89.15%	25.31%	89.42%
	SVMs(quad)	94.14%	90.13%	26.70%	90.28%
	Ensemble Trees	92.02%	92.71%	32.52%	92.68%
II** (test)	SVMs(linear)	95.21%	89.74%	24.69%	89.92%
	SVMs(quad)	94.38%	89.51%	24.12%	89.67%
	Ensemble Trees	87.70%	93.26%	28.60%	92.11%

on the original data set II (without the removal of the similar sequences) is quite similar. See Table 2 for a summary of these two results.

Remarkably, this cross-validation spanning the two data sets shows a much better sensitivity on data set II than that achieved in the 3-fold cross-validation on this data set. A reason may be that only 3.41% ATGs in data set II are true TIS, which leads to an extremely unbalanced numbers of samples between the two classes. However, this bias is rectified significantly by the model built on data set I where the population size of true TIS v.s. false TIS is more balanced.

Incorporation of scanning model

Hatzigeorgiou[8] reported a high accuracy on data set II by an integrated method which combines a consensus ANN with a coding ANN together with a ribosome scanning model. The model suggests to scan from the 5' end of a cDNA sequence and predicts TIS at the first ATG in a good context.[1,4,11] The rest of the ATGs in the cDNA sequence to the right of this ATG are then automatically classified as non-TIS. Thus, one and only one ATG is predicted as TIS per cDNA sequence.

We also incorporate this scanning model into our experiment. This time, in a sequence, we test ATGs in turn from left to right, utill one of them is classified as TIS. A prediction on a sequence is correct if and only if the TIS itself is predicted as a TIS. Since the scanning model indicates that the first ATG that in an optimal nucleotide context would be TIS, a higher prediction accuracy is expected if only up-stream ATGs and true TIS are used. Thus, we ignore all down-stream ATGs in data set I and obtain a new training set containing only true TISs and their up-stream ATGs. Then feature selection and classification model learning are based on this new training data. Table 3 shows our results with scanning model being used.

Under this scanning model idea, Artemis reported that 94% of the TIS were correctly predicted on data set II.[8] Since in her paper,[8] the data set was split into training and testing parts in some way, the results reported there are not directly comparable with our results.

Testing on genomic sequences

In order to further evaluate the feasibility and robustness of our method, we apply our model built in experiment-b to our own prepared data (data set III), which containing gene sequences of Chromosome X and Chromosome 21. Using the simple linear kernel function, SVMs gives 397 correct prediction out of a total of 565 true TISs found in Chromosome X while 132 correct prediction out of a total of 180 true TISs in Chromosome 21. The prediction rates

Table 3: Classification accuracy under scanning model when using data set I (3312 sequences) as training and data set II (188 sequences) as testing (experiment-c). The row of II** is the testing accuracy on data set II before similar sequences being removed (480 sequences). NoCorPred is the number of sequences whose TIS is correctly predicted.

Data	Algorithm	NoCorPred	Accuracy
I	SVMs(linear)	3161	95.44%
(train)	SVMs(quad)	3156	95.29%
	Ensemble Trees	3083	93.09%
II	SVMs(linear)	174	92.55%
(test)	SVMs(quad)	172	91.49%
	Ensemble Trees	176	93.62%
II**	SVMs(linear)	453	94.38%
(test)	SVMs(quad)	450	93.75%
	Ensemble Trees	452	94.17%

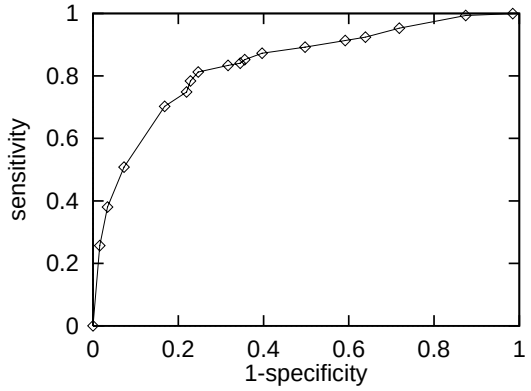

Figure 1: ROC curve of our model on prediction TIS in genomic data Chromosome X and Chromosome 21 (experiment-d). Our SVM model is built on the linear kernel function.

are 70.27% and 73.33%, respectively. One point needs to be addressed here is that in this validation, we removed the feature built on the ribosome scanning model since that model is not true for genomic data.

To illustrate the tradeoff between the prediction sensitivity and specificity, we randomly selected same number of sequences containing non-start ATGs (false TIS) from our own extracted negative data set. Figure 1 gives the ROC curve showing the changes of prediction accuracy on true and false TIS samples.

Discussion

In this study, we proposed a method for recognition of TIS in cDNA or mRNA sequences via generating amino acid patterns. We designed a series of experiments by applying our method to some public data sets as well as our own extracted sequences. Our testing accuracy are better than the previously reported best ones (where available) on the same data sets. Most importantly, we not only conducted the cross validation within the individual data sets separately, but also established the validation across the different data sets, including genomic data. The success of such a validation indicates that there are common feature motifs around true TIS in cDNA or mRNA sequences.

Significant Features

"What are the key features to predict TIS?" To answer this question, let us have a look of an interesting discovery on the features selected in the 3-fold cross validation on data set I in our experiment-a. Table 4 shows the ranking positions of the 10 top-ranked features selected by the entropy-based algorithm for the each fold. Observe that they are the same features though their ordering is slightly different from one fold to another. This suggest that these features, or exactly amino acid patterns, are indeed motifs around true or false TISs. Furthermore, "UP-ATG" can be explained by the ribosome scanning model[1,4] — seeing such an up-stream ATG makes the candidate ATG less likely to be the TIS. "DOWN-STOP" is the in-frame stop codons down-stream from the target ATG and it is consistent with the biological process of translating in-frame codons into amino acids stops upon encountering an in-frame stop codon — seeing such a down-stream stop codon makes the candidate protein improbably short. "UP3-AorG" is correspondence to the well-known Kozak consensus sequence.[10] Most of the other features were also identified in our previous study,[24] in which the feature space is built directly on nucleotides. Remarkably, these amino acid patterns, except "DOWN-L", all contain "G" residue. Note also that "UP-M" is one of 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 significance of these features is further verified when we find that both sensitivity and specificity drop down greatly if these features are excluded from prediction. However, we do not observe obvious decrease when we remove any one of them from the classification model.

In addition to the result when the 100 top-ranked features are used, we also obtained cross-validation results when the whole feature space (i.e. without feature selection) and other numbers (such as 5, 10, 20, 50, 200 and 300) of top-ranked features are used. We found that (all the results are on the basis of SVMs using linear kernel function), (1) using the whole feature space could not let us achieve best results. In fact, we got an accuracy of only 90.94% at 79.86% sensitivity and 94.58% specificity for data set I when running 3-fold cross validation on data set I. This result is not

Table 4: Ranking of the top 10 features selected by the entropy-based algorithm as relevant in each of the 3 folds of data set I. Feature "UP-ATG" indicates whether an in-frame up-stream ATG exists (boolean type). Feature "UP3-AorG" tests whether purine A or G tends to be found 3 nucleotides up-stream of a true TIS (boolean type). Feature "UP(DOWN)-X" counts the occurrence that an in-frame (relative to the candidate ATG) triplet coding for the amino acid letter X appears in the up-stream (down-stream) part of a candidate ATG. Feature "DOWN-STOP" is the occurrence of in-frame stop condons down-stream of a candidate ATG.)

Fold	UP-ATG	DOWN-STOP	UP3-AorG	DOWN-A	DOWN-V	UP-A	DOWN-L	DOWN-D	DOWN-E	UP-G
1	1	2	4	3	6	5	8	9	7	10
2	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	8	9	7	10

as good as that on the 100 top-ranked features. (2) using a small number of features could not achieve good results, either. For example, the accuracy is only 87.44% if only top 5 features are used on data set I; (3) results by top 200, 300 or more features show no much difference with the result by the 100 features. All these observations indicate that the results achieved by using the top 100 features is reasonable. This also suggests that in real biological process of translation there are some factors other than Kozak consensus that may regulate the recognition of TIS.

Classification Algorithms

For the classification methods, overall speaking, SVMs performs slightly better than our ensembles of decision trees method, in terms of prediction accuracy. However, our tree committee achieves very good sensitivity when running 3-fold cross validation on data set II where the number of true TISs is much less than the number of false TISs. Besides, decision trees can output comprehensive rules to disclose the essence of learning and prediction. Some discovered interesting and biologically sensible rules with larger coverage are listed below.

1. If *UP-ATG='Y'* and *DOWN-STOP>0*, then prediction is *false TIS*.

2. If *UP3-AorG='N'* and *DOWN-STOP>0*, then prediction is *false TIS*.

3. If *UP-ATG='N'* and *DOWN-STOP<0* and *UP3-AorG='Y'*, then prediction is *true TIS*.

On the other hand, in our series of experiments, SVMs built on quadratic polynomial kernels do not show their advantages over those built on simple linear kernel functions. Note that quadratic kernels need much more time on training process.

Comparison with ATGpr

As mentioned earlier, ATGpr[17,21] is a TIS prediction program that makes use of a linear discriminant function, several statistical measures derived from the sequence and the ribosome scanning model. It can be accessed via `http://www.hri.co.jp/atgpr/`. When searching TIS in a given sequence, the system will output several (5 by default) ATGs in the order of decreasing confidence but we always predict the ATG with highest confidence as TIS. For the 3312 sequences in our data set I, ATGpr can predict correctly true TIS in 2941 (88.80%) of them. This accuracy is 6.64% lower than that we achieved. For our data set II, true TIS in 442 (92.0%) of 480 sequences are properly recognized, which is about 2.38% lower than the accuracy obtained by us. Our results quoted here are based on SVM model using the linear kernel function.

When we feed the genomic data used in our experiment-d to ATGpr, the program can give correct TIS prediction on 128 (71.11%) of 180 Chromosome 21 gene sequences and 417 (73.81%) of 565 Chromosome X gene sequences, giving the overall sensitivity as 73.15%. On the other hand, ATGpr achieves 70.47% specificity on the same number of negative sequences that were used in our experiment-d. From the ROC curve in Figure 1, we can find our prediction specificity is around 80% when sensitivity is set to 73.15% — 9.5% higher than that of ATGpr on specificity. This indicates that our program may also outperform ATGpr when dealing with genomic data sequences.

Materials and Methods

Data

The first data set (data set I) is provided by Dr. Pedersen. It consists of vertebrate sequences extracted from GenBank (release 95). The sequences are further processed by removing possible introns and joining the remaining exon parts to obtain the corresponding mRNA sequences. [18] From these sequences, only those with an annotated TIS,

and with at least 10 upstream nucleotides as well as 150 downstream nucleotides are considered in our studies. The sequences are then filtered to remove homologous genes from different organisms, sequences added multiple times to the database, and those belonging to same gene families. Since the data are 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. We are aware that some TIS sites may be non-ATG; however, this is reported to be rare in eukaryotes[13] and is not considered in this paper.

An example entry from this data set is given in Figure 2. There are 4 ATGs in this example. The second ATG is the TIS. The other 3 ATGs are non-TIS (false 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. The entire data set contains 3312 sequences. In these sequences, there are a total number of 13375 ATGs, of which 3312 ATGs (24.76%) are true TIS, while 10063 (75.24%) are false. Of the false TISs, 2077 (15.5%) are up-stream ATGs.

The second data set (data set II) is provided by Dr. Hatzigeorgiou. The data collection was first made on the protein database Swissprot. All the human proteins whose N-terminal sites are sequenced at the amino acid level were collected and manually checked.[8] Then the full-length mRNAs for these proteins, whose TIS had been indirectly experimentally verified, were retrieved. The data set consists of 480 human cDNA sequences in standard FASTA format. In these sequences, there are as many as 13581 false TIS, 96.59% of total number of ATGs. However, only 241 (1.8%) of them are up-stream ATGs.

To reduce the *similarity* between the training and testing data, a *BLAST* search between the data set I and II is performed. Two sequences are considered similar if they produce a BLAST hit with an identity $> 75\%$. We found 292 similar sequences and removed them from data set II. As a result, after being removed similar sequences, data set II contains 188 real TIS, while there are total number of 5111 candidates.

Besides these two data sets that have been analyzed by others, we also formed our own genomic ATG data set (data set III) by extracting a number of well-characterized and annotated human genes of Chromosome X and Chromosome 21 from Human Genome Build30. Note that we eliminated those genes that were generated by other prediction tools. The resulting set consists of 565 sequences from Chromosome X and 180 sequences from Chromosome 21. These 745 sequences containing true TIS are used as positive data in our experiment-d. For negative data, we extracted a set of sequences around all ATGs in these two chromosomes but excluded annotated ones.

Methods

Our method comprises three steps: (1) generating candidate features from the original sequence data; (2) selecting relevant features using an entropy-based algorithm; and (3) integrating the selected features by classification algorithm to build a system to correctly recognize true TIS. An important component of our method is to generate a new feature space. After that, we follow some standard feature selection and feature integration ideas to make TIS predictions.

Feature generation

We generate the new feature space using k-gram ($k = 1, 2, 3, ...$) *amino-acid patterns*. A k-gram is simply a pattern of k consecutive letters, which can be amino acid symbols or nucleotide symbols.[16,25] We use each k-gram amino acid pattern as a new feature. For example, "AR" is a 2-gram pattern constituted by an alanine followed by an arginine. Our aim is to recognize a TIS from large amount of candidate ATGs by analysing k-gram amino acid patterns around it. In our study, up-stream and down-stream k-gram amino acid patterns of an ATG (every ATG is a candidate of TIS) are treated as different features. Since there are 20 standard amino acids plus 1 stop symbol, there are 2×21^k possible combinations of k-gram patterns for each k.

The *frequency* of the k-gram amino acid patterns are used as the values of the features. For example, (1) UP-X (DOWN-X), which counts the number of times the amino acid letter X appears in the up-stream (down-stream) part of an ATG in its amino acid sequence, for X ranging over the standard 20 amino acid letters and the special stop symbol. (2) UP-XY (DOWN-XY), which counts the number of times the two amino acid letters XY appear as a substring in the up-stream (down-stream) part of an ATG in its amino acid sequence, for X and Y ranging over the standard 20 amino acid letters and the special stop symbol. In this paper, we use 1-gram and 2-gram patterns only. Thus, there are 924 ($= (21 + 21^2) \times 2$) possible amino acid patterns, i.e. new features.

In the framework of the new feature space, the initial nucleotide sequences need to be transformed. The transformation is as follows. Given a cDNA or mRNA nucleotide sequence containing ATGs, a window is set for each ATG with the ATG in the center and 99 bases up-stream and 99 bases down-stream (excluding the ATG itself) aside. If an ATG does not have enough up-stream or down-stream context—that is, there are less than 99 nucleotides to its left or to its right—we pad the missing context with the appropriate number of dont-care ("?") symbols. As such, for data set I, we get 3312 sequence windows containing true TIS and 10063 containing false TIS; for data set II, 480 sequence windows containing true TIS and 13581 containing false TIS. All the windows have same size, i.e. contain-

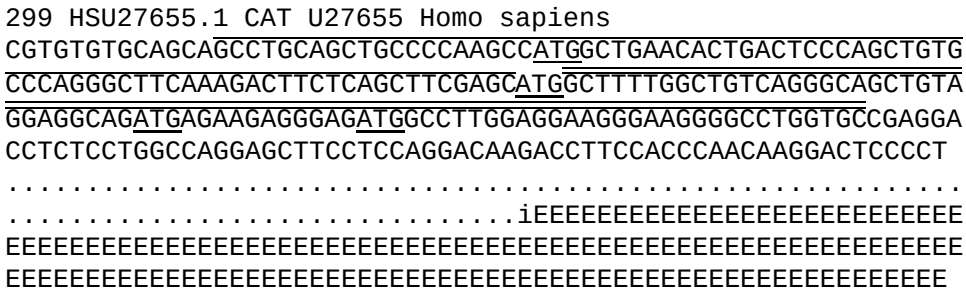

Figure 2: An example annotated sequence from data set I. The 4 occurrences of ATG are underlined. The second ATG is the TIS. The other 3 ATGs are non-TIS. The 99 nucleotides up-stream of the TIS are marked by an overline. The 99 nucleotides down-stream of the TIS are marked by a double overline. The ".", "i", and "E" are annotations indicating whether the corresponding nucleotide is up-stream (.), TIS (i), or down-stream (E).

ing 201 nucleotides. For ease of discussion, given a sequence window, we refer to each position in the sequence window relative to the target ATG of that window. 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 window. [16]

Next, we code every triplet nucleotides, at both up-stream and down-stream of the centered ATG in a sequence window, into an *amino acid* using the standard codon table. A triplet that corresponds to a stop codon is translated into a special "stop" symbol. Thus, every nucleotide sequence window is coded into another sequence consisting of amino acid symbols and "stop" symbol.

Then the amino acid sequences are converted into frequency sequence data under the description of our new features. Later, the classification model will be applied to the frequency sequence data, rather than the original cDNA sequence data or the intermediate amino acid sequence data.

Apart from these k-gram amino acid patterns, we also make use of 3 bio-knowledge patterns as new features. From the original work for the identification of the TIS in cDNA sequences, Kozak developed the first weight matrix from an extended collection of data. [10] The consensus motif from this matrix is GCC[AG]CCATGG, where (1) a G residue tends to follow a true TIS, which indicates that a "G" appears in position +4 of the original sequence window; (2) purine (A or G) tends to be found 3 nucleotides up-stream of a true TIS, which indicates that an "A" or a "G" appears in position -3 of the original sequence window. Also, according to the ribosome scanning model, [1,4,11] a mRNA sequence is scanned from left (5') to right (3'), and the scanning stops as soon as an ATG is recognized as TIS. The rest of the ATGs in the mRNA sequence to the right of this ATG are then treated as non-TIS. To incorporate these knowledge to our feature space, we add three Boolean features "DOWN4-G","UP3-AorG" and "UP-ATG" (whether an in-frame up-stream ATG exists). Thus, there are 927 features in the new feature space.

After this process of feature generation and data transformation, we get 3312 true TIS samples and 10063 false TIS samples from data set I, 480 true TIS samples and 13581 false TIS samples from data set II. Each sample is a vector of 924 integers and three boolean values. Figure 3 presents a diagram for the data transformation with respect to our new feature space.

Feature selection

Since the number of candidate features in the feature space is relatively big, we expect that some of the features would be irrelevant to our prediction problem. So the next step of our method is to apply a feature selection technique to the feature space to pick those features that most likely to help in distinguishing true TIS from false TIS.

In this study, we use a simple and efficient entropy-based algorithm to select important features. The basic idea of this algorithm[5] is to filter out those features whose values are relatively random. For the remaining features, the algorithm can automatically find some cut points in these features' value ranges such that the resulting intervals of every feature can be maximally distinguished. If every interval induced by the cut points of a feature contains only the same class of samples (such as true TIS), then this partitioning by the cut points of this feature has an entropy value of zero. This is an ideal case.

This algorithm is outlined in the following. 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

$$Ent(f, \mathcal{U}, S) = -\sum_{\mathcal{C} \in \mathcal{U}} P(f, \mathcal{C}, S) log(P(f, \mathcal{C}, S))$$

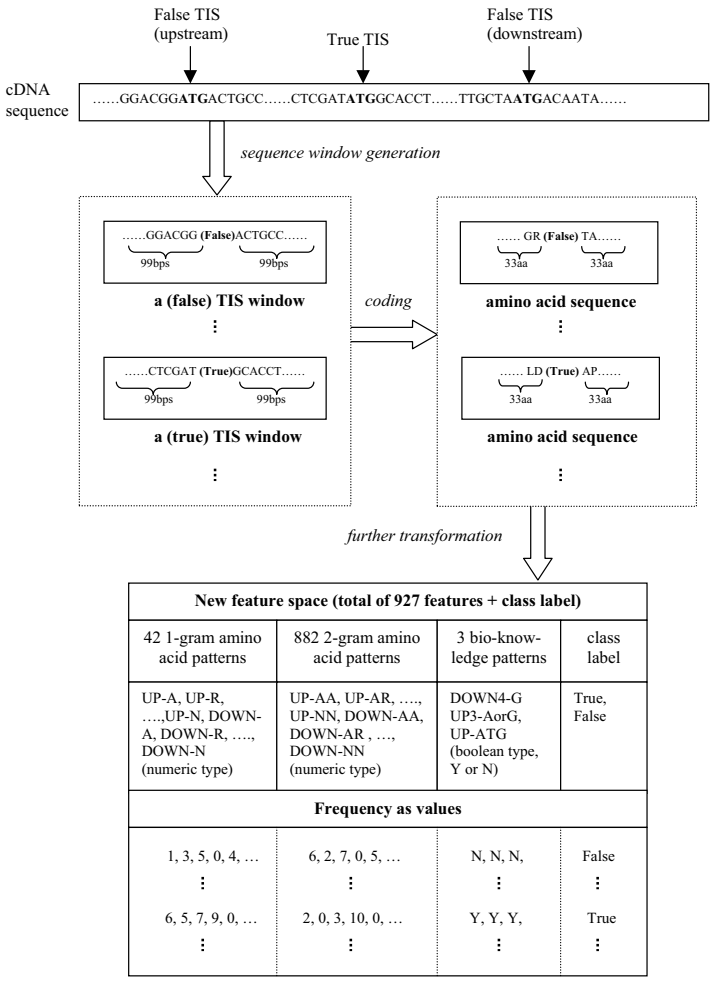

Figure 3: A diagram for data transformation aiming for the description of the new feature space.

Let T partition 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 entropy* of partition (S_1, S_2). Its definition is given below, where $n(f, \mathcal{U}, S)$ means the number of samples in the class $\mathcal{U}$ 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. A commonly used stopping criteria is the so-called minimal description length principle given in.[5]

In the selection process, all features are first ranked according to their class entropy values in an ascending order, then some certain number of top-ranked features are considered to build the model.[15]

Feature integration

To achieve the ultimate goal of predicting true TIS, our next step is to integrate the selected features by a classification algorithm. At this step, we consider support vector machines (SVMs) and our own developed method of constructing ensembles of decision trees.[14] SVMs is chosen as

our main classifier since it is known to have good classification performance in the biological domain.[7,26] On the other hand, from the experiments of using decision trees, we expect to get some interesting rules for TIS prediction.

SVMs SVM is a kind of blend of linear modeling and instance-based learning.[23] It originates from research in statistical learning theory.[22] A SVM selects a small number of critical boundary samples from each class and builds a linear discriminant function (also called maximum margin hyperplane) that separates them as widely as possible. In the case that no linear separation is possible, the technique of *kernel* will be used to automatically inject the training samples into a higher-dimensional space, and to learn a separator in that space.[23] A maximum margin hyperplane $G(T)$ for a test sample T is a linear combination of kernels computed at the training data points and is constructed as

$$G(T) = sign(\sum_i [\alpha]_i * [Y]_i * k(T, [X]_i) + b)$$

where $[X]_i$ are the training data points, $[Y]_i$ are the class labels (which are assumed to have been mapped to 1 or -1) of these data points, $k(\cdot,\cdot)$ is the kernel function, b and $[\alpha]_i$ are parameters that determine the hyperplane and can be learned from the training data. The training of a SVM is a quadratic programming problem and here, we omit the detailed description about this. A good tutorial to understand SVMs is the one written by Burges.[3]

There are several ways for training support vector machines. One of the fastest algorithms is developed by Platt,[9,20] which solves the above quadratic programming problem by sequential minimal optimization (SMO). In our experiments, we use the implementation of SMO in *Weka* (version 3.2), a free machine learning software package written in Java and developed at University of Waikato in New Zealand.[27] The kernel is the polynomial function and the transformation of the output of SVM into probabilities is conducted by a standard sigmoid function. In most of cases in this paper, we present our results obtained by linear and quadratic polynomial kernels.

Ensembles of decision trees Ensemble methods, such as constructing committees of decision trees, have shown significant effectiveness in improving the accuracy of single base classifiers.[2,6,14] Compared with SVMs, approaches of constructing decision trees organize what they learned from data in a more comprehensible way — tree format. Furthermore, every branch of a decision tree can be easily translated into a rule in the format of "If$\cdots$, then$\cdots$".[23]

The main idea of this classification algorithm is to use different top-ranked features as the root node of a member tree. Different from Bagging or Boosting[2,6] which uses bootstrapped data, we always build decision trees using exactly the same set of training samples. In detail, to construct k number of decision trees $k \leq n$ (n is the number of features describe the data), we have following steps:

(1) Ranking all the n features according to certain criterion, with the best feature at the first position.

(2) $i = 1$.

(3) Using the ith feature as root node to construct ith decision tree.

(4) If $i < k$, increasing i by 1 and goto (3); otherwise, stop.

In this paper, *information gain ratio* is used as the measure to rank features and the number of decision trees k is set as 20. When doing classification, we define the *coverage* of a rule in a tree as the percentage of the samples in its class satisfying the rule. Suppose we have discovered k decision trees from our training set containing true TIS and false TIS samples. Then, all the rules derived from k trees can be categorized into two groups: one group only containing rules for true TIS samples, another containing rules for false TIS samples. In each group, we rank the rules in descending order according to their coverage, such as

$$rule_1^{true}, rule_2^{true}, \cdots, rule_k^{true},$$

and

$$rule_1^{false}, rule_2^{false}, \cdots, rule_k^{false}.$$

Given a test sample T, each of the k trees will have a rule to fit this sample and therefore, give a prediction for this sample. Suppose that T satisfies the following k_1 true TIS rules and k_2 false TIS rules:

$$rule(T)_1^{true}, rule(T)_2^{true}, \cdots, rule(T)_{k_1}^{true},$$

and

$$rule(T)_1^{false}, rule(T)_2^{false}, \cdots, rule(T)_{k_2}^{false}.$$

Where $0 \leq k_1, k_2 \leq k$ and $k_1 + k_2 = k$. The order of these rules is also based on their coverage. When we make a prediction for T, two scores will be calculated as following:

$$Score(T)^{true} = \sum_{i=1}^{k_1} \frac{coverage(rule(T)_i^{true})}{coverage(rule_i^{true})}/k,$$

$$Score(T)^{false} = \sum_{i=1}^{k_2} \frac{coverage(rule(T)_i^{false})}{coverage(rule_i^{false})}/k.$$

If $Score(T)^{true} \geq Score(T)^{false}$, then T will be predicted as a true TIS; Otherwise, T predicted as a false TIS. In practice, the tie-score case occurs rarely.[14]

Model evaluation

We adopt standard performance measures defined as follows. *Sensitivity* measures the proportion of TIS that are correctly recognized as TIS. *Specificity* measures the proportion of false TIS that are correctly recognized as false TIS. *Precision* measures the proportion of the claimed TIS that are indeed TIS. *Accuracy* measures the proportion of predictions, both for true TIS and false TIS, that are correct. Let TP be the true positives, TN the true negatives, FP the false positives, and FN the false negatives. Then the above measures are defined as: $sensitivity = TP/(TP + FN)$, $specificity = TN/(TN + FP)$, $precision = TP/(TP + FP)$, and $accuracy = (TP + TN)/(TP + FN + TN + FP)$. Besides, we also plot *ROC* (Receiver Operating Characteristic) curve for the testing on genomic sequences where we select same number of true TIS and false TIS samples. From a ROC curve, the tradeoff between sensitivity and specificity can be illustrated clearly.

Acknowledgement

We wish to thank Dr Anders G. Pedersen and Dr Artemis G. Hatzigeorgiou for providing their data sets.

References

[1] Agarwal, P. & Bafna, V., The ribosome scanning model for translation initiation: implications for gene prediction and full-length cDNA detection. *Intelligent Systems for Molecular Biology*, **6**, 2-7, 1988.

[2] Breiman, L., Bagging predictors. *Machine Learning*, **24**, 123-140, 1996.

[3] Burges, C.J.C., A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, **2(2)**, 121-167, 1998

[4] Cigan, A., Feng, L. & Donahue, T., tRNA functions in directing the scanning ribosome to the start site of translation. *Science*, **242**, 93-97, 1988

[5] Fayyad, U. & Irani, K., Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, 1022–1029, Morgan Kaufmann, 1993.

[6] Freund, Y. and Schapire, R.E., Experiments with a new boosting algorithm. In Saitta, L. editor, *Machine Learning: Proceedings of the 13th International Conference*, 148–156, Morgan Kaufmann, 1996.

[7] Furey, T. S., Cristianini, N., Duffy, N., Bednarski, D. W., Schummer, M. & Haussler, D., Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, **16**, 906-914, 2000.

[8] Hatzigeorgiou, A.G., Translation initiation start prediction in human cDNAs with high accuracy. *Bioinformatics*, **18**, 343-350, 2002.

[9] Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., & Murthy, K.R.K, Improvements to Platt's SMO Algorithm for SVM Classifier Design. *Technical Report CD-99-14*. Control Division, Dept of Mechanical and Production Engineering, National University of Singapore, 1999.

[10] Kozak, M., An analysis of 5'-noncoding sequences from 699 vertebrate messenger RNAs. *Nucleic Acids Research*, **15**, 8125-8148, 1987

[11] Kozak, M., The scanning model for translation: an update. *Journal of Cell Biology*, **108**(2), 229-241, 1989.

[12] Kozak, M., Interpreting cDNA sequences: some insights from studies on translation. *Mamalian Genome*, **7**, 563-574, 1996

[13] Kozak, M., Initiation of translation in prokaryotes and eukaryotes. *Gene*, **234**, 187-208, 1999.

[14] Li, J., & Liu, H., Ensembles of Cascading Trees. Accepted by *3rd IEEE Int. Conf. on Data Mining*, Melbourne, Florida, USA, Nov., 2003.

[15] Liu, H., Li, J., & Wong, L., A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. In *Proceedings of 13th Workshop on Genome Informatics*. Universal Academy Press, 51-60, 2002.

[16] Liu, H., & Wong, L., Data Mining Tools for Biological Sequences. *Journal of Bioinformatics and Computational Biology*, **1(1)**, 139-168, Imperial College Press, April 2003.

[17] Nishikawa, T., Ota, T., & Isogai, T., Prediction whether a human cDNA sequence contains initiation codon by combining statistical information and similarity with protein sequences. *Bioinformatics* **16**, 960-967, 2000

[18] Pedersen, A.G., & Nielsen, H., Neural network prediction of translation initiation sites in eukaryotes: perspectives for EST and genome analysis. In *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, 226-233, 1997.

[19] Peri, S., & Pandey, A., A reassessment of the translation initiation codon in vertebrates. *TRENDS in Genetics*, **17(12)**, 2001.

[20] Platt, J., Fast Training of Support Vector Machines using Sequential Minimal Optimization. In *Advances in Kernel Methods - Support Vector Learning*. Edited by B. Schlkopf, C. Burges, and A. Smola, MIT Press, 1998

[21] Salamov, A.A., Nishikawa, T., & Swindells, M.A., Assessing protein coding region integrity in cDNA sequencing projects. *Bioinformatics*, **14**, 384-390, 1998.

[22] Vapnik, V.N., *The Natural of Statistical Learning Theory*, Springer, 1995

[23] Witten, H. & Frank, E., Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation. *Morgan Kaufmann*, San Mateo, CA, 2000.

[24] Zeng, F., Yap, H.C., & Wong, L., Using feature generation and feature selection for accurate prediction of translation initiation sites. In *Proceedings of 13th Workshop on Genome Informatics*. Universal Academy Press, 192-200, 2002.

[25] Zhang, M.Q., Identification of human gene core promoter in silico. *Genome Research*, 8: 319-326, 1998

[26] Zien, A., Raetsch, G., Mika, S., Schoelkopf, B., Lemmen, C., Smola, A., Lengauer, T. & Mueller, K.-R., Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, **16**, 799-807, 2000.

[27] http://www.cs.waikato.ac.nz/ml/weka/.