An *In-Silico* Method for Prediction of Polyadenylation Signals in Human Sequences

Huiqing Liu huiqing@i2r.a-star.edu.sg
Hao Han hanhao@i2r.a-star.edu.sg
Jinyan Li jinyan@i2r.a-star.edu.sg
Limsoon Wong limsoon@i2r.a-star.edu.sg
Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613

Abstract

This paper presents a machine learning method to predict polyadenylation signals (PASes) in human DNA and mRNA sequences by analysing features around them. This method consists of three sequential steps of feature manipulation: generation, selection and integration of features. In the first step, new features are generated using k-gram nucleotide acid or amino acid patterns. In the second step, a number of important features are selected by an entropy-based algorithm. In the third step, support vector machines are employed to recognize true PASes from a large number of candidates. Our study shows that true PASes in DNA and mRNA sequences can be characterized by different features, and also shows that both upstream and downstream sequence elements are important for recognizing PASes from DNA sequences. We tested our method on several public data sets as well as our own extracted data sets. In most cases, we achieved better validation results than those reported previously on the same data sets. The important motifs observed are highly consistent with those reported in literature.

Keywords: polyadenylation signals, machine learning, feature selection, support vector machines

1 Introduction

The general polyadenylation machinery of mammalian cells has been well studied for decades. The polyadenylation (poly(A)) reaction of mammalian pre-mRNAs proceeds in two stages: the cleavage of pre-mRNA and the addition of poly(A) tail to the newly formed 3' end. The cleavage reaction requires the cleavage/poly(A) specificity factor (CPSF), the cleavage stimulation factor (CStF), the cleavage factors I and II (CF I and CF II), and poly(A) polymerase (PAP) in most cases. CPSF, PAP and poly(A) binding protein 2 are involved in poly(A) [22]. The assembly of the cleavage/poly(A) complex, which contains most or all of the processing factors and the substrate RNA, occurs cooperatively. CPSF consists of four subunits and binds to the highly conserved AAUAAA hexamer upstream of the cleavage site. CStF, which is necessary for cleavage but not for addition of poly(A) tail, interacts with the U/GU rich element located downstream of the AAUAAA hexamer. Two additional factors, the cleavage factor I and II (CF I and CF II) act only in the cleavage step. CF I has been purified to homogeneity and shown to be an RNA-binding factor. CF II has been only partially purified so far, and its function is not known.

After the formation of the cleavage/polyadenylation complex, the selection of poly(A) site is primarily determined by the distance between a hexameric poly(A) signal (PAS) of sequence AAUAAA (or a one-base variant) and the downstream element(denoted as DSE). The spacing requirements for the PAS and DSE reflect the spatial requirements for a stable interaction between CPSF and CstF. The DSE is poorly conserved and two main types have been described as a U-rich, or GU-rich element, which locates 20 to 40 bases downstream of the cleavage site (for reviews, please refer to [5, 20, 22]).

DSE is present in a large proportion of genes and can affect the efficiency of cleavage [11, 20]. Although in few cases, an upstream element (denoted as USE) is required for the poly(A) signal to be fully activated [1, 2, 13], the position and sequence of the USE are undefined. In summary, the organization of mammalian poly(A) sites may have an unexpected flexibility and their activity depends on not only the hexameric signal but also the up/down elements. Figure 1 is a schematic representation of PAS in human mRNA 3'end processing site [22].

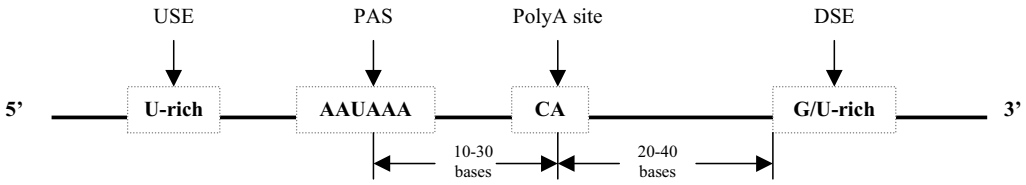

Figure 1: Schematic representation of PAS in human mRNA 3'end processing site. Distances are as described in [5].

There are several software programs that have been developed to detect PASes in human DNA and mRNA sequences by analysing the characteristics of upstream and downstream sequence elements around PASes. In one of early studies, Tabaska and Zhang [16] developed a program named *Polyadq*, which finds PASes using a pair of quadratic discriminant functions. Besides, they also created a database of known active poly(A) sites and trained their program on 280 mRNA sequences and 136 DNA sequences. In their tests of finding PASes, they claimed a correlation coefficient of 0.413 on whole genes and 0.512 in the last two exons of genes. Polyadq is available at `http://argon.cshl.org/tabaska/polyadq_form.html`. Recently, Legendre and Gautheret [8] used bioinformatics analysis of EST and genomic sequences to characterize biases in the regions encompassing 600 nucleotides around the cleavage site. The computer program they developed is called *Erpin* which uses 2-gram position-specific nucleotide acid patterns to analyse 300 bases upstream and downstream region of a candidate PAS. Being trained by 2327 terminal sequences, Erpin was reported to achieve a prediction specificity of 69% to 85% for a sensitivity of 56% on several sets of validation data. The program can be found at `http://tagc.univ-mrs.fr/pub/erpin/`.

In this paper, we present a machine learning methodology to characterize the features in the regions encompassing 200 nucleotides around the PAS. Since we only consider the features of sequence around putative PAS, our program can detect all NNUANA-types poly(A) signals. Our method has three steps of feature manipulation: feature space generation from the original sequence data using k-gram nucleotide acid patterns or amino acid patterns, feature selection via an entropy-based algorithm, and feature integration with a classification algorithm — support vector machines — to build a model that can correctly recognize true PASes. We train and test our model using public data as well as our own extracted sequences. Due to the different features of DNA and mRNA sequences, we build different classification models to fit them individually. When we apply our models to the data sets that have been tested on Erpin, we obtain specificity of 73% to 93% at the same sensitivity (56%). In most of cases, our models outperform other published programs on the same data sets. Besides, the significant features captured by our method are highly consistent with those reported motifs.

2 Data

In this study, a large number of sequences are used to train and test our classification model. They are from two sources.

(1) Training and testing sequences used by program Erpin [8]. The training set contains 2327 terminal sequences including 1632 "unique" and 695 "strong" polyA sites. The testing set consists of 982 positive sequences containing annotated PASes from EMBL and four sets of same sized negative sequences: 982 CDS sequences, 982 intronic sequences of the first intron,

982 randomized UTR sequences of same 1^{st} order Markov model as human 3' UTRs, and 982 randomized UTR sequences of same mono nucleotide composition as human 3' UTRs. The 2327 training sequences can be downloaded from `http://tagc.univ-mrs.fr/pub/erpin/` and have been trimmed in accordance to our window segments described in section 3, i.e. every sequence contains 206 bases, having a PAS in the center. We obtained testing data sets from Dr Gautheret via personal communication.

(2) Human RefSeq mRNA data set: we obtained 312 human mRNA sequences from RefSeq [15] release 1. Each of these sequences contains a "polyA-signal" feature tag carrying an "evidence=experimental" label. We use these sequences to build model for PAS prediction in mRNA sequences. Besides, we also extracted a set of human mRNA sequences from RefSeq containing a "polyA-site" feature tag carrying an "evidence=experimental" label. In this set, we removed the sequences that have been included in the training set used in building our model. We use these sequences for testing purpose and assume that the annotated PAS positions are correct. Our negative data set was generated by scanning for the occurrences of AATAAA at coding region and those AATAAA sites near the end of sequence were excluded purposely.

3 Method

We apply machine learning methodology to this PAS classification and prediction problem by analysing features in upstream and downstream sequence elements around the PAS. Our method includes three steps: feature space generation, feature selection and feature integration.

3.1 Feature Generation

We generate a feature space using k-gram ($k = 1, 2, 3, ...$) *nucleotide acid patterns*. A k-gram is simply a pattern of k consecutive letters, which can be nucleotide symbols or amino acid symbols [21, 10]. At first, we use each k-gram nucleotide acid pattern as a new feature. For example, AT is a 2-gram pattern and ATC is a 3-gram pattern. In order to separate upstream and downstream sequence elements, same pattern but appears in the different side of a candidate PAS is treated as two different features. Thus, there are 2×4^k possible combinations of k-gram nucleotide acid patterns for each k. Our aim is to characterize some important k-gram patterns (motifs) around a PAS so that these motifs can be used to distinguish true PASes from false PASes.

The *frequency* of the k-gram nucleotide acid patterns are used as the values of the features. For examples, (1) UP-T (DOWN-T) counts the number of times the nucleotide acid letter T appears in the upstream (downstream) part of a candidate PAS. (2) UP-TG (DOWN-TG) counts the number of times the two nucleotide acid letters TG appear as a substring in the upstream (downstream) part of a candidate PAS. In this paper, we present our results based on 1-gram, 2-gram and 3-gram patterns. Thus, there are 168 ($= (4 + 4^2 + 4^3) \times 2$) possible nucleotide acid patterns, i.e. features. Our patterns are non-position-specific patterns since their positions in the sequence are not considered when their frequencies are counted.

In the framework of the new feature space, the initial nucleotide sequences need to be transformed. The transformation is as follows. Given a DNA or mRNA nucleotide sequence containing candidate PAS(es), a window is set for each candidate PAS with the candidate PAS in the center and 100 bases upstream and 100 bases downstream (excluding the candidate PAS hexamer) aside. If a candidate PAS does not have enough upstream or downstream 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. Thus, all the window segments have same size, i.e. containing 206 nucleotides. Next, nucleotide acid window segments are further converted into frequency sequence data under the description of our features. Later, the classification model will be applied to the frequency sequence data. Note that

there are *two classes* of data, true PASes (the corresponding window segments have true PASes in the center) and false PASes (the corresponding window segments have false PASes in the center), from machine learning point of view.

3.2 Feature Selection

In order to find explicit features (motifs) that can distinguish true PASes from candidates, the second step of our method is to conduct feature selection. There are various ways [9] to conduct feature selection, for examples, by t-statistics, by signal-to-noise statistics or by entropy measure. Here, we introduce a simple and efficient entropy-based algorithm to select important features. The basic idea of this algorithm originates from [6] in which a discretization method was addressed. According to the method, some numeric features can not be discretized since their values are randomly distributed between the two-class data. Reasonably, these kind of features should be excluded from our classification and prediction task. 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 data (such as true PAS), then this partitioning by the cut points of this feature has an entropy value of zero. In contrast to this ideal case, no proper cut points can be found for randomly distributed features and their entropy value is 1 in the two-class 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 *entropy* of a range S with respect to feature f and a collection of classes $\mathcal{U}$ is defined as

$$Ent(f,\mathcal{U},S) \equiv -\sum_{\mathcal{C}\in\mathcal{U}} P(f,\mathcal{C},S)log_2(P(f,\mathcal{C},S))$$

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 refer to T as a *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 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. A commonly used stopping criteria is the so-called minimal description length principle given in [6].

In this study, we pick up all the features whose entropy value is less than 1, i.e. discard all the features without cut points.

3.3 Feature Integration

To achieve the ultimate goal of predicting true PASes, our third step is to integrate the selected features by a classification algorithm. In this paper, we consider support vector machines (SVMs) as our classifier since it is known to have good classification performance in the biological domain, such as gene expression profile analysis [3, 7, 19] and translation initiation site prediction in DNA sequences [10, 23].

SVMs is a kind of blend of linear modeling and instance-based learning [18]. It originates from research in statistical learning theory [17]. An SVM selects a small number of critical boundary samples (called support vectors) from each class of training data and builds a linear discriminant function that separates them as widely as possible. In the case that no linear separation is possible,

the training data will be mapped into a higher-dimensional space $\mathcal{H}$ and an optimal hyperplane (also called maximum margin hyperplane) will be constructed there. The mapping is performed by a kernel function $K(\cdot,\cdot)$ which defines an inner product in $\mathcal{H}$. The decision function given by an SVM is given like:

$$f(x) = \sum_i \alpha_i^0 y_i K(x_i, x) + b$$

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, b and α_i^0 are parameters to be determined. The training of a SVM is a quadratic programming and here, we omit the detailed description about this. Please refer to the tutorial [4] for a better understanding of SVMs.

There are several ways to train support vector machines. One of the fastest algorithms was developed by Platt [14], which solves the above quadratic programming problem by sequential minimal optimization (SMO) algorithm. 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 [24]. The kernel is a polynomial function and the transformation of the output of SVMs into probabilities is conducted by a standard sigmoid function. We adopt the default setting of Weka's implementation where linear kernel functions are used.

4 Experiments and Results

To test the performance of our method, we first select features and build classification model (i.e. training SVMs) on some training data, and then validate the well-trained model on testing data. To evaluate the performance of model, we adopt standard performance measures defined as follows. *Sensitivity* measures the proportion of true PASes that are correctly recognized as PASes. *Specificity* measures the proportion of the claimed true PASes that are indeed PASes. Besides, we also plot *ROC* (Receiver Operating Characteristic) curve for each validation so that the tradeoff between true positive rate (i.e. sensitivity) and false positive rate can be illustrated clearly.

On the other hand, the training accuracy is indicated by 10-fold cross-validation results. In 10-fold cross-validation, training data is divided randomly into 10 disjoint subsets of approximately equal size, in each of which the class is represented in approximately the same proportions as in the full training data set. Then the above process of training (including feature selection and model construction) and validating will be repeated 10 times. In each iteration, (1) one of the subsets is held out in turn, (2) feature selection and SVMs training are conducted on the remaining 9 subsets, (3) the model is evaluated on the holdout set. After all subsets being tested, an overall performance is yielded.

4.1 Preliminary Results

In the first experiment, we use the 2327 sequences introduced in [8] (see data source (1) in Section 2) as our true PAS training data. To obtain negative sequences, same sized false PAS data is randomly selected from our own extracted negative data set (see data source (2) in Section 2). Using entropy-based feature selection algorithm and SVM classifier, the sensitivity and specificity of 10-fold cross-validation on training data are 89.3% and 80.5%, respectively. In order to compare with other programs, we test our model on the same validation sets whose testing results on programs Erpin and Polyadq were reported in [8]. As described in Section 2 data source (1) , these validation sets include true PASes sequences came from 982 annotated UTRs and four same sized control sets known not to contain PASes: coding sequences (CDS), introns and randomized UTRs (simply shuffled UTRs and 1^{st} order Markov model UTRs). For a direct comparison, we also adjust the prediction sensitivity on the 982 true PASes at around 56.0% so that evaluation can be made on prediction specificities using those four control sets.

Table 1 shows the validation results on true PASes and Table 2 illustrates the results on four control sets. Figure 2 is the ROC curve for this series of tests. All the numbers regarding to the performance of programs Erpin and Polyadq in Table 1 and Table 2 are copied or derived from [8]. The results in Table 2 demonstrate that our model can give better performance than Erpin and Polyadq did on false PASes prediction of CDS, intron and simple shuffling sequences, and almost same prediction accuracy on sequences with 1^{st} order Markov randomization.

Table 1: Validation results by different programs on a set of 982 annotated UTR sequences from the EMBL database [8]. TP is the number of true positives. FN is the number of false negatives. SN is sensitivity, and $SN = TP/(TP + FN)$.

Program	TP	FN	SN
Erpin	549	433	55.9%
Polyadq	547	435	55.7%
Ours	553	429	56.3%

Table 2: Validation results by different programs on different sequences not containing PASes: coding sequences (CDS), introns, and two types of randomized UTR sequences (simple shuffling and 1^{st} order Markov simulation) [8]. TN is the number of true negatives. FP is the number of false positives. TNR is the true negative rate that measures the proportion of false PASes that are correctly recognized as false PASes, and $TNR = TN/(TN + FP)$. FPR is the false positive rate that measures the proportion of false PASes that are misclassified as true PASes, and $FPR = FP/(TN + FP) = 1 - TNR$. SP is specificity, and $SP = TP/(TP + FP)$. CC is correlation coefficient, and $CC = \frac{(TP*TN-FP*FN)}{\sqrt{(TP+FP)*(TP+FN)*(TN+FP)*(TN+FN)}}$. Calculations of SP and CC use TP and FN from Table 1.

Data set	Program	TN	FP	TNR	FPR	SP	CC
CDS	Erpin	880	102	89.6%	10.4%	84.3%	0.483
	Polyadq	862	120	87.8%	12.2%	82.0%	0.459
	Ours	887	95	90.3%	9.7%	85.4%	0.497
Introns	Erpin	741	241	75.5%	24.5%	69.5%	0.320
	Polyadq	718	264	73.1%	26.9%	67.5%	0.293
	Ours	775	207	78.9%	21.1%	72.8%	0.363
Simple shuffling	Erpin	888	94	90.4%	9.6%	85.4%	0.494
	Polyadq	826	156	84.1%	15.9%	77.8%	0.415
	Ours	942	40	95.9%	4.1%	93.3%	0.570
Markov 1^{st} order	Erpin	772	210	78.6%	21.4%	72.3%	0.354
	Polyadq	733	249	74.6%	25.4%	68.7%	0.309
	Ours	765	217	77.9%	22.1%	71.9%	0.351

In this experiment, there are 113 features having their entropy values less than 1. They are selected to integrate with SVMs to form the classification and prediction model. Table 3 lists the top 10 of these features ranking by their entropy values (the less the entropy value is, the more important the feature is). Some of these top features can be interpreted by those reported motifs, for example, it is clearly to visualize both USE and DSE are well-characterized by G/U rich segments since UP-TGT, UP-T, DOWN-TGT, DOWN-T, UP-TG and UP-TT are among top features.

When we apply our model to 312 true PASes that was extracted from mRNA sequences by ourselves (see data source (2) in Section 2), the results obtained is not good — only around 20% of them can be predicted correctly. Besides, the program Erpin performs even worse on these PASes — with

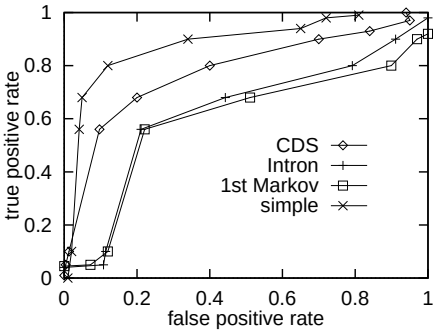

Figure 2: ROC curve of our model on the validation sets described in [8] (please see data source (1) of Section 2).

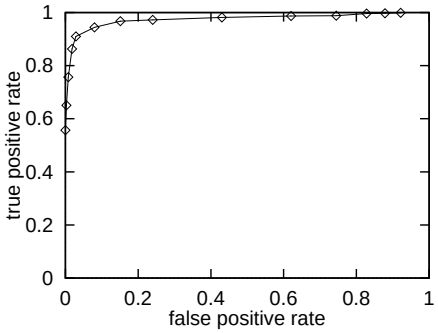

Figure 3: ROC curve of our model on PAS prediction in mRNA sequences.

Table 3: The top 10 features selected by entropy-based feature selection method for PAS classification and prediction in human DNA sequences.

Rank	1	2	3	4	5	6	7	8	9	10
Feature	UP -TGT	DOWN -A	UP -T	UP -AG	DOWN -TGT	DOWN -T	UP -TG	UP -TT	DOWN -AA	UP -A

prediction accuracy at only 13%. It may indicate that the good features used in the model for PAS prediction in DNA sequences are not efficient for mRNA. Therefore, we decide to build another model for mRNA sequences without poly(A) tails. This model is also expected to provide a new way for predicting the mRNA cleavage site/poly(A) addition site.

Since the new model is aimed to predict PASes from mRNA sequences, we only consider the upstream elements around a candidate PAS. Therefore, there are only 84 features (instead of 168 features). To train the model, we use 312 experimental verified true PASes and same number of false PASes that randomly selected from our prepared negative data set. The validation set comprises 767 annotated PASes and same number of false PASes also from our negative data set but different from those used as training (data source (2) in Section 2). This time, we achieve reasonably good results. Sensitivity and specificity for 10-fold cross-validation on training data are 79.5% and 81.8%, respectively. Validation result is 79.0% sensitivity at 83.6% specificity. Besides, we observe that the top ranked features selected via entropy-based feature selection method are different from those listed in Table 3 (features not shown).

Since every 3 nucleotides code for an amino acid when DNA sequences translate to mRNA sequences, it is legitimate to investigate if an alternative approach that generating features based on amino acids can produce more effective PASes prediction for mRNA sequence data. In fact, this idea has been implemented to recognize translation initiation sites in human DNA sequences [10]. Next, let's explore more about this on PAS prediction.

4.2 A Refinement for PAS Prediction in mRNA Sequences

After getting a 206 bases nucleotide acid window segment for each candidate PAS, we code every triplet nucleotides at upstream 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.

Instead of nucleotide acid patterns, we generate the new feature space using k-gram ($k = 1, 2, 3, ...$) *amino acid patterns*. For example, AR is a 2-gram pattern constituted by an alanine followed by an arginine. Apart from these k-gram amino acid patterns, we also present existing knowledge via an

additional feature: denoting number of T residue in upstream as "UP-T-Number". Since there are 20 standard amino acids plus 1 stop symbol, there are 21^k possible combinations of k-gram patterns for each k. If we choose k as 1 and 2, then there are total 463 $(= 21 + 21^2 + 1)$ features in the new feature space.

Similarly, we use *frequency* of the k-gram amino acid patterns as the values of the features, and the amino acid sequences are then converted into frequency sequence data under the description of our new features. Figure 4 presents a diagram for the mRNA data transformation with respect to our new feature space.

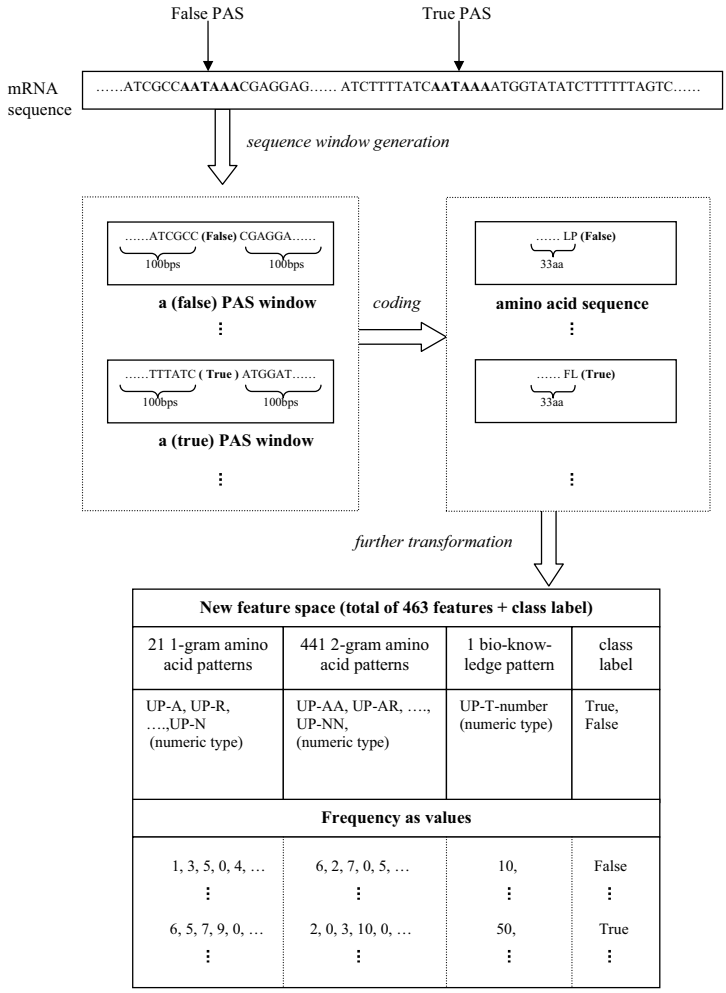

Figure 4: A diagram for mRNA data transformation aiming for the description of the new feature space. In practice, we drop the left-most nucleotide so that 99 bases are used in the coding.

In the new feature space, we conduct feature selection and train SVMs on 312 true PASes and same number of false PASes. The 10-fold cross-validation results on training data are 81.7% sensitivity with 94.1% specificity. When apply the trained model to our validation set containing 767 true PASes and 767 false PASes, we achieve 94.4% sensitivity with 92.2% specificity (correlation coefficient is as high as 0.865). Figure 3 is the ROC curve of this validation. In this experiment, there are only 13 selected features and UP-T-Number is the top 1 feature. This indicates that upstream sequence of PAS in mRNA sequence may also contain T-rich segments.

5 Conclusion

We have described a machine learning methodology for recognition of polyadenylation signals (PASes) in human DNA and mRNA sequences. The method comprises three steps: (1) generating candidate features from the original sequence data using k-gram nucleotide acid patterns or amino acid patterns;

(2) selecting relevant features using an entropy-based algorithm; and (3) integrating the selected features by SVMs to build a system to correctly recognize true PASes.

We train our classification models using some public data sets, including 2327 true PASes in DNA sequences that were used to train program Erpin [8]. When applying our well-trained models to the same validation data sets that have been tested on Erpin and Polyadq, in most of cases, our models outperform other programs — with specificity of 73% to 93% at the sensitivity of 56%. Our experimental results show that PASes in DNA and mRNA sequences may have different characteristics in their upstream and downstream sequence elements so that different classification models should be considered to fit them individually.

To predict PASes in mRNA sequences, we code each nucleotide acid sequence to amino acid sequence and use k-gram amino acid patterns as new features. To train such a model, we extract experimental verified PASes from mRNA sequences as well as some negative data (RefSeq release 1). The model built for PAS prediction in mRNA sequences also achieves very good validation performance, with 94.4% sensitivity at 92.2% specificity for predicting PASes in a set of sequences containing experimental verified poly(A) sites.

To obtain explicit important motifs around true PASes, we use entropy-based feature selection method to filter out those unimportant features. A reduced feature dimensionality will not only greatly shorten the running time of classification program, but also lead to a more accurate prediction. The performances using all generated features are not as good as what we present in this paper (detailed results not shown). The significant features output are highly consistent to those reported motifs in literature.

Currently, we are considering to include patterns containing "dont care" symbols into our feature space so that more general motifs might be found. Meanwhile, some other classification algorithms, such as ensemble decision trees, are being tested to output comprehensive and interesting rules. To further test the feasibility and robustness of our method, we will test our models on EST and genomic sequences.

Acknowledgments

We wish to thank Dr. Daniel Gautheret and Dr. Matthieu Legendre for providing their data sets.

References

[1] Aissouni, Y., Perez, C., Calmels, B., and Benech, P.D., The cleavage/polyadenylation activity triggered by a U-rich motif sequence is differently required depending on the poly(A) site location at either the first or last 3'-terminal exon of the 2'-5' oligo(A) synthetase gene, *J. Biol. Chem.*, 277:35808–35814, 2002.

[2] Brackenridge, S. and Proudfoot, N.J., Recruitment of a basal polyadenylation factor by the upstream sequence element of the human lamin B2 polyadenylation signal, *Mol. Cell. Biol.*, 20:2660–2669, 2000.

[3] Brown, M.P., Grundy, W.N., Lin, D., Cristianini, N., Sugnet, C.W., Furey, T.S., Ares, M.Jr., and Haussler D., Knowledge-based analysis of microarray gene expression data using support vector machines, *Proceedings of the National Academy of Science*, 97(1):262–267, 2000.

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

[5] Colgan, D.F. and Manley J.L., Mechanism and regulation of mRNA polyadenylation, *Gens Development*, 11:2755–2766, 1997.

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

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

[8] Legendre, M. and Gautheret, D., Sequence determinants in human polyadenylation site selection, *BMC Genomics*, 4(1):7, 2003.

[9] Liu, H., Li, J., and Wong, L., A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns, *Genome Informatics*, 13:51–60, 2002.

[10] Liu, H. and Wong, L., Data mining tools for biological sequences, *Journal of Bioinformatics and Computational Biology*, 1(1):139–168, 2003.

[11] McDevitt, M.A., Hart, R.P., Wong, W.W., and Nevins, J.R., Sequence capable of restoring poly(A) site function define two distinct downstream element, *EMBO J.*, 1(5):2907–2931, 1986.

[12] Minvielle-Sebastia, L. and Keller, W., mRNA polyadenylation and its coupling to other RNA processing reactions and to transcription, *Current Opinion in Cell Biology*, 11:352–357, 1999.

[13] Moreira, A., Takagaki, Y., Brackenridge, S., Wollerton, M., Manley, J.L., and Proudfoot, N.J., The upstream sequence element of the C2 complement poly(A) signal activates mRNA 3' end formation by two distinct mechanisms, *Genes Dev.*, 12:2522–2534, 1998.

[14] Platt, J., Fast training of support vector machines using sequential minimal optimization, In *Advances in Kernel Methods - Support Vector Learning*, Edited by Schölkopf, B., Burges, C., and Smola, A., MIT Press, 1998.

[15] Pruitt, K.D. *et al.*, Introducing RefSeq and LocusLink: curated human genome resources at the NCBI, *Trends. Genet.*, 1(16):44–47, 2000.

[16] Tabaska, J.E. and Zhang, M.Q., Detection of polyadenylation signals in human DNA sequences, *Gene*, 231:77–86, 1999.

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

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

[19] Yeoh, E.-J., Ross, M.E., Shurtleff, S.A., Williams, W.K., Patel, D., Mahfouz, R., Behm, F.G., Raimondi, S.C., Relling, M.V., Patel, A., Cheng, C., Campana, D., Wilkins, D., Zhou, X., Li, J., Liu, H., Pui, C.-H., Evans, W.E., Naeve, C., Wong, L., and Downing, J.R., Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling, *Cancer Cell*, 1:133–143, 2002.

[20] Zarudnaya, M.I., Kolomiets, I.M., Potyahaylo, A.L., and Hovorun, D.M., Downstream elements of mammalian pre-mRNA polyadenylation signals: primary, secondary and higher-order structures *Nucl. Acids Res.*, 1(31):1375–1386, 2003.

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

[22] Zhao, J., Hyman, L., and Moore, C., Formation of mRNA 3' ends in eukaryotes: mechanism, regulation, and interrelationships with othe steps in mRNA synthesis, *Microbiology and Molecular Biology Reviews*, 63(2):405–445, 1999.

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

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