Marginalized Kernels for RNA Sequence Data Analysis

Taishin Kin **Koji Tsuda** **Kiyoshi Asai**
taishin@cbrc.jp tsuda@cbrc.jp asai@cbrc.jp

Computational Biology Research Center, National Institute of Advanced Industrial Science and Technology, 2-41-6 Aomi, Koto-ku, Tokyo 135-0064, Japan

Abstract

We present novel kernels that measure similarity of two RNA sequences, taking account of their secondary structures. Two types of kernels are presented. One is for RNA sequences with known secondary structures, the other for those without known secondary structures. The latter employs stochastic context-free grammar (SCFG) for estimating the secondary structure. We call the latter the *marginalized count kernel* (MCK). We show computational experiments for MCK using 74 sets of human tRNA sequence data: (i) kernel principal component analysis (PCA) for visualizing tRNA similarities, (ii) supervised classification with support vector machines (SVMs). Both types of experiment show promising results for MCKs.

Keywords: kernel, feature vector, count kernel, marginalized count kernel, SCFG, SVM, kernel PCA

1 Introduction

Due to the recent success of the support vector machines (SVMs) in many practical problems, emerging kernel methods are appealing in the area of bioinformatics. For instance, SVMs are well known for their performance superiority in many real-world problems such as text-categorization, pattern recognition and, especially now, biological sequence data analysis. Examples for biological sequence data analysis are protein family clustering [4], promoter classification [10] and homology detection [3]. However, to our knowledge, kernel methods have not been applied to RNA sequence data analysis. The primary reason is that it is hard to define a kernel function to reflect the secondary structures of RNAs. For defining the similarity (or kernel) between two RNAs, simple sequence comparison (e.g BLAST) is never enough because the sequences can form stems, hairpins, bulges etc. In addition, string kernels [14] including the spectrum kernel [7] do not define RNA similarities because they do not take into account the secondary structures.

In this paper, we propose a new method for designing a kernel for RNAs. First we will consider a case in which the secondary structure of two RNAs is known and represented using context-free grammar (CFG) where a state (or a non-terminal) is associated with one or two base(s) in an RNA sequence. A feature vector is constructed by counting the base-state combinations and the kernel is defined as the dot product between two vectors. We call this the "count kernel for RNAs". This concept can be generalized to take into account the consecutive two base-state combinations. We call this kernel the "2nd order count kernel for RNAs". However, it is often the case that the RNA secondary structure is not known, but estimated with some probabilistic model such as stochastic context-free grammar (SCFG). In such cases, we use the expectation of the count kernel with respect to the secondary structure. We call this the "marginalized count kernel for RNAs". Similarly, a second-order version can be obtained. Note that these kernels are generalized versions of the kernels proposed for HMMs by Tsuda *et al.* [13]. We performed computational experiments using human tRNA sequence data, which are a visualization of sequence similarities using kernel PCA and a supervised classification using SVMs. For the latter, we compared the performance of the classifications with and without MCKs.

2 Kernels for RNAs

2.1 Representing RNA Secondary Structures

We use CFG to represent RNA secondary structures because it is capable of representing common RNA structural patterns e.g. stems, hairpins, and bulges. Consider a tiny RNA sequence as shown in Figure 1(a) for example. The secondary structure of the RNA can be written as generative rules of the CFG:

$$S \to R_1, R_1 \to P_1 a, P_1 \to gP_2 c, P_2 \to gP_3 c, P_3 \to gL_1 c, L_1 \to cL_2, L_2 \to aL_3, L_3 \to uE,$$

where P represents the state that emits a canonical base pair. R and L correspond to the states that emit a single base to right and left, respectively. S and E are special states that do not associate with any symbols but represent start and end, respectively. The generative rules are interpreted as R_1 emits "a" to the right and makes transition to P_1, then P_1 emits "g" to the left and "c" to the right simultaneously, followed by a move to P_2, and so on.

The generative rules can be represented as a state-path in a matrix. Figure 1(c) shows a matrix representation of the generative rules. We call it the *CFG matrix* that represents how CFG states are associated with the sequence. For example, P at $(1, 9)$ corresponds to a pair of 1st base "g" and 9th base "c", L at $(4, 6)$ corresponds to 4th base "c" and R at $(1, 10)$ corresponds to 10th base "a".

2.2 Kernel Design for Known RNA Structures

1st order count kernel We represent an RNA secondary structure as a vector by counting the occurrence of each base-state combination. We devise a base-state vector for the sample RNA sequence presented in Figure 1(a) as: $(a, R) \times 1, (gc, P) \times 3, (c, L) \times 1, (a, L) \times 1, (u, L) \times 1$, while the other combinations such as $(c, R), (au, P)$, etc. simply count as zero. These numbers build up a 12 $(= 4 \times 3)$ dimensional real-value vector (for the dimension, 4 bases for each of L and R, and 4 canonical base pairs for P). We call this vector the *count feature vector*.

Let us represent the CFG matrix as W and an RNA sequence as $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ where x_i corresponds to a base in the sequence. The count feature vectors are defined as follows:

$$c_P^{ab}(\boldsymbol{z}) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \delta\left[W(i,j) = P, x_i = a, x_j = b\right] \tag{1}$$

$$c_L^{a}(\boldsymbol{z}) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=i}^{n} \delta\left[W(i,j) = L, x_i = a\right] \tag{2}$$

$$c_R^{b}(\boldsymbol{z}) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=i}^{n} \delta\left[W(i,j) = R, x_j = b\right], \tag{3}$$

where $\boldsymbol{z} \equiv \{\boldsymbol{x}, W\}$ is a sequence combined with a CFG matrix, $1/n$ is a normalization factor in regard to the length of sequence n, and $\delta\left[x = y\right] = 1$ if $x = y$ else 0.

We define the *1st order count kernel* which is a dot product of the two count feature vectors. It represents the similarity between two sequences:

$$K_z(\boldsymbol{z}, \boldsymbol{z}') = \sum_{V \in \{P,L,R\}} C_V(\boldsymbol{z}, \boldsymbol{z}'),$$

where

$$C_V(\boldsymbol{z}, \boldsymbol{z}') = \begin{cases} V = P: & \sum_{ab \in \boldsymbol{\Omega}} c_P^{ab}(\boldsymbol{z}) c_P^{ab}(\boldsymbol{z}'), \quad \boldsymbol{\Omega} = \{AU, UA, CG, GC\} \\ V = L: & \sum_{a \in \boldsymbol{B}} c_L^{a}(\boldsymbol{z}) c_L^{a}(\boldsymbol{z}'), \quad \boldsymbol{B} = \{A, U, C, G\} \\ V = R: & \sum_{a \in \boldsymbol{B}} c_R^{a}(\boldsymbol{z}) c_R^{a}(\boldsymbol{z}') \end{cases}$$

Table 1: Values of $\triangle^V_{\ell|r}$ which represent if the state V emits a symbol to the left or the right. "1" indicates the occurrence of emission while "0" indicates no emission.

V	P	L	R
$\triangle^V_\ell$	1	1	0
$\triangle^V_r$	1	0	1

2nd order count kernel We also define the 2nd order count kernel where we count the combination of two consecutive states and the associated bases. The 2nd order count kernel corresponds to a bi-gram model of an RNA sequence with the secondary structure information. The bi-gram model is known to be able to take account of implicit but essential information for identifying biological sequences [1, 5].

We define the 2nd order count feature vector which has 144 ($3^2 \times 4^2$) dimensions. The scheme of the counting is shown in Figure 2. There are nine types of the 2nd order count feature vectors with respect to base-state combinations. We denote these vectors as follows (only three of them are listed here):

$$
\begin{aligned}
c^{abcd}_{PP}(\boldsymbol{z}) &= \frac{2}{n}\sum_{i=1}^{n-1-\triangle^P_\ell}\sum_{j=i+1+\triangle^P_r}^{n} d^{abcd}_{PP}(i,j) && (4)\\
d^{abcd}_{PP}(i,j) &\equiv \delta\left[W(i,j)=P, W(i+\triangle^P_\ell, j-\triangle^P_r)=P, x_i=a, x_j=b, x_{i+\triangle^P_\ell}=c, x_{j-\triangle^P_r}=d\right]\\
c^{abc}_{PL}(\boldsymbol{z}) &= \frac{2}{n}\sum_{i=1}^{n-1-\triangle^P_\ell}\sum_{j=i+1+\triangle^P_r}^{n} d^{abc}_{PL}(i,j) && (5)\\
d^{abc}_{PL}(i,j) &\equiv \delta\left[W(i,j)=P, W(i+\triangle^P_\ell, j-\triangle^P_r)=P, x_i=a, x_j=b, x_{i+\triangle^P_\ell}=c\right]\\
c^{abd}_{PR}(\boldsymbol{z}) &= \frac{2}{n}\sum_{i=1}^{n-1-\triangle^P_\ell}\sum_{j=i+1+\triangle^P_r}^{n} d^{abd}_{PR}(i,j) && (6)\\
d^{abd}_{PR}(i,j) &\equiv \delta\left[W(i,j)=P, W(i+\triangle^P_\ell, j-\triangle^P_r)=P, x_i=a, x_j=b, x_{j-\triangle^P_r}=d\right],
\end{aligned}
$$

where $\triangle^V_\ell$ and $\triangle^V_r$ are binary flags indicating whether V emits a symbol to the left ($\triangle^V_\ell$) and the right ($\triangle^V_r$). See Table 1 for actual values. We define the 2nd order count kernel using the vectors as follows:

$$
\begin{aligned}
K(\boldsymbol{z},\boldsymbol{z}') &= \sum_{VY\in\boldsymbol{\Psi}} C_{VY}(\boldsymbol{z},\boldsymbol{z}'), && (7)\\
\boldsymbol{\Psi} &= \{PP, PL, PR, LP, LL, LR, RP, RL, RR\}
\end{aligned}
$$

where

$$
C_{VY}(\boldsymbol{z},\boldsymbol{z}') = \begin{cases} VY=PP: & \sum_{abcd\in\boldsymbol{\Omega}\times\boldsymbol{\Omega}} c^{abcd}_{PP}(\boldsymbol{z}) c^{abcd}_{PP}(\boldsymbol{z}') \\ VY=PL: & \sum_{abc\in\boldsymbol{\Omega}\times\boldsymbol{B}} c^{abc}_{PL}(\boldsymbol{z}) c^{abc}_{PL}(\boldsymbol{z}') \\ VY=PR: & \sum_{abd\in\boldsymbol{\Omega}\times\boldsymbol{B}} c^{abd}_{PR}(\boldsymbol{z}) c^{abd}_{PR}(\boldsymbol{z}') \\ & \vdots \end{cases}
$$

The limitation of these kernels is that computation is only possible if the states, i.e. the secondary structures, are known. However, since several RNA prediction algorithms are available, we can use such algorithms with our kernels.

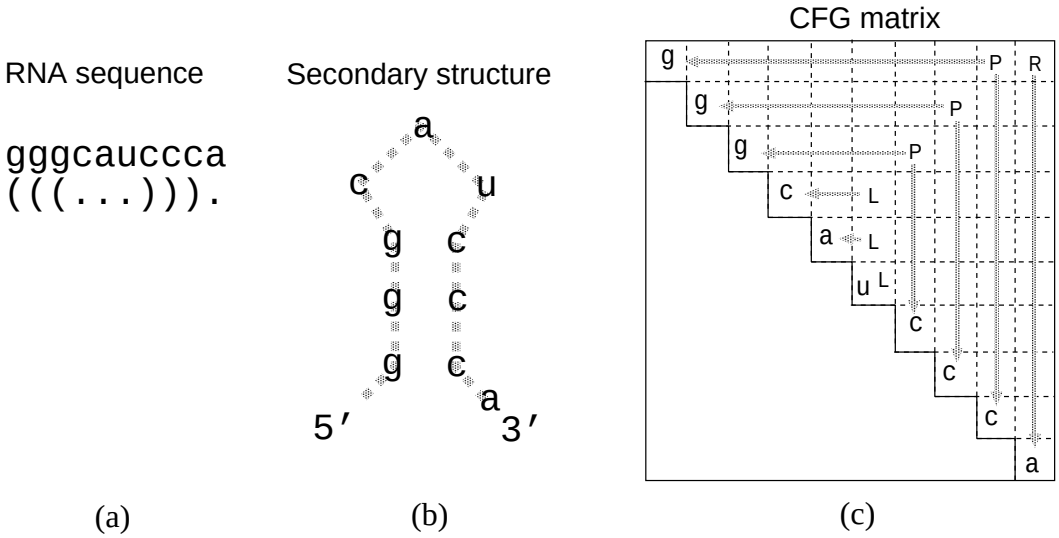

Figure 1: An example of an RNA sequence and a representation of the secondary structure using the *CFG matrix*. (a) a tiny RNA sequence and its secondary structure. The parentheses represent base-pairings. (b) a hairpin-like structure of the RNA. (c) the secondary structure represented in the *CFG matrix*. The arrows show that one or two symbols correspond to a certain state.

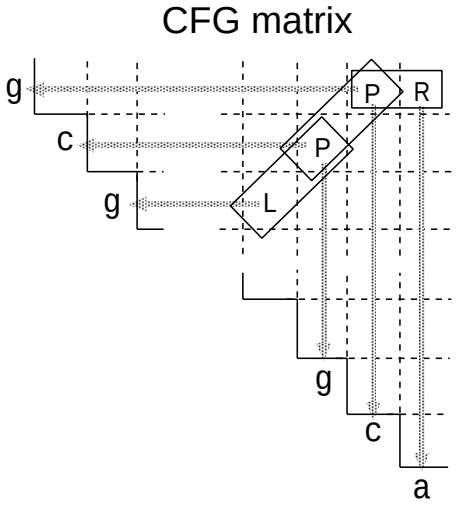

Figure 2: The counting scheme of the 2nd order count feature vector from a CFG matrix. An exemple sate-path is shown as R, P, P and L. Note that each state is associated with one or two bases (arrows). The rectangles surrounding each pair of states indicate how consecutive two states are counted. It reads $[(R, a), (P, gc)]$, $[(P, gc), (P, cg)]$, $[(P, gc), (L, g)]$, and so on. There are 144 possible combinations to be counted. Thus, the 2nd order count feature vector has 144 dimensions.

3 Kernel Design for Unknown RNA Structures

Using algorithms for secondary structure prediction such as Nussinov [9] or Zuker algorithms [17, 15, 16] is an approach to build the CFG matrix without the secondary structure information *a priori*. However, since we do not know the correct structure, we can take account of all probable structures. Instead of having the explicit states in the CFG matrix W, we can estimate probabilities for each state placed at (i, j) by using a stochastic prediction algorithm such as SCFG. We use SCFG for this purpose; the details of SCFG are presented in next section. Here we will coninue to focus on our kernels.

3.1 Marginalized Count Kernel (MCK)

In this case, it is not possible to compute $\delta\left[W(i,j)=V\right]$ as used in the count feature vectors (Eqs. 1 to 3). Instead, we compute a probability for the condition $W(i,j)=V$ with SCFG. Then we replace $\delta\left[W(i,j)=V\right]$ with a probability $0 \leq p(W(i,j)=V) \leq 1$. Therefore, the probabilistic versions of the count feature vectors are written as:

$$g_P^{ab}(\boldsymbol{x}) = \frac{2}{n}\sum_{i=1}^{n}\sum_{j=i+1}^{n} p(W(i,j)=P|\boldsymbol{x})\delta\left[x_i=a, x_j=b\right] \tag{8}$$

$$g_L^{a}(\boldsymbol{x}) = \frac{2}{n}\sum_{i=1}^{n}\sum_{j=i}^{n} p(W(i,j)=L|\boldsymbol{x})\delta\left[x_i=a\right] \tag{9}$$

$$g_R^{a}(\boldsymbol{x}) = \frac{2}{n}\sum_{i=1}^{n}\sum_{j=i}^{n} p(W(i,j)=R|\boldsymbol{x})\delta\left[x_j=a\right]. \tag{10}$$

Let us call this vector the 1st order marginalized count feature vector. The *1st order marginalized count kernel* can be defined as:

$$K(\boldsymbol{x},\boldsymbol{x}') = \sum_{V\in\{P,L,R\}} G_V(\boldsymbol{x},\boldsymbol{x}') \tag{11}$$

where

$$G_V(\boldsymbol{x},\boldsymbol{x}') = \begin{cases} V=P: & \sum_{ab\in\boldsymbol{\Omega}} g_P^{ab}(\boldsymbol{x}) g_P^{ab}(\boldsymbol{x}') \\ V=L: & \sum_{a\in\boldsymbol{B}} g_L^{a}(\boldsymbol{x}) g_L^{a}(\boldsymbol{x}') \\ V=R: & \sum_{a\in\boldsymbol{B}} g_R^{a}(\boldsymbol{x}) g_R^{a}(\boldsymbol{x}') \end{cases} \tag{12}$$

3.2 2nd Order MCK

We use a joint probability instead of the δ functions used in the 2nd order count feature vectors (Eqs. 4 to 6). The joint probability represents a probability to have a state V at $W(i,j)$ succeeded by a state Y at $W(i+\triangle_\ell^V, j-\triangle_r^V)$:

$$\xi_{VY}(i,j) \equiv p(W(i+\triangle_\ell^V, j-\triangle_r^V)=Y, W(i,j)=V|\boldsymbol{x}), \tag{13}$$

which is a parameter commonly computed for the SCFG learning algorithm. The 2nd order marginalized count feature vectors are defined as follows (only three equations out of nine are listed here):

$$g_{PP}^{abcd}(\boldsymbol{x}) = \frac{2}{n}\sum_{i=1}^{n-1-\triangle_\ell^P}\sum_{j=i+1+\triangle_r^P}^{n}\xi_{PP}(i,j)\delta\left[x_i=a, x_j=b, x_{i+\triangle_\ell^P}=c, x_{j-\triangle_r^P}=d\right] \quad (14)$$

$$g_{PL}^{abc}(\boldsymbol{x}) = \frac{2}{n}\sum_{i=1}^{n-1-\triangle_\ell^P}\sum_{j=i+1+\triangle_r^P}^{n}\xi_{PL}(i,j)\delta\left[x_i=a, x_j=b, x_{i+\triangle_\ell^P}=c\right] \quad (15)$$

$$g_{PR}^{abd}(\boldsymbol{x}) = \frac{2}{n}\sum_{i=1}^{n-1-\triangle_\ell^P}\sum_{j=i+1+\triangle_r^P}^{n}\xi_{PR}(i,j)\delta\left[x_i=a, x_j=b, x_{j-\triangle_r^P}=d\right] \quad (16)$$

The other varieties of $g(\boldsymbol{x})$s are derived in the same manner. Hence, the *2nd order MCK* is defined as:

$$K(\boldsymbol{x},\boldsymbol{x}') = \sum_{VY\in\boldsymbol{\Psi}} G_{VY}(\boldsymbol{x},\boldsymbol{x}'), \quad (17)$$

where

$$G_{VY}(\boldsymbol{x},\boldsymbol{x}') = \begin{cases} VY=PP: & \sum_{abcd\in\boldsymbol{\Omega}\times\boldsymbol{\Omega}} g_{PP}^{abcd}(\boldsymbol{x}) g_{PP}^{abcd}(\boldsymbol{x}') \\ VY=PL: & \sum_{abc\in\boldsymbol{\Omega}\times\boldsymbol{B}} g_{PL}^{abc}(\boldsymbol{x}) g_{PL}^{abc}(\boldsymbol{x}') \\ VY=PR: & \sum_{abd\in\boldsymbol{\Omega}\times\boldsymbol{B}} g_{PR}^{abd}(\boldsymbol{x}) g_{PR}^{abd}(\boldsymbol{x}') \\ & \vdots \end{cases}$$

4 SCFG

SCFG can be defined with two sets of parameters. One is a set of transition probabilities and the other a set of emission probabilities. Each state transition has a transition probability. We denote a transition probability from state V to Y as $t_V(Y)$. We write the emission probability for state P as $e_P(a,b)$. Since L and R emit a single base, their emission probability is written as $e_{L|R}(a)$.

SCFG parameters are trained with expectation-maximization learning using a standard inside-outside algorithm [6]. Maximizing the likelihood of each sequence leads to maximizing the expectation to use the state P because the pair-wise emission is more economic than are the single emission states such as L and R to annotate the sequence. Therefore, the expectation-maximization learning corresponds to the Nussinov algorithm which estimates a secondary structure in a way to maximize the occurrence of base-pairs. However, the same learning algorithm behaves like the Zuker algorithm with appropriate parameter settings in regard to the stacking energies.

We replace the CFG matrix with a probability matrix in which each element corresponds to a probability to have a state V at $W(i,j)$. We denote the probability as $p(W(i,j)=V|\boldsymbol{x})$, which can be computed using the inside parameter $\alpha_V(i,j)$ and the outside parameter $\beta_V(i,j)$:

$$p(W(i,j)=V|\boldsymbol{x}) = \frac{1}{p(\boldsymbol{x}|\boldsymbol{\theta})}\alpha_V(i,j)\beta_V(i,j). \quad (18)$$

In addition, the probability used for the 2nd order MCK (Equation 13) can be represented as follows:

$$\xi_{VY}(i,j) = \frac{1}{p(\boldsymbol{x}|\boldsymbol{\theta})}\beta_V(i,j)t_V(Y)\alpha_Y(i+\triangle_\ell^V, j-\triangle_r^V). \quad (19)$$

For full details of SCFG, please refer to an appropriate textbook (e.g. [2]) for a complete description.

5 Computational Experiments

We present some computational experiments using human tRNA sequence data to evaluate the performance of MCKs.

5.1 Data Set

We use 74 human tRNAs: 25 Ala-AGC, 26 Asn-GTT and 23 Cys-GCA (i.e. three sequence classes), retrieved from GtRDB [8] (`http://rna.wustl.edu/GtRDB/`). These amino-acid/codon combinations are chosen because they count major populations in human tRNAs of GtRDB. The sequence data are processed not to include any identical sequences so that all sequences become unique. While most of the sequence lengths are 73 bp, they range from 71 to 76 bp.

5.2 SCFG

We use a simple SCFG for modeling generic RNA features; this involves three non symbol emitting states in addition to the state described in Section 2. They are a bifurcation state B and its child states S_L and S_R. The bifurcation state is needed for representing multi-loop structures. S_L and S_R are unrelated to the secondary structures; they are included for implementation convenience of our SCFG software. Thus, 8 states are used for SCFG. Non-canonical base pairs such as G-U and U-G are not considered because the occurrence of such base pairs is rare and makes little contributions to model improvements while at the same time making the model more complex.

SCFG is represented as a 8×8 transition matrix and three 4×4 emission matrices which are bound to three emission states i.e. P, L and R. The initial transition/emission probabilities for these states are uniform e.g. transition probabilities from P to the other states are all equally set. No constraints in regard to the stacking energy are applied. No information of secondary structures is given *a priori*.

5.3 Kernel Principal Component Analysis

Method Kernel PCA is performed so that we can visualize similarities of tRNA sequences in terms of the 1st and 2nd order MCKs. In order to suppress the performance contribution from SCFG, SCFG is trained using the *whole* 74 sequences as the learning data set. Therefore, SCFG is expected to learn a common feature shared by the sequences, i.e. a common secondary structure of tRNAs. Since the 1st and 2nd order MCKs are computed over the SCFG, a similarity is expected to be computed within a context of a common tRNA secondary structure. Thus, the major difference is thought to come from essentially different compositions per each sequence or each class such as anti-codon stems. The 1st/2nd order MCK is generated as a 74×74 square matrix where each element represents a pair-wise similarity between each pair of tRNAs. These kernel matrices are used for the computation of kernel PCA.

Results Figure 3 shows the results of kernel PCA based on the 1st order MCK (left) and the 2nd order MCK (right). The two figures clearly demonstrate performance differences of the 1st and 2nd order MCKs. The 1st order MCK discriminates almost all of the Asn-GTT tRNAs from the other tRNAs. The 2nd order MCK correctly identified Cys-GCA tRNAs from Ala-AGC tRNAs. Another notable observation is featured in the black box located at left most-side of both figures and the white box located beneath the "+" cluster. The black and white boxes correspond to the following sequences: "chrNA random.trna1-AsnGTT Sc: 25.89 Pseudo" and "chr8.trna11-AlaAGC Sc: 22.95 Pseudo", respectively. Since the GtRDB entries are prediction-results generated with a fast and accurate program named tRNA-scan-SE [8], fairly low prediction scores of the outliers (avg. score: Asn-GTT 59.8, Ala-AGC 58.6) and their remarks as "Pseudo" (3 "Pseudo"s in Ala-AGC, 4 in Asn-GTT) indicate the possibility of their being false positives, which is known to be rare for

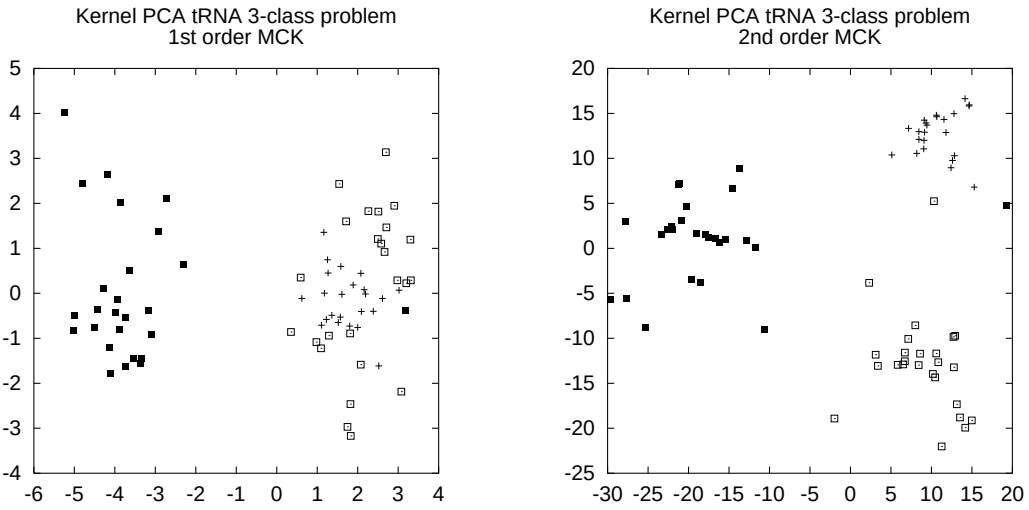

Figure 3: Two results of kernel PCA for 74 human tRNA sequences. White boxes, black boxes and plus symbols represent Ala-AGC, Asn-GTT and Cys-GCA, respectively. The left figure shows a result based on the 1st order MCK where only two classes are recognized while the Cys-GCA tRNAs are totally submerged in the Ala-AGC cluster. The figure on the right represents the result of the 2nd order MCK where three clusters are almost clearly separated.

tRNA-scan-SE. It is worth mentioning that ours is a novel method for validating results of automated structural RNA prediction programs.

5.4 Supervised Classification

Method In supervised classification based on SCFG likelihood, an SCFG is trained with the training sequences per sequence class. In the classification of a test sequence, the likelihoods of all SCFGs are computed and the sequence is assigned to the class with the largest likelihood.

On the other hand, when SVMs are used for the classification, the MCKs (1st and 2nd order) are first computed as in the kernel PCA case (i.e. SCFG is trained using 74 sequences). Then SVMs are trained due to the "one-against-others" scheme: each SVM is trained with the training sequences of a class as positive samples and those of the other classes as negative samples. In classification of a test sequence, the outputs of all SVMs are computed and the sequence is assigned to the class with the largest output.

For the cross-validation, we divided 74 sequences into training/testing data as follows: 10 sequences per class, thus 30 sequences, are randomly chosen for testing; the rest are used for training. In both the likelihood and SVM experiments, the training/testing stage is iterated 250 times.

Results In Figure 4, the averaged ROC curves for each of the three classes are shown. Note that ROC curves are plots of the fraction of false-positives (FFP) $\frac{fp}{tn+fp}$ versus the fraction of true-positives (FTP) $\frac{tp}{tp+fn}$. The 2nd order MCK yielded the best results. Although the curves tend to fluctuate due to small number of test data, overall performance differences are obvious. The 2nd order MCK exibited stable performance which constantly scores over 0.8 FTP at zero false positives (FFP=0). The performance of the 1st order MCK is not as strong. This result corresponds with the observations we made on the kernel PCA. In fact, the 1st order MCK tends to perform worse than the raw likelihood classification at the point of zero false positives; and in the case of Cys-GCA against others, it performs

worse than the raw likelihood classification.

6 Discussion and Conclusion

Our kernel design was applied to RNA sequences for which no other kernels have been used. It is worth noting that the 2nd order MCK possibly exploits the stacking energy which is indispensable for the secondary structure (Figure 5). Although a detailed modeling of RNAs requires massive implementation of SCFG states [11], the 2nd order MCK facilitates feature-extraction considering the stacking energy even with a simple SCFG that does not have the energy information. This is because, in terms of the marginalized kernel, the design of a kernel and a model are separated [13]. Besides, the introduction of MCKs sheds a new light on RNA sequence data analysis. Without the kernel, the performance of a stochastic model such as SCFG relies on a simple discriminator called likelihood, which is essentially a weighted average of all parameters. However, it is apparent that the whole information contained in parameters in such stochastic models is far more complicated and richer than the likelihood. In fact, the limitation of the likelihood is the major issue when it comes to a practical analysis against large scale data, for instance, the bold challenge issued by Rivas and Eddy [12]. Therefore, allowing further exploitation of the parameters using MCK makes sense.

7 Software

All computations introduced here are realized as a computer program: **SOKOS**-*a SOftware for Kernel computation Over SCFG*; an SCFG program that is capable of computing kernels as well as basic SCFG functions for learning and evaluation. This software is available at `http://www.cbrc.jp/~taishin/sokos/`. For a notion of the actual computating situation, we used a personal computer equipped with dual Pentium-III 1GHz and 2GB memory. SOKOS (multi-threaded) finishes SCFG training and kernel computations (1st and 2nd) within 10 minutes at 2–3 MB memory usage for the tRNA data set.

8 Acknowledgment

This work is partially supported by the Grant-in-Aid for Scientific Research on Priority Areas C "Genome Information Science."

References

[1] Asai, K., Hayamizu, S., and Handa, K., Prediction of protein secondary structure by the hidden markov model, *CABIOS*, 9(2):141–146, 1993.

[2] Durbin, R., Eddy, S., Krogh, A., and Mitchison, G., *Biological Sequence Analysis*, Cambridge University Press, 1998.

[3] Jaakkola, T., Diekhans, M., and Haussler, D., A discriminative framework for detecting remote protein homologies, *Journal of Computational Biology*, 7(1-2):95–114, 2000.

[4] Karchin, R., Karplus, K., and Haussler, D., Classifying G-protein coupled receptors with support vector machines, *Bioinformatics*, 18:147–159, 2002.

[5] Kim, C., Asai, K., and Konagaya, A., A generic criterion for gene recognitions in genomic sequences, *Genome Informatics*, 10:13–22, 1999.

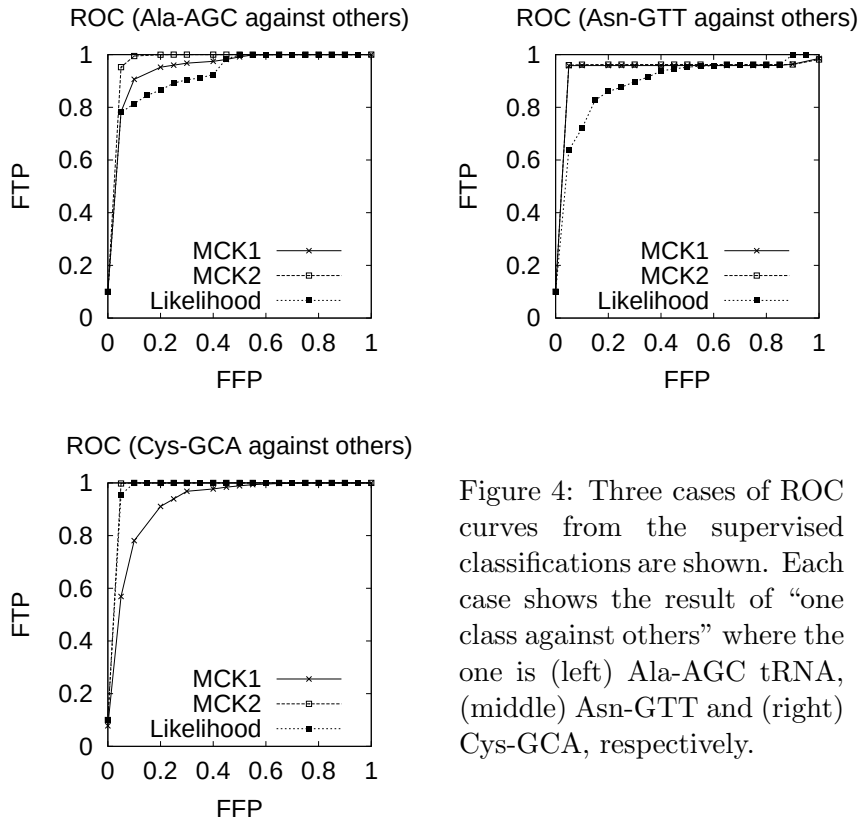

Figure 4: Three cases of ROC curves from the supervised classifications are shown. Each case shows the result of "one class against others" where the one is (left) Ala-AGC tRNA, (middle) Asn-GTT and (right) Cys-GCA, respectively.

[6] Lari, K. and Young, S., The estimation of stochastic context-free grammars using the inside-outside algorithm, *Computer Speech and Language*, 4:35–56, 1990.

[7] Leslie, C., Eskin, E., and Noble, W., The spectrum kernel: A string kernel for SVM protein classification, *Proc. Pacific Symposium on Biocomputing 2002*, 564–575, 2002.

[8] Lowe, T. and Eddy, S., tRNAscan-SE: A program for improved detection of transfer RNA genes in genomic sequence, *Nucleic Acids Res.*, 25:955–964, 1997.

[9] Nussinov, R., Pieczenk, G., Griggs, J., and Kleitman, D., Algorithms for loop matchings, *SIAM Journal of Applied Mathematics*, 35:68–82, 1978.

[10] Pavlidis, P., Furey, T., Liberto, M., Haussler, D., and Grundy, W., Promoter region-based classification of genes, *Proc. Pacific Symposium on Biocomputing 2001*, 151–163, 2001.

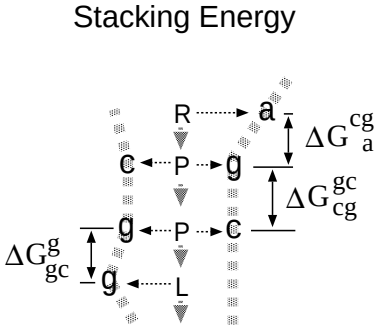

Figure 5: This figure shows that the stacking energy corresponds to the two consecutive states. For instance, $\triangle G_a^{cg}$ corresponds to the base-state combination $[(R, a), (R, cq)]$. It is likely that the 2nd order feature vector is capable of capturing essential structural information of RNAs.

[11] Rivas, E. and Eddy, S., A dynamic programming algorithm for RNA structure prediction including pseudoknots, *Journal of Molecular Biology*, 283:1168–1171, 1999.

[12] Rivas, E. and Eddy, S., Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs, *Bioinformatics*, 16:573–585, 2000.

[13] Tsuda, K., Kin, T., and Asai, K., Marginalized kernels for biological sequences, *Bioinformatics*, 18:S268–S275, 2002.

[14] Vert, J., Support vector machine prediction of signal peptide cleavage site using a new class of kernels for strings, In *Proc. Pacific Symposium on Biocomputing 2002*, 649–660, 2002.

[15] Zuker, M., Computer prediction of RNA structure, *Methods in Enzymology*, 180:262–288, 1989.

[16] Zuker, M., On folding all suboptimal foldings of an RNA molecule, *Science*, 244:48–52, 1989.

[17] Zuker, M. and Stiegler, P., Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information, *Nucleic Acids Res.*, 9:133–148, 1981.