ICOT Technical Memorandum: TM-1277

TM-1277

Classification of Proteins via Successive State Splitting of Hidden Markov Network

by
H. Tanaka, K. Onizuka & K. Asai (ETL)

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191 ~5

Institute for New Generation Computer Technology

Classification of Proteins via Successive State Splitting of Hidden Markov Network

Hidetoshi Tanaka and **Kentaro Onizuka**
Institute for New Generation Computer Technology (ICOT)
1-4-28, Mita, Minato-ku, Tokyo 108 Japan
htanaka@icot.or.jp, oniznka@icot.or.jp

Kiyoshi Asai
Electrotechnical Laboratory (ETL)
1-1-4, Umezono, Tsukuba
Ibaraki 305 Japan
asai@etl.go.jp

Abstract

The proteins will be classified by *Successive State Splitting (SSS)* algorithm without previous knowledge of the proteins. The SSS for *allophone modeling*, which was proposed by Takami and Sagayama, enables to obtain an appropriate *Hidden Markov Network* automatically, and the *Hidden Markov Model* simultaneously. We apply it to the protein classification by regarding amino acids as phonemes. The Hidden Markov Model can be a representation of the classified proteins and used as an index of similarity search in protein databases.

Introduction

Protein Classification – Superfamily in PIR It is one of the most important issues in protein analysis to research how to classify protein sequences appropriately. Proteins are classified by various factors such as their functions, structures, or organizations, but we consider it most invaluable to classify proteins by their sequences. Protein sequence classifications are used for an index system for similarity searches, which contain the similarity of amino acid sequences and the similarity of protein structures, so that we can predict functions and structures of unknown proteins.

There is a conventional classification called *superfamily* in a protein sequence database, PIR[6]. It classifies protein sequences by similarity scores of *DP-matching* using some similarity score system between amino acids. It is however inefficient in practical use by two reasons. One is because of the imbalance in frequencies. There are many superfamilies of single protein while several superfamilies of large amount of proteins. We should emphasize local similarities with various scores instead of such total similarities with scalar scores as superfamily classifications. The other is the computational cost. It is necessary to do many DP-matching procedures in order to determine the appropriate class. Each DP-matching costs $O(N^2)$ where N is the length of the sequences.

Protein Representation – Motifs in ProSite It is another important issue to research how to represent the classified proteins. Through the alignment of the protein sequences, we determine a specific consensus pattern (*motif*)[8], or a variation of protein sequences (*profile*)[4] to represent the classified proteins. The former uses only regular-expression-like patterns of amino acids as representation of the groups of protein sequences while the latter uses a kind of probability distribution of amino acids as the representation.

From the information retrieval point of view, the *recall* and *precision* [3] of motifs are rather sufficient. For example, the motif database system, ProSite[1], classifies each protein into five sorts on the relationship between motifs and proteins in Swiss-Prot[2]: true-positive(true, for short), false-negative(missed), potential-hit, unknown, and false-positive(wrong), where false-negative proteins are true but missed by the motif matching while false-positive proteins are picked but wrong. A rough estimation is shown in Table 1. That estimates its recall by true/(true+missed) and its precision by true/(true+wrong).

Table 1: Estimation of Recall & Precision Rate of Motifs

Version	true	missed	wrong	unknown	potential	recall	precision
ProSite 7	10787	260	571	75	630	97.6 %	94.9 %
ProSite 9	12736	352	661	86	727	97.3 %	95.0 %

However, it seems insufficient for the biological practical use. There is a trade-off between complexity of patterns and improvement of the recall and precision. For example, a motif "apple domain" in ProSite shows quite complicated pattern:

C-x(3)-[LIVMFY]-x(5)-[LIVMFY]-x(3)-[DENQ]-[LIVMFY]-x(10)-C-x(3)-C-T-x(4)-C-x-[LIVMFY]-F-x-[FY]-x(13,14)-C-x-[LIVMFY]-[RK]-x-[ST]-x(14,15)-S-G-x-[ST]-[LIVMFY]-x(2)-C.

Capital letters correspond to amino acids, letters in square brackets indicate alternatives, and the string "$x(n)$" means n-residues of any amino acids. "(m, n)" is a representation of a region. It is natural that such a complex pattern scores 100% in both its recall and precision. It is not appropriate for the comparison between motifs or the recognition of the relationship between patterns and functions.

SSS and HMM We use *Hidden Markov Model (HMM)* [7] as the representation of the protein classification [10] and *Successive State Splitting (SSS)* algorithm [9] as their classification algorithm. It is important to generate a precise and robust HMM: representation capability is necessary for the precision of the model, whereas simplicity is necessary for the robustness of the model. In order to build the robust and precise HMM, it is quite essential to determine an appropriate *Hidden Markov Network (HMNet)*. The SSS algorithm automatically builds an appropriate HMNet with the *maximum likelihood* criterion.

It is essential to introduce probability distribution to the representation of the protein classification. It improves representation capability and reduces description complexity. There are several related works that employ probability in the motif representation, for example, probability with *MDL* criteria [5]. We consider the Hidden Markov Model is the most suitable for representing the protein classification. It represents the protein set by the network whose nodes and arcs have probability distributions. It can be regarded as flexible multiple consensus patterns for the protein set.

SSS Algorithm in Protein Classification

We show an overview of the SSS algorithm [9]. At every step, we explain SSS algorithm for phonemes (allophone modeling), and show the difference between phonemes and amino acids.

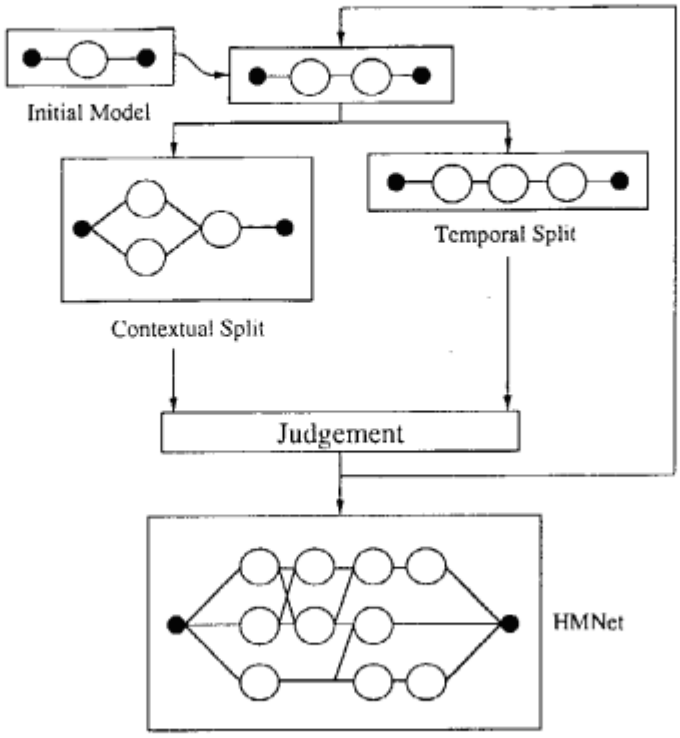

Figure 1: **SSS Algorithm** [9] — *Every box indicates the Hidden Markov Network (HMNet) of respective stages of the SSS algorithm. Each circle represents a state and each arc between circles represents a state-transition in the HMNet. Every state has a self transition, but the corresponding looped arcs are omitted in this figure for the simplicity.*

Step 0: Training of the Initial Model Let the initial model be the HMM of one state, whose output is according to the probability density function (PDF) of the diagonal-covariance 2-mixture Gaussian. The model is trained by all data with a learning algorithm such as *forward-backward (Baum-Welch)* algorithm or *Viterbi* algorithm.

As amino acids have discrete distribution while phonemes have continuous distribution, we should find out common grounds between them to apply SSS algorithm to the protein classification. There are two approaches: to put amino acids into a continuous space and to develop a discrete SSS algorithm. We chose former approach.

We provide a continuous space out of 20 amino acids using *Dayhoff PAM-250* as a distance matrix, so that amino acids can be mapped onto the continuous space (a *continuous protein SSS*). It is a kind of *Multi-Dimensional Scaling (MDS)*. It is the easiest way though PAM-250 should be properly biased to make such space. PDFs of the diagonal-covariance 2-mixture Gaussian are applied to the continuous space of amino acids, so that we can apply the continuous SSS algorithm to the proteins.

Step 1: Calculation of Distribution Size We should choose one state to split. For each state i, a criterion d_i by Eq.(1) is calculated in the original phoneme SSS. It represents a *size* of the output PDF of 2-mixture Gaussian on the state S(i). It is applicable as it is to the

continuous protein SSS.

$$d_i = \sum_k^K \frac{{\sigma_{ik}}^2}{{\sigma_{Tk}}^2} n_i, \qquad {\sigma_{ik}}^2 = \lambda_{i1}{\sigma_{i1k}}^2 + \lambda_{i2}{\sigma_{i2k}}^2 + \lambda_{i1}\lambda_{i2}(\mu_{i1k} - \mu_{i2k})^2 \tag{1}$$

K:	parameter dimension,
$\lambda_{1i}, \lambda_{2i}$:	weight coefficients of mixture Gaussians of S(i),
μ_{i1k}, μ_{i2k}:	k-th means of S(i),
${\sigma_{i1k}}^2, {\sigma_{i2k}}^2$:	k-th variances of S(i),
n_i:	the number of training samples for S(i),
${\sigma_{Tk}}^2$:	k-th variance of all samples.

Step 2: Split of the State The state S(m) of the largest d_m is split into two states, S'(m) and S(M), where M is the current number of the states. Both S'(m) and S(M) have the output PDFs of single Gaussian while S(m) had a PDF of 2-mixture Gaussian. Each PDF of single Gaussian corresponds to each Gaussian of the 2-mixture PDF.

There are parallel splitting called *contextual split* and sequential one called *temporal split* in the phoneme SSS. We choose either by estimating *maximum likelihood* of both contextual split P_c and temporal split P_t.

— **Contextual Split** S'(m) and S(M) are concatenated in parallel by the contextual split (see Figure 1). Since all paths on S(m) should be split into two, all training samples should also be split. The samples are split by a contextual factor (such as preceding phonemes or succeeding phonemes) to maximize P_c, which is estimated by Eq.(2).

$$P_c = \max_J \sum_l \max\left(p_m(y_{Jl}), p_M(y_{Jl})\right), \tag{2}$$

$y_{1l}y_{2l}\cdots y_{Tl}$:	l-th sequence (sample),
T:	the length of l-th sequence,
J:	a series of positions of l-th sequence on S(m),
y_{Jl}:	a subsequence of $y_{1l}y_{2l}\cdots y_{Tl}$ on S(m),
$p_m(y_{Jl})$:	total likelihood for y_{Jl} on S'(m),
$p_M(y_{Jl})$:	total likelihood for y_{Jl} on S(M).

After maximum J is determined, corresponding residues e_{Jl} are distributed to the states S'(m) and S(M) by Eq.(3).

$$\begin{aligned} e_{Jl} &\in E_{m_J}(p_m(y_{Jl}) \le p_M(y_{Jl})), \\ e_{Jl} &\in E_{M_J}(p_m(y_{Jl}) > p_M(y_{Jl})), \end{aligned} \tag{3}$$

E_{m_J}:	a serics of residues on S'(m),
E_{M_J}:	a series of residues on S(M).

— **Temporal Split** S'(m) and S(M) are concatenated in series by the temporal split (see Figure 1). There are two models: S'(m)–S(M) and S(M)–S'(m). Thus, P_t is estimated by Eq.(4).

$$P_t = \max\left(p_{mM}(Y), p_{Mm}(Y)\right), \tag{4}$$

Y:	all samples on S(m),
$p_{mM}(Y)$:	total likelihood of the model S'(m)–S(M),
$p_{Mm}(Y)$:	total likelihood of the model S(M)–S'(m).

Step 3: Retraining of the Model The model which contains two states with single Gaussian PDF, S'(m) and S(M), should be retrained by all samples, so that all states have PDFs of 2-mixture Gaussian again.

Step 4: Change of Distribution When the number of states is increased to a prescribed number, the model is retrained as all states have single Gaussian PDFs.

Computational Cost of SSS

The computational cost of the original phoneme SSS algorithm is practical. An upper limit is set to the number of the states, which increases by one when the split occurs at every cycle. Thus the number of iteration of this algorithm is limited.

It needs moderate computation at every step of the cycle: estimations of the size of the distribution at all states (Step 1), calculations of P_c and P_t (Step 2), and a training of the HMM (Step 3). Step 1 and Step 2 are rather negligible, and Step 3 has several practical algorithms such as the Baum-Welch algorithm we chose. The cost of the continuous protein SSS will be practical, because the protein SSS is almost the same algorithm as the original one.

Future Work

Discrete Protein SSS When we apply SSS algorithm to the protein classification, we can think of a *discrete protein SSS* using multiple alignment as another idea. It is essential that proteins can be split into two groups while it is not so essential that proteins can be regarded as mixture of any parametric PDF such as Gaussian. At Step 0, it is rather reasonable to split proteins according to the multiple alignment scores, though the search for optimal partition of size N needs at least $2^{N-1}-1$ trials of two group alignment. It is invaluable to discover good estimation to reduce computation.

At Step 1, we can use entropy for the decision criterion instead of d_i in Eq.(1). It can be estimated by D_m in Eq.(5):

$$D_m = \sum_r p_r \log p_r, \tag{5}$$

where p_r is the frequency of the residue r.

At Step 2, the parallel split corresponds to the distribution of protein sequences into two groups. We should somehow reduce the number of trials of such splitting, because there are $2^{N-1}-1$ ways where N is the number of proteins. It is the same problem as the one at Step 0.

The sequential split corresponds to the vertical partition of the alignment. There are not so more than L ways of partitions where L is the length of the alignment. In other words, there are not more than $2L'$ ways where L' is the length of the longest sequence. It is the rather reasonable number of trials than the parallel split, though every trial needs one alignment procedures for each part of sequences.

The discrete protein SSS algorithm is impractical unless we have any idea to reduce computation at Step 0 and Step 2.

Continuous Protein SSS: other scalings for the amino acids We can use various indexes such as the electric charge or the hydrophobicity instead of the space made from PAM-250. Through an investigation of the resultant HMM, we can recognize what kind of indexes are

influential for every specific characteristics, such as alpha-helix domain of secondary structures or DNA binding sites of functional domains.

Concluding Remarks

The SSS algorithm enables to obtain an appropriate *Hidden Markov Network* automatically, and the *Hidden Markov Model* simultaneously. Its effectiveness is confirmed in phonemes [9]. The proteins will be classified by SSS algorithm without previous knowledge of the proteins. We apply it to the protein classification by regarding amino acids as a continuous variant and implemented the continuous protein SSS algorithm.

References

[1] Bairoch, A.: *PROSITE: A Dictionary of Protein Sites and Patterns, User's Manual*, (1991).

[2] Bairoch, A.: *EMBL Data Library: SWISS-PROT Protein Sequence Database Release Notes*, Release 20, (1991).

[3] Frakes, W.B. and Baeza-Yates, R.(ed.): *Information Retrieval, Data Structures & Algorithms*, Prentice Hall, (1992).

[4] Gribskov, M., McLachlan, A.D. and Eisenberg, D.: "Profile Analysis: Detection of Distantly Related Proteins", *Proc. Natl. Acad. Sci. USA*, Vol.84, pp.4355-4358 (1987).

[5] Konagaya, A. and Kondo, H.: "Stochastic Motif Extraction using a Genetic Algorithm with the MDL principle", *IIICSS 26*, pp.746-753 (1993).

[6] National Biomedical Research Foundation: *PIR Document CXFSD-1091: CODATA Exchange Format Specification*, Ver.2.1, (1991).

[7] Rabiner, L.R.: "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proc. IEEE*, Vol.77, No.2, pp.257-286 (1989).

[8] Staden, R.: "Methods to Define and Locate Patterns of Motifs in Sequences", *CABIOS*, Vol.4, No.1, pp.53-60 (1988).

[9] Takami, J. and Sagayama, S.: "A Successive State Splitting Algorithm for Efficient Allophone Modeling", *IEEE*, (1992).

[10] Tanaka, H., Asai, K., Ishikawa, M. and Konagaya, A.: "Hidden Markov Models and Iterative Aligners: Study of their Equivalence and Possibilities", *ISMB 1*, (1993).