社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－MPS－58 (13)
2006／3／17

局所アラインメントカーネルを用いたアミノ酸置換行列の最適化

西郷浩人[1] ジャンフィリップ・ヴェール[2] 阿久津達也[1]
1. 京都大学化学研究所 バイオインフォマティクスセンター
2. パリ高等鉱山学校 計算生物学グループ

生物学的配列間の類似性検出はバイオインフォマティクスにおける重要な問題で、特に微弱な類似性を検出するのは難しいことが知られている。我々は先の研究において局所アラインメントカーネルを提案して、良い結果を示した。局所アラインメントカーネルの性能はアミノ酸置換行列に依存する。我々は局所アラインメントカーネルのアミノ酸置換行列に対する導関数が解析的に求まり、かつ動的計画法で効率良く計算できることを示す。さらに導関数を利用する最適化法と組み合わせることにより真の類似配列を偽の類似配列から区別するようにアミノ酸置換行列を最適化する。局所アラインメントカーネルは、Smith-Waterman アルゴリズムによって最適化されたアミノ酸置換行列よりも局所アラインメントカーネルによって最適化されたアミノ酸置換行列において良い性能を示した。更にこの行列は Smith-Waterman アルゴリズムと組み合わせてもうまく利用することが出来る。

Optimizing amino acid substitution matrices with local alignment kernels

Hiroto Saigo[1] Jean－Philippe Vert[2] Tatsuya Akutsu[1]
1. Bioinformatics Center, Institute for Chemical Research, Kyoto University
2. Center for Comutational Biology, Ecole des minde de Paris

Detecting similarity between protein sequence is one of the core problems in bioinformatics, and detecting weak similarities is known as a hard problem. We have proposed a local alignmnet kernel for this purpose and showed good performance in the previous research. The local alignment kernel depdends on amino acid substitution matrices. We show that we can analytically calculate the derivatives of the local alignment kernels with respect to amino acid substitution matrix as well as their efficient calculation through dynamic programming. Then we plug them into the gradient based optimization procedure which is designed to discriminate true homologs from non-homologs. The local alignment kernel exhibits better performance when it uses the matrices and gap parameters optimized by this procedure than when it uses the matrices optimized for the Smith-Waterman algorithm. Furthermore, the matrices and gap parameters optimized for the local alignment kernel can also be used successfully by the Smith-Waterman algorithm.

Introduction

Sequence comparison for homology detection remains one of the core tools in bioinformatics. Especially, the identification of *remote homologs* is a challenging task because divergences in sequences can prevent sequence comparison algorithms from recognizing those homologies.

We developed previously a score to compare protein sequences, called the *local alignment kernels* (LA kernels) [2], which in combination with support vector machines could better detect remote homology than several state-of-the-art methods, including the Smith-Waterman (SW) algorithm, in a benchmark experiment based on the SCOP database.

Both the SW algorithm and the LA kernel depend critically on gap parameters, and on a substitution matrix (also called score matrix) that quantifies the contribution in the score of an alignment between any two given amino acids.

Kann et al.[1] investigated the possibility to automatically optimize a substitution matrix to improve the performance of the final score in terms of homology detection. Kann et al. prepared a dataset of homologous pairs from the Clusters of Orthlogous

Group(COG) database, and maximized the average C(confidence)-values of the pairs, where the C value is designed to be large when the expected number of non-homologous sequences scoring higher than the candidate pair is small.

However, the method by Kann et al. suffer from the difficulty to optimize the SW score with respect to the substitution matrix. Indeed, the fact that the SW score only takes into account the maximum scoring alignment makes it non-differentiable with respect to the substitution matrix. As a result, the final objective function which is based on SW scores is itself not differentiable with respect to the substitution matrix, and therefore difficult to optimize. Hence the authors had to introduce a strong assumption that the alignment remains unchanged while changing the substitution matrix.

This study is devoted to the extension of this approach to the LA kernel, instead of the SW local alignment score. The motivations for this work are twofold. First, the LA kernel was previously shown to be a more sensitive measure of similarity for remote homologs, suggesting that it could also remain competitive with an optimized substitution matrix. Second, contrary to the SW score, the LA kernel is differentiable with respect to the elements of the substitution matrix and the gap parameters, and these derivatives can be computed efficiently by dynamic programming. This means that any objective function that is itself differentiable with respect to the LA kernel is differentiable with respect to the substitution matrix and can be optimized by simple gradient descent methods, without the need to alternate between the gradient descent steps and alignment steps used in the optimization of the SW score. Applying this procedure to the objective function used in [1], we optimized the substitution matrix as well as the gap parameters to separate true homologs from non-homologs in a dataset of protein sequences extracted from the COG database, and evaluated the performance of the resulting methods for homology detection on several independent test sets. We compared these results with those obtained after optimizing the substitution matrix with the Smith-Waterman algorithm [1], and compared how each scoring algorithm performs with each optimized matrix.

Methods

The LA kernel

The LA kernel [2] between two sequences $\mathbf{x}$ and $\mathbf{y}$ is defined by

$$K_{LA}(\mathbf{x},\mathbf{y}) = \sum_{\pi} e^{\beta s(\mathbf{x},\mathbf{y},\pi)} , \qquad (1)$$

where β is a parameter, π runs over the possible local alignment between $\mathbf{x}$ and $\mathbf{y}$, and $s(\mathbf{x},\mathbf{y},\pi)$ is the score of the alignment π between x and y. The score of an alignment π is itself given by the sum of substitution scores for the letters paired together, minus an affine gap penalty:

$$\begin{aligned} s(\mathbf{x},\mathbf{y},\pi) &= \sum_{a,b} n_{a,b}(\mathbf{x},\mathbf{y},\pi) S(a,b) \\ &- n_{g_d}(\mathbf{x},\mathbf{y},\pi) g_d - n_{g_e}(\mathbf{x},\mathbf{y},\pi) g_e , \end{aligned}$$

where $n_{a,b}(\mathbf{x},\mathbf{y},\pi)$ represents the number of times that the amino acid a is aligned with the amino acid b, $S(a,b)$ denotes the substitution score between amino acids a and b, $n_{g_d}(\mathbf{x},\mathbf{y},\pi)$ and $n_{g_e}(\mathbf{x},\mathbf{y},\pi)$ are the counts for every single opening and extension of a gap , and g_d and g_e are penalties for gap opening and gap extension, respectively.

As shown in (1) the LA kernel takes into account all possible alignments between two strings by summing the scores, and can be computed by the following algorithm.

Algorithm 1: local alignment kernels

Initialization : for $i = 0, \ldots, |\mathbf{x}|$ and $j = 0, \ldots, |\mathbf{y}|$:

$$\begin{aligned} M_{i,0} &= M_{0,j} = X_{i,0} = X_{0,j} = X2_{i,0} = X2_{0,j} = 0, \\ & Y_{i,0} = Y_{0,j} = Y2_{i,0} = Y2_{0,j} = 0, \end{aligned}$$

Iteration : for $i = 1, \ldots, |\mathbf{x}|$ and $j = 1, \ldots, |\mathbf{y}|$:

$$\begin{aligned} M_{i,j} &= e^{\beta S(x_i,y_j)}(1 + \\ & X_{i-1,j-1} + Y_{i-1,j-1} + M_{i-1,j-1}), \\ X_{i,j} &= e^{\beta g_d}(M_{i-1,j}) + e^{\beta g_e}(X_{i-1,j}), \\ Y_{i,j} &= e^{\beta g_d}(M_{i,j-1} + X_{i,j-1}) + e^{\beta g_e}(Y_{i,j-1}), \\ X2_{i,j} &= M_{i-1,j} + X2_{i-1,j}, \\ Y2_{i,j} &= M_{i,j-1} + X2_{i,j-1} + Y2_{i,j-1}, \end{aligned}$$

Termination :

$$K_{LA}(\mathbf{x},\mathbf{y}) = 1 + X2_{|\mathbf{x}|,|\mathbf{y}|} + Y2_{|\mathbf{x}|,|\mathbf{y}|} + M_{|\mathbf{x}|,|\mathbf{y}|}$$

In the above algorithm, M stands for the matching state between amino acid, while $X, Y, X2, Y2$ are for states corresponding to insertions or deletions.

The score of the LA kernels is then described as the logarithm of (1):

$$\hat{K}_{LA}(\mathbf{x}, \mathbf{y}) = \log K_{LA}(\mathbf{x}, \mathbf{y}) = \log \sum_{\pi} e^{\beta s(\mathbf{x}, \mathbf{y}, \pi)} . \tag{2}$$

The derivative of (2) with respect to $S(a, b)$ can therefore be written as:

$$\frac{\partial \hat{K}_{LA}(\mathbf{x}, \mathbf{y})}{\partial S(a, b)} = \frac{\frac{\partial}{\partial S(a,b)} \sum_{\pi} e^{\beta s(\mathbf{x}, \mathbf{y}, \pi)}}{\sum_{\pi} e^{\beta s(\mathbf{x}, \mathbf{y}, \pi)}}. \tag{3}$$

Note that the denominator of (3) is the same as (1), and can therefore be calculated by the Algorithm 1 above, while the numerator of (3) is also calculated by dynamic programming similar to Algorithm 1.

To assess the significance of the score on a database search, the Z-score is widely used:

$$Z = \frac{s - \mu}{\sqrt{\langle s^2 \rangle - \mu^2}} = \frac{s - \mu}{\sigma} ,$$

where s is the score of the candidate homolog in the database against the query, and μ and σ are the mean and variance of the scores of other sequences in the database against the query. For extreme values (maxima or minima) such as the SW score, the extreme value distribution (EVD) is commonly used to assess the statistical significance of the scores. The probability that a given random score is equal or greater than s is given by $p(\mu > s) = 1 - e^{-e^{-\alpha Z - \beta}}$, where α, β are parameters for extreme value distribution. Then for the search of a database of size D, the expected number of scores which are higher than s is $E = Dp(\mu > s)$. A natural objective function to quantify the performance of an algorithm for remote homology is therefore to minimize the E-values obtained on pairs of distant homologs. Following Kann et al. [1], we consider the confidence value $C = 1/(1+E)$, setting $D = 100000$ for the computation of the E-value, and define our objective function to be maximized as the average of C over a training set of homologous pairs. We used both the algorithms of the Smith-Waterman and the LA kernels together with their derivatives, then maximized the objective function using gradient descent until Armijo's rule was satisfied. Since there is no guarantee of reaching the global optimum, we used several starting matrices such as BLOSUM62, PAM250, GCB, JTT, with default gap parameters (12 and 2 for open and extension).

Training and testing to discriminate homologs from non-homologs are performed on COG database. We are interested in homologs whose homology is hard to detect, thus collected sequences with less than 20% identity only from the COG database, resulting in 395 pairs of protein sequences. We used 300 of them for training and left the rest for the evaluation. Also, we prepared sequences pairs of distant homologs from the PFAM database in a similar manner, resulting in 200 additional pairs, and used as the second test set. The third and the fourth data sets are the COG close and PFAM close dataset, which are prepared keeping the identity between 25% and 40%.

Results

The performance of the SW algorithm and the LA kernel with various substitution matrices on four independent test sets (COG distant COG close, PFAM distant, PFAM close) are shown on Table 1. For each dataset, we observe that the best substitution matrix is always the one optimized with the LA kernel starting from BLOSUM62, both for the LA kernel and the SW score, and that the LA kernel always outperforms the SW score with this matrix.

Interestingly, the matrix optimized with the LA kernel starting from BLOSUM62 is better than the matrix optimized with the SW algorithm starting from BLOSUM62 with the SW algorithm as well, although it was optimized with the LA kernel. This highlights the fact that the optimization based on the SW score has more difficulty to find a good maximum than the optimization based on the LA ker-

nel. Finally we observe that the LA kernel outperforms the SW algorithm with both optimized matrices, confirming its superiority over a large choice of parameters.

Conclusion and Discussion

We proposed a method to optimize amino acid substitution matrices for the LA kernels, based on the properties of differentiability of the LA kernel with respect to the substitution matrix. The main contribution of this study is to propose an optimization framework for substitution matrices based on an exact gradient descent method. This approach is made possible by the fact that the alignment score we consider, the LA kernel, is differentiable with respect to the substitution matrix, contrary to the SW alignment score. The fact that the matrix optimized with this approach outperforms the matrix optimized with the SW score even for the SW algorithm itself suggests that there is an important benefit for this approach compared to the more heuristic nature of the optimization in the case of the SW score. Intuitively, the optimization of the LA kernel can be thought of as the optimization of a smooth approximation of the SW score, which can more easily find good local optima. This suggests, in the spirit of simulated annealing, that further improvements for the SW algorithms might be obtained by optimizing the LA kernel.

A second point to be highlighted is the good performance of the LA kernel as a similarity score compared to the SW score. While it was shown in [2] that the LA kernel outperforms the SW score as a kernel function for support vector machines, the present studies validate the relevance of the LA kernel as measure of similarity. It should be pointed out that the advantage of the LA kernel over the SW score is expected to increase for remote homologs, because when the sequence similarity is low the best local alignment computed by the SW score is likely not to be the correct one, and in this case an average over a large number of candidate alignments might provide safer evidences for homology .

Finally, we can conclude that the derived matrices may be useful when standard methods fail to detect homologs.

参考文献

[1] Maricel Kann, Bin Qian and Richard A. Goldstein :Optimization of a New Score Function for the Detection of Remote Homologs. *Proteins* 2000, 41:498-503.

[2] Hiroto Saigo, Jean-Philippe Vert, Nobuhisa Ueda and Tatsuya Akutsu :Protein homology detection using string alignment kernels. *Bioinformatics* 2004, 20:1682-1689.

	< C >							
Score matrix	COG distant		COG close		PFAM distant		PFAM close	
	SW	LA	SW	LA	SW	LA	SW	LA
BLOSUM62	0.314	0.338	0.718	0.731	0.454	0.486	0.764	0.781
BLOSUM62SWOPT	0.333	0.361	0.716	0.727	0.479	0.506	0.765	0.781
BLOSUM62LAOPT	0.348	**0.418**	**0.734**	**0.742**	**0.508**	**0.583**	**0.780**	**0.805**
PAM250	0.277	0.280	0.531	0.518	0.426	0.426	0.619	0.611
PAM250SWOPT	0.269	0.270	0.530	0.514	0.423	0.417	0.621	0.514
PAM250LAOPT	0.277	0.236	0.531	0.517	0.427	0.425	0.620	0.610
GCB	0.106	0.065	0.329	0.320	0.371	0.109	0.371	0.364
GCBSWOPT	0.162	0.164	0.455	0.459	0.131	0.140	0.494	0.503
GCBLAOPT	0.268	0.278	0.559	0.566	0.256	0.275	0.569	0.580
JTT	0.311	0.295	0.525	0.514	0.474	0.460	0.605	0.597
JTTSWOPT	**0.369**	0.372	0.588	0.459	0.504	0.140	0.660	0.503
JTTLAOPT	0.312	0.228	0.536	0.522	0.479	0.466	0.617	0.606

Table 1. Comparison of various scoring matrices and scoring algorithms. The first column shows the scoring matrices. For example, BLOSUM62SWOPT is the matrix optimized for the SW algorithm starting from BLOSUM62 matrix. The second column shows the performance of each score matrix by the SW algorithm on COG distant test set. The following columns show the performance, in terms of average C, of each matrix used in combination with either the SW algorithm or the LA kernel on four different datasets. The best < C > in each column is highlighted in bold font.