Position-aware string kernels with weighted shifts and a general framework to apply string kernels to other structured data

Kilho Shin

Carnegie Mellon CyLab Japan

Abstract. In combination with efficient kernel-base learning machines such as Support Vector Machine (SVM), string kernels have proven to be significantly effective in a wide range of research areas (*e.g.* bioinformatics, text analysis, voice analysis). Many of the string kernels proposed so far take advantage of simpler kernels such as trivial comparison of characters and/or substrings, and are classified into two classes: the *position-aware* string kernel which takes advantage of positional information of characters/substrings in their parent strings, and the *position-unaware* string kernel which does not. Although the positive semidefiniteness of kernels is a critical prerequisite for learning machines to work properly, a little has been known about the positive semidefiniteness of the position-aware string kernel. The present paper is the first paper that presents easily checkable sufficient conditions for the positive semidefiniteness of a certain useful subclass of the position-aware string kernel: the similarity/matching of pairs of characters/substrings is evaluated with weights determined according to *shifts* (the differences in the positions of characters/substrings). Such string kernels have been studied in the literature but insufficiently. In addition, by presenting a general framework for converting positive semidefinite string kernels into those for richer data structures such as trees and graphs, we generalize our results.

1 Introduction

The string kernel, in combination with efficient kernel-base learning machines such as Support Vector Machine (SVM), has been applied to a wide range of research areas (*e.g.* bioinformatics ([1–3]), text analysis ([4]), voice recognition), and proves to be significantly effective for clustering string-type data.

Many of the known string kernels were engineered based on simpler kernels such as trivial comparison of characters and/or substrings, and therefore are classified into two classes: the *position-aware* string kernel which takes advantage of positional information of characters/substrings in their parent strings, and the *position-unaware* string kernel which does not (*e.g.* the spectrum kernel ([1]), the string subsequence kernel ([4])). Some of position-aware string kernels evaluate only those pairs of characters/substrings whose positions in their parent strings coincide with each other (*e.g.* the locality-improved kernel ([2]) and the weighted-degree kernel ([3])). This constraint, however, is too restrictive for

some applications, and it may be desirable to allow *weighted shifts* to improve generality — a pair of characters/substrings whose positions differ by a shift s is also evaluated but with a weight $\bar{w}_s$ determined according to s. The codon-improved kernel ([2]) and the weighed-degree kernel with shifts ([3]) are examples of kernels of this type. A brief survey of string kernels is given in Section 2.

Theoretically, string kernels with weighted shifts are expected to be effective when applied to string-type data such that discriminative patterns intensively distribute around plural particular positions (*e.g.* written/spoken texts, html documents). On the other hand we have to pay careful attention when engineering kernels of the type, since it is known that naïve selection of weights could easily harm the positive semidefiniteness of the resulting kernels. A kernel $K(x, y)$ is said to be positive semidefinite, if, and only if, for an arbitrary set of data $\{x_1, \ldots, x_N\}$, the derived Gram matrix $[K(x_i, x_j)]_{(i,j)\in\{1,\ldots,N\}^2}$ is positive semidefinite (*i.e.* all the eigenvalues are non-negative), and the positive semidefiniteness of a kernel is a critical prerequisite for kernel-base classifiers to work properly.

The present paper is the first paper that presents easily checkable sufficient conditions for position-aware string kernels with weighted shifts to be positive semidefinite (see Section 3 for the conditions, and Section 5 and Section 6 for proofs). A limited part of the results of the present paper is presented in [5] without proofs.

Furthermore, we describe a general framework for transforming positive semidefinite string kernels into positive semidefinite kernels for richer data structures such as trees and graphs (Section 4)

2 A survey of string kernels

In this section, we review 6 important string kernels in the literature, namely the spectrum kernel ([1]), the string subsequence kernel ([4]), the locality-improved kernel ([2]), the weighted-degree kernel ([3]), the codon-improved kernel ([2]), and the weighted-degree kernel with shifts ([3]). The spectrum kernel and the string subsequence kernel are examples of position-unaware kernels, while the others are examples of position-aware kernels. In particular, the codon-improved kernel and the shifted-weighted-degree kernel allow weighted sifts (see Section 1). Also, a good survey of kernels for structured data is available in [6].

In this section, we use the following notations. We define $C_{n,m}$ and $D_{n,m}$ as follows for integers $n \leq m$.

$$C_{n,m} = \{(i, i+1, \ldots, i+n-1) \mid 1 \leq i \leq m-n+1\}$$
$$D_{n,m} = \{(i_1, \ldots, i_n) \mid 1 \leq i_1 < \cdots < i_n \leq m\}$$

Strings are defined over an alphabet $\mathcal{A}$, and, for a string $x = x_1 x_2 \ldots x_{|x|}$ and a vector $\boldsymbol{i} \in D_{n,|x|}$, we let $x[\boldsymbol{i}]$ denote the n-length string $x_{i_1} x_{i_2} \ldots x_{i_n}$. The function $\delta(a, b)$ indicates Kronecker's delta function — it returns 1 if a and b are "identical", and 0 otherwise.

2.1 Position-unaware string kernels

Fix an integer $n \geq 1$ in this subsection. The spectrum kernel ([1]) was introduced for protein classification, and counts up the contiguous substrings of length n shared between input strings x and y.

$$K(x,y) = \sum_{\boldsymbol{i} \in C_{n,|x|}} \sum_{\boldsymbol{j} \in C_{n,|y|}} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$$

On the other hand, the string subsequence kernel ([4]) has proven to be effective for text classification. Compared with the spectrum kernel, the restriction on substrings with respect to contiguity is relaxed, and instead, the decay factor $\lambda \in (0,1)$ is introduced to emphasize contiguous substrings.

$$K(x,y) = \sum_{\boldsymbol{i} \in D_{n,|x|}} \sum_{\boldsymbol{j} \in D_{n,|y|}} \lambda^{i_n - i_1 + j_n - j_1} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$$

Both kernels only evaluate matching of substrings as sequences of characters, and don't take the positions of substrings into account at all.

2.2 Position-aware string kernels with precise position matching

In some applications, it has proven true that positional information of substrings is an important factor to improve generality of machine learning.

The locality-improved kernel ([2]) is a successful example of string kernels exploiting positional information, and compares between nucleotide sequences x and y in order to recognize translation initiation sites (TIS). The locality-improved kernel $K(x,y)$ inspects matching between nucleotides at the same position in x and y, and is defined as follows with non-negative weights $\gamma_{|k|}$.

$$\mathrm{win}_p(x,y) = \left(\sum_{k=-\ell}^{\ell} \gamma_{|k|} \delta(x_{p+k}, y_{p+k}) \right)^{d_1}, \quad K(x,y) = \left(\sum_{p=\ell+1}^{L-\ell} \mathrm{win}_p(x,y) \right)^{d_2}$$

The weighted degree kernel ([3]) is another successful example of position-aware kernels, and is defined as follows.

$$K(x,y) = \sum_{n=1}^{d} \beta_n \sum_{\boldsymbol{i} \in C_{n,L}} \delta(x[\boldsymbol{i}], y[\boldsymbol{i}])$$

The kernels are different from each other in that the former performs character-by-character comparison, while the latter compares substrings. They, however are the same in that the character/substring pairs to be compared with each other are located at the same position in their parent strings.

2.3 Position-aware string kernels with weighted shifts

The codon-improved kernel ([2]) is a modification of the locality-improved kernel so as to exploit the *a priori* knowledge "a coding sequence (CDS) shifted by three nucleotides still looks like CDS ([2])". In fact, in addition to the matches of nucleotides placed at the same position, it compares the pairs of nucleotides whose positions differ exactly by 3. When T_3 denotes the 3-shift operator that chops off the leading 3 nucleotides, the window score $\mathrm{win}_p(x, y)$ at position p of the locality-improved kernel is modified as follows.

$$k_p(x, y) = \sum_{k=-\ell}^{\ell} \gamma_{|k|} \delta(x_{p+k}, y_{p+k})$$

$$\mathrm{win}_p(x, y) = \left[k_p(x, y) + \bar{w}\left\{k_p(T_3 x, y) + k_p(x, T_3 y)\right\}\right]^{d_1}$$

Although it is claimed in [2] that the codon-improved kernel is unconditionally positive semidefinite, the fact is that the weights should be chosen carefully. For simplicity, assume $\gamma_{|k|} = 1$, $d_1 = 1$, $\ell = 3q$ and $p = 3q+1$. When x and y are the strings of length $6q + 1$ defined as follows, $\mathrm{win}_p(x, x) = \mathrm{win}_p(y, y) = 6q + 1$ and $\mathrm{win}_p(x, y) = 4q + 4\bar{w}q$ hold.

$$x = \underbrace{\texttt{ATGCGT ATGCGT}\ldots\texttt{ATGCGT}}_{6q}\texttt{A}, \quad y = \underbrace{\texttt{CTGAGT CTGAGT}\ldots\texttt{CTGAGT}}_{6q}\texttt{C}$$

Therefore, the determinant of the corresponding Gram matrix is $(1 - 2(2\bar{w} - 1)q)(1 + 2(2\bar{w} + 5)q)$, and is not always non-negative for $\bar{w} > \frac{1}{2}$.

In [3], the weighted degree kernel is modified along the same line as the codon-improved kernel except that the modified kernel includes $\bar{w}_s$ for plural s. Let $k(x, y) = \sum_{n=1}^{d} \beta_n \sum_{\boldsymbol{i} \in C_{n,L}} w_{i_1} \delta(x[\boldsymbol{i}], y[\boldsymbol{i}])$. When T_s denotes the s-shift operator, the following kernel is introduced in [3][1].

$$K(x, y) = k(x, y) + \sum_{s=1}^{S} \bar{w}_s \{k(T_s x, y) + k(x, T_s y)\} \tag{1}$$

The positive semidefiniteness of $K(x, y)$ was investigated as follows in [3]. Since $k(x, y)$ is positive semidefinite, so is $k(x, y) + k(T_s x, y) + k(x, T_s y) + k(T_s x, T_s y)$. If w_i remain constant irrespective of i[2], it follows that $2k(x, y) + k(T_s x, y) + k(x, T_s y)$ is positive semidefinite. Therefore, $\sum_{s=1}^{S} 2\bar{w}_s \leq 1$ is a sufficient condition for $K(x, y)$ to be positive semidefinite. We can relax the constraint of $w_1 = \cdots = w_L$ to $w_1 \leq w_2 \leq \cdots \leq w_L$: The key property of $2k(x, y) + k(T_s x, y) + k(x, T_s y)$ remains true, since $k(x, y) - k(T_s x, T_s y) = \sum_{n=1}^{d} \beta_n \sum_{\boldsymbol{i} \in C_{n,L}} (w_{i_1} - w_{i_1 - s}) \delta(x[\boldsymbol{i}], y[\boldsymbol{i}])$ and $w_{i_1} - w_{i_1 - s} \geq 0$ hold with $w_i = 0$ for $i \leq 0$.

[1] Although S varies according to n in [3], we assume that S is a constant just for simplicity.

[2] This seems to be assumed in [3] with no declaration.

3 Our contributions

As seen in the previous section, many existing string kernels are based on evaluation of comparison between characters/substrings of input strings. Also, to improve the performance of such character/substring-base string kernels, exploitation of the positional information of the characters/substrings is important, and, in fact, has proven to be effective at least in certain applications.

On the other hand, positive semidefiniteness of kernels, in principle, must be guaranteed, since kernel-based learning machines may not treat them properly, otherwise. In contrast to the position-aware string kernel which requires precise matching of the positions of characters/substrings (2.2), the positive semidefiniteness of the position-aware kernel with weighted shifts (2.3) is subtly affected by choice of weights, and, yet worse, only a little has been known about conditions on *good* weights.

The first contribution of this paper is to give an answer to the problem. In the remaining of this paper, without loss of generality, we assume that any strings are of length L over an alphabet $\mathcal{A}$. Given positional weights w_i, a positive shift s, a shift weight $\bar{w}_s$ and a kernel k over $\mathcal{A}$, we define $K(x, y)$ as follows.

$$K(x,y) = \sum_{i=1}^{L} w_i \left[k(x_i, y_i) + \bar{w}_s \left\{k(x_{i+s}, y_i) + k(x_i, y_{i+s})\right\}\right] \tag{2}$$

For non-negative integers a and b such that $a \in \{1, \ldots, s\}$ and $s(b-1)+a \leq L$, we define $\gamma_b^{(a)}$ by the recurrence formulas described below.

$$\gamma_0^{(a)} = 1, \quad \gamma_1^{(a)} = w_a, \quad \gamma_b^{(a)} = w_{s(b-1)+a}\gamma_{b-1}^{(a)} - \bar{w}_s^2 w_{s(b-2)+a}^2 \gamma_{b-2}^{(a)} \tag{3}$$

Then, our main theorem is stated as follows, and its proof is given in Section 5.

Theorem 1. *If $\gamma_b^{(a)} > 0$ holds for every (a, b) such that $a \in \{1, \ldots, s\}$ and $s(b-1)+a \leq L$, the character-base string kernel $K(x, y)$ defined by Eq.(2) is positive semidefinite for an arbitrary positive semidefinite kernel $k(x_i, y_j)$.*

Conversely, if $\gamma_b^{(a)} < 0$ holds for some (a, b), there exists a positive semidefinite kernel $k(x_i, y_j)$ such that the resulting $K(x, y)$ is not positive semidefinite.

The sufficient condition presented in Theorem 1 is *very close* to a necessary condition, since the positive semidefiniteness of $K(x, y)$ is left undetermined only in the marginal cases where $\gamma_b^{(a)} \geq 0$ for all (a, b) and $\gamma_b^{(a)} = 0$ for some (a, b).

On the other hand, when $w_1, \ldots, w_L$ are fixed, the condition is reduced to an equivalent inequality of $0 \leq \bar{w}_s < b_{w_1,\ldots,w_L}^{(s)}$ for some $b_{w_1,\ldots,w_L}^{(s)}$. While it is not easy to determine the actual values for $b_{w_1,\ldots,w_L}^{(s)}$, Corollary 1 gives an easily computable lower bound for $b_{w_1,\ldots,w_L}^{(s)}$.

Corollary 1. *Assume that all the weights are positive. The kernel K defined by Eq.(2) is positive semidefinite for an arbitrary positive semidefinite $k(x_i, y_j)$, if the following inequality holds for $w_1, \ldots, w_L$ and $\bar{w}_s$.*

$$\bar{w}_s \leq \min\left\{ \frac{w_i}{w_{i-s} + w_i} \middle| \, i = s+1, \ldots, L \right\} \tag{4}$$

Now, let us consider the kernel of the following form. In the same way as in the above, $k(x_i, y_j)$ is a positive semidefinite kernel over $\mathcal{A}$.

$$K(x,y) = \sum_{i=1}^{L} w_i \left[k(x_i, y_i) + \sum_{s=1}^{S} \bar{w}_s \left\{ k(x_{i+s}, y_i) + k(x_i, y_{i+s}) \right\} \right] \tag{5}$$

Let $b'^{(s)}_{w_1,\ldots,w_L}$ be positive numbers such that, if $0 \leq \bar{w}_s \leq b'^{(s)}_{w_1,\ldots,w_L}$, the kernel of Eq. (2) is positive semidefinite. If we have $\sum_{s=1}^{S} \alpha_s = 1$ such that $0 \leq \bar{w}_s \leq \alpha_s b'^{(s)}_{w_1,\ldots,w_L}$, $K_s(x,y)$ defined below is positive semidefinite, and therefore so is $K(x,y) = \sum_{s=1}^{S} K_s(x,y)$.

$$K_s(x,y) = \sum_{i=1}^{L} w_i \left[\alpha_s k(x_i, y_i) + \bar{w}_s \left\{ k(x_{i+s}, y_i) + k(x_i, y_{i+s}) \right\} \right]$$

Thus, we have obtained Theorem 2.

Theorem 2. *If the following inequality holds for $\bar{w}_s$, the character-base string kernel of Eq. (5) is positive semidefinite for an arbitrary positive semidefinite $k(x_i, y_j)$.*

$$\sum_{s=1}^{S} \frac{\bar{w}_s}{b'^{(s)}_{w_1,\ldots,w_L}} \leq 1$$

Proof. We have only to take α_s such that $\frac{\bar{w}_s}{b'^{(s)}_{w_1,\ldots,w_L}} \leq \alpha_s$ and $\sum_s^S \alpha_s = 1$. □

The sufficient condition by [3], which was also described in 2.3, is obtained as a corollary to Corollary 1 and Theorem 2.

Corollary 2. *If $w_1 \leq \cdots \leq w_L$, the character-base string kernel of Eq. (5) with $\sum_{s=1}^{S} \bar{w}_s \leq \frac{1}{2}$ is positive semidefinite.*

In Section 4, we introduce a general framework to transform character-base string kernels into not only substring-base string kernels but also kernels for richer data structures than strings. Here, with the framework, we derive from Eq. (5) two types of position-aware substring-base string kernels with weighted shifts: one is the weighted-degree kernel with shift described in 2.3 (Eq.(1) and [3]), and the other is its variation for non-contiguous substrings (Eq.(6)).

4 A framework for transforming character-base string kernels into kernels over other structured data

In this section, we present a general framework to transform given *character-base* string kernels into not only *substring-base* string kernels but also kernels for richer data structures such as trees and graphs.

We start with defining the framework in a formal manner, and then look closely at it using examples. Let $\chi, \chi', \{\chi'_x \mid x \in \chi\}, \mu$ and k' be as follows.

– χ is a space of data points.
– χ' is a space of *subparts* (*e.g.* characters, substrings, subtrees, subgraphs)
– Per each $x \in \chi$, a finite set $\chi'_x \subseteqq \chi'$ is assigned.
– $\mu : \chi' \to \mathbb{N}$ is a positioning mapping. Further, we denote $\max \mu(\chi'_x)$ by $|x|$.
– $k' : \chi' \times \chi' \to \mathbb{R}$ is a positive semidefinite kernel.

Given $(\chi, \chi', \{\chi'_x\}, \mu, k')$, we define an alphabet $\mathfrak{A}$, a mapping $\mathfrak{L} : \chi \to \mathfrak{A}^*$ and a kernel $k : \mathfrak{A} \times \mathfrak{A} \to \mathbb{R}$ as follows.

– The alphabet $\mathfrak{A}$ is the power set $\mathfrak{P}(\chi')$ of χ'.
– For $x \in \chi$, the *lift* of x, denoted by $\mathfrak{L}(x)$, is the string of length $|x|$ whose i-th character $\mathfrak{L}(x)_i$ is $\{x' \in \chi'_x \mid \mu(x') = i\} \in \mathfrak{A}$.
– A kernel $k : \mathfrak{A} \times \mathfrak{A} \to \mathbb{R}$ is defined by $k(X, Y) = \sum_{x' \in X} \sum_{y' \in Y} k'(x', y')$.

Let $K(\xi, \eta)$ be an arbitrary character-base string kernel, which includes a *character kernel* $k(\xi_i, \eta_j)$. Furthermore, we assume that $K(\xi, \eta)$ has the property that it is positive semidefinite, if so is $k(\xi_i, \eta_j)$ (as Theorem 1, Corollary 1 and Theorem 2 assert). We define $K(x, y)$ for $x, y \in \chi$ by substituting $k(\mathfrak{L}(x)_i, \mathfrak{L}(y)_j)$ for $k(\xi_i, \eta_j)$ in $K(\xi, \eta)$. Since Haussler's theorem ([7]) asserts that $k(\mathfrak{L}(x)_i, \mathfrak{L}(y)_j)$ is positive semidefinite, $K(x, y)$ remains positive semidefinite.

With this framework, the substring-base string kernels described in Section 2, namely the spectrum kernel (Sp), the string subsequence kernel (SSs), the weighted-degree kernel (WD), the weighted-degree kernel with shifts (WDwS) and the non-contiguous substring version of WDwS (Eq. (6)), are all derived from some of the character-base string kernels that we discuss in the present paper. Table 1 describes the necessary settings for the derivation. In particular, Theorem 1, Corollary 1 and Theorem 2 provide sufficient conditions on the weights for $K(x, y)$ of WDwS and Eq. (6) to be positive semidefinite.

$$\bar{w}_0 = 0,\ K(x, y) = \sum_{\boldsymbol{i} \in D_{n,L}} \sum_{\boldsymbol{j} \in D_{n,L}} \sum_{s=0}^{S} w_{\min\{i_1, j_1\}} \bar{w}_s \delta(|i_1 - j_1|, s) \delta(x[\boldsymbol{i}], y[\boldsymbol{j}]) \quad (6)$$

Furthermore, we can apply the framework to structured data other than strings. For example, let χ be a set of rooted trees, and let χ'_x denote the set of the subtrees of x. When the *depth* of a vertex v of a tree x is defined as the number of edges of the upward path from v to the root of x, we define $\mathsf{dpth}(x')$ for $x' \in \chi'_x$ as the depth of the root of x' in x. Then, with the setting of $\mu(x') = \mathsf{dpth}(x')$ and $k'(x', y') = \delta(x', y')$, we will obtain from the character-base string kernel of Eq. (5) a tree kernel that counts isomorphic subtree pairs $(x', y') \in \chi'_x \times \chi'_y$ with the weights $w_{\min\{\mathsf{dpth}(x'), \mathsf{dpth}(y')\}} \cdot \bar{w}_{|\mathsf{dpth}(x') - \mathsf{dpth}(y')|}$.

$$K(x, y) = \sum_{x' \in \chi'_x} \sum_{y' \in \chi'_y} \sum_{s=0}^{S} w_{\min\{\mathsf{dpth}(x'), \mathsf{dpth}(y')\}} \bar{w}_s \delta(|\mathsf{dpth}(x') - \mathsf{dpth}(y')|, s) \delta(x', y)$$

This tree kernel would be useful to classify web page trees, where the distance of a page from its root page has significance. Also, we can use the order $\mathsf{preodr}(x')$

Table 1. Settings for applying the framework to respective string kernels

Kernel	$\chi'_x =$	$\mu((\boldsymbol{i}, x)) =$	$k'((\boldsymbol{i}, x), (\boldsymbol{j}, y)) =$	Char.-base kernel
Sp	$C_{n,\lvert x\rvert} \times \{x\}$	i_1	$\delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	$\sum_{i=1}^{\lvert\xi\rvert} \sum_{j=1}^{\lvert\eta\rvert} k(\xi_i, \eta_j)$
SSs	$D_{n,\lvert x\rvert} \times \{x\}$	i_1	$\lambda^{i_n - i_1 + j_n - j_1} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	$\sum_{i=1}^{\lvert\xi\rvert} \sum_{j=1}^{\lvert\eta\rvert} k(\xi_i, \eta_j)$
WD	$\left(\bigcup_{n=1}^{d} C_{n,\lvert x\rvert}\right) \times \{x\}$	i_1	$\beta_{\lvert\boldsymbol{i}\rvert} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	$\sum_{i=1}^{\min\{\lvert x\rvert, \lvert y\rvert\}} k(\xi_i, \eta_i)$
WDwS	$\left(\bigcup_{n=1}^{d} C_{n,\lvert x\rvert}\right) \times \{x\}$	i_1	$\beta_{\lvert\boldsymbol{i}\rvert} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	Eq. (5)
Eq. (6)	$\left(\bigcup_{n=1}^{d} D_{n,\lvert x\rvert}\right) \times \{x\}$	i_1	$\beta_{\lvert\boldsymbol{i}\rvert} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	Eq. (5)

derived from the pre-order traversal of trees instead of the depth $\mathsf{dpth}(x')$. The resulting tree kernel would be useful to classify parse trees of natural languages, for example, where the word order in sentences has important meaning.

5 Proof of Theorem 1

Here, we will take advantage of the result of [5]. Indeed, our key lemma, namely, Lemma 1, is a degenerated corollary to Theorem 1 in [5], which gives a general sufficient condition of multivariate polynomials of arbitrary degrees such that *polynomial kernels* derived from the polynomials become positive semidefinite.

Lemma 1 ([5]). *Let $\mathcal{A}$ be an alphabet, and let x_i denote the i-th character of an L-length string $x \in \mathcal{A}^L$. For an L-dimensional real matrix $C = [c_{i,j}]_{(i,j)\in\{1,\ldots,L\}^2}$, the following are equivalent to each other.*

1. *C is positive semidefinite.*
2. *For an arbitrary positive semidefinite $k : \mathcal{A} \times \mathcal{A} \to \mathbb{R}$, the kernel that returns $\sum_{i=1}^{L} \sum_{j=1}^{L} c_{i,j} k(x_i, y_j)$ on input of $(x, y) \in \mathcal{A}^L \times \mathcal{A}^L$ is also positive semidefinite.*

When we define $c_{i,j}$ by: $c_{i,j} = w_i$, if $i = j$; $c_{i,j} = \bar{w}_s w_i$, if $j = i+s$; $c_{i,j} = \bar{w}_s w_j$, if $i = j + s$; and $c_{i,j} = 0$, otherwise. Then, $K(x, y) = \sum_{i=1}^{L} \sum_{j=1}^{L} c_{i,j} k(x_i, y_j)$ holds for $K(x, y)$ of Eq.(2). Therefore, Lemma 1 implies that, to prove the first assertion of Theorem 1, it suffices to show that the matrix $C = [c_{i,j}]_{(i,j)\in\{1,\ldots,L\}^2}$ is positive semidefinite.

We let a be one of $\{1, 2, \ldots, s\}$, and let the submatrix $C_b^{(a)}$ denote the b-dimensional matrix $[c_{s(i-1)+a, s(j-1)+a}]_{(i,j)=\{1,\ldots,b\}^2}$. For $L = sq + r$ such that $r \in \{1, 2, \ldots, s\}$, b moves in the interval $[1, q + 1]$ if $a \leq r$, and does in the interval $[1, q]$ if $a > r$. For example, when $b = 4$, $C_4^{(a)}$ looks as follows.

$$C_4^{(a)} = \begin{bmatrix} w_a & \bar{w}_s w_a & 0 & 0 \\ \bar{w}_s w_a & w_{s+a} & \bar{w}_s w_{s+a} & 0 \\ 0 & \bar{w}_s w_{s+a} & w_{2s+a} & \bar{w}_s w_{2s+a} \\ 0 & 0 & \bar{w}_s w_{2s+a} & w_{3s+a} \end{bmatrix}$$

By applying the same permutation to the rows and the columns of C if necessary, C is decomposed into a direct sum of its sub-matrices as follows.

$$C = C_{q+1}^{(1)} \oplus \cdots \oplus C_{q+1}^{(r)} \oplus C_q^{(r+1)} \oplus \cdots \oplus C_q^{(s)}$$

Therefore, C is positive definite (*i.e.* C has only positive eigenvalues), if, and only if, so are $C_{q+1}^{(a)}$ for $a \leq r$ and $C_q^{(a)}$ for $a > r$.

On the other hand, $C_{q+1}^{(a)}$ for $a \leq r$ (resp. $C_q^{(a)}$ for $a > r$) is positive definite, if, and only if, $\det(C_b^{(a)}) > 0$ for all $1 \leq b \leq q+1$ (resp. $1 \leq b \leq q$)[3]. Since, by the Laplacian determinant expansion by minors, we have the recurrence formula (7) for $\det(C_0^{(a)}) = 1$ and $\det(C_1^{(a)}) = w_a$. This indicates that $\det(C_b^{(a)})$ coincides with $\gamma_b^{(a)}$. Thus, the first assertion of Theorem 1 has been proved.

$$\det(C_b^{(a)}) = w_{(b-1)s+a}\det(C_{b-1}^{(a)}) - (w_{(b-2)s+a}\bar{w}_s)^2\det(C_{b-2}^{(a)}) \tag{7}$$

The second assertion is also derived from Lemma 1. If $\gamma_b^{(a)} < 0$ for some (a, b), $\det(C)$ is negative, and hence C is not positive semidefinite. By Lemma 1, there exists a positive semidefinite kernel $k(x_i, y_j)$ defined over the alphabet $\mathcal{A}$ such that $K(x, y)$ is not positive semidefinite.

6 Proof of Corollary 1

In this section, assuming that $\bar{w}_s$ satisfies the inequality (4), we will prove that $\gamma_b^{(a)} > 0$ holds for an arbitrary pair of non-negative integers (a, b) such that $a \in \{1, \ldots, s\}$ and $s(b-1) + a \leq L$.

To start with, we define $B_b^{(a)}$ as the matrix obtained by replacing the (b, b)-element $w_{s(b-1)+a}$ of $C_b^{(a)}$ with $\bar{w}_s w_{s(b-1)+a}$, and let $\beta_b^{(a)} = \det(B_b^{(a)})$. For example, $B_4^{(a)}$ looks as follows (compare with $C_4^{(a)}$).

$$B_4^{(a)} = \begin{bmatrix} w_a & \bar{w}_s w_a & 0 & 0 \\ \bar{w}_s w_a & w_{s+a} & \bar{w}_s w_{s+a} & 0 \\ 0 & \bar{w}_s w_{s+a} & w_{2s+a} & \bar{w}_s w_{2s+a} \\ 0 & 0 & \bar{w}_s w_{2s+a} & \bar{w}_s w_{3s+a} \end{bmatrix}$$

In the rest of this section, we fix $a \in \{1, \ldots, s\}$, and prove $\gamma_b^{(a)} > 0$ and $\beta_b^{(a)} > 0$ by induction on b. Furthermore, we can assume $b > 1$, since $\gamma_1^{(a)} = w_a > 0$ and $\beta_1^{(a)} = \bar{w}_s w_a > 0$ hold.

First, we confirm a few key properties.

– The hypothesis (4) implies $\bar{w}_s < 1$.

[3] A symmetric real matrix $[a_{i,j}]_{(i,j)\in\{1,\ldots n\}^2}$ is positive definite, if, and only if, $\det([a_{i,j}]_{(i,j)\in\{1,\ldots m\}^2}) > 0$ for all $1 \leq m \leq n$. It is easy to prove it by induction on m. Also, the reader may refer to [8] for a proof.

– Therefore, $\gamma_b^{(a)} > \beta_b^{(a)}$ follows from the hypothesis of induction $\gamma_{b-1}^{(a)} > 0$. This implies that we have only to show $\beta_b^{(a)} > 0$ to complete the proof.
– The inequality $\bar{w}_s \leq w_{s(b-1)+a}/(w_{s(b-2)+a}+w_{s(b-1)+a})$ implies the following.

$$1 - \frac{\bar{w}_s w_{s(b-2)+a}}{w_{s(b-1)+a}} \geq 1 - \bar{w}_s \left(\frac{1}{\bar{w}_s} - 1 \right) = \bar{w}_s \tag{8}$$

To show $\beta_b^{(a)} > 0$, we first expand $\beta_b^{(a)}$ and $\gamma_{b-1}^{(a)}$ by Laplacian determinant expansion, apply the inequality of (8) (note that $\gamma_{b-2}^{(a)} > 0$ holds by the hypothesis of induction), and then collect up the terms into $\beta_{b-1}^{(a)}$ by applying Laplacian determinant expansion in reverse. The assertion follows from the hypothesis of induction $\beta_{b-1}^{(a)} > 0$.

$$\begin{aligned}
\beta_b^{(a)} &= \bar{w}_s w_{s(b-1)+a} \gamma_{b-1}^{(a)} - (\bar{w}_s w_{s(b-2)+a})^2 \gamma_{b-2}^{(a)} \\
&= \bar{w}_s w_{s(b-1)+a} \left\{ \left(1 - \frac{\bar{w}_s w_{s(b-2)+a}}{w_{s(b-1)+a}} \right) w_{s(b-2)+a} \gamma_{b-2}^{(a)} - \bar{w}_s^2 w_{s(b-3)+a}^2 \gamma_{b-3}^{(a)} \right\} \\
&\geq \bar{w}_s w_{s(b-1)+a} \left(\bar{w}_s w_{s(b-2)+a} \gamma_{b-2}^{(a)} - \bar{w}_s^2 w_{s(b-3)+a}^2 \gamma_{b-3}^{(a)} \right) \\
&= \bar{w}_s w_{s(b-1)+a} \beta_{b-1}^{(a)}
\end{aligned}$$

References

1. Leslie, C., Eskin, E., Noble, W.: The spectrum kernel: a string kernel for svm protein classification. In: 7th Pacific Symposium of Biocomputing. (2002)
2. Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T., Müller, K.R.: Engineering support vector machne kernels that recognize translation initiation sites. Bioinformatics **16**(9) (2000) 799 – 807
3. Rätsch, G., Sonnenburg, S., Schölkopf, B.: Rase: recognition of alternatively spliced exons in c.elegans. Bioinformatics **21** (2005) i369 – i377
4. Lodhi, H., Shawe-Taylor, J., Cristianini, N., Watkins, C.J.C.H.: Text classificatio using string kernels. Advances in Neural Information Processing Systems **13** (2001)
5. Shin, K., Kuboyama, T.: Polynomial summaries of positive semidefinite kernels. In: The 18th International Conference on Algorithmic Learning Theory (ALT 07). (to appear)
6. Gärtner, T.: A survey of kernels for structured data. SIGKDD Explorations **5**(1) (2003) 49–58
7. Haussler, D.: Convolution kernels on discrete structures. UCSC-CRL 99-10, Dept. of Computer Science, University of California at Santa Cruz (1999)
8. Berg, C., Christensen, J.P.R., Ressel, R.: Harmonic Analysis on semigroups. Theory of positive definite and related functions. Springer (1984)