5 Kernel Methods

Kernel methods are widely used in machine learning. They are flexible techniques that can be used to extend algorithms such as SVMs to define non-linear decision boundaries. Other algorithms that only depend on inner products between sample points can be extended similarly, many of which will be studied in future chapters.

The main idea behind these methods is based on so-called *kernels* or *kernel functions*, which, under some technical conditions of symmetry and *positive-definiteness*, implicitly define an inner product in a high-dimensional space. Replacing the original inner product in the input space with positive definite kernels immediately extends algorithms such as SVMs to a linear separation in that high-dimensional space, or, equivalently, to a non-linear separation in the input space.

In this chapter, we present the main definitions and key properties of positive definite symmetric kernels, including the proof of the fact that they define an inner product in a Hilbert space, as well as their closure properties. We then extend the SVM algorithm using these kernels and present several theoretical results including general margin-based learning guarantees for hypothesis sets based on kernels. We also introduce *negative definite symmetric kernels* and point out their relevance to the construction of positive definite kernels, in particular from distances or metrics. Finally, we illustrate the design of kernels for non-vectorial discrete structures by introducing a general family of kernels for sequences, *rational kernels*. We describe an efficient algorithm for the computation of these kernels and illustrate them with several examples.

5.1 Introduction

In the previous chapter, we presented an algorithm for linear classification, SVMs, which is both effective in applications and benefits from a strong theoretical justification. In practice, linear separation is often not possible. Figure 5.1a shows an example where any hyperplane crosses both populations. However, one can use more complex functions to separate the two sets as in figure 5.1b. One way to define such a non-linear decision boundary is to use a non-linear mapping Φ from the input

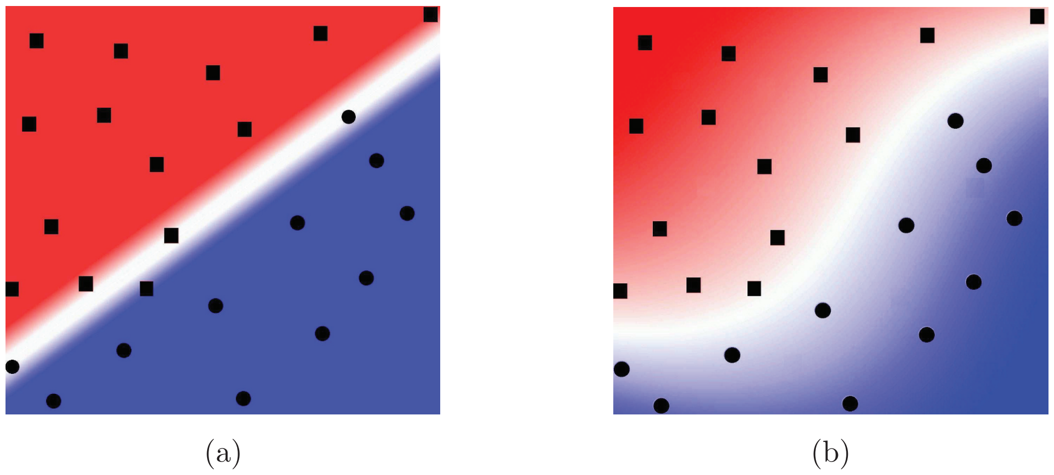

Figure 5.1 Non-linearly separable case. The classification task consists of discriminating between solid squares and solid circles. (a) No hyperplane can separate the two populations. (b) A non-linear mapping can be used instead.

space $\mathcal{X}$ to a higher-dimensional space $\mathbb{H}$, where linear separation is possible.

The dimension of $\mathbb{H}$ can truly be very large in practice. For example, in the case of document classification, one may wish to use as features sequences of three consecutive words, i.e., *trigrams*. Thus, with a vocabulary of just 100,000 words, the dimension of the feature space $\mathbb{H}$ reaches 10^{15}. On the positive side, the margin bounds presented in section 4.4 show that, remarkably, the generalization ability of large-margin classification algorithms such as SVMs do not depend on the dimension of the feature space, but only on the margin ρ and the number of training examples m. Thus, with a favorable margin ρ, such algorithms could succeed even in very high-dimensional space. However, determining the hyperplane solution requires multiple inner product computations in high-dimensional spaces, which can become be very costly.

A solution to this problem is to use *kernel methods*, which are based on *kernels* or *kernel functions*.

Definition 5.1 **Kernels**
A function $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a kernel over $\mathcal{X}$.

The idea is to define a kernel K such that for any two points $x, x' \in \mathcal{X}$, $K(x, x')$ be equal to an inner product of vectors $\Phi(x)$ and $\Phi(y)$:[1]

$$\forall x, x' \in \mathcal{X}, \quad K(x, x') = \langle \Phi(x), \Phi(x') \rangle , \tag{5.1}$$

for some mapping $\Phi\colon \mathcal{X} \to \mathbb{H}$ to a Hilbert space $\mathbb{H}$ called a *feature space*. Since an inner product is a measure of the similarity of two vectors, K is often interpreted as a similarity measure between elements of the input space $\mathcal{X}$.

An important advantage of such a kernel K is efficiency: K is often significantly more efficient to compute than Φ and an inner product in $\mathbb{H}$. We will see several common examples where the computation of $K(x, x')$ can be achieved in $O(N)$ while that of $\langle \Phi(x), \Phi(x') \rangle$ typically requires $O(\dim(\mathbb{H}))$ work, with $\dim(\mathbb{H}) \gg N$. Furthermore, in some cases, the dimension of $\mathbb{H}$ is infinite.

Perhaps an even more crucial benefit of such a kernel function K is flexibility: there is no need to explicitly define or compute a mapping Φ. The kernel K can be arbitrarily chosen so long as the existence of Φ is guaranteed, i.e. K satisfies *Mercer's condition* (see theorem 5.1).

Theorem 5.1 **Mercer's condition**
Let $\mathcal{X} \subset \mathbb{R}^N$ be a compact set and let $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a continuous and symmetric function. Then, K admits a uniformly convergent expansion of the form

$$K(x, x') = \sum_{n=0}^{\infty} a_n \phi_n(x) \phi_n(x'),$$

with $a_n > 0$ iff for any square integrable function c ($c \in L_2(\mathcal{X})$), the following condition holds:

$$\int\int_{\mathcal{X} \times \mathcal{X}} c(x) c(x') K(x, x') dx dx' \geq 0.$$

This condition is important to guarantee the convexity of the optimization problem for algorithms such as SVMs and thus convergence guarantees. A condition that is equivalent to Mercer's condition under the assumptions of the theorem is that the kernel K be *positive definite symmetric* (PDS). This property is in fact more general since in particular it does not require any assumption about $\mathcal{X}$. In the next section, we give the definition of this property and present several commonly used examples of PDS kernels, then show that PDS kernels induce an inner product in a Hilbert space, and prove several general closure properties for PDS kernels.

1. To differentiate that inner product from the one of the input space, we will typically denote it by $\langle \cdot, \cdot \rangle$.

5.2 Positive definite symmetric kernels

5.2.1 Definitions

Definition 5.2 **Positive definite symmetric kernels**
A kernel $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *is said to be* positive definite symmetric *(PDS) if for any* $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$*, the matrix* $\mathbf{K} = [K(x_i, x_j)]_{ij} \in \mathbb{R}^{m\times m}$ *is symmetric positive semidefinite (SPSD).*

$\mathbf{K}$ is SPSD if it is symmetric and one of the following two equivalent conditions holds:

- the eigenvalues of $\mathbf{K}$ are non-negative;
- for any column vector $\mathbf{c} = (c_1, \ldots, c_m)^\top \in \mathbb{R}^{m\times 1}$,

$$\mathbf{c}^\top \mathbf{K} \mathbf{c} = \sum_{i,j=1}^{n} c_i c_j K(x_i, x_j) \geq 0. \tag{5.2}$$

For a sample $S = (x_1, \ldots, x_m)$, $\mathbf{K} = [K(x_i, x_j)]_{ij} \in \mathbb{R}^{m\times m}$ is called the *kernel matrix* or the *Gram matrix* associated to K and the sample S.

Let us insist on the terminology: the kernel matrix associated to a *positive definite kernel* is *positive semidefinite* . This is the correct mathematical terminology. Nevertheless, the reader should be aware that in the context of machine learning, some authors have chosen to use instead the term *positive definite kernel* to imply a *positive definite* kernel matrix or used new terms such as *positive semidefinite kernel*.

The following are some standard examples of PDS kernels commonly used in applications.

Example 5.1 Polynomial kernels
For any constant $c > 0$, a *polynomial kernel of degree* $d \in \mathbb{N}$ is the kernel K defined over $\mathbb{R}^N$ by:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^N, \quad K(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + c)^d. \tag{5.3}$$

Polynomial kernels map the input space to a higher-dimensional space of dimension $\binom{N+d}{d}$ (see exercise 5.9). As an example, for an input space of dimension $N = 2$, a second-degree polynomial ($d = 2$) corresponds to the following inner product in

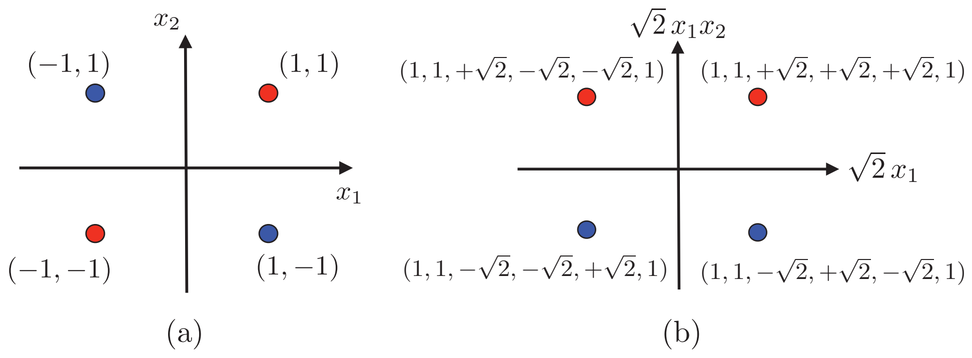
x_2
x_1
$(-1, 1)$
$(1, 1)$
$(-1, -1)$
$(1, -1)$
(a)
$\sqrt{2}\,x_1x_2$
$\sqrt{2}\,x_1$
$(1, 1, +\sqrt{2}, -\sqrt{2}, -\sqrt{2}, 1)$
$(1, 1, +\sqrt{2}, +\sqrt{2}, +\sqrt{2}, 1)$
$(1, 1, -\sqrt{2}, -\sqrt{2}, +\sqrt{2}, 1)$
$(1, 1, -\sqrt{2}, +\sqrt{2}, -\sqrt{2}, 1)$
(b)

Figure 5.2 Illustration of the XOR classification problem and the use of polynomial kernels. (a) XOR problem linearly non-separable in the input space. (b) Linearly separable using second-degree polynomial kernel.

dimension 6:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^2, \quad K(\mathbf{x}, \mathbf{x}') = (x_1x'_1 + x_2x'_2 + c)^2 = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}\,x_1x_2 \\ \sqrt{2c}\,x_1 \\ \sqrt{2c}\,x_2 \\ c \end{bmatrix} \cdot \begin{bmatrix} x'^2_1 \\ x'^2_2 \\ \sqrt{2}\,x'_1x'_2 \\ \sqrt{2c}\,x'_1 \\ \sqrt{2c}\,x'_2 \\ c \end{bmatrix}. \tag{5.4}$$

Thus, the features corresponding to a second-degree polynomial are the original features (x_1 and x_2), as well as products of these features, and the constant feature. More generally, the features associated to a polynomial kernel of degree d are all the monomials of degree at most d based on the original features. The explicit expression of polynomial kernels as inner products, as in (5.4), proves directly that they are PDS kernels.

To illustrate the application of polynomial kernels, consider the example of figure 5.2a which shows a simple data set in dimension two that is not linearly separable. This is known as the XOR problem due to its interpretation in terms of the exclusive OR (XOR) function: the label of a point is blue iff exactly one of its coordinates is 1. However, if we map these points to the six-dimensional space defined by a second-degree polynomial as described in (5.4), then the problem becomes separable by the hyperplane of equation $x_1x_2 = 0$. Figure 5.2b illustrates that by showing the projection of these points on the two-dimensional space defined by their third and fourth coordinates.

Example 5.2 Gaussian kernels

For any constant $\sigma > 0$, a *Gaussian kernel* or *radial basis function (RBF)* is the kernel K defined over $\mathbb{R}^N$ by:

$$\forall\, \mathbf{x}, \mathbf{x}' \in \mathbb{R}^N, \quad K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}' - \mathbf{x}\|^2}{2\sigma^2}\right). \tag{5.5}$$

Gaussians kernels are among the most frequently used kernels in applications. We will prove in section 5.2.3 that they are PDS kernels and that they can be derived by *normalization* from the kernels $K' \colon (\mathbf{x}, \mathbf{x}') \mapsto \exp\left(\frac{\mathbf{x}\cdot\mathbf{x}'}{\sigma^2}\right)$. Using the power series expansion of the function exponential, we can rewrite the expression of K' as follows:

$$\forall\, \mathbf{x}, \mathbf{x}' \in \mathbb{R}^N, \quad K'(\mathbf{x}, \mathbf{x}') = \sum_{n=0}^{+\infty} \frac{(\mathbf{x} \cdot \mathbf{x}')^n}{\sigma^n\, n!},$$

which shows that the kernels K', and thus Gaussian kernels, are positive linear combinations of polynomial kernels of all degrees $n \geq 0$.

Example 5.3 Sigmoid kernels
For any real constants $a, b \geq 0$, a *sigmoid kernel* is the kernel K defined over $\mathbb{R}^N$ by:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^N, \quad K(\mathbf{x}, \mathbf{x}') = \tanh\big(a(\mathbf{x} \cdot \mathbf{x}') + b\big). \tag{5.6}$$

Using sigmoid kernels with SVMs leads to an algorithm that is closely related to learning algorithms based on simple neural networks, which are also often defined via a sigmoid function. When $a < 0$ or $b < 0$, the kernel is not PDS and the corresponding neural network does not benefit from the convergence guarantees of convex optimization (see exercise 5.15).

5.2.2 Reproducing kernel Hilbert space

Here, we prove the crucial property of PDS kernels, which is to induce an inner product in a Hilbert space. The proof will make use of the following lemma.

Lemma 5.1 **Cauchy-Schwarz inequality for PDS kernels**
Let K be a PDS kernel. Then, for any $x, x' \in \mathcal{X}$,

$$K(x, x')^2 \leq K(x, x)K(x', x'). \tag{5.7}$$

Proof Consider the matrix $\mathbf{K} = \begin{pmatrix} K(x,x) & K(x,x') \\ K(x',x) & K(x',x') \end{pmatrix}$. By definition, if K is PDS, then $\mathbf{K}$ is SPSD for all $x, x' \in \mathcal{X}$. In particular, the product of the eigenvalues of $\mathbf{K}$, $\det(\mathbf{K})$, must be non-negative, thus, using $K(x', x) = K(x, x')$, we have

$$\det(\mathbf{K}) = K(x, x)K(x', x') - K(x, x')^2 \geq 0,$$

which concludes the proof. ■

The following is the main result of this section.

Theorem 5.2 **Reproducing kernel Hilbert space (RKHS)**
Let $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PDS kernel. Then, there exists a Hilbert space $\mathbb{H}$ and a mapping Φ from $\mathcal{X}$ to $\mathbb{H}$ such that:

$$\forall x, x' \in \mathcal{X}, \quad K(x, x') = \langle \Phi(x), \Phi(x') \rangle . \tag{5.8}$$

Furthermore, $\mathbb{H}$ has the following property known as the reproducing property*:*

$$\forall h \in \mathbb{H}, \forall x \in \mathcal{X}, \quad h(x) = \langle h, K(x, \cdot) \rangle . \tag{5.9}$$

$\mathbb{H}$ is called a reproducing kernel Hilbert space *(RKHS) associated to K.*

Proof For any $x \in \mathcal{X}$, define $\Phi(x)\colon \mathcal{X} \to \mathbb{R}$ as follows:

$$\forall x' \in \mathcal{X},\ \Phi(x)(x') = K(x, x').$$

We define $\mathbb{H}_0$ as the set of finite linear combinations of such functions $\Phi(x)$:

$$\mathbb{H}_0 = \Big\{ \sum_{i \in I} a_i \Phi(x_i) \colon a_i \in \mathbb{R}, x_i \in \mathcal{X}, \mathrm{card}(I) < \infty \Big\}.$$

Now, we introduce an operation $\langle \cdot, \cdot \rangle$ on $\mathbb{H}_0 \times \mathbb{H}_0$ defined for all $f, g \in \mathbb{H}_0$ with $f = \sum_{i \in I} a_i \Phi(x_i)$ and $g = \sum_{j \in J} b_j \Phi(x_j)$ by

$$\langle f, g \rangle = \sum_{i \in I, j \in J} a_i b_j K(x_i, x'_j) = \sum_{j \in J} b_j f(x'_j) = \sum_{i \in I} a_i g(x_i).$$

By definition, $\langle \cdot, \cdot \rangle$ is symmetric. The last two equations show that $\langle f, g \rangle$ does not depend on the particular representations of f and g, and also show that $\langle \cdot, \cdot \rangle$ is bilinear. Further, for any $f = \sum_{i \in I} a_i \Phi(x_i) \in \mathbb{H}_0$, since K is PDS, we have

$$\langle f, f \rangle = \sum_{i, j \in I} a_i a_j K(x_i, x_j) \geq 0.$$

Thus, $\langle \cdot, \cdot \rangle$ is positive semidefinite bilinear form. This inequality implies more generally using the bilinearity of $\langle \cdot, \cdot \rangle$ that for any $f_1, \ldots, f_m$ and $c_1, \ldots, c_m \in \mathbb{R}$,

$$\sum_{i,j=1}^{m} c_i c_j \langle f_i, f_j \rangle = \Big\langle \sum_{i=1}^{m} c_i f_i, \sum_{j=1}^{m} c_j f_j \Big\rangle \geq 0.$$

Hence, $\langle \cdot, \cdot \rangle$ is a PDS kernel on $\mathbb{H}_0$. Thus, for any $f \in \mathbb{H}_0$ and any $x \in \mathcal{X}$, by lemma 5.1, we can write

$$\langle f, \Phi(x)\rangle^2 \leq \langle f, f\rangle \langle \Phi(x), \Phi(x)\rangle.$$

Further, we observe the reproducing property of $\langle \cdot, \cdot \rangle$: for any $f = \sum_{i\in I} a_i \Phi(x_i) \in \mathbb{H}_0$, by definition of $\langle \cdot, \cdot \rangle$,

$$\forall x \in \mathcal{X}, \quad f(x) = \sum_{i\in I} a_i K(x_i, x) = \langle f, \Phi(x)\rangle\,. \tag{5.10}$$

Thus, $[f(x)]^2 \leq \langle f, f\rangle K(x, x)$ for all $x \in \mathcal{X}$, which shows the definiteness of $\langle \cdot, \cdot \rangle$. This implies that $\langle \cdot, \cdot \rangle$ defines an inner product on $\mathbb{H}_0$, which thereby becomes a pre-Hilbert space. $\mathbb{H}_0$ can be completed to form a Hilbert space $\mathbb{H}$ in which it is dense, following a standard construction. By the Cauchy-Schwarz inequality , for any $x \in \mathcal{X}$, $f \mapsto \langle f, \Phi(x)\rangle$ is Lipschitz, therefore continuous. Thus, since $\mathbb{H}_0$ is dense in $\mathbb{H}$, the reproducing property (5.10) also holds over $\mathbb{H}$. ■

The Hilbert space $\mathbb{H}$ defined in the proof of the theorem for a PDS kernel K is called *the reproducing kernel Hilbert space (RKHS) associated to K*. Any Hilbert space $\mathbb{H}$ such that there exists $\Phi\colon \mathcal{X} \to \mathbb{H}$ with $K(x, x') = \langle \Phi(x), \Phi(x')\rangle$ for all $x, x' \in \mathcal{X}$ is called a *feature space* associated to K and Φ is called a *feature mapping*. We will denote by $\|\cdot\|_{\mathbb{H}}$ the norm induced by the inner product in feature space $\mathbb{H}$: $\|\mathbf{w}\|_{\mathbb{H}} = \sqrt{\langle \mathbf{w}, \mathbf{w}\rangle}$ for all $\mathbf{w} \in \mathbb{H}$. Note that the feature spaces associated to K are in general not unique and may have different dimensions. In practice, when referring to the *dimension of the feature space* associated to K, we either refer to the dimension of the feature space based on a feature mapping described explicitly, or to that of the RKHS associated to K.

Theorem 5.2 implies that PDS kernels can be used to implicitly define a feature space or feature vectors. As already underlined in previous chapters, the role played by the features in the success of learning algorithms is crucial: with poor features, uncorrelated with the target labels, learning could become very challenging or even impossible; in contrast, good features could provide invaluable clues to the algorithm. Therefore, in the context of learning with PDS kernels and for a fixed input space, the problem of seeking useful features is replaced by that of finding useful PDS kernels. While features represented the user's prior knowledge about the task in the standard learning problems, here PDS kernels will play this role. Thus, in practice, an appropriate choice of PDS kernel for a task will be crucial.

5.2.3 Properties

This section highlights several important properties of PDS kernels. We first show that PDS kernels can be *normalized* and that the resulting normalized kernels are also PDS. We also introduce the definition of *empirical kernel maps* and describe their properties and extension. We then prove several important closure properties of PDS kernels, which can be used to construct complex PDS kernels from simpler ones.

To any kernel K, we can associate a *normalized kernel* K' defined by

$$\forall x, x' \in \mathcal{X}, \quad K'(x, x') = \begin{cases} 0 & \text{if } (K(x,x) = 0) \wedge (K(x',x') = 0) \\ \frac{K(x,x')}{\sqrt{K(x,x)K(x',x')}} & \text{otherwise.} \end{cases} \tag{5.11}$$

By definition, for a normalized kernel K', $K'(x, x) = 1$ for all $x \in \mathcal{X}$ such that $K(x, x) \neq 0$. An example of normalized kernel is the Gaussian kernel with parameter $\sigma > 0$, which is the normalized kernel associated to $K'\colon (\mathbf{x}, \mathbf{x}') \mapsto \exp\left(\frac{\mathbf{x}\cdot\mathbf{x}'}{\sigma^2}\right)$:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^N, \frac{K'(\mathbf{x}, \mathbf{x}')}{\sqrt{K'(\mathbf{x}, \mathbf{x})K'(\mathbf{x}', \mathbf{x}')}} = \frac{e^{\frac{\mathbf{x}\cdot\mathbf{x}'}{\sigma^2}}}{e^{\frac{\|\mathbf{x}\|^2}{2\sigma^2}} e^{\frac{\|\mathbf{x}'\|^2}{2\sigma^2}}} = \exp\left(-\frac{\|\mathbf{x}' - \mathbf{x}'\|^2}{2\sigma^2}\right). \tag{5.12}$$

Lemma 5.2 **Normalized PDS kernels**
Let K be a PDS kernel. Then, the normalized kernel K' associated to K is PDS.

Proof Let $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ and let $\mathbf{c}$ be an arbitrary vector in $\mathbb{R}^m$. We will show that the sum $\sum_{i,j=1}^m c_i c_j K'(x_i, x_j)$ is non-negative. By lemma 5.1, if $K(x_i, x_i) = 0$ then $K(x_i, x_j) = 0$ and thus $K'(x_i, x_j) = 0$ for all $j \in [1, m]$. Thus, we can assume that $K(x_i, x_i) > 0$ for all $i \in [1, m]$. Then, the sum can be rewritten as follows:

$$\sum_{i,j=1}^m \frac{c_i c_j K(x_i, x_j)}{\sqrt{K(x_i, x_i)K(x_j, x_j)}} = \sum_{i,j=1}^m \frac{c_i c_j \langle \Phi(x_i), \Phi(x_j) \rangle}{\|\Phi(x_i)\|_{\mathbb{H}} \|\Phi(x_j)\|_{\mathbb{H}}} = \left\| \sum_{i=1}^m \frac{c_i \Phi(x_i)}{\|\Phi(x_i)\|_{\mathbb{H}}} \right\|_{\mathbb{H}}^2 \geq 0,$$

where Φ is a feature mapping associated to K, which exists by theorem 5.2. ■

As indicated earlier, PDS kernels can be interpreted as a similarity measure since they induce an inner product in some Hilbert space $\mathbb{H}$. This is more evident for a normalized kernel K since $K(x, x')$ is then exactly the cosine of the angle between the feature vectors $\Phi(x)$ and $\Phi(x')$, provided that none of them is zero: $\Phi(x)$ and $\Phi(x')$ are then unit vectors since $\|\Phi(x)\|_{\mathbb{H}} = \|\Phi(x')\|_{\mathbb{H}} = \sqrt{K(x, x)} = 1$.

While one of the advantages of PDS kernels is an implicit definition of a feature mapping, in some instances, it may be desirable to define an explicit feature mapping based on a PDS kernel. This may be to work in the primal for various optimization and computational reasons, to derive an approximation based on an explicit mapping, or as part of a theoretical analysis where an explicit mapping is more convenient. The *empirical kernel map* Φ associated to a PDS kernel K is a feature mapping that can be used precisely in such contexts. Given a training sample containing points $x_1, \ldots, x_m \in \mathcal{X}$, $\Phi\colon \mathcal{X} \to \mathbb{R}^m$ is defined for all $x \in \mathcal{X}$ by

$$\Phi(x) = \begin{bmatrix} K(x, x_1) \\ \vdots \\ K(x, x_m) \end{bmatrix}.$$

Thus, $\Phi(x)$ is the vector of the K-similarity measures of x with each of the training points. Let $\mathbf{K}$ be the kernel matrix associated to K and $\mathbf{e}_i$ the ith unit vector. Note that for any $i \in [1, m]$, $\Phi(x_i)$ is the ith column of $\mathbf{K}$, that is $\Phi(x_i) = \mathbf{K}\mathbf{e}_i$. In particular, for all $i, j \in [1, m]$,

$$\langle \Phi(x_i), \Phi(x_j) \rangle = (\mathbf{K}\mathbf{e}_i)^\top (\mathbf{K}\mathbf{e}_j) = \mathbf{e}_i^\top \mathbf{K}^2 \mathbf{e}_j.$$

Thus, the kernel matrix $\mathbf{K}'$ associated to Φ is $\mathbf{K}^2$. It may desirable in some cases to define a feature mapping whose kernel matrix coincides with $\mathbf{K}$. Let $\mathbf{K}^{\dagger\frac{1}{2}}$ denote the SPSD matrix whose square is $\mathbf{K}^\dagger$, the pseudo-inverse of $\mathbf{K}$. $\mathbf{K}^{\dagger\frac{1}{2}}$ can be derived from $\mathbf{K}^\dagger$ via singular value decomposition and if the matrix $\mathbf{K}$ is invertible, $\mathbf{K}^{\dagger\frac{1}{2}}$ coincides with $\mathbf{K}^{-1/2}$ (see appendix A for properties of the pseudo-inverse). Then, Ψ can be defined as follows using the empirical kernel map Φ:

$$\forall x \in \mathcal{X}, \quad \Psi(x) = \mathbf{K}^{\dagger\frac{1}{2}} \Phi(x).$$

Using the identity $\mathbf{K}\mathbf{K}^\dagger\mathbf{K} = \mathbf{K}$ valid for any symmetric matrix $\mathbf{K}$, for all $i, j \in [1, m]$, the following holds:

$$\langle \Psi(x_i), \Psi(x_j) \rangle = (\mathbf{K}^{\dagger\frac{1}{2}} \mathbf{K}\mathbf{e}_i)^\top (\mathbf{K}^{\dagger\frac{1}{2}} \mathbf{K}\mathbf{e}_j) = \mathbf{e}_i^\top \mathbf{K}\mathbf{K}^\dagger\mathbf{K}\mathbf{e}_j = \mathbf{e}_i^\top \mathbf{K}\mathbf{e}_j.$$

Thus, the kernel matrix associated to Ψ is $\mathbf{K}$. Finally, note that for the feature mapping $\Omega\colon \mathcal{X} \to \mathbb{R}^m$ defined by

$$\forall x \in \mathcal{X}, \quad \Omega(x) = \mathbf{K}^\dagger \Phi(x),$$

for all $i, j \in [1, m]$, we have $\langle \Omega(x_i), \Omega(x_j) \rangle = \mathbf{e}_i^\top \mathbf{K}\mathbf{K}^\dagger\mathbf{K}^\dagger\mathbf{K}\mathbf{e}_j = \mathbf{e}_i^\top \mathbf{K}\mathbf{K}^\dagger\mathbf{e}_j$, using the identity $\mathbf{K}^\dagger\mathbf{K}^\dagger\mathbf{K} = \mathbf{K}^\dagger$ valid for any symmetric matrix $\mathbf{K}$. Thus, the kernel matrix associated to Ω is $\mathbf{K}\mathbf{K}^\dagger$, which reduces to the identity matrix $\mathbf{I} \in \mathbb{R}^{m \times m}$ when $\mathbf{K}$ is invertible, since $\mathbf{K}^\dagger = \mathbf{K}^{-1}$ in that case.

As pointed out in the previous section, kernels represent the user's prior knowledge about a task. In some cases, a user may come up with appropriate similarity measures or PDS kernels for some subtasks — for example, for different subcategories of proteins or text documents to classify. But how can he combine these PDS kernels to form a PDS kernel for the entire class? Is the resulting combined kernel guaranteed to be PDS? In the following, we will show that PDS kernels are closed under several useful operations which can be used to design complex PDS kernels. These operations are the sum and the product of kernels, as well as the *tensor product* of two kernels K and K', denoted by $K \otimes K'$ and defined by

$$\forall x_1, x_2, x'_1, x'_2 \in \mathcal{X}, \quad (K \otimes K')(x_1, x'_1, x_2, x'_2) = K(x_1, x_2)K'(x'_1, x'_2).$$

They also include the pointwise limit: given a sequence of kernels $(K_n)_{n\in\mathbb{N}}$ such that for all $x, x' \in \mathcal{X}$ $(K_n(x, x'))_{n\in\mathbb{N}}$ admits a limit, the pointwise limit of $(K_n)_{n\in\mathbb{N}}$ is the kernel K defined for all $x, x' \in \mathcal{X}$ by $K(x, x') = \lim_{n\to+\infty}(K_n)(x, x')$. Similarly, if $\sum_{n=0}^{\infty} a_n x^n$ is a power series with radius of convergence $\rho > 0$ and K a kernel taking values in $(-\rho, +\rho)$, then $\sum_{n=0}^{\infty} a_n K^n$ is the kernel obtained by composition of K with that power series. The following theorem provides closure guarantees for all of these operations.

Theorem 5.3 **PDS kernels — closure properties**
PDS kernels are closed under sum, product, tensor product, pointwise limit, and composition with a power series $\sum_{n=0}^{\infty} a_n x^n$ with $a_n \geq 0$ for all $n \in \mathbb{N}$.

Proof We start with two kernel matrices, $\mathbf{K}$ and $\mathbf{K}'$, generated from PDS kernels K and K' for an arbitrary set of m points. By assumption, these kernel matrices are SPSD. Observe that for any $\mathbf{c} \in \mathbb{R}^{m\times 1}$,

$$(\mathbf{c}^\top \mathbf{K}\mathbf{c} \geq 0) \wedge (\mathbf{c}^\top \mathbf{K}'\mathbf{c} \geq 0) \Rightarrow \mathbf{c}^\top (\mathbf{K} + \mathbf{K}')\mathbf{c} \geq 0.$$

By (5.2), this shows that $\mathbf{K} + \mathbf{K}'$ is SPSD and thus that $K + K'$ is PDS. To show closure under product, we will use the fact that for any SPSD matrix $\mathbf{K}$ there exists $\mathbf{M}$ such that $\mathbf{K} = \mathbf{M}\mathbf{M}^\top$. The existence of $\mathbf{M}$ is guaranteed as it can be generated via, for instance, singular value decomposition of $\mathbf{K}$, or by Cholesky decomposition. The kernel matrix associated to KK' is $(\mathbf{K}_{ij}\mathbf{K}'_{ij})_{ij}$. For any $\mathbf{c} \in \mathbb{R}^{m\times 1}$, expressing $\mathbf{K}_{ij}$ in terms of the entries of $\mathbf{M}$, we can write

$$\begin{aligned}
\sum_{i,j=1}^{m} c_i c_j (\mathbf{K}_{ij}\mathbf{K}'_{ij}) &= \sum_{i,j=1}^{m} c_i c_j \Big(\Big[\sum_{k=1}^{m} \mathbf{M}_{ik}\mathbf{M}_{jk} \Big] \mathbf{K}'_{ij} \Big) \\
&= \sum_{k=1}^{m} \Big[\sum_{i,j=1}^{m} c_i c_j \mathbf{M}_{ik}\mathbf{M}_{jk}\mathbf{K}'_{ij} \Big] \\
&= \sum_{k=1}^{m} \mathbf{z}_k^\top \mathbf{K}'\mathbf{z}_k \geq 0,
\end{aligned}$$

with $\mathbf{z}_k = \begin{bmatrix} c_1\mathbf{M}_{1k} \\ \vdots \\ c_m\mathbf{M}_{mk} \end{bmatrix}$. This shows that PDS kernels are closed under product. The tensor product of K and K' is PDS as the product of the two PDS kernels $(x_1, x'_1, x_2, x'_2) \mapsto K(x_1, x_2)$ and $(x_1, x'_1, x_2, x'_2) \mapsto K'(y_1, y_2)$. Next, let $(K_n)_{n\in\mathbb{N}}$ be a sequence of PDS kernels with pointwise limit K. Let $\mathbf{K}$ be the kernel matrix associated to K and $\mathbf{K}_n$ the one associated to K_n for any $n \in \mathbb{N}$. Observe that

$$(\forall n, \mathbf{c}^\top \mathbf{K}_n \mathbf{c} \geq 0) \Rightarrow \lim_{n \to \infty} \mathbf{c}^\top \mathbf{K}_n \mathbf{c} = \mathbf{c}^\top \mathbf{K} \mathbf{c} \geq 0.$$

This shows the closure under pointwise limit. Finally, assume that K is a PDS kernel with $|K(x, x')| < \rho$ for all $x, x' \in \mathcal{X}$ and let $f \colon x \mapsto \sum_{n=0}^{\infty} a_n x^n, a_n \geq 0$ be a power series with radius of convergence ρ. Then, for any $n \in \mathbb{N}$, K^n and thus $a_n K_n$ are PDS by closure under product. For any $N \in \mathbb{N}$, $\sum_{n=0}^{N} a_n K^n$ is PDS by closure under sum of $a_n K_n$s and $f \circ K$ is PDS by closure under the limit of $\sum_{n=0}^{N} a_n K^n$ as N tends to infinity. ■

The theorem implies in particular that for any PDS kernel matrix K, $\exp(K)$ is PDS, since the radius of convergence of exp is infinite. In particular, the kernel $K' \colon (\mathbf{x}, \mathbf{x}') \mapsto \exp\left(\frac{\mathbf{x} \cdot \mathbf{x}'}{\sigma^2}\right)$ is PDS since $(\mathbf{x}, \mathbf{x}') \mapsto \frac{\mathbf{x} \cdot \mathbf{x}'}{\sigma^2}$ is PDS. Thus, by lemma 5.2, this shows that a Gaussian kernel, which is the normalized kernel associated to K', is PDS.

5.3 Kernel-based algorithms

In this section we discuss how SVMs can be used with kernels and analyze the impact that kernels have on generalization.

5.3.1 SVMs with PDS kernels

In chapter 4, we noted that the dual optimization problem for SVMs as well as the form of the solution did not directly depend on the input vectors but only on inner products. Since a PDS kernel implicitly defines an inner product (theorem 5.2), we can extend SVMs and combine it with an arbitrary PDS kernel K by replacing each instance of an inner product $x \cdot x'$ with $K(x, x')$. This leads to the following general form of the SVM optimization problem and solution with PDS kernels extending (4.32):

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{5.13}$$

$$\text{subject to: } 0 \leq \alpha_i \leq C \wedge \sum_{i=1}^{m} \alpha_i y_i = 0, i \in [1, m].$$

In view of (4.33), the hypothesis h solution can be written as:

$$h(x) = \operatorname{sgn}\Big(\sum_{i=1}^{m} \alpha_i y_i K(x_i, x) + b\Big), \tag{5.14}$$

with $b = y_i - \sum_{j=1}^{m} \alpha_j y_j K(x_j, x_i)$ for any x_i with $0 < \alpha_i < C$. We can rewrite the optimization problem (5.13) in a vector form, by using the kernel matrix $\mathbf{K}$ associated to K for the training sample $(x_1, \ldots, x_m)$ as follows:

$$\max_{\boldsymbol{\alpha}} 2\, \mathbf{1}^\top \boldsymbol{\alpha} - (\boldsymbol{\alpha} \circ \mathbf{y})^\top \mathbf{K} (\boldsymbol{\alpha} \circ \mathbf{y}) \tag{5.15}$$
$$\text{subject to: } \mathbf{0} \leq \boldsymbol{\alpha} \leq \mathbf{C} \wedge \boldsymbol{\alpha}^\top \mathbf{y} = 0.$$

In this formulation, $\boldsymbol{\alpha} \circ \mathbf{y}$ is the Hadamard product or entry-wise product of the vectors $\boldsymbol{\alpha}$ and $\mathbf{y}$. Thus, it is the column vector in $\mathbb{R}^{m \times 1}$ whose ith component equals $\alpha_i y_i$. The solution in vector form is the same as in (5.14), but with $b = y_i - (\boldsymbol{\alpha} \circ \mathbf{y})^\top \mathbf{K} \mathbf{e}_i$ for any x_i with $0 < \alpha_i < C$.

This version of SVMs used with PDS kernels is the general form of SVMs we will consider in all that follows. The extension is important, since it enables an implicit non-linear mapping of the input points to a high-dimensional space where large-margin separation is sought.

Many other algorithms in areas including regression, ranking, dimensionality reduction or clustering can be extended using PDS kernels following the same scheme (see in particular chapters 8, 9, 10, 12).

5.3.2 Representer theorem

Observe that modulo the offset b, the hypothesis solution of SVMs can be written as a linear combination of the functions $K(x_i, \cdot)$, where x_i is a sample point. The following theorem known as the *representer theorem* shows that this is in fact a general property that holds for a broad class of optimization problems, including that of SVMs with no offset.

Theorem 5.4 **Representer theorem**
Let $K: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PDS kernel and $\mathbb{H}$ its corresponding RKHS. Then, for any non-decreasing function $G: \mathbb{R} \to \mathbb{R}$ and any loss function $L: \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$, the optimization problem

$$\operatorname*{argmin}_{h \in \mathbb{H}} F(h) = \operatorname*{argmin}_{h \in \mathbb{H}} G(\|h\|_{\mathbb{H}}) + L\big(h(x_1), \ldots, h(x_m)\big)$$

admits a solution of the form $h^* = \sum_{i=1}^{m} \alpha_i K(x_i, \cdot)$. If G is further assumed to be increasing, then any solution has this form.

Proof Let $\mathbb{H}_1 = \operatorname{span}(\{K(x_i, \cdot) : i \in [1, m]\})$. Any $h \in \mathbb{H}$ admits the decomposition $h = h_1 + h^\perp$ according to $\mathbb{H} = \mathbb{H}_1 \oplus \mathbb{H}_1^\perp$, where $\oplus$ is the direct sum. Since G is non-decreasing, $G(\|h_1\|_{\mathbb{H}}) \leq G(\sqrt{\|h_1\|_{\mathbb{H}}^2 + \|h^\perp\|_{\mathbb{H}}^2}) = G(\|h\|_{\mathbb{H}})$. By the reproducing property, for all $i \in [1, m]$, $h(x_i) = \langle h, K(x_i, \cdot) \rangle = \langle h_1, K(x_i, \cdot) \rangle = h_1(x_i)$. Thus, $L\big(h(x_1), \ldots, h(x_m)\big) = L\big(h_1(x_1), \ldots, h_1(x_m)\big)$ and $F(h_1) \leq F(h)$. This proves the first part of the theorem. If G is further increasing, then $F(h_1) < F(h)$ when $\|h^{\perp}\|_{\mathbb{H}} > 0$ and any solution of the optimization problem must be in $\mathbb{H}_1$. ■

5.3.3 Learning guarantees

Here, we present general learning guarantees for hypothesis sets based on PDS kernels, which hold in particular for SVMs combined with PDS kernels.

The following theorem gives a general bound on the empirical Rademacher complexity of kernel-based hypotheses with bounded norm, that is a hypothesis set of the form $H = \{h \in \mathbb{H} \colon \|h\|_{\mathbb{H}} \leq \Lambda\}$, for some $\Lambda \geq 0$, where $\mathbb{H}$ is the RKHS associated to a kernel K. By the reproducing property, any $h \in H$ is of the form $x \mapsto \langle h, K(x, \cdot)\rangle = \langle h, \Phi(x)\rangle$ with $\|h\|_{\mathbb{H}} \leq \Lambda$, where Φ is a feature mapping associated to K, that is of the form $x \mapsto \langle \mathbf{w}, \Phi(x)\rangle$ with $\|\mathbf{w}\|_{\mathbb{H}} \leq \Lambda$.

Theorem 5.5 **Rademacher complexity of kernel-based hypotheses**
Let $K \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PDS kernel and let $\Phi \colon \mathcal{X} \to \mathbb{H}$ be a feature mapping associated to K. Let $S \subseteq \{x \colon K(x,x) \leq r^2\}$ be a sample of size m, and let $H = \{\mathbf{x} \mapsto \mathbf{w} \cdot \Phi(x) \colon \|\mathbf{w}\|_{\mathbb{H}} \leq \Lambda\}$ for some $\Lambda \geq 0$. Then

$$\widehat{\mathfrak{R}}_S(H) \leq \frac{\Lambda\sqrt{\mathrm{Tr}[\mathbf{K}]}}{m} \leq \sqrt{\frac{r^2\Lambda^2}{m}}. \tag{5.16}$$

Proof The proof steps are as follows:

$$\begin{aligned}
\widehat{\mathfrak{R}}_S(H) &= \frac{1}{m} \mathop{\mathrm{E}}_{\boldsymbol{\sigma}}\left[\sup_{\|\mathbf{w}\| \leq \Lambda} \left\langle \mathbf{w}, \sum_{i=1}^{m} \sigma_i \Phi(x_i)\right\rangle\right] \\
&= \frac{\Lambda}{m} \mathop{\mathrm{E}}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^{m} \sigma_i \Phi(x_i)\right\|_{\mathbb{H}}\right] && \text{(Cauchy-Schwarz , eq. case)} \\
&\leq \frac{\Lambda}{m}\left[\mathop{\mathrm{E}}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^{m} \sigma_i \Phi(x_i)\right\|_{\mathbb{H}}^2\right]\right]^{1/2} && \text{(Jensen's ineq.)} \\
&= \frac{\Lambda}{m}\left[\mathop{\mathrm{E}}_{\boldsymbol{\sigma}}\left[\sum_{i=1}^{m} \|\Phi(x_i)\|_{\mathbb{H}}^2\right]\right]^{1/2} && (i \neq j \Rightarrow \mathop{\mathrm{E}}_{\boldsymbol{\sigma}}[\sigma_i\sigma_j] = 0) \\
&= \frac{\Lambda}{m}\left[\mathop{\mathrm{E}}_{\boldsymbol{\sigma}}\left[\sum_{i=1}^{m} K(x_i, x_i)\right]\right]^{1/2} \\
&= \frac{\Lambda\sqrt{\mathrm{Tr}[\mathbf{K}]}}{m} \leq \sqrt{\frac{r^2\Lambda^2}{m}}.
\end{aligned}$$

The initial equality holds by definition of the empirical Rademacher complexity (definition 3.2). The first inequality is due to the Cauchy-Schwarz inequality and $\|\mathbf{w}\|_{\mathbb{H}} \leq \Lambda$. The following inequality results from Jensen's inequality (theorem B.4) applied to the concave function $\sqrt{\cdot}$. The subsequent equality is a consequence of

$\mathrm{E}_{\boldsymbol{\sigma}}[\sigma_i\sigma_j] = \mathrm{E}_{\boldsymbol{\sigma}}[\sigma_i]\,\mathrm{E}_{\boldsymbol{\sigma}}[\sigma_j] = 0$ for $i \neq j$, since the Rademacher variables σ_i and σ_j are independent. The statement of the theorem then follows by noting that $\mathrm{Tr}[\mathbf{K}] \leq mr^2$. ■

The theorem indicates that the trace of the kernel matrix is an important quantity for controlling the complexity of hypothesis sets based on kernels. Observe that by the Khintchine-Kahane inequality (D.22), the empirical Rademacher complexity $\widehat{\mathfrak{R}}_S(H) = \frac{\Lambda}{m}\,\mathrm{E}_{\boldsymbol{\sigma}}[\|\sum_{i=1}^m \sigma_i \Phi(x_i)\|_{\mathbb{H}}]$ can also be lower bounded by $\frac{1}{\sqrt{2}}\frac{\Lambda\sqrt{\mathrm{Tr}[\mathbf{K}]}}{m}$, which only differs from the upper bound found by the constant $\frac{1}{\sqrt{2}}$. Also, note that if $K(x,x) \leq r^2$ for all $x \in \mathcal{X}$, then the inequalities 5.16 hold for all samples S.

The bound of theorem 5.5 or the inequalities 5.16 can be plugged into any of the Rademacher complexity generalization bounds presented in the previous chapters. In particular, in combination with theorem 4.4, they lead directly to the following margin bound similar to that of corollary 4.1.

Corollary 5.1 **Margin bounds for kernel-based hypotheses**
Let $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PDS kernel with $r = \sup_{x\in\mathcal{X}} K(x,x)$. Let $\Phi\colon \mathcal{X} \to \mathbb{H}$ be a feature mapping associated to K and let $H = \{\mathbf{x} \mapsto \mathbf{w}\cdot\Phi(x)\colon \|\mathbf{w}\|_{\mathbb{H}} \leq \Lambda\}$ for some $\Lambda \geq 0$. Fix $\rho > 0$. Then, for any $\delta > 0$, each of the following statements holds with probability at least $1-\delta$ for any $h \in H$:

$$R(h) \leq \widehat{R}_\rho(h) + 2\sqrt{\frac{r^2\Lambda^2/\rho^2}{m}} + \sqrt{\frac{\log\frac{1}{\delta}}{2m}} \tag{5.17}$$

$$R(h) \leq \widehat{R}_\rho(h) + 2\frac{\sqrt{\mathrm{Tr}[\mathbf{K}]\Lambda^2/\rho^2}}{m} + 3\sqrt{\frac{\log\frac{2}{\delta}}{2m}}\,. \tag{5.18}$$

5.4 Negative definite symmetric kernels

Often in practice, a natural distance or metric is available for the learning task considered. This metric could be used to define a similarity measure. As an example, Gaussian kernels have the form $\exp(-d^2)$, where d is a metric for the input vector space. Several natural questions arise such as: what other PDS kernels can we construct from a metric in a Hilbert space? What technical condition should d satisfy to guarantee that $\exp(-d^2)$ is PDS? A natural mathematical definition that helps address these questions is that of *negative definite symmetric (NDS) kernels*.

Definition 5.3 **Negative definite symmetric (NDS) kernels**
A kernel $K\colon \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ is said to be negative-definite symmetric (NDS) *if it is symmetric and if for all $\{x_1,\ldots,x_m\} \subseteq \mathcal{X}$ and $\mathbf{c} \in \mathbb{R}^{m\times 1}$ with $\mathbf{1}^\top\mathbf{c} = 0$, the*

following holds:

$$\mathbf{c}^\top \mathbf{K}\mathbf{c} \leq 0.$$

Clearly, if K is PDS, then $-K$ is NDS, but the converse does not hold in general. The following gives a standard example of an NDS kernel.

Example 5.4 Squared distance — NDS kernel
The squared distance $(x, x') \mapsto \|x' - x\|^2$ in $\mathbb{R}^N$ defines an NDS kernel. Indeed, let $\mathbf{c} \in \mathbb{R}^{m\times 1}$ with $\sum_{i=1}^m c_i = 0$. Then, for any $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$, we can write

$$\begin{aligned}
\sum_{i,j=1}^m c_i c_j ||\mathbf{x}_i - \mathbf{x}_j||^2 &= \sum_{i,j=1}^m c_i c_j (\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2\mathbf{x}_i \cdot \mathbf{x}_j)\\
&= \sum_{i,j=1}^m c_i c_j (\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2) - 2\sum_{i=1}^m c_i \mathbf{x}_i \cdot \sum_{j=1}^m c_j \mathbf{x}_j\\
&= \sum_{i,j=1}^m c_i c_j (\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2) - 2\Big\|\sum_{i=1}^m c_i \mathbf{x}_i\Big\|^2\\
&\leq \sum_{i,j=1}^m c_i c_j (\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2)\\
&= \Big(\sum_{j=1}^m c_j\Big)\Big(\sum_{i=1}^m c_i (\|\mathbf{x}_i\|^2\Big) + \Big(\sum_{i=1}^m c_i\Big)\Big(\sum_{j=1}^m c_j \|\mathbf{x}_j\|^2\Big) = 0.
\end{aligned}$$

The next theorems show connections between NDS and PDS kernels. These results provide another series of tools for designing PDS kernels.

Theorem 5.6
Let K' be defined for any x_0 by

$$K'(x, x') = K(x, x_0) + K(x', x_0) - K(x, x') - K(x_0, x_0)$$

for all $x, x' \in \mathcal{X}$. Then K is NDS iff K' is PDS.

Proof Assume that K' is PDS and define K such that for any x_0 we have $K(x, x') = K(x, x_0) + K(x_0, x') - K(x_0, x_0) - K'(x, x')$. Then for any $\mathbf{c} \in \mathbb{R}^m$ such that $\mathbf{c}^\top \mathbf{1} = 0$ and any set of points $(x_1, \ldots, x_m) \in \mathcal{X}^m$ we have

$$\begin{aligned}
\sum_{i,j=1}^m c_i c_j K(x_i, x_j) &= \Big(\sum_{i=1}^m c_i K(x_i, x_0)\Big)\Big(\sum_{j=1}^m c_j\Big) + \Big(\sum_{i=1}^m c_i\Big)\Big(\sum_{j=1}^m c_j K(x_0, x_j)\Big)\\
&\quad - \Big(\sum_{i=1}^m c_i\Big)^2 K(x_0, x_0) - \sum_{i,j=1}^m c_i c_j K'(x_i, x_j) = -\sum_{i,j=1}^m c_i c_j K'(x_i, x_j) \leq 0\,.
\end{aligned}$$

which proves K is NDS.

Now, assume K is NDS and define K' for any x_0 as above. Then, for any $\mathbf{c} \in \mathbb{R}^m$, we can define $c_0 = -\mathbf{c}^\top \mathbf{1}$ and the following holds by the NDS property for any points $(x_1, \ldots, x_m) \in \mathcal{X}^m$ as well as x_0 defined previously: $\sum_{i,j=0}^m c_i c_j K(x_i, x_j) \leq 0$. This implies that

$$\Big(\sum_{i=0}^m c_i K(x_i, x_0)\Big)\Big(\sum_{j=0}^m c_j\Big) + \Big(\sum_{i=0}^m c_i\Big)\Big(\sum_{j=0}^m c_j K(x_0, x_j)\Big)$$
$$- \Big(\sum_{i=0}^m c_i\Big)^2 K(x_0, x_0) - \sum_{i,j=0}^m c_i c_j K'(x_i, x_j) = - \sum_{i,j=0}^m c_i c_j K'(x_i, x_j) \leq 0\,,$$

which implies $2\sum_{i,j=1}^m c_i c_j K'(x_i, x_j) \geq -2c_0 \sum_{i=0}^m c_i K'(x_i, x_0) + c_0^2 K'(x_0, x_0) = 0$. The equality holds since $\forall x \in \mathcal{X}, K'(x, x_0) = 0$. ■

This theorem is useful in showing other connections, such the following theorems, which are left as exercises (see exercises 5.14 and 5.15).

Theorem 5.7
Let $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric kernel. Then, K is NDS iff $\exp(-tK)$ is a PDS kernel for all $t > 0$.

The theorem provides another proof that Gaussian kernels are PDS: as seen earlier (Example 5.4), the squared distance $(x, x') \mapsto \|x - x'\|^2$ in $\mathbb{R}^N$ is NDS, thus $(x, x') \mapsto \exp(-t||x - x'||^2)$ is PDS for all $t > 0$.

Theorem 5.8
Let $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be an NDS kernel such that for all $x, x' \in \mathcal{X}, K(x, x') = 0$ iff $x = x'$. Then, there exists a Hilbert space $\mathbb{H}$ and a mapping $\Phi\colon \mathcal{X} \to \mathbb{H}$ such that for all $x, x' \in \mathcal{X}$,

$$K(x, x') = \|\Phi(x) - \Phi(x')\|^2.$$

Thus, under the hypothesis of the theorem, $\sqrt{K}$ defines a metric.

This theorem can be used to show that the kernel $(x, x') \mapsto \exp(-|x - x'|^p)$ in $\mathbb{R}$ is not PDS for $p > 2$. Otherwise, for any $t > 0$, $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ and $\mathbf{c} \in \mathbb{R}^{m\times 1}$, we would have:

$$\sum_{i,j=1}^m c_i c_j e^{-t|x_i - x_j|^p} = \sum_{i,j=1}^m c_i c_j e^{-|t^{1/p}x_i - t^{1/p}x_j|^p} \geq 0.$$

This would imply that $(x, x') \mapsto |x - x'|^p$ is NDS for $p > 2$, which can be proven (via theorem 5.8) not to be valid.

5.5 Sequence kernels

The examples given in the previous sections, including the commonly used polynomial or Gaussian kernels, were all for PDS kernels over vector spaces. In many learning tasks found in practice, the input space $\mathcal{X}$ is not a vector space. The examples to classify in practice could be protein sequences, images, graphs, parse trees, finite automata, or other discrete structures which may not be directly given as vectors. PDS kernels provide a method for extending algorithms such as SVMs originally designed for a vectorial space to the classification of such objects. But, how can we define PDS kernels for these structures?

This section will focus on the specific case of *sequence kernels*, that is, kernels for sequences or strings. PDS kernels can be defined for other discrete structures in somewhat similar ways. Sequence kernels are particularly relevant to learning algorithms applied to computational biology or natural language processing, which are both important applications.

How can we define PDS kernels for sequences, which are similarity measures for sequences? One idea consists of declaring two sequences, e.g., two documents or two biosequences, as similar when they share common substrings or subsequences. One example could be the kernel between two sequences defined by the sum of the product of the counts of their common substrings. But which substrings should be used in that definition? Most likely, we would need some flexibility in the definition of the matching substrings. For computational biology applications, for example, the match could be imperfect. Thus, we may need to consider some number of mismatches, possibly gaps, or wildcards. More generally, we might need to allow various substitutions and might wish to assign different weights to common substrings to emphasize some matching substrings and deemphasize others.

As can be seen from this discussion, there are many different possibilities and we need a general framework for defining such kernels. In the following, we will introduce a general framework for sequence kernels, *rational kernels*, which will include all the kernels considered in this discussion. We will also describe a general and efficient algorithm for their computation and will illustrate them with some examples.

The definition of these kernels relies on that of *weighted transducers*. Thus, we start with the definition of these devices as well as some relevant algorithms.

5.5.1 Weighted transducers

Sequence kernels can be effectively represented and computed using *weighted transducers*. In the following definition, let Σ denote a finite input alphabet, Δ a finite output alphabet, and ϵ the *empty string* or null label, whose concatenation with

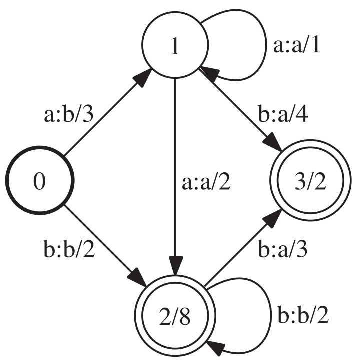

Figure 5.3 Example of weighted transducer.

any string leaves it unchanged.

Definition 5.4
A weighted transducer *T is a 7-tuple $T = (\Sigma, \Delta, Q, I, F, E, \rho)$ where Σ is a finite input alphabet, Δ a finite output alphabet, Q is a finite set of states, $I \subseteq Q$ the set of initial states, $F \subseteq Q$ the set of final states, E a finite multiset of transitions elements of $Q \times (\Sigma \cup \{\epsilon\}) \times (\Delta \cup \{\epsilon\}) \times \mathbb{R} \times Q$, and $\rho : F \to \mathbb{R}$ a final weight function mapping F to $\mathbb{R}$. The* size *of transducer T is the sum of its number of states and transitions and is denoted by $|T|$.*[2]

Thus, weighted transducers are finite automata in which each transition is labeled with both an input and an output label and carries some real-valued weight. Figure 5.3 shows an example of a weighted finite-state transducer. In this figure, the input and output labels of a transition are separated by a colon delimiter, and the weight is indicated after the slash separator. The initial states are represented by a bold circle and final states by double circles. The final weight $\rho[q]$ at a final state q is displayed after the slash.

The input label of a path π is a string element of Σ^* obtained by concatenating input labels along π. Similarly, the output label of a path π is obtained by concatenating output labels along π. A path from an initial state to a final state is an *accepting path*. The weight of an accepting path is obtained by multiplying the weights of its constituent transitions and the weight of the final state of the path.

A weighted transducer defines a mapping from $\Sigma^* \times \Delta^*$ to $\mathbb{R}$. The weight associated by a weighted transducer T to a pair of strings $(x, y) \in \Sigma^* \times \Delta^*$ is denoted by $T(x, y)$ and is obtained by summing the weights of all accepting paths

2. A multiset in the definition of the transitions is used to allow for the presence of several transitions from a state p to a state q with the same input and output label, and even the same weight, which may occur as a result of various operations.

with input label x and output label y. For example, the transducer of figure 5.3 associates to the pair (aab, baa) the weight $3 \times 1 \times 4 \times 2 + 3 \times 2 \times 3 \times 2$, since there is a path with input label aab and output label baa and weight $3 \times 1 \times 4 \times 2$, and another one with weight $3 \times 2 \times 3 \times 2$.

The sum of the weights of all accepting paths of an acyclic transducer, that is a transducer T with no cycle, can be computed in linear time, that is $O(|T|)$, using a general *shortest-distance* or forward-backward algorithm. These are simple algorithms, but a detailed description would require too much of a digression from the main topic of this chapter.

Composition An important operation for weighted transducers is *composition*, which can be used to combine two or more weighted transducers to form more complex weighted transducers. As we shall see, this operation is useful for the creation and computation of sequence kernels. Its definition follows that of composition of relations. Given two weighted transducers $T_1 = (\Sigma, \Delta, Q_1, I_1, F_1, E_1, \rho_1)$ and $T_2 = (\Delta, \Omega, Q_2, I_2, F_2, E_2, \rho_2)$, the result of the composition of T_1 and T_2 is a weighted transducer denoted by $T_1 \circ T_2$ and defined for all $x \in \Sigma^*$ and $y \in \Omega^*$ by

$$(T_1 \circ T_2)(x, y) = \sum_{z \in \Delta^*} T_1(x, z) \cdot T_2(z, y), \tag{5.19}$$

where the sum runs over all strings z over the alphabet Δ. Thus, composition is similar to matrix multiplication with infinite matrices.

There exists a general and efficient algorithm to compute the composition of two weighted transducers. In the absence of ϵs on the input side of T_1 or the output side of T_2, the states of $T_1 \circ T_2 = (\Sigma, \Delta, Q, I, F, E, \rho)$ can be identified with pairs made of a state of T_1 and a state of T_2, $Q \subseteq Q_1 \times Q_2$. Initial states are those obtained by pairing initial states of the original transducers, $I = I_1 \times I_2$, and similarly final states are defined by $F = Q \cap (F_1 \times F_2)$. The final weight at a state $(q_1, q_2) \in F_1 \times F_2$ is $\rho(q) = \rho_1(q_1)\rho_2(q_2)$, that is the product of the final weights at q_1 and q_2. Transitions are obtained by matching a transition of T_1 with one of T_2 from appropriate transitions of T_1 and T_2:

$$E = \biguplus_{\substack{(q_1, a, b, w_1, q_2) \in E_1 \\ (q_1', b, c, w_2, q_2') \in E_2}} \left\{ \Big((q_1, q_1'), a, c, w_1 \otimes w_2, (q_2, q_2') \Big) \right\}.$$

Here, $\uplus$ denotes the standard join operation of multisets as in $\{1, 2\} \uplus \{1, 3\} = \{1, 1, 2, 3\}$, to preserve the multiplicity of the transitions.

In the worst case, all transitions of T_1 leaving a state q_1 match all those of T_2 leaving state q_1', thus the space and time complexity of composition is quadratic: $O(|T_1||T_2|)$. In practice, such cases are rare and composition is very efficient. Figure 5.4 illustrates the algorithm in a particular case.

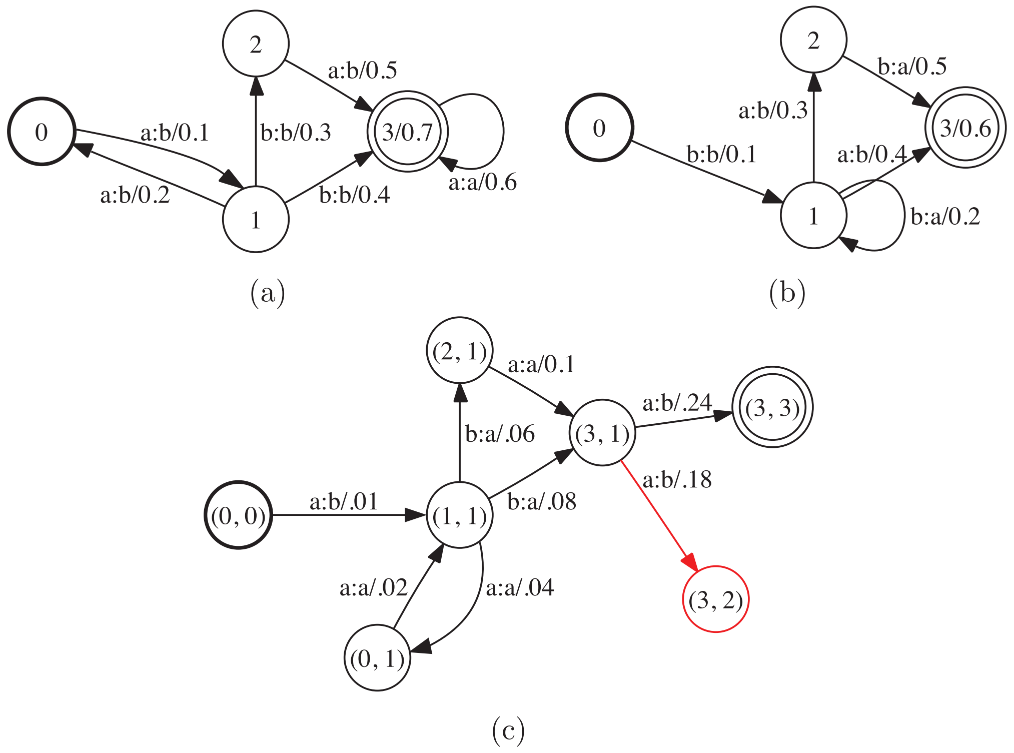

Figure 5.4 (a) Weighted transducer T_1. (b) Weighted transducer T_2. (c) Result of composition of T_1 and T_2, $T_1 \circ T_2$. Some states might be constructed during the execution of the algorithm that are not *co-accessible*, that is, they do not admit a path to a final state, e.g., $(3, 2)$. Such states and the related transitions (in red) can be removed by a trimming (or connection) algorithm in linear time.

As illustrated by figure 5.5, when T_1 admits output ϵ labels or T_2 input ϵ labels, the algorithm just described may create redundant ϵ-paths, which would lead to an incorrect result. The weight of the matching paths of the original transducers would be counted p times, where p is the number of redundant paths in the result of composition. To avoid with this problem, all but one ϵ-path must be filtered out of the composite transducer. Figure 5.5 indicates in boldface one possible choice for that path, which in this case is the shortest. Remarkably, that filtering mechanism itself can be encoded as a finite-state transducer F (figure 5.5b).

To apply that filter, we need to first augment T_1 and T_2 with auxiliary symbols that make the semantics of ϵ explicit: let $\tilde{T_1}$ ($\tilde{T_2}$) be the weighted transducer obtained from T_1 (respectively T_2) by replacing the output (respectively input) ϵ labels with ϵ_2 (respectively ϵ_1) as illustrated by figure 5.5. Thus, matching with the symbol ϵ_1 corresponds to remaining at the same state of T_1 and taking a transition of T_2 with input ϵ. ϵ_2 can be described in a symmetric way. The filter transducer F disallows a matching (ϵ_2, ϵ_2) immediately after (ϵ_1, ϵ_1) since this can be done instead via (ϵ_2, ϵ_1).

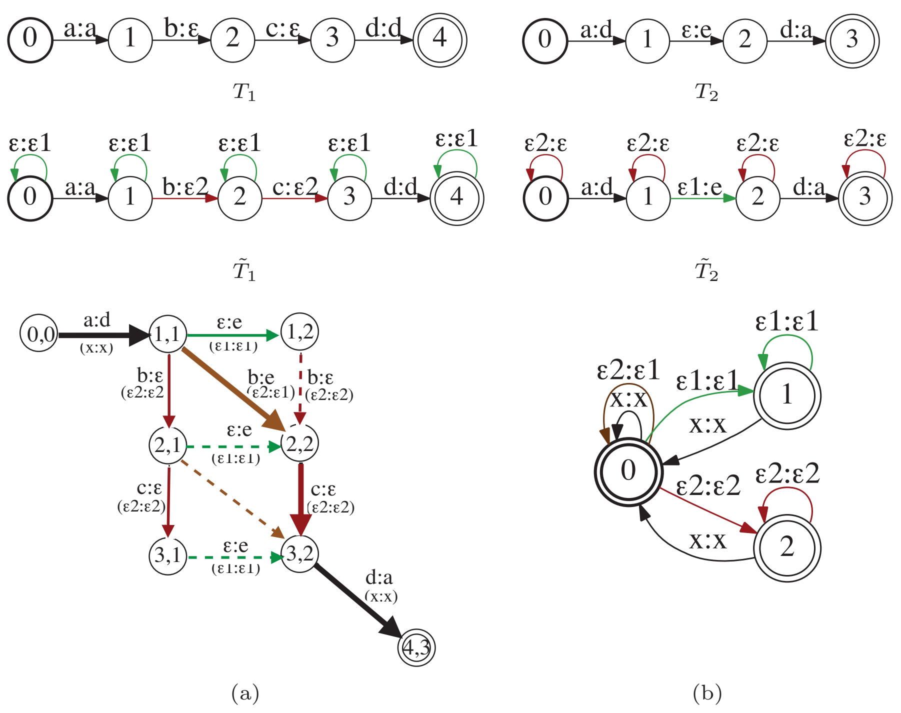

Figure 5.5 Redundant ϵ-paths in composition. All transition and final weights are equal to one. (a) A straightforward generalization of the ϵ-free case would generate all the paths from $(1,1)$ to $(3,2)$ when composing T_1 and T_2 and produce an incorrect results in non-idempotent semirings. (b) Filter transducer F. The shorthand x is used to represent an element of Σ.

By symmetry, it also disallows a matching (ϵ_1, ϵ_1) immediately after (ϵ_2, ϵ_2). In the same way, a matching (ϵ_1, ϵ_1) immediately followed by (ϵ_2, ϵ_1) is not permitted by the filter F since a path via the matchings $(\epsilon_2, \epsilon_1)(\epsilon_1, \epsilon_1)$ is possible. Similarly, $(\epsilon_2, \epsilon_2)(\epsilon_2, \epsilon_1)$ is ruled out. It is not hard to verify that the filter transducer F is precisely a finite automaton over pairs accepting the complement of the language

$$L = \sigma^*((\epsilon_1, \epsilon_1)(\epsilon_2, \epsilon_2) + (\epsilon_2, \epsilon_2)(\epsilon_1, \epsilon_1) + (\epsilon_1, \epsilon_1)(\epsilon_2, \epsilon_1) + (\epsilon_2, \epsilon_2)(\epsilon_2, \epsilon_1))\sigma^*,$$

where $\sigma = \{(\epsilon_1, \epsilon_1), (\epsilon_2, \epsilon_2), (\epsilon_2, \epsilon_1), x\}$. Thus, the filter F guarantees that exactly one ϵ-path is allowed in the composition of each ϵ sequences. To obtain the correct result of composition, it suffices then to use the ϵ-free composition algorithm already described and compute

$$\tilde{T}_1 \circ F \circ \tilde{T}_2. \tag{5.20}$$

Indeed, the two compositions in $\tilde{T}_1 \circ F \circ \tilde{T}_2$ no longer involve ϵs. Since the size of the filter transducer F is constant, the complexity of general composition is the same as that of ϵ-free composition, that is $O(|T_1||T_2|)$. In practice, the augmented transducers $\tilde{T}_1$ and $\tilde{T}_2$ are not explicitly constructed, instead the presence of the auxiliary symbols is simulated. Further filter optimizations help limit the number of non-coaccessible states created, for example, by examining more carefully the case of states with only outgoing non-ϵ-transitions or only outgoing ϵ-transitions.

5.5.2 Rational kernels

The following establishes a general framework for the definition of sequence kernels.

Definition 5.5 **Rational kernels**
A kernel $K\colon \Sigma^* \times \Sigma^* \to \mathbb{R}$ is said to be rational *if it coincides with the mapping defined by some weighted transducer U: $\forall x, y \in \Sigma^*, K(x, y) = U(x, y)$.*

Note that we could have instead adopted a more general definition: instead of using weighted transducers, we could have used more powerful sequence mappings such as *algebraic transductions*, which are the functional counterparts of context-free languages, or even more powerful ones. However, an essential need for kernels is an efficient computation, and more complex definitions would lead to substantially more costly computational complexities for kernel computation. For rational kernels, there exists a general and efficient computation algorithm.

Computation We will assume that the transducer U defining a rational kernel K does not admit any ϵ-cycle with non-zero weight, otherwise the kernel value is infinite for all pairs. For any sequence x, let T_x denote a weighted transducer with just one accepting path whose input and output labels are both x and its weight equal to one. T_x can be straightforwardly constructed from x in linear time $O(|x|)$. Then, for any $x, y \in \Sigma^*$, $U(x, y)$ can be computed by the following two steps:

1. Compute $V = T_x \circ U \circ T_y$ using the composition algorithm in time $O(|U||T_x||T_y|)$.
2. Compute the sum of the weights of all accepting paths of V using a general shortest-distance algorithm in time $O(|V|)$.

By definition of composition, V is a weighted transducer whose accepting paths are precisely those accepting paths of U that have input label x and output label y. The second step computes the sum of the weights of these paths, that is, exactly $U(x, y)$. Since U admits no ϵ-cycle, V is acyclic, and this step can be performed in linear time. The overall complexity of the algorithm for computing $U(x, y)$ is then in $O(|U||T_x||T_y|)$. Since U is fixed for a rational kernel K and $|T_x| = O(|x|)$ for any x, this shows that the kernel values can be obtained in quadratic time $O(|x||y|)$. For some specific weighted transducers U, the computation can be more efficient, for example in $O(|x| + |y|)$ (see exercise 5.17).

PDS rational kernels For any transducer T, let T^{-1} denote the *inverse* of T, that is the transducer obtained from T by swapping the input and output labels of every transition. For all x, y, we have $T^{-1}(x, y) = T(y, x)$. The following theorem gives a general method for constructing a PDS rational kernel from an arbitrary weighted transducer.

Theorem 5.9
For any weighted transducer $T = (\Sigma, \Delta, Q, I, F, E, \rho)$, the function $K = T \circ T^{-1}$ is a PDS rational kernel.

Proof By definition of composition and the inverse operation, for all $x, y \in \Sigma^*$,

$$K(x, y) = \sum_{z \in \Delta^*} T(x, z)\, T(y, z).$$

K is the pointwise limit of the kernel sequence $(K_n)_{n \geq 0}$ defined by:

$$\forall n \in \mathbb{N}, \forall x, y \in \Sigma^*, \quad K_n(x, y) = \sum_{|z| \leq n} T(x, z)\, T(y, z),$$

where the sum runs over all sequences in Δ^* of length at most n. K_n is PDS since its corresponding kernel matrix $\mathbf{K}_n$ for any sample $(x_1, \ldots, x_m)$ is SPSD. This can be see form the fact that $\mathbf{K}_n$ can be written as $\mathbf{K}_n = \mathbf{A}\mathbf{A}^\top$ with $\mathbf{A} = (K_n(x_i, z_j))_{i \in [1,m], j \in [1,N]}$, where $z_1, \ldots, z_N$ is some arbitrary enumeration of the set of strings in Σ^* with length at most n. Thus, K is PDS as the pointwise limit of the sequence of PDS kernels $(K_n)_{n \in \mathbb{N}}$. ■

The sequence kernels commonly used in computational biology, natural language processing, computer vision, and other applications are all special instances of rational kernels of the form $T \circ T^{-1}$. All of these kernels can be computed efficiently using the same general algorithm for the computational of rational kernels presented in the previous paragraph. Since the transducer $U = T \circ T^{-1}$ defining such PDS rational kernels has a specific form, there are different options for the computation of the composition $T_x \circ U \circ T_y$:

- compute $U = T \circ T^{-1}$ first, then $V = T_x \circ U \circ T_y$;
- compute $V_1 = T_x \circ T$ and $V_2 = T_y \circ T$ first, then $V = V_1 \circ V_2^{-1}$;
- compute first $V_1 = T_x \circ T$, then $V_2 = V_1 \circ T^{-1}$, then $V = V_2 \circ T_y$, or the similar series of operations with x and y permuted.

All of these methods lead to the same result after computation of the sum of the weights of all accepting paths, and they all have the same worst-case complexity. However, in practice, due to the sparsity of intermediate compositions, there may be substantial differences between their time and space computational costs. An

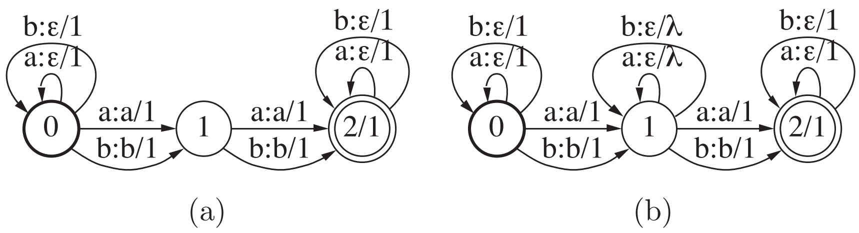

Figure 5.6 (a) Transducer T_{bigram} defining the bigram kernel $T_{\text{bigram}} \circ T_{\text{bigram}}^{-1}$ for $\Sigma = \{a, b\}$. (b) Transducer $T_{\text{gappy\_bigram}}$ defining the gappy bigram kernel $T_{\text{gappy\_bigram}} \circ T_{\text{gappy\_bigram}}^{-1}$ with gap penalty $\lambda \in (0, 1)$.

alternative method based on an *n-way composition* can further lead to significantly more efficient computations.

Example 5.5 Bigram and gappy bigram sequence kernels
Figure 5.6a shows a weighted transducer T_{bigram} defining a common sequence kernel, the *bigram sequence kernel*, for the specific case of an alphabet reduced to $\Sigma = \{a, b\}$. The bigram kernel associates to any two sequences x and y the sum of the product of the counts of all bigrams in x and y. For any sequence $x \in \Sigma^*$ and any bigram $z \in \{aa, ab, ba, bb\}$, $T_{\text{bigram}}(x, z)$ is exactly the number of occurrences of the bigram z in x. Thus, by definition of composition and the inverse operation, $T_{\text{bigram}} \circ T_{\text{bigram}}^{-1}$ computes exactly the bigram kernel.

Figure 5.6b shows a weighted transducer $T_{\text{gappy\_bigram}}$ defining the so-called *gappy bigram kernel*. The gappy bigram kernel associates to any two sequences x and y the sum of the product of the counts of all gappy bigrams in x and y penalized by the length of their *gaps*. Gappy bigrams are sequences of the form aua, aub, bua, or bub, where $u \in \Sigma^*$ is called the gap. The count of a gappy bigram is multiplied by $|u|^\lambda$ for some fixed $\lambda \in (0, 1)$ so that gappy bigrams with longer gaps contribute less to the definition of the similarity measure. While this definition could appear to be somewhat complex, figure 5.6 shows that $T_{\text{gappy\_bigram}}$ can be straightforwardly derived from T_{bigram}. The graphical representation of rational kernels helps understanding or modifying their definition.

Counting transducers The definition of most sequence kernels is based on the counts of some common patterns appearing in the sequences. In the examples just examined, these were bigrams or gappy bigrams. There exists a simple and general method for constructing a weighted transducer counting the number of occurrences of patterns and using them to define PDS rational kernels. Let X be a finite automaton representing the set of patterns to count. In the case of bigram kernels with $\Sigma = \{a, b\}$, X would be an automaton accepting exactly the set of strings $\{aa, ab, ba, bb\}$. Then, the weighted transducer of figure 5.7 can be used to compute exactly the number of occurrences of each pattern accepted by X.

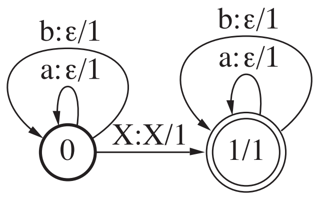

Figure 5.7 Counting transducer T_{count} for $\Sigma = \{a, b\}$. The "transition" $X : X/1$ stands for the weighted transducer created from the automaton X by adding to each transition an output label identical to the existing label, and by making all transition and final weights equal to one.

Theorem 5.10
For any $x \in \Sigma^*$ and any sequence z accepted by X, $T_{count}(x, z)$ is the number of occurrences of z in x.

Proof Let $x \in \Sigma^*$ be an arbitrary sequence and let z be a sequence accepted by X. Since all accepting paths of T_{count} have weight one, $T_{\text{count}}(x, z)$ is equal to the number of accepting paths in T_{count} with input label x and output z.

Now, an accepting path π in T_{count} with input x and output z can be decomposed as $\pi = \pi_0\,\pi_{01}\,\pi_1$, where π_0 is a path through the loops of state 0 with input label some prefix x_0 of x and output label ϵ, π_{01} an accepting path from 0 to 1 with input and output labels equal to z, and π_1 a path through the self-loops of state 1 with input label a suffix x_1 of x and output ϵ. Thus, the number of such paths is exactly the number of distinct ways in which we can write sequence x as $x = x_0 z x_1$, which is exactly the number of occurrences of z in x. ■

The theorem provides a very general method for constructing PDS rational kernels $T_{\text{count}} \circ T_{\text{count}}^{-1}$ that are based on counts of some patterns that can be defined via a finite automaton, or equivalently a regular expression. Figure 5.7 shows the transducer for the case of an input alphabet reduced to $\Sigma = \{a, b\}$. The general case can be obtained straightforwardly by augmenting states 0 and 1 with other self-loops using other symbols than a and b. In practice, a lazy evaluation can be used to avoid the explicit creation of these transitions for all alphabet symbols and instead creating them on-demand based on the symbols found in the input sequence x. Finally, one can assign different weights to the patterns counted to emphasize or deemphasize some, as in the case of gappy bigrams. This can be done simply by changing the transitions weight or final weights of the automaton X used in the definition of T_{count}.

5.6 Chapter notes

The mathematical theory of PDS kernels in a general setting originated with the fundamental work of Mercer [1909] who also proved the equivalence of a condition similar to that of theorem 5.1 for continuous kernels with the PDS property. The connection between PDS and NDS kernels, in particular theorems 5.8 and 5.7, are due to Schoenberg [1938]. A systematic treatment of the theory of reproducing kernel Hilbert spaces was presented in a long and elegant paper by Aronszajn [1950]. For an excellent mathematical presentation of PDS kernels and positive definite functions we refer the reader to Berg, Christensen, and Ressel [1984], which is also the source of several of the exercises given in this chapter.

The fact that SVMs could be extended by using PDS kernels was pointed out by Boser, Guyon, and Vapnik [1992]. The idea of kernel methods has been since then widely adopted in machine learning and applied in a variety of different tasks and settings. The following two books are in fact specifically devoted to the study of kernel methods: Schölkopf and Smola [2002] and Shawe-Taylor and Cristianini [2004]. The classical representer theorem is due to Kimeldorf and Wahba [1971]. A generalization to non-quadratic cost functions was stated by Wahba [1990]. The general form presented in this chapter was given by Schölkopf, Herbrich, Smola, and Williamson [2000].

Rational kernels were introduced by Cortes, Haffner, and Mohri [2004]. A general class of kernels, *convolution kernels*, was earlier introduced by Haussler [1999]. The convolution kernels for sequences described by Haussler [1999], as well as the pair-HMM string kernels described by Watkins [1999], are special instances of rational kernels. Rational kernels can be straightforwardly extended to define kernels for finite automata and even weighted automata [Cortes et al., 2004]. Cortes, Mohri, and Rostamizadeh [2008b] study the problem of *learning* rational kernels such as those based on counting transducers.

The composition of weighted transducers and the filter transducers in the presence of ϵ-paths are described in Pereira and Riley [1997], Mohri, Pereira, and Riley [2005], and Mohri [2009]. Composition can be further generalized to the *N-way composition* of weighted transducers [Allauzen and Mohri, 2009]. N-way composition of three or more transducers can substantially speed up computation, in particular for PDS rational kernels of the form $T \circ T^{-1}$. A generic *shortest-distance algorithm* which can be used with a large class of semirings and arbitrary queue disciplines is described by Mohri [2002]. A specific instance of that algorithm can be used to compute the sum of the weights of all paths as needed for the computation of rational kernels after composition. For a study of the class of languages linearly separable with rational kernels , see Cortes, Kontorovich, and Mohri [2007a].

5.7 Exercises

5.1 Let $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PDS kernel, and let $\alpha\colon \mathcal{X} \to \mathbb{R}$ be a positive function. Show that the kernel K' defined for all $x, y \in \mathcal{X}$ by $K'(x, y) = \frac{K(x,y)}{\alpha(x)\alpha(y)}$ is a PDS kernel.

5.2 Show that the following kernels K are PDS:

(a) $K(x, y) = \cos(x - y)$ over $\mathbb{R} \times \mathbb{R}$.

(b) $K(x, y) = \cos(x^2 - y^2)$ over $\mathbb{R} \times \mathbb{R}$.

(c) $K(x, y) = (x + y)^{-1}$ over $(0, +\infty) \times (0, +\infty)$.

(d) $K(\mathbf{x}, \mathbf{x}') = \cos \angle(\mathbf{x}, \mathbf{x}')$ over $\mathbb{R}^n \times \mathbb{R}^n$, where $\angle(\mathbf{x}, \mathbf{x}')$ is the angle between $\mathbf{x}$ and $\mathbf{x}'$.

(e) $\forall \lambda > 0,\ K(x, x') = \exp\big(-\lambda[\sin(x' - x)]^2\big)$ over $\mathbb{R} \times \mathbb{R}$. (*Hint*: rewrite $[\sin(x' - x)]^2$ as the square of the norm of the difference of two vectors.)

5.3 Show that the following kernels K are NDS:

(a) $K(x, y) = [\sin(x - y)]^2$ over $\mathbb{R} \times \mathbb{R}$.

(b) $K(x, y) = \log(x + y)$ over $(0, +\infty) \times (0, +\infty)$.

5.4 Define a *difference kernel* as $K(x, x') = |x - x'|$ for $x, x' \in \mathbb{R}$. Show that this kernel is not positive definite symmetric (PDS).

5.5 Is the kernel K defined over $\mathbb{R}^n \times \mathbb{R}^n$ by $K(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^{3/2}$ PDS? Is it NDS?

5.6 Let H be a Hilbert space with the corresponding dot product $\langle \cdot, \cdot \rangle$. Show that the kernel K defined over $H \times H$ by $K(x, y) = 1 - \langle x, y \rangle$ is negative definite.

5.7 For any $p > 0$, let K_p be the kernel defined over $\mathbb{R}_+ \times \mathbb{R}_+$ by

$$K_p(x, y) = e^{-(x+y)^p}. \tag{5.21}$$

Show that K_p is positive definite symmetric (PDS) iff $p \leq 1$. (*Hint*: you can use the fact that if K is NDS, then for any $0 < \alpha \leq 1$, K^α is also NDS.)

5.8 Explicit mappings.

(a) Denote a data set $x_1, \ldots, x_m$ and a kernel $K(x_i, x_j)$ with a Gram matrix $\mathbf{K}$. Assuming $\mathbf{K}$ is positive semidefinite, then give a map $\Phi(\cdot)$ such that

$K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$.

(b) Show the converse of the previous statement, i.e., if there exists a mapping $\Phi(x)$ from input space to some Hilbert space, then the corresponding matrix $\mathbf{K}$ is positive semidefinite.

5.9 Explicit polynomial kernel mapping. Let K be a polynomial kernel of degree d, i.e., $K\colon \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$, $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + c)^d$, with $c > 0$, Show that the dimension of the feature space associated to K is

$$\binom{N+d}{d}. \tag{5.22}$$

Write K in terms of kernels $k_i\colon (\mathbf{x}, \mathbf{x}') \mapsto (\mathbf{x} \cdot \mathbf{x}')^i$, $i \in [0, d]$. What is the weight assigned to each k_i in that expression? How does it vary as a function of c?

5.10 High-dimensional mapping. Let $\Phi\colon \mathcal{X} \to H$ be a feature mapping such that the dimension N of H is very large and let $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PDS kernel defined by

$$K(x, x') = \operatorname*{E}_{i \sim D}\big[[\Phi(x)]_i [\Phi(x')]_i\big], \tag{5.23}$$

where $[\Phi(x)]_i$ is the ith component of $\Phi(x)$ (and similarly for $\Phi'(x)$) and where D is a distribution over the indices i. We shall assume that $|[\Phi(x)]_i| \leq R$ for all $x \in \mathcal{X}$ and $i \in [1, N]$. Suppose that the only method available to compute $K(x, x')$ involved direct computation of the inner product (5.23), which would require $O(N)$ time. Alternatively, an approximation can be computed based on random selection of a subset I of the N components of $\Phi(x)$ and $\Phi(x')$ according to D, that is:

$$K'(x, x') = \frac{1}{n} \sum_{i \in I} D(i) [\Phi(x)]_i [\Phi(x')]_i, \tag{5.24}$$

where $|I| = n$.

(a) Fix x and x' in X. Prove that

$$\Pr_{I \sim D^n}[|K(x, x') - K'(x, x')| > \epsilon] \leq 2e^{\frac{-n\epsilon^2}{2r^2}}. \tag{5.25}$$

(*Hint*: use McDiarmid's inequality).

(b) Let $\mathbf{K}$ and $\mathbf{K}'$ be the kernel matrices associated to K and K'. Show that for any $\epsilon, \delta > 0$, for $n > \frac{r^2}{\epsilon^2} \log \frac{m(m+1)}{\delta}$, with probability at least $1 - \delta$, $|\mathbf{K}'_{ij} - \mathbf{K}_{ij}| \leq \epsilon$ for all $i, j \in [1, m]$.

5.11 Classifier based kernel. Let S be a training sample of size m. Assume that

S has been generated according to some probability distribution $D(x, y)$, where $(x, y) \in X \times \{-1, +1\}$.

(a) Define the Bayes classifier $h^*\colon X \to \{-1, +1\}$. Show that the kernel K^* defined by $K^*(x, x') = h^*(x)h^*(x')$ for any $x, x' \in X$ is positive definite symmetric. What is the dimension of the natural feature space associated to K^*?

(b) Give the expression of the solution obtained using SVMs with this kernel. What is the number of support vectors? What is the value of the margin? What is the generalization error of the solution obtained? Under what condition are the data linearly separable?

(c) Let $h : X \to \mathbb{R}$ be an arbitrary real-valued function. Under what condition on h is the kernel K defined by $K(x, x') = h(x)h(x')$, $x, x' \in X$, positive definite symmetric?

5.12 Image classification kernel. For $\alpha \geq 0$, the kernel

$$K_\alpha\colon (\mathbf{x}, \mathbf{x}') \mapsto \sum_{k=1}^{N} \min(|x_k|^\alpha, |x'_k|^\alpha) \tag{5.26}$$

over $\mathbb{R}^N \times \mathbb{R}^N$ is used in image classification. Show that K_α is PDS for all $\alpha \geq 0$. To do so, proceed as follows.

(a) Use the fact that $(f, g) \mapsto \int_{t=0}^{+\infty} f(t)g(t)dt$ is an inner product over the set of measurable functions over $[0, +\infty)$ to show that $(x, x') \mapsto \min(x, x')$ is a PDS kernel. (*Hint*: associate an indicator function to x and another one to x'.)

(b) Use the result from (a) to first show that K_1 is PDS and similarly that K_α with other values of α is also PDS.

5.13 Fraud detection. To prevent fraud, a credit-card company decides to contact Professor Villebanque and provides him with a random list of several thousand fraudulent and non-fraudulent *events*. There are many different types of events, e.g., transactions of various amounts, changes of address or card-holder information, or requests for a new card. Professor Villebanque decides to use SVMs with an appropriate kernel to help predict fraudulent events accurately. It is difficult for Professor Villebanque to define relevant features for such a diverse set of events. However, the risk department of his company has created a complicated method to estimate a probability $\Pr[U]$ for any event U. Thus, Professor Villebanque decides to make use of that information and comes up with the following kernel defined over all pairs of events (U, V):

$$K(U, V) = \Pr[U \wedge V] - \Pr[U] \Pr[V]. \tag{5.27}$$

Help Professor Villebanque show that his kernel is positive definite symmetric.

5.14 Relationship between NDS and PDS kernels. Prove the statement of theorem 5.7. (*Hint*: Use the fact that if K is PDS then $\exp(K)$ is also PDS, along with theorem 5.6.)

5.15 Metrics and Kernels. Let $\mathcal{X}$ be a non-empty set and $K \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a negative definite symmetric kernel such that $K(x, x) = 0$ for all $x \in \mathcal{X}$.

(a) Show that there exists a Hilbert space $\mathbb{H}$ and a mapping $\Phi(x)$ from $\mathcal{X}$ to $\mathbb{H}$ such that:

$$K(x, y) = ||\Phi(x) - \Phi(x')||^2 .$$

Assume that $K(x, x') = 0 \Rightarrow x = x'$. Use theorem 5.6 to show that $\sqrt{K}$ defines a metric on $\mathcal{X}$.

(b) Use this result to prove that the kernel $K(x, y) = \exp(-|x - x'|^p)$, $x, x' \in \mathbb{R}$, is not positive definite for $p > 2$.

(c) The kernel $K(x, x') = \tanh(a(x \cdot x') + b)$ was shown to be equivalent to a two-layer neural network when combined with SVMs. Show that K is not positive definite if $a < 0$ or $b < 0$. What can you conclude about the corresponding neural network when $a < 0$ or $b < 0$?

5.16 Sequence kernels. Let $X = \{a, c, g, t\}$. To classify DNA sequences using SVMs, we wish to define a kernel between sequences defined over X. We are given a finite set $I \subset X^*$ of non-coding regions (introns). For $x \in X^*$, denote by $|x|$ the length of x and by $F(x)$ the set of factors of x, i.e., the set of subsequences of x with contiguous symbols. For any two strings $x, y \in X^*$ define $K(x, y)$ by

$$K(x, y) = \sum_{z \in (F(x) \cap F(y)) - I} \rho^{|z|}, \tag{5.28}$$

where $\rho \geq 1$ is a real number.

(a) Show that K is a rational kernel and that it is positive definite symmetric.

(b) Give the time and space complexity of the computation of $K(x, y)$ with respect to the size s of a minimal automaton representing $X^* - I$.

(c) Long common factors between x and y of length greater than or equal to

n are likely to be important coding regions (exons). Modify the kernel K to assign weight $\rho_2^{|z|}$ to z when $|z| \geq n$, $\rho_1^{|z|}$ otherwise, where $1 \leq \rho_1 \ll \rho_2$. Show that the resulting kernel is still positive definite symmetric.

5.17 n-gram kernel. Show that for all $n \geq 1$, and any n-gram kernel K_n, $K_n(x, y)$ can be computed in linear time $O(|x| + |y|)$, for all $x, y \in \Sigma^*$ assuming n and the alphabet size are constants.

5.18 Mercer's condition. Let $\mathcal{X} \subset \mathbb{R}^N$ be a compact set and $K \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a continuous kernel function. Prove that if K verifies Mercer's condition (theorem 5.1), then it is PDS. (*Hint*: assume that K is not PDS and consider a set $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ and a column-vector $c \in \mathbb{R}^{m \times 1}$ such that $\sum_{i,j=1}^m c_i c_j K(x_i, x_j) < 0$.)