Diffusion Kernels on Graphs and Other Discrete Input Spaces

Risi Imre Kondor KONDOR@CMU.EDU
John Lafferty LAFFERTY@CS.CMU.EDU
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA

Abstract

The application of kernel-based learning algorithms has, so far, largely been confined to real-valued data and a few special data types, such as strings. In this paper we propose a general method of constructing natural families of kernels over discrete structures, based on the matrix exponentiation idea. In particular, we focus on generating kernels on graphs, for which we propose a special class of exponential kernels called diffusion kernels, which are based on the heat equation and can be regarded as the discretization of the familiar Gaussian kernel of Euclidean space.

1. Introduction

Kernel-based algorithms, such as Gaussian processes (Mackay, 1997), support vector machines (Burges, 1998), and kernel PCA (Mika et al., 1998), are enjoying great popularity in the statistical learning community. The common idea behind these methods is to express our prior beliefs about the correlations, or more generally, the similarities, between pairs of points in data space $\mathcal{X}$ in terms of a kernel function $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, and thereby to implicitly construct a mapping $\phi : \mathcal{X} \mapsto \mathcal{H}_K$ to a Hilbert space $\mathcal{H}_K$, in which the kernel appears as the inner product,

$$K(x,x') = \langle \phi(x), \phi(x') \rangle \tag{1}$$

(Schölkopf & Smola, 2001). With respect to a basis of $\mathcal{H}_K$, each datapoint then splits into (a possibly infinite number of) *independent* features, a property which can be exploited to great effect.

Graph-like structures occur in data in many guises, and in order to apply machine learning techniques to such discrete data it is desirable to use a kernel to capture the long-range relationships between data points induced by the local structure of the graph. One obvious example of such data is a graph of documents related to one another by links, such as the hyperlink structure of the World Wide Web. Other examples include social networks, citations between scientific articles, and networks in linguistics (Albert & Barabási, 2002).

Graphs are also sometimes used to model complicated or only partially understood structures in a first approximation. In chemistry or molecular biology, for example, it might be anticipated that molecules with similar chemical structures will have broadly similar properties. While for two arbitrary molecules it might be very difficult to quantify exactly how similar they are, it is not so difficult to propose rules for when two molecules can be considered "neighbors," for example, when they only differ in the presence or absence of a single functional group, movement of a bond to a neigboring atom, etc. Representing such relationships by edges gives rise to a graph, each vertex corresponding to one of our original objects. Finally, adjacency graphs are sometimes used when data is expected to be confined to a manifold of lower dimensionality than the original space (Saul & Roweis, 2001; Belkin & Niyogi, 2001) and (Szummer & Jaakkola, 2002). In all of these cases, the challenge is to capture in the kernel K the local and global structure of the graph.

In addition to adequately expressing the known or hypothesized structure of the data space, the function K must satisfy two mathematical requirements to be able to serve as a kernel: it must be symmetric ($K(x,x') = K(x',x)$) and positive semi-definite. Constructing appropriate positive definite kernels is not a simple task, and this has largely been the reason why, with a few exceptions, kernel methods have mostly been confined to Euclidean spaces ($\mathcal{X} = \mathbb{R}^m$), where several families of provably positive semi-definite and easily interpretable kernels are known. When dealing with intrinsically discrete data spaces, the usual approach has been either to map the data to Euclidean space first (as is commonly done in text classification, treating integer word counts as real numbers (Joachims, 1998)) or, when no such simple mapping is forthcoming, to forgo using kernel methods altogether. A notable exception to this is the line of work stemming from the convolution kernel idea introduced in (Haussler, 1999) and related but independently conceived ideas on string kernels first presented in (Watkins, 1999). Despite the promise of these ideas, relatively little work has been done on discrete kernels since the publication of these articles.

In this paper we use ideas from spectral graph theory to propose a natural class of kernels on graphs, which we refer to as *diffusion kernels*. We start out by presenting in the following section a more general class of kernels, called *exponential kernels*, applicable to a wide variety of discrete objects. In Section 3 we discuss the interpretation of diffusion kernels on graphs. In Section 4 we show how diffusion kernels can be computed for some special families of graphs. Experiments using diffusion kernels for classification of categorical data are presented in Section 5, and we conclude and summarize our results in Section 6.

2. Exponential Kernels and Diffusion Kernels

This section shows that the exponentiation operation on matrices naturally yields the crucial positive-definite criterion of kernels, describes how to build kernels on the direct product of graphs, and introduces diffusion kernels on graphs as a distinguished example of exponential kernels.

2.1 Exponential kernels

Recall that in the discrete case, positive semi-definiteness amounts to

$$\sum_{x \in \mathcal{X}} \sum_{x' \in \mathcal{X}} f_x \, f_{x'} \, K(x, x') \geq 0 \tag{2}$$

for all sets of real coefficients $\{f_x\}$, and in the continuous case,

$$\int_{\mathcal{X}} \int_{\mathcal{X}} f(x) \, f(x') \, K(x, x') \, dx \, dx' \quad \geq \quad 0$$

for all square integrable real functions $f \in L_2(\mathcal{X})$; the latter is sometimes referred to as Mercer's condition.

In the discrete case, for finite $\mathcal{X}$, the kernel can be uniquely represented by an $|\mathcal{X}| \times |\mathcal{X}|$ matrix (which we shall denote by the same letter K) with rows and columns indexed by the elements of $\mathcal{X}$, and related to the kernel by $K_{xx'} = K(x, x')$. Since this matrix, called the Gram matrix, and the function $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ are essentially equivalent (in particular, the matrix inherits the properties of symmetry and positive semi-definiteness), we can refer to one or the other as the "kernel" without risk of confusion.

The exponential of a square matrix H is defined as

$$e^{\beta H} = \lim_{n \to \infty} \left(1 + \frac{\beta H}{n} \right)^n , \tag{3}$$

where the limit always exists and is equivalent to

$$e^{\beta H} = I + \beta H + \frac{\beta^2}{2!} \, H^2 + \frac{\beta^3}{3!} \, H^3 + \ldots \, . \tag{4}$$

It is well known that any even power of a symmetric matrix is positive semi-definite, and that the set of positive semi-definite matrices is complete with respect to limits of sequences under the Frobenius norm. Taking H to be symmetric and replacing n by $2n$ shows that the exponential of any symmetric matrix is symmetric and positive semi-definite, hence it is a candidate for a kernel.

Conversely, it is easy to show that any infinitely divisible kernel K can be expressed in the exponential form (3). Infinite divisibility means that K can be written as an n-fold convolution

$$K = K^{1/n} \cdot K^{1/n} \cdot \ldots \cdot K^{1/n}$$

for any $n \in \mathbb{Z}$ (Haussler, 1999). Such kernels form continuous families $\{K(\beta) = K(1)^\beta\}$, indexed by a real parameter β, and are related to infinitely divisible probability distributions, which are the limits of sums of independent random variables (Feller, 1971). The tautology $K(\beta) = [K(1)^{\beta/n}]^n$ becomes, as n goes to infinity,

$$K = \lim_{n \to \infty} \left(1 + \frac{\beta}{n} \, \frac{dK}{d\beta} \Big|_{\beta=0} \right)^n ,$$

which is equivalent to (3) for $H = \frac{dK}{d\beta}|_{\beta=0}$.

The above already suggests looking for kernels over finite sets in the form

$$K = e^{\beta H} , \tag{5}$$

guaranteeing positive definiteness without seriously restricting our choice of kernel. Furthermore, differentiating (5) with respect to β and examining the resulting differential equation

$$\frac{d}{d\beta} \, K_\beta = H K_\beta , \tag{6}$$

with the accompanying initial conditions $K(0) = I$, lends itself naturally to the interpretation that $K(\beta)$ is the product of a continuous process, expressed by H, gradually transforming it from the identity matrix ($K(0)$) to a kernel with stronger and stronger off-diagonal effects as β increases. We shall see in the examples below that by virtue of this relationship, choosing H to express the local structure of $\mathcal{X}$ will result in the global structure of $\mathcal{X}$ naturally emerging in K. We call (5) an exponential family of kernels, with generator H and bandwidth parameter β.

Note that the exponential kernel construction is *not* related to the result described in (Berg et al., 1984; Haussler, 1999) and (Schölkopf & Smola, 2001), based on Schoenberg's pioneering work in the late 1930's in the theory of positive definite functions (Schoenberg, 1938). This work shows that any positive semi-definite K can be written as

$$K(x, x') = e^{M(x, x')} \tag{7}$$

where M is a *conditionally* positive semi-definite kernel; that is, it satisfies (2) under the additional constraint that

$\sum_{i \in \mathcal{X}} f_i = 0$.[1] Whereas (5) involves matrix exponentiation via (3), formula (7) prescribes the more straightforward componentwise exponentiation. On the other hand, conditionally positive definite matrices are somewhat elusive mathematical objects, and it is not clear where Schoenberg's beautiful result will find application in statistical learning theory. The advantage of our relatively brute-force approach to constructing positive definite objects is that it only requires that the generator H be symmetric (more generally, self-adjoint) and guarantees the positive semi-definiteness of the resulting kernel K.

2.2 Tensor products and conjugacy

There is a canonical way of building exponential kernels over direct products of sets, which will prove useful in what follows. Let $\{K_1(\beta)\}$ be a family of kernels over the set $\mathcal{X}_1$ with generator H_1, and let $\{K_2(\beta)\}$ be a family of kernels over $\mathcal{X}_2$ with generator H_2. To construct an exponential kernels over the pairs (x_1, x_2), with $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$, it is natural to use the generator

$$[H]_{x_1,x_2;x_1',x_2'} = [H_1]_{x_1,x_1'}\, \delta(x_2,x_2') + [H_2]_{x_2,x_2'}\, \delta(x_1,x_1')$$

where $\delta(x,y) = 1$ if $x = y$ and 0 otherwise. In other words, we take the generator over the product set $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2$ to be $H = H_1 \otimes I_2 + H_2 \otimes I_1$, where I_1 and I_2 are the $|\mathcal{X}_1|$ and $|\mathcal{X}_2|$ dimensional diagonal kernels, respectively. Plugging into (6) shows that the corresponding kernels will be given simply by

$$[K(\beta)]_{x_1,x_2;x_1',x_2'} = [K_1(\beta)]_{x_1,x_1'}\, [K_2(\beta)]_{x_2,x_2'} \ ;$$

that is, $K(\beta) = K_1(\beta) \otimes K_2(\beta)$. In particular, we can lift any exponential kernel K on $\mathcal{X}$ to an exponential kernel K^n over length n sequences $\{\ \boldsymbol{x} = (x_1, x_2, \ldots, x_n) \ : \ x_i \in \mathcal{X}\ \}$ by

$$K^n(\boldsymbol{x}, \boldsymbol{x}') = \prod_{i=1}^{n} K(x_i, x_i') \ , \qquad (8)$$

or, using the tensor product notation, $K^n = \bigotimes_{i=1}^n K$.

Finally, we note that another method for creating kernels is to conjugate the Gram matrix by a not necessarily square matrix T

$$K^{(T)} = T^\top K\, T \qquad (9)$$

which yields a new positive semi-definite kernel $K : \mathcal{X}_2 \times \mathcal{X}_2 \mapsto \mathbb{R}$ of the form

$$K^{(T)}(x_i, x_j) = \sum_q \sum_p T_{ip} T_{jq} K(z_p, z_q) \ . \qquad (10)$$

[1] Instead of using the term "conditionally positive definite," this type of object is sometimes referred to by saying that $-M$ is "negative definite." Confusingly, a negative definite kernel is then *not* the same as the negative of a positive definite kernel, so we shall avoid using this terminology.

2.3 Diffusion kernels on graphs

An undirected, unweighted graph Γ is defined by a vertex set V and an edge set E, the latter being the set of unordered pairs $\{v_1, v_2\}$, where $\{v_1, v_2\} \in V$ whenever the vertices v_1 and v_2 are joined by an edge (denoted $v_1 \sim v_2$). Equation (6) suggests using an exponential kernel with generator

$$H_{ij} = \begin{cases} 1 & \text{for } i \sim j \\ -d_i & \text{for } i = j \\ 0 & \text{otherwise} \ , \end{cases} \qquad (11)$$

where d_i is the degree of vertex i (number of edges emanating from vertex i).

The negative of this matrix (sometimes up to normalization) is called the Laplacian of Γ, and it plays a central role in spectral graph theory (Chung, 1997). It is instructive to note that for any vector $w \in \mathbb{R}^{|V|}$,

$$w^\top H\, w = -\sum_{\{i,j\} \in E} (w_i - w_j)^2,$$

showing that H is, in fact, negative semi-definite. Acting on functions $\{\ f \ : \ V \ \mapsto \ \mathbb{R}\ \}$ by $(Hf)(x) = \sum_{x'} H_{x,x'} f(x')$, H can also be regarded as an operator. In fact, it is easy to show that on a square grid in m-dimensional Euclidean space with grid spacing h, H/h^2 is just the finite difference approximation to the familiar continuous Laplacian

$$\Delta = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \ldots + \frac{\partial^2}{\partial x_m^2} \ ,$$

and that in the limit $h \to 0$ this approximation becomes exact. In analogy with classical physics, where equations of the form

$$\frac{\partial}{\partial t}\, \psi = \mu\, \Delta \psi$$

are used to describe the diffusion of heat and other substances through continuous media, our equation

$$\frac{d}{d\beta}\, K_\beta = H K_\beta \qquad (12)$$

with H as defined in (11) is called the *heat equation* on Γ, and the resulting kernels are called *diffusion kernels* or *heat kernels*. In the context of learning theory, the principal components of the diffusion kernel were used in Belkin and Niyogi (2001) to find optimal embeddings of data manifolds. To the best of our knowledge, diffusion kernels have not been proposed previously for direct use in kernel-based learning algorithms.

We remark that diffusion kernels are not restricted to simple unweighted graphs. For multigraphs or weighted symmetric graphs, all we need to do is to set H_{ij}, $i \neq j$ to be the total weight of all edges between i and j and reweight the diagonal terms accordingly. The rest of the analysis carries through as before.

3. Interpretation

To motivate the use of diffusion kernels as a natural way of quantifying the structure of discrete input spaces in learning algorithms, in this section we discuss some of the many interpretations of diffusion kernels on graphs.

3.1 A stochastic and a physical model

There is a natural class of stochastic processes on graphs whose covariance structure yields diffusion kernels. Consider the random field obtained by attaching independent, zero mean, variance σ^2 random variables $Z_i(0)$ to each vertex $i \in V$. Now let each of these random variables "send" a fraction $\alpha \ll 1$ of their value to each of their respective neighbors at discrete time steps $t = 1, 2, \ldots$; that is, let

$$Z_i(t+1) = Z_i(t) + \alpha \sum_{j \in V:\, j \sim i} \left(Z_j(t) - Z_i(t) \right) \ .$$

Introducing the time evolution operator

$$T(t) = (1 + \alpha H)^t \ ,$$

$Z(t) = (Z_1(t), Z_2(t), \ldots, Z_{|V|}(t))^\top$ can be written as

$$Z(t) = T(t) Z(0) \ . \qquad (13)$$

The covariance of the random field at time t is

$$\begin{aligned} \mathrm{Cov}_{ij}(t) &= E\left[(Z_i(t) - EZ_i(t)) \, (Z_j(t) - EZ_j(t)) \right] \\ &= E\left[Z_i(t) Z_j(t) \right] \\ &= E\Big[\Big(\sum_{i'} T_{ii'}(t) Z_{i'}(0) \Big) \Big(\sum_{j'} T_{jj'}(t) Z_{j'}(0) \Big) \Big], \end{aligned}$$

which simplifies to

$$\begin{aligned} \mathrm{Cov}_{ij}(t) &= \sigma^2 \sum_k T_{ik}(t) T_{kj}(t) \\ &= \sigma^2 \, [T(t)^2]_{ij} = \sigma^2 \, T_{ij}(2t) \qquad (14) \end{aligned}$$

by independence at time zero, $E[Z_i Z_j] = \sigma^2 \delta(i,j)$. Note that (14) holds regardless of the particular distribution of the $\{Z_i(0)\}$, as long as their mean is zero and their variance is σ^2.

Now we can decrease the time step from 1 to Δt by replacing t by $t/(\Delta t)$ and α by $\alpha \Delta t$ in (13) [2], giving

$$T(t) = \left(1 + \frac{\alpha H}{1/(\Delta t)} \right)^{t/\Delta t}$$

which, in the limit $\Delta t \to 0$, is exactly of the form (3). In particular, the covariance becomes the diffusion kernel $\mathrm{Cov}(t) = \sigma^2 e^{2\alpha t H}$. Since kernels are in some sense nothing but "generalized" covariances (in fact, in the case of Gaussian Processes, they *are* the covariance), this example supports the contention that diffusion kernels are a natural choice on graphs.

[2] Note that Δ is here used to denote infinitesimals and not the Laplacian.

Closely related to the above is an electrical model. Differentiating (13) with respect to t yields the differential equations

$$\frac{d}{dt} \, Z_i(t) = \alpha \sum_{j \in V:\, i \sim j} \left(Z_j(t) - Z_i(t) \right) .$$

These equations are the same as those describing the relaxation of a network of capacitors of unit capacitance, where one plate of each capacitor is grounded, and the other plates are connected according to the graph structure, each edge corresponding to a connection of resistance $1/\alpha$. The $\{Z_i(t)\}$ then measure the potential at each capacitor at time t. In particular, $K(i,j) = \exp(\alpha t H)$ is the potential at capacitor i, time t after having initialized the system by decharging every capacitor, except for capacitor j, which starts out at unit potential.

3.2 The continuous limit

As a special case, it is instructive to again consider the infinitely fine square grid on $\mathbb{R}^m$. Introducing the similarity function $k_x(x') = K(x, x')$, the heat equation (12) gives

$$\frac{d}{d\beta} \, k_x(x') = \int H(x, x'') \, k_{x''}(x') \, dx'' \ .$$

Since the Laplacian is a local operator in the sense that $\Delta f(x)$ is only affected by the behavior of f in the neighborhood of x, as long as $k_x(x')$ is continuous in x, the above can be rewritten as simply

$$\frac{d}{d\beta} \, k_x(x') = \Delta \, k_x(x') \ .$$

It is easy to verify that the solution of this equation with Dirac spike initial conditions $k_x(x') = \delta(x - x')$ is just the Gaussian

$$k_x(x') = \frac{1}{\sqrt{4\pi\beta}} \, e^{-|x-x'|^2/(4\beta)},$$

showing that similarity to any given point x', as expressed by the kernel, really does behave as some substance diffusing in space, and also that the familiar Gaussian kernel on $\mathbb{R}^m$,

$$K(x, x') = \frac{1}{\sqrt{2\pi\sigma^2}} \, e^{-|x-x'|^2/(2\sigma^2)}$$

is just a diffusion kernel with $\beta = \sigma^2/2$. In this sense, diffusion kernels can be regarded as a generalization of Gaussian kernels to graphs.

3.3 Relationship to random walks

It is well known that diffusion is closely related to random walks. A random walk on an unweighted graph Γ is a stochastic process generating sequences $z_0, z_1, z_2, \ldots$ where $z_l \in V$ in such a way that $p\,(\, z_{l+1} = j \mid z_l = i\,) = 1/d_i$ if $i \sim j$ and zero otherwise.

A lazy random walk on Γ with parameter $\beta \leq 1/(\max_i d_i)$ is very similar, except that when at vertex i, the process will take each of the edges emanating from i with fixed probability β, i.e. $p\,(\, z_{l+1} = j \mid z_l = i\,) = \beta$ for $i \sim j$, and will remain in place with probability $1 - d_i\beta$. Considering the distribution $p\,(\, z_N \mid z_0\,)$ in the limit $\Delta t \to 0$ with $N = 1/(\Delta t)$ and $\beta = \beta_0 \Delta t$ leads exactly to (3) showing that diffusion kernels are the continuous time limit of lazy random walks.

This analogy also shows that $K(i,j)$ can be regarded as a sum over paths from i to j, namely the sum of the probabilities that the lazy walk takes each path. For graphs in which every vertex is of the same degree $d_i = d$, mapping each vertex i to every path starting at i weighted by the square root of the probability of a lazy random walk starting at i taking that path,

$$\tau(i) = e^{-d\beta/2} \sum_{\boldsymbol{z} \in \mathcal{Z}\,:\, z_0 = i} \left(\frac{\beta^{|\boldsymbol{z}|}}{|\boldsymbol{z}|!} \right)^{1/2} \boldsymbol{z}\,,$$

where $\mathcal{Z} = \{\, \boldsymbol{z} = (z_0, z_1, \ldots, z_{|\boldsymbol{z}|}) \;:\; z_l \in V,\;\; z_l \sim z_{l+1} \,\}$ is the set of all paths on Γ, gives a representation of the kernel in the space $F(\mathcal{Z})$ of linear combinations of paths of the form

$$K_{F(\mathcal{Z})}(\boldsymbol{z}, \boldsymbol{z}') = \begin{cases} \delta(\boldsymbol{z}, \boldsymbol{z}') & \text{for loops, i.e., } z_0 = z_{|\boldsymbol{z}|} \\ \delta(\overline{\boldsymbol{z}}, \boldsymbol{z}')/\sqrt{2} & \text{otherwise,} \end{cases}$$

where $\overline{\boldsymbol{z}} = z_{|\boldsymbol{z}|}, z_{|\boldsymbol{z}|-1}, \ldots, z_0$ is the reverse of $\boldsymbol{z}$. In the basis of loops $\{\, \boldsymbol{\xi} \in \mathcal{Z} \;:\; \xi_0 = \xi_{|\boldsymbol{\xi}|} \}$ and linear combinations

$$\boldsymbol{\xi}^{(1)}_{\boldsymbol{z}} = \frac{1}{\sqrt{2}}\,(\boldsymbol{z} + \overline{\boldsymbol{z}})$$

$$\boldsymbol{\xi}^{(2)}_{\boldsymbol{z}} = \frac{1}{\sqrt{2}}\,(\boldsymbol{z} - \overline{\boldsymbol{z}})$$

for all pairs $\{\boldsymbol{z}, \overline{\boldsymbol{z}}\}$ of non-loops, this does give a diagonal representation of K, but not a representation satisfying (1), because there are alternating $+1$'s and -1's on the diagonal.

4. Some Special Graphs

In general, computing exponential kernels involves diagonalizing the generator

$$H = T^{-1} D\, T,$$

which is always possible because H is symmetric, and then computing

$$e^{\beta H} = T^{-1}\, e^{\beta D}\, T,$$

which is easy, because $e^{\beta D}$ will be diagonal with $[e^{\beta D}]_{ii} = \exp(\beta D_{ii})$. The diagonalization process is computationally expensive, however, and the kernel matrix must be stored in memory during the whole time the learning algorithm operates. Hence there is interest in the few special cases for which the kernel matrix can be computed directly.

4.1 k-regular trees

An infinite k-regular tree is an undirected, unweighted graph with no cycles, in which every vertex has exactly k neighbors. Note that this differs from the notion of a rooted n-ary tree in that no special node is designated the root. Any vertex can function as the root of the tree, but that too must have exactly k neighbors. Hence a 3-regular tree looks much like a rooted binary tree, except that at the root it splits into three branches and not two.

Because of the graph's symmetry, $K(i,j) = \exp(\beta H)$ can only depend on the relative positions of i and j, namely the length of the unique path between them, $d(i,j)$. Chung and Yau (1999) show that

$$K(i,j) = K^{(k)}_{\mathrm{R}}(\, d(i,j)) \;= \tag{15}$$
$$\frac{2}{\pi(k-1)} \int_0^{\pi} \frac{e^{-\beta\left(1 - \frac{2\sqrt{k-1}}{k}\cos x\right)}}{k^2 - 4\,(k-1)\,\cos^2 x} \;\times$$
$$\times\; \sin x\,[\,(k-1)\sin(d+1)\,x - \sin(d-1)\,x\,]\; dx$$

for $d = d(i,j) \geq 1$, and

$$K(i,i) = K^{(k)}_{\mathrm{R}}(0) \;= \tag{16}$$
$$\frac{2k(k-1)}{\pi} \int_0^{\pi} \frac{\exp\left(-\beta\,(1 - \frac{2\sqrt{k-1}}{k}\cos x)\right)\,\sin^2 x}{k^2 - 4\,(k-1)\,\cos^2 x}\, dx$$

for the diagonal elements.

4.2 Complete graphs

In the unweighted complete graph with n vertices, any pair of vertices is joined by an edge, hence $H_{ij} = 1 - \delta(i,j)n$. It is easy to verify that the corresponding solution to (12) is

$$K(i,j) = \begin{cases} \dfrac{1 + (n-1)\,e^{-n\beta}}{n} & \text{for } i = j \\[2ex] \dfrac{1 - e^{-n\beta}}{n} & \text{for } i \neq j\,, \end{cases} \tag{17}$$

showing that with increasing β, the kernel relaxes exponentially to $K(i,j) = 1/n$. The asymptotically exponential character of this solution, and the convergence to the uniform kernel for finite $\mathcal{X}$, are direct consequences of the fact that H is a linear operator, and we shall see this type of behavior recur in other examples.

4.3 Closed chains

When Γ is a single closed chain of length n, $K(i,j)$ will clearly only depend on the distance $d(i,j)$ along the chain between i and j. Labeling the vertices consecutively from 0 to $n-1$, the similarity function at a particular vertex (without loss of generality, vertex zero) can be expressed in terms of its discrete Fourier transform

$$K(0,j) = k_0(j) = \frac{1}{\sqrt{n}} \sum_{\nu=0}^{n-1} \widehat{k}_0(j) \, \cos \frac{2\pi\nu j}{n} \,.$$

The heat equation implies

$$\frac{d}{d\beta}\, k_0(j) \\ = \quad k_0(j+1 \bmod n) - 2k_0(j) + k_0(j-1 \bmod n),$$

which after some trigonometry translates into

$$\frac{d}{d\beta}\, \widehat{k}_0(\nu) = -\,2 \left(1 - \cos \frac{2\pi\nu}{n}\right) \widehat{k}_0(\nu),$$

showing that the Fourier coefficients decay independently of one another. Using the inverse Fourier transform, the solution corresponding to the initial condition $k_0(i) = \delta(i,0)$ at $\beta = 0$ will be

$$k_0(j) = \frac{1}{n} \sum_{\nu=0}^{n-1} e^{-\omega_\nu \beta} \cos \frac{2\pi\nu j}{n}$$

where $\omega_\nu = 2\left(1 - \cos \frac{2\pi\nu}{n}\right)$, and the kernel itself will be

$$K(i,j) = \frac{1}{n} \sum_{\nu=0}^{n-1} e^{-\omega_\nu \beta} \cos \frac{2\pi\nu(i-j)}{n} \,.$$

4.4 The hypercube and tensor products of complete graphs

Kernels on the special graphs considered above can serve as building blocks for tensor product kernels, as in (8). For example, it is natural to identify binary strings $\boldsymbol{x} = x_0 x_1 x_2 \ldots x_m, \;\; x_i \in \{0,1\}$ of length m with the vertices $(x_1, x_2, \ldots, x_m)$ of the m-dimensional hypercube. Constructing a diffusion kernel on the hypercube regarded as a graph amounts to asserting that two sequences are neighbors if they only differ in a single digit. From (17) and (8), the diffusion kernel on the hypercube will be

$$\begin{aligned} K(\boldsymbol{x},\boldsymbol{x}') \quad &\propto \quad \left(\frac{1-e^{-2\beta}}{1+e^{-2\beta}}\right)^{d(\boldsymbol{x},\boldsymbol{x}')} \\ &= \quad (\tanh\beta)^{d(\boldsymbol{x},\boldsymbol{x}')} \,, \end{aligned}$$

which only depends on the Hamming distance $d(\boldsymbol{x},\boldsymbol{x}')$ between $\boldsymbol{x}$ and $\boldsymbol{x}'$, and is extremely easy to compute. Similarly, the diffusion kernel on strings over an alphabet $\mathcal{A}$ of size $|\mathcal{A}|$ will be

$$K(\boldsymbol{x},\boldsymbol{x}') \quad \propto \quad \left(\frac{1-e^{-|\mathcal{A}|\beta}}{1+(|\mathcal{A}|-1)\,e^{-|\mathcal{A}|\beta}}\right)^{d(\boldsymbol{x},\boldsymbol{x}')}$$

where $d(\boldsymbol{x},\boldsymbol{x}')$ is the number of character places at which $\boldsymbol{x}$ and $\boldsymbol{x}'$ differ.

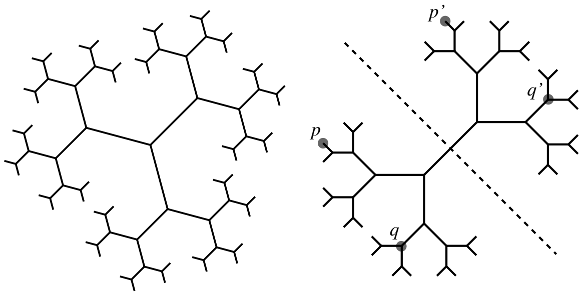

Figure 1. The three-regular tree (left), which extends to infinity in all directions. A little bending of the branches shows that it is isomorphic to two rooted binary trees joined at the root (right). The method of images enables us to compute the diffusion kernel between vertices of the binary tree by mapping each to a pair of vertices (p,p') and (q,q') on the three-regular tree and summing the contributions from $K(p,q)$, $K(p,q')$, $K(p',q)$ and $K(p',q')$.

4.5 Rooted trees

We have noted above that the distinction between k-regular trees and infinite $(k-1)$-ary rooted trees is that arbitrarily designating a vertex in the former as the root, we find that it has an extra branch emanating from it (Figure 1). Taking for simplicity $k=3$, the analytical formulæ (15) and (16) are hence not directly applicable to binary rooted trees, because if we simply try to ignore this branch by not mapping any data points to it, in the language of the electrical analogy of Section 3.1, we find that some current will "seep away" through it. The kernel can be obtained, however, using the method of conjugation.

The crucial observation is that the graph possesses mirror symmetry about the edge connecting this errant branch to the rest of the graph. Mapping each vertex i of the binary tree to the analogous vertex on one side of this plane of symmetry $p = \tau(i)$ in the 3-regular tree *and* its mirror image $p' = \tau'(i)$ on the other side solves the problem, because, by symmetry, in the electrical analogy, the flow of current across the critical edge connecting the two halves of the graph will be zero. This construction, called the method of images, corresponds to a transformation matrix of the form

$$T = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & & & & & \cdots \\ & & \frac{1}{2} & \frac{1}{2} & & & \\ & & & & \frac{1}{2} & \frac{1}{2} & \\ \vdots & & & & & & \ddots \end{pmatrix}$$

			Hamming kernel		*Diffusion kernel*			*Improvement*	
Data Set	#Attr	$\max\lvert\mathcal{A}\rvert$	error	$\lvert SV\rvert$	error	$\lvert SV\rvert$	β	Δerror	$\Delta\lvert SV\rvert$
`Breast Cancer`	9	10	7.44 ± 1.70%	206.0	3.70 ± 0.83%	43.3	0.30	50%	80%
`Hepatitis`	13	2	19.50 ± 3.90%	420.0	18.80 ± 4.13%	192.0	1.80	4%	57%
`Income`	11	42	19.19 ± 1.20%	1149.5	18.50 ± 1.27%	1033.4	0.40	4%	8%
`Mushroom`	22	10	1.40 ± 0.44%	117.7	0.007 ± 0.018%	27.2	0.40	99%	77%
`Votes`	16	2	4.79 ± 1.16%	176.5	4.53 ± 1.44%	60.6	1.5	6%	66%

Table 1. Results on five UCI data sets. For each data set, only the categorical features are used. The column marked $\max|\mathcal{A}|$ indicates the maximum number of values for an attribute; thus the `Votes` data set has binary attributes. Results are reported for the setting of the diffusion coefficient β achieving the best cross-validated error rate.

and yields in analytical form the diffusion kernel for infinite binary trees

$$K(i,j) = \frac{1}{2} K_{\mathrm{R}}^{(3)}(d(i,j)) + \frac{1}{2} K_{\mathrm{R}}^{(3)}(d(i,r)+d(r,j)+1)$$

where r designates the root and d measures distances on the binary tree.

4.6 String kernels

Another application of conjugated diffusion kernels is in the construction of string kernels sensitive to matching non-contiguous substrings. The usual approach to this is to introduce "blank" characters $\square$ into the strings $\boldsymbol{x}$ and $\boldsymbol{x}'$ to be compared, so that the characters of the common substring are aligned..

Using the tensor product of complete graphs approach developed above, it is easy to add an extra character to the alphabet $\mathcal{A}$ to represent $\square$. We can then map $\boldsymbol{x}$ and $\boldsymbol{x}'$ to a generalized hypercube $\mathcal{A}^n$ of dimensionality $n \geq |\boldsymbol{x}|+|\boldsymbol{x}'|$ by mapping each string to the vertices corresponding to all its posssible extensions by $\square$'s. Let us represent an alignment between $\boldsymbol{x}$ and $\boldsymbol{x}'$ by the vector of matches $\boldsymbol{s} = ((u_1,v_1),(u_2,v_2),\ldots,(u_{|\boldsymbol{s}|},v_{|\boldsymbol{s}|}))$ where $1 \leq u_1 < u_2 < \ldots < u_{|\boldsymbol{s}|} \leq |\boldsymbol{x}|, \quad 1 \leq v_1 < v_2 < \ldots < v_{|\boldsymbol{s}|} \leq |\boldsymbol{x}'|$, $x_{u_l} = x'_{v_l}$ and let $\mathcal{S}(\boldsymbol{x},\boldsymbol{x}')$ be the set of all alignments between $\boldsymbol{x}$ and $\boldsymbol{x}'$. Assuming that all virtual strings are weighted equally, the resulting kernel will be

$$K(\boldsymbol{x},\boldsymbol{x}') = \sum_{\boldsymbol{s}\in\mathcal{S}(\boldsymbol{x},\boldsymbol{x}')} c(\boldsymbol{x},\boldsymbol{x}')\; B^{\,n-|\boldsymbol{s}|} \qquad (18)$$

for some combinatorial factor $c(\boldsymbol{x},\boldsymbol{x}')$ and

$$B(\beta) = \frac{1-e^{-(|\mathcal{A}|+1)\beta}}{1+|\mathcal{A}|\; e^{-(|\mathcal{A}|+1)\beta}}\,.$$

In the special case that $n \to \infty$, the combinatorial factor becomes constant for all pairs of strings and (18) becomes computable by dynamic programming by the recursion $k_{i+1,j+1} = B(k_{i,j+1}+k_{i+1,j}) + \kappa_{i+1,j+1}k_{i,j}$, where $\kappa_{i,j} = 1$ when $x_{i+1} = x'_{j+1}$ and B otherwise. For the derivation of recursive formulæ such as this, and comparison to other measures of similarity between strings, see (Durbin et al., 1998).

5. Experiments on UCI Datasets

In this section we describe preliminary experiments with diffusion kernels, focusing on the use of kernel-based methods for classifying categorical data. For such problems, it is often quite unnatural to encode the data as vectors in Euclidean space to allow the use of standard kernels. However as our experiments show, even simple diffusion kernels on the hypercube, as described in Section 4.4, can result in good performance for such data.

For ease of experimentation we use a large margin classifier based on the voted perceptron, as described in (Freund & Schapire, 1999). In each set of experiments we compare models trained using a diffusion kernel and a kernel based on the Hamming distance, $K_H(\boldsymbol{x},\boldsymbol{x}') = n - \sum_{i=1}^n \delta(x_i,x'_i)$.

Data sets having a majority of categorical variables were chosen; any continuous features were ignored. The diffusion kernels used are given by the natural extension of the hypercube kernels given in Section 4.4, namely

$$K_\beta(\boldsymbol{x},\boldsymbol{x}') \quad\propto\quad \prod_{i=1}^n \left(\frac{1-e^{-|\mathcal{A}_i|\beta}}{1+(\,|\mathcal{A}_i|-1\,)\,e^{-|\mathcal{A}_i|\beta}}\right)^{\delta(x_i,x'_i)}$$

where $|\mathcal{A}_i|$ is the number of values in the alphabet of the i-th attribute.

Table 1 shows sample results of experiments carried out on five UCI data sets having a majority of categorical features. In each experiment, a voted perceptron was trained using 10 rounds for each kernel. Results are reported for the setting of the diffusion coefficient β achieving the best cross-validated error rate. The results are averaged across 40 random 80%/20% splits of the training/test data. In addition to the error rates, also shown is the average number of support vectors (or perceptrons) used. We see that even when the difference between the two kernels is not statistically significant, as for the `Hepatitis` dataset, the diffusion kernel results in a significantly sparser representation. The reduction in error rate varies, but the simple diffusion kernel generally performs well.

The performance over a range of values of β on the `Mushroom` data set is shown in Figure 2. We note that this is a

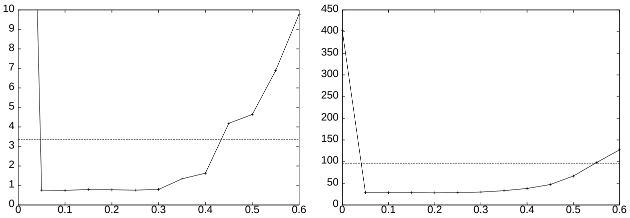

Figure 2. The average error rate (left) and number of support vectors (right) as a function of the diffusion coefficient β on the `Mushroom` data set. The horizontal line is the baseline performance using the Hamming kernel.

very easy data set for a symbolic learning algorithm, since it can be learned to a high accuracy with a few simple logical rules. However, standard kernels perform poorly on this data set, and the Hamming kernel has an error rate of 1.40%. The simple diffusion kernel reduces the error to 0.007%.

6. Conclusions

We have presented a natural approach to constructing kernels on graphs and related discrete objects by using the analogue on graphs of the heat equation on Riemannian manifolds. The resulting kernels are easily shown to satisfy the crucial positive semi-definiteness criterion, and they come with intuitive interpretations in terms of random walks, electrical circuits, and other aspects of spectral graph theory. We showed how the explicit calculation of diffusion kernels is possible for specific families of graphs, and how the kernels correspond to standard Gaussian kernels in a continuous limit. Preliminary experiments on categorical data, where the standard kernel methods of Euclidean space were previously not applicable, indicate that diffusion kernels can be effectively used with standard margin-based classification schemes. While the tensor product construction allows one to incrementally build up more powerful kernels from simple components, explicit formulas will be difficult to come by in general. Yet the use of diffusion kernels may still be practical when the underlying graph structure is sparse by using standard sparse matrix techniques.

It is often said that the key to the success of kernel-based algorithms is the implicit mapping from a data space to some, usually much higher dimensional, feature space which better captures the structure inherent in the data. The motivation behind the approach to building kernels presented in this paper is the realization that the kernel is a general representation of this inherent structure, independent of how we represent individual data points. Hence, by constructing a kernel directly on whatever object the data points naturally lie on (e.g. a graph), we can avoid the arduous process of forcing the data through any Euclidean space altogether. In effect, the kernel trick is a method for unfolding structures in Hilbert space. It can be used to unfold non-trivial correlation structures between points in Euclidean space, but it is equally valuable for unfolding other types of structures which intrinsically have nothing to do with linear spaces at all.

References

Albert, R., & Barabási, A. (2002). Statistical mechanics of complex networks. *Review of Modern Physics*, *74*, 47.

Belkin, M., & Niyogi, P. (2001). *Laplacian eigenmaps for dimensionality reduction and data representation.* (Technical Report TR-2002-01). Computer Science Department, University of Chicago.

Berg, C., Christensen, J., & Ressel, P. (1984). *Harmonic analysis on semigroups: Theory of positive definite and related functions*. No. 100 in Graduate Texts is Mathematics. Springer-Verlag.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*, 121–167.

Chung, F. R. K. (1997). *Spectral graph theory*. No. 92 in Regional Conference Series in Mathematics. American Mathematical Society.

Chung, F. R. K., & Yau, S.-T. (1999). Coverings, heat kernels and spanning trees. *Electronic Journal of Combinatorics*, *6*.

Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological sequence analysis probabilistic models of proteins and nucleic acids*. Cambridge University Press.

Feller, W. (1971). *An introduction to probability theory and its applications*, vol. II. Wiley. Second edition.

Freund, Y., & Schapire, R. E. (1999). Large margin classification using the perceptron algorithm. *Machine Learning*, *37*, 277–296.

Haussler, D. (1999). *Convolution kernels on discrete structures* (Technical Report UCSC-CRL-99-10). Department of Computer Science, University of California at Santa Cruz.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Proceedings of ECML-98, 10th European Conference on Machine Learning* (pp. 137–142).

Mackay, D. J. C. (1997). Gaussian processes: A replacement for neural networks? *NIPS tutorial*. Available from `http://wol.ra.phy.cam.ac.uk/pub/mackay/`.

Mika, S., Schölkopf, B., Smola, A., Müller, K., Scholz, M., & Rätsch, G. (1998). Kernel PCA and de-noising in feature spaces. *Advances in Neural Information Processing Systems 11*.

Saul, L. K., & Roweis, S. T. (2001). An introduction to locally linear embedding. Available from `http://www.cs.toronto.edu/~roweis/lle/`.

Schoenberg, I. J. (1938). Metric spaces and completely monotone functions. *The Annals of Mathematics*, *39*, 811–841.

Schölkopf, B., & Smola, A. (2001). *Learning with kernels*. MIT Press.

Szummer, M., & Jaakkola, T. (2002). Partially labeled classification with Markov random walks. *Advances in Neural Information Processing Systems*.

Watkins, C. (1999). Dynamic alignment kernels. In A. J. Smola, B. Schölkopf, P. Bartlett, and D. Schuurmans (Eds.), *Advances in kernel methods*. MIT Press.