Available online at www.sciencedirect.com

SCIENCE DIRECT®

Neural Networks 18 (2005) 1087–1092

Neural Networks

www.elsevier.com/locate/neunet

2005 Special Issue

An incremental regression method for graph structured data

Menita Carozza[a], Salvatore Rampone[b,*]

[a]Department PEMEIS, Università del Sannio, Piazzetta Vari, 82100 Benevento, Italy
[b]DSGA and RCOST, Università del Sannio, Via Port'Arsa 11, 82100 Benevento, Italy

Abstract

In this paper, we consider learning problems defined on graph-structured data. We propose an incremental supervised learning algorithm for network-based estimators using diffusion kernels. Diffusion kernel nodes are iteratively added in the training process. For each new node added, the kernel function center and the output connection weight are decided according to an empirical risk driven rule based on an extended chained version of the Nadaraja–Watson estimator. Then the diffusion parameters are determined by a genetic-like optimization technique.

Keywords: Graph structured data; Diffusion kernel; Incremental estimator; Empirical risk; Diffusion parameter; Genetic-like optimization

1. Introduction

Graph like structures occur in data in many guises, and are widely used to model and compare complex structures. In chemistry or molecular biology, for example, it might be anticipated that molecules with 'similar' chemical structures will have broadly similar properties. In the same way, when predicting the functions of unannoted proteins based on a protein network, one relies on some notions of 'closeness' or 'distance' among the nodes. However, like much real-world structured data, graphs have not in general a natural vectorial representation, and the concepts of 'similarity', and 'distance', well defined in the Euclidean spaces, become quite fuzzy.

Starting from the convolution kernel idea introduced by Haussler (Haussler, 1999), recent works (Hammer & Jain, 2004; Kondor & Lafferty, 2002; Nicotra, Micheli, & Starita, 2004; Tan & Wang, 2004; Watkins, 1999) were directly motivated by applying suitable kernel methods to discrete, categorical data. The common idea behind kernel-based algorithms (Aizerman, Braverman, & Rozonoér, 1964; Boser, Guyon, & Vapnik, 1992; Kimeldorf & Wahba, 1971; Schölkopf & Smola, 2002) is to express the similarities between pairs of points in a data space in terms of a kernel function. Obviously, a good kernel should calculate a high or low similarity between examples according to the known or hypothesized structure of the data space.

Recently, the use of *discrete diffusion kernels* (Kondor & Lafferty, 2002) has been suggested for graph-structured data. They have been successfully applied for making predictions from biological networks (Vert & Kanehisa, 2003). While such kernels appear to be appropriate to express similarities between graph-structured data, they evidence the difficulty of correctly modulating the associate *diffusion parameter* (Tsuda & Stafford Noble, 2004).

In this paper, we combine the diffusion kernels with an incremental network-based estimator (Carozza & Rampone, 1999, 2001; Chentouf, Jutten, Maignan, & Kanevski, 1997). Diffusion kernel nodes are iteratively added in the training process. For each new added node, the activation function center and the output connection weight are decided according to an extended chained version of the Nadaraja–Watson estimator (Bishop, 1995; Schioler & Hartmann, 1992; Specht, 1990). Then, the diffusion parameters of the kernel functions are determined by an empirical risk driven rule based on a genetic-like optimization technique (Carozza & Rampone, 1999, 2001). We also consider a theoretical upper bound to the number of kernel nodes.

Assuming noisy data, the residual error must be at least equal to the noise. Then, we stop the adding procedure when

* Corresponding author. Tel.: +39 0824 305151; fax: +39 824 23013.
E-mail addresses: carozza@unisannio.it (M. Carozza), rampone@unisannio.it (S. Rampone).

0893-6080/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2005.07.008

the residual error is under a fixed threshold related to the noise level.

While the resulting system is conceived for function approximation, in the experiments we limit its application to classification problems.

This paper is organized as follows. In Section 2, we introduce the diffusion kernels; in Section 3 and in Section 4, we describe the incremental estimator and the network training mechanism, respectively, bounding the kernel nodes and summarizing all the results in a single learning procedure; finally, in Section 5, we describe the experimental results on some classification problems.

2. Diffusion kernels on graphs

Let Ω be a set and

$$K : \Omega \times \Omega \rightarrow \mathcal{R} \tag{1}$$

According to Haussler (Haussler, 1999), K is a kernel on $\Omega \times \Omega$ if K is symmetrical, i.e.

$$K(x, x') = K(x', x) \quad \forall\, x, x' \in \Omega \tag{2}$$

and K is positive semidefinite. However, in this paper we consider kernels in a more general sense, where the positive semidefiniteness is not necessarily satisfied.

In the discrete case, for finite Ω, the kernel can be uniquely represented by a square matrix K

$$K = \{K(x, x') | x, x' \in \Omega\} \tag{3}$$

In particular, we consider exponential kernels (Kondor & Lafferty, 2002) represented by

$$K_\beta = e^{\beta H} \tag{4}$$

where H is a square and symmetric matrix called *generator matrix.*

An exponential kernel K_β^n over length n sequences $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, can be viewed as

$$K_\beta^n(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^{n} K_\beta(x_i, x_i') \tag{5}$$

Now, let us consider an undirected and unweighted graph defined by the vertex set

$$V = \{x_1, x_2, \ldots, x_n\} \tag{6}$$

and the edge set

$$E = \{e_{i,j} = (x_i, x_j)\} \subset V \times V \tag{7}$$

The generator matrix H of an exponential kernel for such kind of graph can be defined by

$$H_{i,j} = \begin{cases} 1, & e_{i,j} \in E \\ -d_i, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where d_i is the degree of vertex i, i.e. the number of edges emanating from vertex i.

When H is defined as in (8), the equation

$$\frac{d}{d\beta} K_\beta = H K_\beta \tag{9}$$

is the well known heat or *diffusion equation* (Kondor & Lafferty, 2002), and K_β, as solution of this equation, is called *diffusion kernel.*

Starting from the identity matrix $K_0 = I$, K_β can be interpreted as the product of a continuous process, expressed by H, gradually transforming it from the identity matrix to a kernel with increasing off-diagonal effects as β increases. So β is called *diffusion parameter.*

3. Regression method

Let $\mathbf{X}$ be a graph data set and Y a subset of $\mathcal{R}$. We consider the problem of approximating functions

$$f : \mathbf{X} \rightarrow Y \tag{10}$$

given a set of training data

$$\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \ldots, (\mathbf{x}_p, t_p)\} \tag{11}$$

where

$$\mathbf{x}_k \in \mathbf{X}, \quad t_k = f(\mathbf{x}_k) + \text{rand}[-a, a], \tag{12}$$

and *rand*$[-a, a]$ is a random additive noise from a zero mean uniform distribution.

Given the set (11) of training data, the complete description of the statistical properties of the data generator is given by the unknown probability density

$$p(\mathbf{x}, t) \tag{13}$$

in the joint input-target space. Let us consider a subset of (11)

$$\{(\mathbf{x}_{c_1}, t_{c_1}), (\mathbf{x}_{c_2}, t_{c_2}), \ldots, (\mathbf{x}_{c_M}, t_{c_M})\} \tag{14}$$

where $c_j \in \{1, 2, \ldots, p\}$ and $M \le p$, and the diffusion kernel

$$K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j}) \tag{15}$$

where β_j is the diffusion parameter and $\mathbf{x}_{c_j}$ is the center. We use as an estimator of (13) an adaptive mixture model, so having the following expression for the regression

$$\text{net}_M(\mathbf{x}) = \frac{\sum_{j=1}^{M} w_j K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j})}{\sum_{j=1}^{M} K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j})} \tag{16}$$

where w_j will be specified in the following. This can be viewed as a kernel network based on the Nadaraya–Watson estimator (Bishop, 1995), extended to the multivariate case (Ghahramani & Jordan, 1994), and restricted to a subset of data.

The estimator performance is valued by means of the empirical risk

$$E_M = \frac{1}{p}\sum_{k=1}^{p}(t_k - \mathrm{net}_M(\mathbf{x}_k))^2 \tag{17}$$

3.1. Incremental estimator

The estimator (16) can be easily chained to obtain an incremental approach

$$\mathrm{net}_M(\mathbf{x}) = \frac{\mathrm{net}_{M-1}(\mathbf{x})\mathrm{den}_{M-1}(\mathbf{x}) + w_M K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j})}{\mathrm{den}_M(\mathbf{x})} \tag{18}$$

where

$$\mathrm{den}_M(\mathbf{x}) = \sum_{j=1}^{M} K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j}) \tag{19}$$

To explicitly reduce the residual approximation error given by (17), in (18) we assign to w_j the value

$$w_j = t_{c_j} - \mathrm{net}_{j-1}(\mathbf{x}_{c_j}) \tag{20}$$

where c_j is such that

$$(t_{c_j} - \mathrm{net}_{j-1}(\mathbf{x}_{c_j}))^2 = \max_{k=1,\ldots,p}(t_k - \mathrm{net}_{j-1}(\mathbf{x}_k))^2 \tag{21}$$

i.e. the training point in which net_{j-1} has the worst approximation performance.

Moreover, since this quantity is affected by the normalization factor den_j, we multiply w_j by this factor, so having

$$\mathrm{net}_M(\mathbf{x}) = \frac{\sum_{j=1}^{M} w_j \mathrm{den}_j(\mathbf{x}) K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j})}{\sum_{j=1}^{M} K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j})} \tag{22}$$

or, in the chained version

$$\mathrm{net}_M(\mathbf{x}) = \frac{\mathrm{net}_{M-1}(\mathbf{x})\mathrm{den}_{M-1}(\mathbf{x})}{\mathrm{den}_M(\mathbf{x})} + w_M K_{\beta_j}(\mathbf{x}, \mathbf{x}_{c_j}) \tag{23}$$

4. Training

The training phase consists of an iterative process, which adds kernel nodes until a target error level is reached. At each step M, the kernel center is selected by (21). We have only to set the β_j values.

4.1. Diffusion parameters

The training of β_j's consists in two stages: a 'local training', where only the last added node diffusion parameter is updated, and a 'fine tuning' of all the net diffusion parameters. These parameters are optimized by means of a genetic-like algorithm. A population of individuals is generated by applying the following mutation

$$\beta_{j,\mathrm{new}} \leftarrow \beta_j(1+\psi) \tag{24}$$

where ψ is a random number in a fixed range $[-\alpha, \alpha]$. If the empirical risk associated to $\beta_{j,\mathrm{new}}$ is less than the same quantity associated to β_j, then $\beta_j \leftarrow \beta_{j,\mathrm{new}}$. This step is iterated a fixed number of times (tr), and it aims to improve the empirical risk, as evidenced by the following theorem

Theorem 1 Let $I_1, I_2,\ldots,I_\nu$ be disjoint intervals of $\mathfrak{R}$ such that $\cup_{k=1}^{\nu} I_k = I$, and $|I_k| = |I_q|$, $\forall k, q \in \{1,\ldots,\nu\}$. Let I_m be the interval in which $E_M \le E'_M$, where E_M is the empirical risk of net_M with $\beta_j \in I_m$, and E'_M the empirical risk of net_M with $\beta_j \in I_k$, $k \ne m$. Then the expected number of drawings in I necessary to acquire at least a value in I_m is

$$\nu \log \nu \tag{25}$$

Proof (see (Feller, 1970) example IX.3.d, (Carozza & Rampone, 2001; Rampone, 1995))

4.2. Kernel nodes

The average error of net_M estimator, the expected risk (Niyogi & Girosi, 1996), is unknown because our only source of information is the data set (11). So, we approximate the expected risk by the empirical one, i.e. by the sample mean square error (17). Here, we bound the network node number M, and so the size of the subset (14), in order to guarantee the empirical risk converges to the expected one for target functions belonging to a non-trivial class (Niyogi & Girosi, 1996, 1994). Some related definitions are reported in the Appendix A. Let

$$f_{M,p} = \arg\min_{\beta_j}\sum_{k=1}^{p}(t_k - \mathrm{net}_M(\mathrm{x}_k))^2 = \arg\min_{\beta_j} E_M \tag{26}$$

Theorem 2 Let H_M be the class of network based estimators with n input nodes and M hidden nodes as defined in (22) and f be the target function (10), element of the Bessel potential space $\mathcal{L}_1^m(R^n)$ of order m, with $m > n/2$. Given the data set (11) is obtained by randomly sampling f in presence of noise, and that the noise distribution has compact support, then for any $0<\delta<1$, with probability greater than $1-\delta$, the following bound for the generalization error holds

$$\|f - f_{M,p}\|^2_{L^2(P)} \le O\left(\frac{1}{M}\right) + O\left(\left[\frac{Mn\ln(Mp) - \ln\delta}{p}\right]^{1/2}\right) \tag{27}$$

Proof: Niyogi and Girosi (Niyogi & Girosi, 1996, 1994) proved the estimate (27) for functions belonging to the class

$$\left\{ f \middle| f = \sum_{j=1}^{M}\gamma_j \exp(-\|\mathbf{x}-\mu_j\|^2/2\beta_j) \text{ and } \sum_{j=1}^{M}\gamma_j \text{ is bounded} \right\}.$$

We note that the result is also true for the class

$$\left\{f | f = \sum_{j=1}^{M} \gamma_j K_{\beta_j}(\mathbf{x}-\mu_j) \text{ and } \sum_{j=1}^{M} \gamma_j \text{ is bounded}\right\}$$

where $K_{\beta_j}(\mathbf{x}, \mu_j)$ is the kernel we use. We prove the estimator (22) belongs to this class. In fact $\mathrm{net}_M(\mathbf{x})$ can be written as

$$\mathrm{net}_M(\mathbf{x}) = \sum_{j=1}^{M} \frac{w_j \mathrm{den}_j(\mathbf{x})}{\mathrm{den}_M(\mathbf{x})} K_{\beta_j}(\mathbf{x}, \mu_j) \tag{28}$$

Setting

$$\frac{w_j \mathrm{den}_j(\mathbf{x})}{\mathrm{den}_M(\mathbf{x})} = \gamma_j(\mathbf{x}) \tag{29}$$

we have

$$\mathrm{net}_M(\mathbf{x}) = \sum_{j=1}^{M} \gamma_j(\mathbf{x}) K_{\beta_j}(\mathbf{x}, \mu_j) \tag{30}$$

At each step M, the kernel center is the point $\mathbf{x}_{c_M}$ such that

$$(t_{c_M} - \mathrm{net}_{M-1}(\mathbf{x}_{c_M}))^2 = \max_{k=1,\dots,p} (t_k - \mathrm{net}_{M-1}(\mathbf{x}_k))^2 \tag{31}$$

The formula (20) implies

$$|w_j| \le 2|t_{c_1}| \quad j = 1, 2, \dots, M \tag{32}$$

Since

$$\frac{\mathrm{den}_j(\mathbf{x})}{\mathrm{den}_M(\mathbf{x})} \le 1 \quad j = 1, 2, \dots, M \tag{33}$$

we have

$$\sum_{j=1}^{M} \gamma_j(\mathbf{x}) \le 2M|t_{c_1}| = \bar{M} \tag{34}$$

By this result, as in (Niyogi & Girosi, 1996), we can obtain an upper bound for the node number M by viewing it as a function of the given number of examples (11). Namely, for p examples, $M \le p^r$, $r < 1$.

4.3. Learning algorithm

The learning algorithm can be summarized as follows

```
Step 1: Input. $G=\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2),\dots(\mathbf{x}_p, t_p)\}$ the set of
        training instances, $T$ the target training approxi-
        mation error, $r$ the exponent that maximizes $M$, $\alpha$
        the $\beta_j$ mutation parameter, $tr$ the number of search
        iteration of each $\beta_j$.
        Initialization. Set $M=0$.
Step 2: While the empirical risk $E_M > T$ and $M < p^r$ do
    2.1 Choose the center. Select $k$ such that $|t_k - \mathrm{net}_M(\mathbf{x}_k)|$
        is maximum.
    2.2 Add a new node. Set $M \leftarrow M+1$, initialize $\beta_M$,
        $w_M = t_k - \mathrm{net}_{M-1}(\mathbf{x}_k)$, $\mathbf{x}_{c_M} = \mathbf{x}_k$.
    2.3 Adapt $\beta_j$ parameters: For all nodes $j, j \le M$, starting
        from the last do
    2.3.1 While $E_M$ decreases do
    2.3.1.1 For $tr$ times do
            – Mutate $\beta_j$. $\beta_{j,\mathrm{new}} \leftarrow \beta_j(1+rand[-\alpha, \alpha])+\varepsilon$.
              Compute the corresponding risk $E_{M,\mathrm{new}}$.
            – Update $\beta_j$. if $E_{M,\mathrm{new}} < E_M$ then $\beta_j \leftarrow \beta_{j,\mathrm{new}}$.
Step 3: Output $net_M$.
```

5. Experiments

In the reported experiments, we consider an extension of diffusion kernels on the hypercube regarded as a graph. This is in order to compare our results to the Kondor and Lafferty ones (Kondor & Lafferty, 2002).

Namely the diffusion kernel we use here is

$$K_\beta(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^{n} \left(\frac{1 - e^{-|\mathcal{A}_i|\beta}}{1 + (|\mathcal{A}_i| - 1)e^{-|\mathcal{A}_i|\beta}} \right)^{1-\delta(\mathbf{x}_i, \mathbf{x}'_i)} \tag{35}$$

where $\delta(x, y) = 1$ if $x = y$ and 0 otherwise, $|\mathcal{A}_i|$ is the size of $\mathcal{A}_i$ and $\mathcal{A}_i$ is the set of values of the ith attribute.

Experiments are performed on 3 of 5 UCI data sets (Blake & Merz, 1998) used in previous papers to assess the diffusion kernel[1]. Each dataset is randomly divided in training (80%) and test (20%) data. A summary of dataset information is reported in Table 1.

According to the previous usage, for each data set only the categorical features are used. This means that we use only 13 of the 20 Hepatitis data set attributes. Tables 2 and 3 report the results obtained by our method by lowering the target from 0.5000 to 0.0156, while Table 4 (Votes) reports the results in the target range 0.5000–0.0078. The reported results are mean values over 40 random selections of the data into training and test sets. The test error sample standard deviation (SSD) is also reported.

Fig. 1 shows the mean of the diffusion parameters β_j, varying the target, on the Hepatitis data set.

The distribution of the β_j values evidences the existence of singular points, selected by the error contribution, whose β_j values tend to be minimal. Fig. 2 shows the β_j distribution on the Votes data set, where the problem is quite relevant.

Table 5 reports comparative results on UCI data sets. Each row reports the data set name, the number of used attributes, the maximum attribute cardinality, the errors obtained by using a Large Margin Classifier (LMC) (Kondor & Lafferty, 2002) using Hamming Distance and Diffusion Kernel (35), and the results of our method (CRM).

[1] Regarding the remaining 2 data sets, Mushrooms is not considered since diffusion kernels obtain semiperfect recognition independently from the model, and Income has been recently changed.

Table 1
UCI data sets

Data set	Instances (Train+Test)	Classes	Attributes
Breast cancer	286 (228+58)	2	9
Hepatitis	155 (124+31)	2	20
Votes	435 (348+87)	2	16

Table 2
Breast cancer results

Target	Kernels	Training error	Test error	SSD
0.5000	1	0.2205	0.2181	0.0460
0.2500	1	0.1914	0.2038	0.0294
0.1250	21	0.1225	0.1473	0.0196
0.0625	30	0.0588	0.1087	0.0618
0.0312	44	0.0282	0.0729	0.0797
0.0156	49	0.0154	0.0439	0.0575

The number of Support Vector (SV) for the first two methods and the number of centers (M) for CRM are reported in brackets.

Our experiments evidence a reduction of the mean number of diffusion kernel nodes necessary to obtain good performances for the used data. In a case (Hepatitis) this also involves an error reduction of more than 50%.

6. Concluding remarks

In this paper, we combine the diffusion kernels with an incremental network-based estimator. The resulting system can be applied to learning problems defined on graph-structured data. Diffusion kernel nodes are iteratively added in the training process. For each new added node, the kernel function center and the output connection weight are decided according to an extended chained version of the Nadaraja–Watson estimator. The diffusion parameters of the kernel functions are determined by an empirical risk driven rule based on a genetic-like optimization technique. We also considered a theoretical upper bound to the number of kernel nodes.

Experimental results on classification problems are reported to illustrate the validity and effectiveness of the proposed method. Our system evidences a dramatic reduction of the mean number of diffusion kernel nodes necessary to obtain good performances for the used data. In a case this also involves a significant error reduction.

The use of diffusion kernels combined with an incremental learning algorithm tends towards an iterative learning mechanism for Support Vector Machines (SVM) (deKruif & deVries, 2001): SVM tends to prune a set of support vectors during the parameter optimization, while our model grows a small set of kernel nodes related to the generalization ability.

As the number of center grows, the cost of the diffusion parameters fine-tuning, while polynomial, may come expensive. However, the fine-tuning of a diffusion parameter can be limited as a function of the center selection index, and this will be a matter of a future work.

Table 3
Hepatitis results

Target	Kernels	Training error	Test error	SSD
0.5000	1	0.1563	0.1687	0.0414
0.2500	1	0.1731	0.1792	0.0421
0.1250	7	0.1188	0.1577	0.0561
0.0625	15	0.0573	0.1080	0.0451
0.0312	19	0.0118	0.0941	0.0554
0.0156	21	0.0057	0.0807	0.0413

Table 4
Votes results

Target	Kernels	Training error	Test error	SSD
0.5000	1	0.2365	0.2409	0.0385
0.2500	1	0.1517	0.2435	0.0527
0.1250	50	0.1244	0.1710	0.0302
0.0625	86	0.0605	0.1708	0.0800
0.0312	89	0.0277	0.1700	0.0120
0.0156	97	0.0143	0.1649	0.0119
0.0078	131	0.0051	0.1301	0.0278

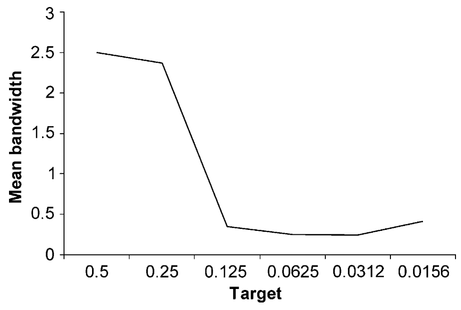

Fig. 1. Mean of the diffusion parameters β_j, varying the target, on the Hepatitis data set.

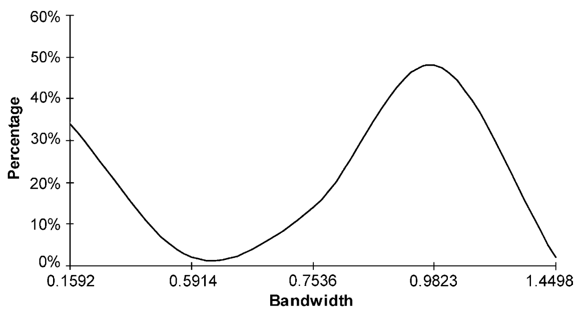

Fig. 2. Diffusion parameter distribution on the Votes data set.

Table 5
Comparative results on UCI data sets

Data set	Used attributes	Max attribute cardinality	Hamming distance LMC error (SV)	Diffusion kernel LMC error (SV)	Diffusion kernel CRM error (M)
Breast cancer	9	10	7.44% (206)	3.70% (43)	4.39% (49)
Hepatitis	13	2	19.50% (420)	18.80% (192)	8.07% (21)
Votes	16	2	4.79% (176)	4.53% (60)	13.01% (131)
Mean			10.58% (267)	9.01% (98)	8.49% (67)

Appendix A

The *Bessel potential space of order m*, $\mathcal{L}_1^m$ is defined as follows:

$$\mathcal{L}_1^m = \{f|f = \lambda * G_m, m > kz, |\lambda|_{R^k} \le M\} \tag{A1}$$

where

M is a positive number;
λ is a signed Radon measure on the Borel sets of R^k;
G_m is the Bessel–Macdonald kernel, i.e., the inverse Fourier transform of $\bar{G}_m(s) = (1 + ||s||^2)^{-m/2}$;
the symbol * stands for the convolution operation;
$|\lambda|_{R^k}$ is the total variation of the measure λ.

- $L^2(P)$ is the set of functions whose square is integrable with respect to the measure defined by a probability distribution P.
- The norm in $L^2(P)$ is defined by

$$||f||^2_{L^2(P)} = \int_{R^k} f^2(\mathbf{x})P(\mathbf{x})\mathrm{d}\mathbf{x} \tag{A2}$$

References

Aizerman, M. A., Braverman, E. M., & Rozonoér, L. I. (1964). Theoretical foundations of the potential function method in pattern recognition and learning. *Automation and Remote Control*, *25*, 821–837.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford: Clarendon Press.

Blake, C. L., & Merz, C. J. (1998). *UCI Repository of machine learning databases*. Irvine, CA: University of California, Department of Information and Computer Science http://www.ics.uci.edu/~mlearn/MLRepository.html.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In D. Haussler, *Proceedings of the 5th annual ACM workshop on computational learning theory* (pp. 144–152). Pittsburgh: ACM Press.

Carozza, M., & Rampone, S. (1999). Function approximation from noisy data by an incremental RBF network. *Pattern Recognition*, *32*(12), 2081–2083.

Carozza, M., & Rampone, S. (2001). An incremental multivariate regression method for function approximation from noisy data. *Pattern Recognition*, *34*(3), 179–186.

Chentouf, R., Jutten, C., Maignan, M., & Kanevski, M. (1997). Incremental neural networks for function approximation. *Nuclear Instruments and Methods in Physics Research*, *A389*, 268–270.

Feller, W. (1970). (3rd ed.) *An introduction to probability theory and its applications*, Vol. 1. New York: Wiley.

Ghahramani, Z., & Jordan, M. I. (1994). Supervised learning from incomplete data via an EM approach. In J. D. Cowan, G. T. Tesauro, & J. A. lspector, *Advances in neural information processing systems* (Vol. 6) (pp. 120–127). San Mateo: Morgan Kaufmann.

Hammer, B., & Jain, B. J. (2004). Neural methods for non-standard data. In M. Verleysen (Ed.), *European symposium at artificial neural networks'2004* (pp. 281–292). D-side publications.

Haussler, D. (1999). Convolution kernels on discrete structures. Technical report. Santa Cruz: Department of Computer Science, University of California.

Kimeldorf, G., & Wahba, G. (1971). Some results on Tchebychean spline functions. *Journal of Mathematical Analysis and Applications*, *33*, 82–95.

Kondor, R. I., & Lafferty, J. (2002). Diffusion kernels on graphs and other discrete input spaces. In C. Sammut, & A. Hoffmann, *Proceedings of the nineteenth international conference on machine learning (ICML)* (pp. 315–322). San Francisco: Morgan Kaufmann.

deKruif, B. J., & deVries, T. J. A. (2001). On using a Support Vector Machine in Learning Feed-Forward Control. In 2001 IEEE/ASME international conference on advanced intelligent mechatronics (AIM'01), Vol. 1 (pp. 272–277). Como: IEEE Press.

Nicotra, L., Micheli, A., & Starita, A. (2004). Fisher Kernel for tree structured data. In *Proceedings of the IEEE international joint conference on neural networks IJCNN'2004* (pp. 1917–1922). Budapest: IEEE press.

Niyogi, P., & Girosi, F. (1994). On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions. MIT AI LAB memo no. 1467.

Niyogi, P., & Girosi, F. (1996). On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions. *Neural Computation*, *8*, 819–842.

Rampone, S. (1995). Linear codes interpolation from noisy patterns by means of a vector quantization process. *Computers and Mathematics with Applications*, *30*(11), 91–106.

Schölkopf, B., & Smola, A. J. (2002). *Learning with Kernels*. Boston: The MIT Press.

Schioler, H., & Hartmann, U. (1992). Mapping neural network derived from the parzen window estimator. *Neural Networks*, *5*(6), 903–909.

Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks*, *3*(1), 109–118.

Tan, Y., & Wang, J. (2004). A support vector machine with a hybrid kernel and minimal Vapnik-Chervonenkis dimension. *IEEE Transactions on Knowledge and Data Engineering*, *16*(4), 385–395.

Tsuda, K., & Stafford Noble, W. (2004). Learning kernels from biological networks by maximizing entropy. *Bioinformatics*, *20*(Suppl. 1), i326–i333.

Vert, J. P., & Kanehisa, M. (2003). Graph-driven features extraction from microarray data using diffusion kernels and kernel CCA. In S. Becker, S. Thrun, & K. Obermayer, *Advances in neural information processing systems* (Vol. 15) (pp. 1425–1432). Cambridge: MIT Press.

Watkins, C. (1999). Dynamic alignment kernels. In A. J. Smola, B. Schölkopf, P. Bartlett, & D. Schuurmans (Eds.), *Advances in large margin classifiers* (pp. 39–50). Cambridge: MIT press.