Proc. International Conference on Soft Computing (SOCO'99), Genoa, Italy, pp.610-619 (1999).

A gradient based technique for generating sparse representation in function approximation

Sethu Vijayakumar*and Si Wu
RIKEN Brain Science Institute,
Hirosawa 2-1, Wako-shi, Saitama, Japan
{sethu,phwusi}@brain.riken.go.jp

1 Abstract

We provide an RKHS based inverse problem formulation[15] for analytically deriving the optimal function approximation when probabilistic information about the underlying regression is available in terms of the associated correlation functions as used in [9, 8]. On the lines of Poggio and Girosi[9], we show that this solution can be sparsified using principles of SVM and provide an implementation of this sparsification using a novel, conceptually simple and robust gradient based sequential method instead of the conventional quadratic programming routines.

2 Introduction

In this paper, we consider the standard regression task of estimating an underlying multivariate function (or representation) f from a given set of finite training data, $\{\boldsymbol{x}_m, y_m\}_{m=1}^{M}$. This general framework encompasses the problems of signal reconstruction and image representation in artificial as well as biological systems. Here, we assume that we are given some information about the probabilistic distribution of the underlying function f in form of the associated correlation function R [9], which we will formalize mathematically in the next section. Then, the regression problem, from the standpoint of image processing, can be stated as one of reconstructing a specific image f given it's pixel values at discrete locations, where f corresponds to the input image and $\boldsymbol{x}$ represents a vector in the image plane.

It has been argued that one of the major goals of sensory processing should be to reduce dimensionality of the input space. There is experimental and statistical evidence [5, 10, 11] which show that representation of natural images uses a parsimonious parametrization of a suitable subspace to represent the images during sensory processing.

In this work we will, at first, focus on providing a framework for analytically obtaining optimal (not necessarily sparse or parsimonious) approximations to the underlying regression which incorporates the apriori knowledge in form of correlation functions. We will show that for a particular choice of the original function space, the approximation reduces to the linear combination of local correlation kernels-moreover, this provides a direct justification for the use of correlation kernels in image reconstruction as done in [9, 8]. This particular choice of the function space enables us to use the principles of sparsification described in Poggio and Girosi[9] to find a more parsimonious representation of the solution. Traditionally, the sparsification based on the principles of SVM is carried out using quadratic programming routines [12, 13]. Here, we present a novel gradient based method [16] to arrive at the sparsified solution, an alternative that retains all the guarantees of the Structural Risk Minimization (SRM) principle while being conceptually much simpler to implement. The algorithm is assured of convergence to global maxima within theoretically derived bounds of the learning rate, does not suffer from the numerical instabilities of the quadratic programming packages and is computationally very efficient.

3 Function approximation as an inverse problem

In this section, we will review the inverse problem formulation, details of which can be found in [15]. Let H be the set of functions which includes $f(\boldsymbol{x})$, the function to be approximated. Assume that H is a Reproducing Kernel Hilbert Space(RKHS) with a *reproducing kernel* $K(\boldsymbol{x}, \boldsymbol{x}')$. The reproducing ker-

*corresponding author

nel satisfies the Mercer's condition[4], i.e.,

$$K(\boldsymbol{x}, \boldsymbol{x}') = K(\boldsymbol{x}', \boldsymbol{x}) \text{ and}$$
$$\int\int K(\boldsymbol{x}, \boldsymbol{x}') f(\boldsymbol{x}) f(\boldsymbol{x}') d\boldsymbol{x} d\boldsymbol{x}' \geq 0$$

for every $f \in H$ and has the following properties:

1. For all $\boldsymbol{x}'$ in the domain of f, $K(\boldsymbol{x}, \boldsymbol{x}')$ is a function in H.

2. For any function f in H, it holds that

$$< f(\boldsymbol{x}), K(\boldsymbol{x}, \boldsymbol{x}') >= f(\boldsymbol{x}'), \tag{1}$$

   where the left hand side of eq.(1) denotes the inner product in H.

In the theory of Hilbert space, arguments are developed by regarding a function as a point in that space. Thus, things such as 'value of a function at a point' cannot be discussed under the general framework of Hilbert space. However, if the Hilbert space has a reproducing kernel[1], then it is possible to deal with the value of a function at a point. Indeed, if we define functions $\psi_m(\boldsymbol{x})$ as

$$\psi_m(\boldsymbol{x}) = K(\boldsymbol{x}, \boldsymbol{x}_m) : 1 \leq m \leq M, \tag{2}$$

then, the value of f at a sample point $\boldsymbol{x}_m$ is expressed in Hilbert space language as the inner product of f and ψ_m as

$$f(\boldsymbol{x}_m) =< f, \psi_m > . \tag{3}$$

Once the training set $\{\boldsymbol{x}_m\}_{m=1}^M$ is fixed, the vector $\boldsymbol{y} \equiv (y_1\, y_2 \ldots y_M)^T$ is uniquely determined from f. So, we can introduce an operator A which transforms f to $\boldsymbol{y}$:

$$\boldsymbol{y} = Af. \tag{4}$$

The operator A, called the *sampling operator*, becomes a linear operator even when we are concerned with nonlinear approximators. It is expressed by using the Schatten product as

$$A = \sum_{m=1}^{M} e_m \otimes \overline{\psi_m}, \tag{5}$$

where $\{e_m\}_{m=1}^M$ is the so-called natural basis in $\mathbb{R}^M$, i.e., the vector e_m is the M-dimensional vector consisting of zero elements except the element m equal to 1. The Schatten product denoted by $(. \otimes \bar{.})$ is defined by

$$(e_m \otimes \overline{\psi_m}) f =< f, \psi_m > e_m \tag{6}$$

[1] A Hilbert space always possesses a reproducing kernel if it is separable [2]

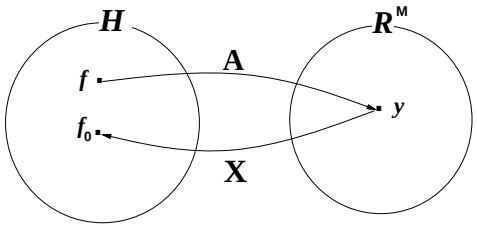

Figure 1: Learning as an inverse problem

and corresponds to an outer product of two vectors when a vectorial representation of the functions are possible. Hence, the approximation problem can be reformulated as the problem of obtaining an estimate, say f_0, to f from $\boldsymbol{y}$ in the model (See Fig.1). This can be considered as an *inverse problem* equivalent to obtaining an operator X which provides f_0 from $\boldsymbol{y}$:

$$f_0 = X\boldsymbol{y}. \tag{7}$$

We will refer to the operator X as the *learning operator*. This operator X can be optimized based on different optimization criteria. We will look at a particular cost function, the Wiener criterion, which utilizes the apriori information on the function ensemble correlation in the next section.

4 The Wiener cost criterion and analytical optimization

Most optimization criteria reduces errors in the sample space, i.e., reduce errors at the training location while using some form of regularization etc. This does not necessarily gaurantee good generalization ability. Since we do not have the knowledge of the original function f, it is expected that one cannot do better. However, when we have apriori information about the function correlation ensemble, we can analytically find optimal approximations which reduce errors in the original function space in an averaged sense over the entire ensemble.

4.1 The Wiener criterion

The functional representing the *Wiener criterion* J_W for the *noiseless* case is given as:

$$\min_X J_W[X] = E_f \|f_0 - f\|^2 = E_f \|XAf - f\|^2, \tag{8}$$

where $\|.\|$ is the norm in H and E_f is the expectation taken over the ensemble $\{f\}$. An operator

X satisfying the above criterion is called a Wiener learning operator. The criterion aims at reducing the difference between the original function f and the function f_0 reconstructed by using the learning operator X. This minimization is done in the *original function* space H and in an *averaged sense* with respect to the function ensemble.

4.2 Analytical batch and incremental solutions

The Wiener criterion can be transformed into a more useful form [15, 7]. Let R be the *correlation operator* of the function ensemble and be defined as

$$R = E_f(f \otimes \overline{f}). \tag{9}$$

This is the ensemble correlation function described in Penev and Atick [8] and used for constructing kernels in Poggio and Girosi[9]. Techniques for computing/generating this correlation function is described in Penev and Atick[8] which involves collecting generic images, scaling, aligning and cropping them and then, computing the correlation. By looking at the saddle points of the functional (8), it can be shown [7, 14] that the necessary and sufficient condition for the Wiener criterion to be satisfied by an operator X is given as

$$XARA^* = RA^*, \tag{10}$$

where A^* is the adjoint operator of A.

Theorem 1 (Batch Wiener approximation) *A general form of the solution of eq.(10) is given as*

$$X = RA^*U^\dagger + W(I - UU^\dagger), \tag{11}$$

where W is any operator from $\mathbb{R}^M$ to H, $U^\dagger$ represents the Moore-Penrose generalized inverse of U [1] and U is defined as

$$U = ARA^*. \tag{12}$$

The approximated function f_0 can be computed based on eq.(11) as

$$f_0 = Xy = RA^*U^\dagger y, \tag{13}$$

since $I - UU^\dagger$ lies in the null space of A.

Once the training set is fixed, A can be calculated using eqs.(5) and (2). Hence, corresponding to this sampling operator, a learning operator X satisfying the Wiener criterion can be obtained using eqs.(11) and (12). The function approximation using the Wiener criterion can also be carried out incrementally. Let A_m and f_m represent the sampling operator and the approximated function using m training data, respectively.

Theorem 2 (Incremental learning) *[15] The approximation due to $m+1$ training data, f_{m+1}, can be expressed as a function of the previous approximation f_m as*

$$f_{m+1} = f_m + \frac{y_{m+1} - f_m(x_{m+1})}{\phi_{mc}(x_{m+1})}\phi_{mc}. \tag{14}$$

where $\phi_m = R\psi_{m+1}$ and ϕ_{mc} is the projection of ϕ_m onto $\mathcal{N}(A_m)$ along $\mathcal{R}(RA_m^*)$.

Here, $\mathcal{R}(.)$ and $\mathcal{N}(.)$ refer to the range and the null space of an operator, respectively. This incremental learning is *exact* in the sense that the function approximation that results from applying this incremental scheme exactly coincides with the results obtained using the batch scheme, i.e., it is not an approximation.

5 Sparsifying the function representation

In using our functional analytic framework, we have so far not specifically dealt with which Hilbert space H to use. Vijayakumar and Ogawa [15] have looked at a variety of possible function spaces with characteristic properties. Here we concentrate on a particular choice of the function space H and it's corresponding kernel suitable for sparsifying the solution.

Let us consider the case of M training data and revert back to the batch approximation notation. It can be shown [14] that the approximated function f_0 using the Wiener optimization criterion is an oblique projection of f onto $\mathcal{R}(RA^*)$ along $\mathcal{R}(R) \cap \mathcal{N}(A)$, i.e., $f_0 \in \mathcal{R}(RA^*)$. However, if we choose a Hilbert function search space using the correlation operator on the lines of Poggio and Girosi[9] such that $H = \mathcal{R}(R)$, it is easily seen that f_0 can now be written as an orthogonal projection of f onto $\mathcal{R}(A^*)$ along $\mathcal{N}(A)$ [2], i.e., $f_0 \in \mathcal{R}(A^*)$. Since

$$A = \sum_{m=1}^{M} e_m \otimes \overline{K(x, x_m)} \quad \text{and} \tag{15}$$

$$A^* = \sum_{m=1}^{M} K(x, x_m) \otimes \overline{e_m}, \tag{16}$$

it is clear from the properties of the schatten operator that the approximated function f_0 can be represented as

$$f_0 = \sum_{i=1}^{M} a_i K(x, x_i), \tag{17}$$

[2] $\mathcal{R}(A^*)$ and $\mathcal{N}(A)$ are orthogonal decompositions of the approximation space H in this case

where a_i is a set of scalar coefficients. This is similar to the correlation kernel based approximation derived in Poggio and Girosi[9] which result from regularization functionals (Appendix B, [6]).

The resulting function approximation has an expansion in terms of the weighted sums of the correlation kernels at each training data location. These kind of kernels generated using the correlation function of the natural images have been shown to be local in nature (refer to LFA of Penev and Atick[8]). Also, the function representation is topographic in nature - the nearby values of x ellicit similar responses - because the kernels are indexed by the grid variable x. Locality and topography may be desirable feature in certain segmentation and pattern analysis tasks and there are evidence to support such properties in the biological sensory processing, at least in the early to intermediate stages of the visual pathway. However, if the number of training data is large, this leads to an overcomplete and redundant dictionary of basis functions.

In order to sparsify our representation, we look at a trade-off functional. A sparse approximation scheme chooses, among all the approximating schemes with similar training error, the one with the minimum non-zero coefficients. Therefore, we look at minimizing the following functional with respect to the coefficients $\boldsymbol{a} = (a_1 \cdots a_M)^T$:

$$J[\boldsymbol{a}] = \frac{1}{2}\|f(\boldsymbol{x}) - \sum_{i=1}^{M} a_i K(\boldsymbol{x}, \boldsymbol{x}_i)\|_H^2 + \epsilon\|\boldsymbol{a}\|_{L_1}. \quad (18)$$

This functional results from the fact that we can write the approximated function in the first part of the functional as an expansion of the kernel correlation due to eq.(17). This error functional is in the spirit of Basis Pursuit De-noising (BPD) of Chen et al. [3]. The difference, as pointed out in [9], is that while BPD uses the L_2 norm to measure the reconstruction error, we use the true distance in form of the H norm. This has been shown in approximation theory to lead to better generalization properties [15] due to its emphasis on reducing errors in the original function space rather than the sampled space or parameter space.

Here, we borrow from the results of Girosi[6] which says that minimizing the functional (18) - under the assumption of noiseless data - is equivalent to solving the following dual minimization problem[3]:

$$\text{Min. } J_D[\alpha, \alpha^*] = \frac{1}{2}\sum_{i,j=1}^{M}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \epsilon\sum_{i=1}^{M}(\alpha_i + \alpha_i^*) - \sum_{i=1}^{M} y_i(\alpha_i^* - \alpha_i) \quad (19)$$

$$\text{subject to} \quad 0 \le \alpha_i, \alpha_i^* \le C, \quad i = 1, \ldots, M, \quad (20)$$

$$\text{and} \quad \sum_{i=1}^{M}(\alpha_i^* - \alpha_i) = 0 \quad (21)$$

where α_i, α_i^* are non-negative Lagrange multipliers and are related to the scalar variables a_i as $a_i = \alpha_i^* - \alpha_i$. Since, we are considering the noiseless case, C can be equated to infinity in line with the argument in [6]. The minimization of functional (19) leads to the solution obtained by the support vector machines for regression [13] and has been traditionally solved using quadratic programming routines.

5.1 Gradient descent based sequential implementation

In this work, we introduce a modification of the bias term and augment the kernel $K(x, x_i)$ with a constant term λ. This augmentation, as analysed in detail in the work on modified margin optimization for sequential SVMS by the authors[16], leads to an margin maximization which is sufficiently justified in the high dimensional learning cases. This helps in absorbing the condition (21) into the cost function, making a sequential implementation possible. The optimal approximation surface using the modified formulation is now given as

$$f(\boldsymbol{x}) = \sum_{i=1}^{M}(\alpha_i^* - \alpha_i)(K(\boldsymbol{x}_i, \boldsymbol{x}) + \lambda^2). \quad (22)$$

Due to sparseness properties of the large margin approximators, very few number of the coeffcients $(\alpha_i^* - \alpha_i)$ are non-zero and hence, the representation of the approximated function is *sparse*.

Using the gradient of the cost function (19), we propose an update rule to approximate the variables α_i and α_i^* iteratively. Let the kernel function $K(x_i, x_j)$ be constructed using the correlation operator R of the learning problem such that the RKHS corresponding to this kernel spans the space $\mathcal{R}(R)$. Fig.2 gives the update rule for iteratively approximating the variables. Here, λ is the augmenting factor, which should be chosen in the scale of the input vectors. ϵ is the user defined error insensitivity parameter which controls the balance between

[3]Strictly speaking, there is an additional constraint that, assuming the target function has zero mean, the approximating function also has zero mean.

Algorithm for Sparse Representation

1. Initialize $\alpha_i = 0, \alpha_i^* = 0$. Compute $[R]_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \lambda^2$ for $i, j = 1, \ldots, M$.

2. For each training point, i=1 to M, compute

2.1 $E_i = y_i - \sum_{j=1}^{l} (\alpha_i^* - \alpha_i) R_{ij}$.

2.2 $\delta\alpha_i^* = \min\{\max[\gamma(E_i - \epsilon), -\alpha_i^*], C - \alpha_i^*\}$.
$\delta\alpha_i = \min\{\max[\gamma(-E_i - \epsilon), -\alpha_i], C - \alpha_i\}$.

2.3 $\alpha_i^* = \alpha_i^* + \delta\alpha_i^*$.
$\alpha_i = \alpha_i + \delta\alpha_i$.

3. If the training has converged, then **stop** else **goto step 2**.

Figure 2: Sequential algorithm for sparseness

the sparseness of the solution and the closeness to the training data and γ is the learning rate. The value of the tradeoff parameter C is set to infinity for the case of noiseless data, hence, not necessitating the outer *min* comparison in the step 2.2 of the algorithm.

If we look at the gradient of the cost function (19) or the change in cost function with small changes in α and α^*, we can write the following relationship.

$$\begin{aligned} \Delta J_D &= \delta\alpha_i^*(-\epsilon + E_i - \frac{1}{2} R_{ii}\delta\alpha_i^*) + \delta\alpha_i^*\delta\alpha_i R_{ii} \\ &\quad + \delta\alpha_i(-\epsilon - E_i - \frac{1}{2} R_{ii}\delta\alpha_i), \end{aligned} \tag{23}$$

where the elements of matrix $\mathbf{R}$ and the scalars E_i are defined as shown in the algorithm. With some additional analysis which is omitted for brevity, we can show that this cost function monotonically decreases to a minimum value and converges provided the learning rate satisfies $\gamma < 1/\max_{\{i\}} R_{ii}$. Faster convergence can be obtained by using a data dependent learning rate like $\gamma_i < 1/R_{ii}$.

6 Illustrative examples

In this section, we look at a synthetic regression task and compare the approximation properties of the RKHS based analytical solution against the more parsimonious sparse representation.

We look at the task of approximating a function $f = 4 - \sin x + \sin 2x - \sin 3x + \sin 4x - \sin 5x$ shown in Fig.3 from a set of 20 uniformly sampled training data. First, we consider approximation using a function space spanned by $H = \{\sin nx, \cos nx\}_{n=0}^{5}$. This ensures that the function to be approximated

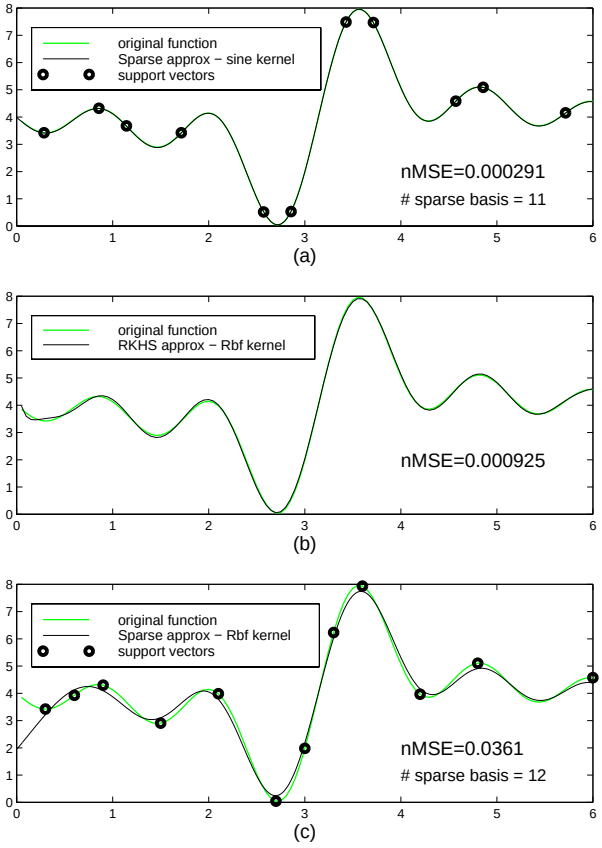

Figure 3: Approximation results using 20 uniform training data (a) Sparse approximation using sine kernels (b) RKHS analytic solution with RBF kernels (c) Sparse approximation using RBF kernels

lies within the model space being considered. The reproducing kernel of this space can be written as

$$\begin{aligned} K(x, x') &= \sum_{n=0}^{5} (\cos nx \cos nx' + \sin nx \sin nx') \\ &= \begin{cases} 6 & \text{if } x = x' \\ \frac{1}{2}[\sin \frac{11*(x-x')}{2} / \sin \frac{x-x'}{2} + 1] & \text{otherwise} \end{cases} \end{aligned} \tag{24}$$

This choice is equivalent to having a correlation operator R which spans the approximation space H. Using the analytical method of solving described in Section 4.2, we achieve perfect generalization i.e., the function is learned exactly. This is expected since we have noiseless data and the function resides in the search space. In comparison, when we sparsify the solution using the same sin kernel $K(x, x')$ (24), we get a parsimonious representation using 11 basis vectors with a normalised mean squared error(nMSE) of .00029 on a test data set as shown in Fig.3(a). The epsilon parameter in the sparse

training was set to $\epsilon = 0.01$.

Next, we consider learning with a space spanned by RBF kernels with reproducing kernels

$$K(x, x') = e^{\|x-x'\|^2/2\sigma^2}, \tag{25}$$

where σ is the variance parameter of the kernel which we set here to a value of $\sigma = 0.5$. Hence, in this case the function we are trying to approximate does not strictly lie in the search space. The result of learning with 20 points using the RKHS analytic method is shown in Fig.3(b) resulting in an nMSE=0.00092. Using the sparseness constraint in this function space leads to an approximation with 12 basis vectors and an nMSE of 0.0361 as shown in Fig.3(c). Here, an $\epsilon = 0.2$ was used. The tradeoff between accuracy and degree of sparseness can be controlled by varying the thickness of the ϵ-tube.

7 Conclusion and Discussion

In this paper, we formulate the problem of learning a mapping as an inverse problem and provide analytical solutions by using an optimization criterion which exploits the apriori knowledge on the probilistic distribution of function ensembles. Although this solution is optimal from a generalization perspective, it is expensive in terms of the resources since it employs one kernel function at every training data location to represent the learned result. However, it is shown that if we choose a particular search space spanned by the correlation operator (apriori knowledge usually accessable) and enforce a sparseness constraint on the solution on the lines of Poggio and Girosi[9], the problem reduces to the same dual problem encountered in support vector regression. Here, we introduce a novel gradient descent based sequential learning algorithm to solve this dual problem for sparsification. This algorithm is simple to implement and assured of convergence within theoretically derived learning rate bounds.

References

[1] A. Albert. *Regression and the Moore-Penrose Pseudoinverse*. Academic Press, 1972.

[2] N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

[3] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. Technical Report 479, Dept. of Statistics, Stanford University, 1995.

[4] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[5] D.J. Field. Relation between the statistics of natural images and the response properties of cortical cells. *Journal of Optical Society Am.*, 4:2379–2394, 1987.

[6] F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Computation*, 10(6):1455–1480, 1998.

[7] H. Ogawa and E. Oja. Projection filter, Wiener filter and Karhunen-Loève subspaces in digital image restoration. *Journal of Mathematical Analysis and Applications*, 114(1):37–51, 1986.

[8] P.S. Penev and J.J. Atick. Local feature analysis: A general statistical thoery for object recognition. *Neural Systems*, 7:477–500, 1996.

[9] T. Poggio and F. Girosi. A sparse representation for function approximation. *Neural Computation*, 10(6):1445–1454, 1998.

[10] L. Sirovich and M. Kirby. Low dimensional procedure for characterization of human faces. *Journal of Optical Society Am.*, 4:519–524, 1987.

[11] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[12] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons,Inc.,New York, 1997.

[13] V. Vapnik, S.E. Golowich, and A. Smola. Support vector method for function approximation, regression estimation and signal processing. In *Advances in Neural Information Processing Systems 9*, pages 281–287, 1997.

[14] S. Vijayakumar. *Computational theory of incremental and active learning for optimal generalization*. PhD thesis, Tokyo Institute of Technology, 1998.

[15] S. Vijayakumar and H. Ogawa. RKHS based functional analysis for exact incremental learning. *Neurocomputing: Special Issue on Theoretical analysis of real valued function classes*, 1999. (in press).

[16] S. Vijayakumar and S. Wu. Sequential support vector classifiers and regression. In *Proceedings, Soft Computing '99 (SOCO'99)*, 1999.