Kernel-based Methods and Function Approximation

BAUDAT G. , ANOUAR F.
MEI, Mars Electronics International
1301 Wilson Drive, West Chester, PA 19380, USA
E-mail: gaston.baudat@eu.effem.com
Tel: (00) 1 610 430 27 51
E-mail: fatiha.anouar@effem.com
Tel: (00) 1 610 430 25 22 Fax: (00) 1 610 430 27 95

Abstract

This paper provides a new insight into neural networks by using the kernel theory drawn from the work on support vector machine (SVM) and related algorithms. The kernel trick is used to extract a relevant data set into the feature space according to a geometrical consideration. Then the data are projected onto the subspace of the selected vectors where classical algorithms are applied without adaptation. This approach covers a wide range of algorithms. In particular, different types of neural network are covered by choosing the appropriate kernel. We investigate the function approximation on a real classification problem and on a regression problem.

1 Introduction

Kernel methods were first used in Support Vector Machines (SVM) [13] [7], and were applied to a wide class of problems such as classification and regression. The possibility of using different kernels allows viewing learning methods like Radial Basis Function (RBF) neural network or multi-layer neural networks as particular cases of SVM despite the fact that the optimized criteria are not the same. As a matter of fact, the SVM optimizes a geometrical criterion which is the margin and is sensitive only to the extreme values and not to the distribution of the data into the feature space. On the other hand, the RBF [5] [3] or multi-layer neural networks optimizes the Mean Square Error MSE and provides a solution dependent on the distribution of all the data.

The kernel methods transform data into a feature space F that usually has a huge dimension. But it has been shown [14] that the capacity of generalization depends on the geometrical characteristics of the training data, not on the dimension of the input space. Therefore, if these characteristics are well chosen, the expected error of generalization can be small even if the feature space has a huge dimensionality.
Moreover, Vapnik's work comparing the SVM and the neural network perceptron showed that the empirical risk, which evaluates the capacity of generalization, is bounded for the SVM by a function of the largest value of the norm of the support vectors. For the perceptron this bound is a function of the largest value of the norm of the vectors where the perceptron makes a correction [14]. Vapnik pointed out that minimizing the ratio between the radius of the sphere that contains the support vectors and the margin can give better generalization than maximizing only the margin [14]. This showed that SVM lacks an important characteristic of the data, which is the structure of the data into F.
We propose to split the problem into two steps. The first step is the data selection that extract the relevant vectors into the feature space F taking advantage of the kernel theory and preserving the geometrical structure of the data into F. The second step optimizes a given criterion into the subspace of the selected vectors.
Consider a mapping function ϕ operating from an input space X into a feature Hilbert space F:

$$\begin{aligned} \phi : X &\rightarrow F \\ x &\rightarrow \phi(x) \end{aligned} \tag{1}$$

Suppose M is the number of samples; then we define the kernel matrix *K* of dot products by:

$$K = (k_{ij})_{1 \le i \le M, 1 \le j \le M} \tag{2}$$

where $k_{ij} = \phi^t(x_i)\phi(x_j)$

The transformed data lie into a subspace of F with a dimension up to M. Usually, the dimension of this subspace is lower than M and equal to the rank of the kernel matrix *K*. We propose to work into the subspace of the data into F. Therefore, we developed a preprocessing method that

selects the relevant data, forming a basis of this subspace and capturing the structure of the entire data into F. The data selection into F uses the kernel trick that manipulates the dot product K without knowing explicitly the mapping function ϕ [1] [6] [11]. Data selection is a common step in all kernel methods. Once the data are projected onto the subspace of the selected data, one can run classical linear algorithms. We focus here on regression and classification problems.

In the following we present the data selection into F that we call Feature Vector Selection (FVS). Then, we investigate the combination of the data selection with the linear regression to get a non-linear function approximation. The latter algorithm is applied to classification and regression problems.

2 Feature Vector Selection

The dimensionality of the data subspace into F is given by the rank of *K*. In practice, the rank of the matrix *K* is often inferior to M, $rank(K) < M$. The aim of the FVS is to select a basis of the data subspace into F.

The FVS is based on a geometrical approach into F. We look for the vectors that are sufficient to express all the remaining data as a linear combination of those selected vectors in the transformed space F.

The Feature Vectors FV are among the samples. We are able to express the subspace of the data into F with samples only, because the data subspace is entirely expressed with the whole data through *K*. Our approach leads to the extraction of the sufficient submatrix of *K*, which reduces significantly the required memory for storage.

Let L be the number of selected vectors.

For notation simplification: for each x_i the mapping is noted $\phi(x_i) = \phi_i \ \ for\ 1 \le i \le M$.

We note the selected vectors by x_{s_j} and its image into F by $\phi(x_{s_j}) = \phi_{s_j} \ \ for\ 1 \le j \le L$, where $L \le M$.

For a given set of selected vectors $S = \{x_{s_1}, \ldots, x_{s_L}\}$ into F, we can estimate the mapping of any vector x_i as a linear combination of S, which is formulated as a dot product:

$$\hat{\phi}_i = \Phi_S . a_i \qquad (3)$$

where $\Phi_S = (\phi_{s_1}, \ldots, \phi_{s_L})$ is the matrix of the selected vectors into F and $a_i = (a_i^1, \ldots, a_i^L)^t$ is the coefficient vector that weighted this matrix.

The goal is to find the coefficients a_i such that the estimated mapping $\hat{\phi}_i$ is as close as possible to the real mapping ϕ_i for a given set S. Therefore, we minimize into F, the normalized Euclidean distance given by the following ratio:

$$\delta_i = \frac{\left\| \phi_i - \hat{\phi}_i \right\|^2}{\left\| \phi_i \right\|^2} \qquad (4)$$

Rewriting (4) in a matrix form and putting, according to a_i , the derivatives to zero; we can show that the minimum of (4) for a given S can be expressed with dot products only and leads to (5):

$$\min \delta_i = 1 - \frac{K_{Si}{}^t K_{SS}{}^{-1} K_{Si}}{k_{ii}} \qquad (5)$$

Where K_{SS} is a square matrix of dot products of the selected vectors: $K_{SS} = (k_{s_p s_q})_{1 \le p \le L, 1 \le q \le L}$.

And $K_{Si} = (k_{s_p i})_{1 \le p \le L}$ is the vector of dot product between x_i and the selected vector set S.

The goal now is to find the set S that minimizes (5) over all samples x_i :

$$\min_S \left(\sum_{x_i \in X} \left(1 - \frac{K_{Si}{}^t K_{SS}{}^{-1} K_{Si}}{k_{ii}}\right)\right) \qquad (6)$$

We define the fitness function for a given set S expressed by:

$$J_S = \frac{1}{M} \sum_{x_i \in X} \left(\frac{K_{Si}{}^t K_{SS}{}^{-1} K_{Si}}{k_{ii}}\right) \qquad (7)$$

(6) is equivalent to:

$$\max_S J_S \qquad (8)$$

Note that for $x_i \in S$, (5) is zero, and the maximum of (7) is one.

The selection algorithm is an iterative process, which is a sequential forward selection: at each step we look for the sample that, when combined with the previously selected vectors, gives the maximum fitness function J_S (7). The algorithm stops when K_{SS} is no longer invertible, which means that S is an actual basis for the data into F. We can also stop when the fitness reaches a given value or when a predefined number of selected vectors is reached.

2.1 FVS Algorithm

- Arguments:
  - Training sample $X = \{x_1, \ldots, x_M\}$
  - Kernel function k_{ij}
  - Stopping criterion: number of feature vectors nbFV or maxFitness or basisFound
- Return: S subset of selected feature vectors

- Initially $S=\varnothing$, at the first iteration add to S the sample that maximizes J_S,
- At the iteration n, if L samples have been selected. Append to S the sample, that combined with the previous L samples, maximizes J_S
- Stop when K_{SS} is no longer invertible, or when the error is below a given small value or when a predefined number of selected vectors is reached.

3 Kernel Function Approximation algorithm

Once the feature vectors are selected, they define a subspace S into F that captures the structure of the entire data. We transform all of the samples into this subspace. The transformation of a sample x_i is given by the dot product projection:

$$z_i = \Phi_S{}^t \phi_i \quad (9)$$

Note that z_i is obtained through a linear transformation (9). Other transformations can be considered, in particular, an orthogonal projection $z_i = (\Phi_S{}^t \Phi_S)^{-1} \Phi_S{}^t \phi_i$, which requires more computation.

Given a set of data (z_i, y_i) where z_i is the projection onto F of the sample x_i, the goal of the function approximation algorithm is to estimate the output function $\hat{y}_i$.

The estimation of this function using linear regression into the subspace of FV, minimizes the MSE error of the learning set T and results to an optimal solution (10) using the Moore-Penrose pseudo-inverse [2]:

$$\hat{y}_i = z_i{}^t A + \beta^t \quad (10)$$

where $A = (Z_T{}^t Z_T)^{-1} Z_T{}^t Y_T$

and $Z_T = (z_j{}^t)_{j \in T}$, $Y_T = (y_j)_{j \in T}$.

β is a vector that can be included in the estimation of A by adding a constant component in each vector z_i.

It has been shown that (10) leads to the posterior probability estimation and gives the best approximation of the Bayes decision function [9].

A classification task into N classes is a particular regression problem for which the output $y_i = (y_{ij})_{1 \le j \le N}$ is a binary row vector such that $y_{ij} = 1$ where j is the class number.

$$y_{ij} = \begin{cases} 1 & if \quad z_i \in j^{th} \ cluster \\ 0 & otherwise \end{cases}$$

Therefore the same algorithm is used to solve classification problems. A sample is assigned to the class c such that $\hat{y}_{ic} = \max_j \hat{y}_{ij}$.

4 Classification

The data selection corresponds in fact to a data quantization into F based on geometrical considerations. This step reduces the size of the manipulated matrix and at the same time preserves the structure of the data into F. It can be viewed as the definition of the hidden layer of a multi-layer neural network. The number of hidden neurons on which the data are projected corresponds to the number of FV selected into F. However, the FVS has the advantage of determining the optimal number of hidden neurons based on the MSE criterion for a given kernel. The trick of the kernel operator makes it possible to work and select data into the feature space F. For a neural network, the feature space remains hidden and is never explored. This main attribute gives a new insight into neural networks.

With this new approach, the classification optimization performed into F is simple and leads to the same formulas as presented above for the regression. The generalization to different kinds of neural networks is obtained by choosing the appropriate kernel function: a Gaussian kernel gives a RBF neural network, while a sigmoid function, which fulfills the Mercer conditions [13], leads to a multi-layer perceptron. Figure 1 shows the architecture of a neural network using the FVS that we called Kernel Function Approximation.

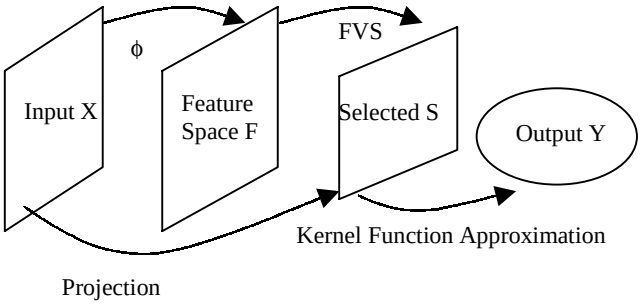


Figure 1: Architecture of Kernel Function Approximation procedure

We can even construct a multi-layer network with more than one hidden layer by repeating the FVS step as many times as the number of hidden layers.

5 Experiments

5.1 Feature space versus input space

This simple example illustrates visually the relation between the input space X and the feature space F when using a polynomial kernel of the 2nd order. The space F for this kernel has a dimension of three and can be displayed.

We can express the mapping function ϕ in this particular case. The polynomial kernel in R^2 is defined by :

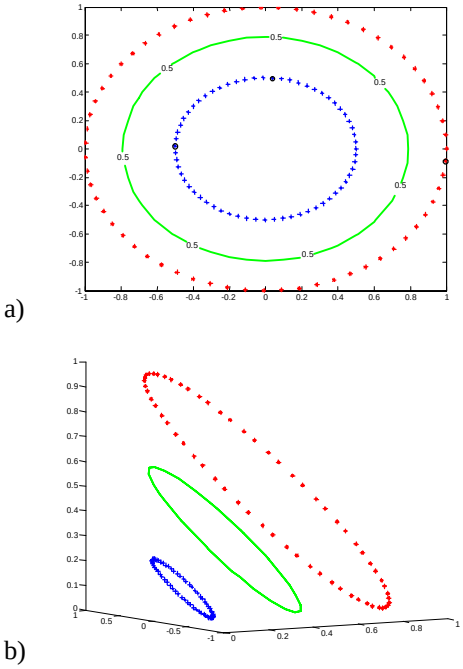
$$k_{ij} = (x_i{}^t x_j)^2 ,$$

And can be rewritten as:

$$(x_i{}^t x_j)^2 = \phi^t(x_i)\phi(x_j)$$

Where $\phi^t(x_i) = (x_{i1}{}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}{}^2)$.

Consider two clusters along two circles of radius 1 and 0.5 plotted with crosses and stars on figure 2 a). The images of these two circles into F are displayed on figure 2 b).



a)

b)

Figure 2: a) Data along two circles, the continuous circle is the separation obtained with the kernel function approximation. b) The 3D graph is the image of the clusters into F.

This example shows that the data need only three samples to be expressed into F. The separation function obtained with the kernel function approximation is given by the continuous circle into X. Its image into F lies on a plan that separates linearly the two circles into F.

5.2 Non separable data

We simulate 2 clusters with normal distributions $N((-1,0), I)$ and $N((1,0), \begin{pmatrix} 1 & 0 \\ 0 & 0.1 \end{pmatrix})$ displayed on figure 3. I is the identity matrix. The learning set contains 100 examples from each class. Since we know the theoretical distribution we computed the optimal separation given by the Bayesian decision function [10]. Therefore we can compare the theoretical results and the results obtained with the kernel function approximation into F. Figure 3 shows the results of the Bayesian classification and kernel function approximation using a Gaussian kernel function $k_{ij} = \exp(-\|x_i - x_j\|^2 / \sigma^2)$ with $\sigma = 1$.

The FV classification gives slightly better results than the SVM (OSU SVM Matlab toolbox) does. Nevertheless, our algorithm reaches the optimum with only 23 selected vectors, while the SVM needs 54 support vectors to reach its best results. Moreover, once the kernel function is fixed, the kernel function approximation requires no parameter adjustment unlike the SVM which needs an adjustment of the parameter C.

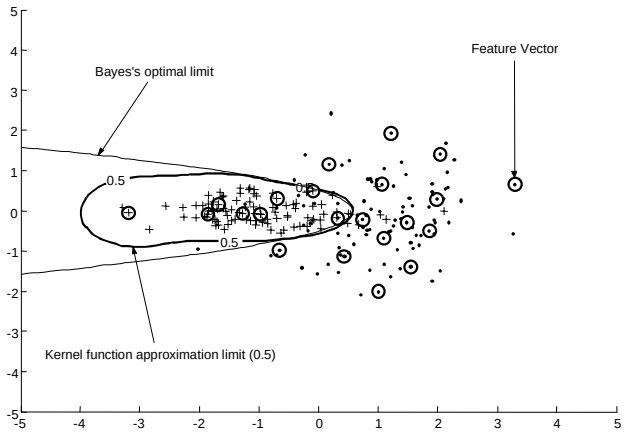


Figure 3: Optimal boundary and kernel function approximation boundary; the feature vectors are represented by circles

We evaluated the performances shown on Table 1 using a testing set of 100000 samples.

Table 1: Test performance comparison

	Bayes	SVM	Kernel function approximation
Testing performance	88.8 %	88.2 %	88.6 %
Number of points	-	54 SV	23 FV

5.3 Separable data

We consider 3 clusters having 128 samples each that are non-linearly separable. Figure 4 shows the learning set with 64 samples by cluster and the 23 selected feature vectors using the FVS algorithm**.** In order to classify the three clusters, we applied the kernel function approximation using a Gaussian kernel. The obtained separation function is shown on figure 4 b).

a)

b)

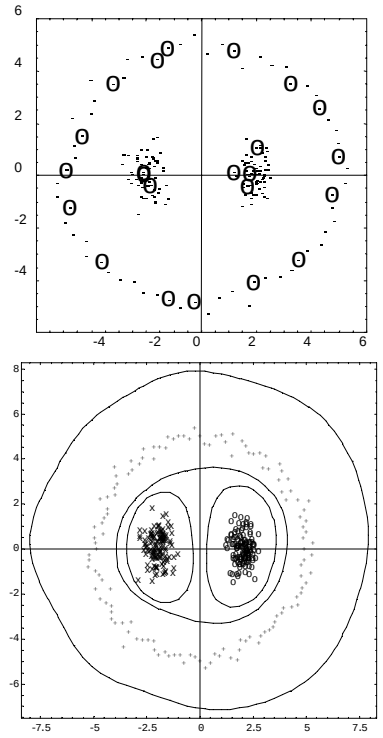


Figure 4: FVS and kernel function approximation, separation function corresponding to a cross section at 0.5

Note that the SVM gives an optimal solution of the margin for a two classes problem. An extension to multi-classes problems is based on a pairwise classification or one class versus all others. In this case, the optimality is no longer guaranteed. Unlike the SVM, the kernel function approximation algorithm provides an optimal solution whatever the number of the studied classes.

5.4 Heart disease data

We studied performances on the Cleveland heart disease data set [8] from UCI repository [16]. The data set contains 303 samples and 13 variables to be classified into 2 classes indicating a presence of disease or not. The two classes are highly overlapped, as shown on figure 5.

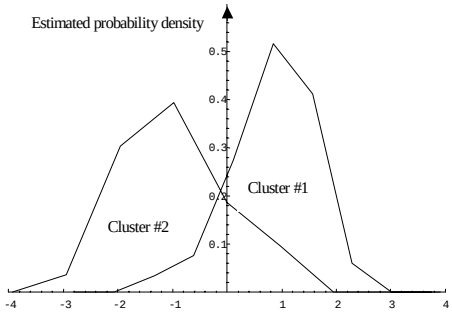


Figure 5: Heart disease probability density functions along the linear discriminant axis

We applied the kernel function approximation algorithm, the RBF neural network of the Matlab toolbox and the SVM to this data set. The performances are evaluated using the leave one out procedure. The results are presented on Table 2 using a Gaussian kernel with $\sigma = 10$ and C=1000 for SVM.

Table 2: Test performance comparison on the heart disease data set

	RBF	SVM	Kernel function approximation
Testing performance	78.4 %	79.7 %	83.2 %
Number of points	70 nodes	77 SV	14 FV

The results are significantly better than the SVM for this data set. In addition, kernel function approximation requires many fewer feature vectors than the number of support vectors required for SVM.

5.5 Function regression

We consider the approximation of the sinc function $f(x) = (\sin \pi x)/\pi x$ corrupted by a normal noise. Results on this function using SVM are given by [15]. We used the same kernel function, which is linear spline:

$$K(x_i, x_j) = 1 + x_i x_j + x_i x_j \min(x_i, x_j) - \frac{(x_i + x_j)}{2}(\min(x_i, x_j))^2 + \frac{(\min(x_i, x_j))^3}{3}$$

The added normal noise has a standard deviation of 0.5. Figure 6 shows the approximated function and the real function.

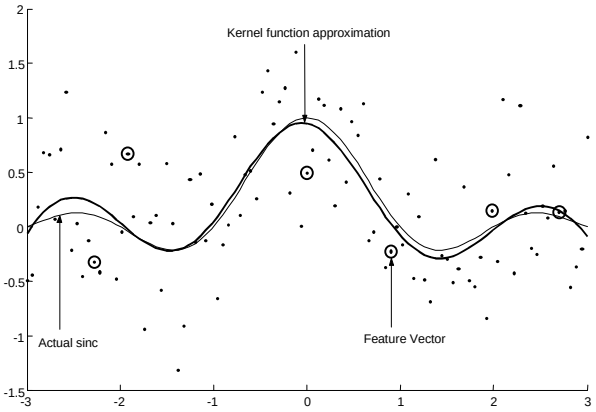


Figure 6: Sinc function corrupted by a normal noise and the approximated function obtained by kernel function approximation

The approximation given in [15] needed 27 support vectors while the kernel function approximation requires only 6 feature vectors and results to better approximation.

6 Discussion

We have presented a new regression and classification algorithm based on a feature vector selection into F using the trick of kernel methods. The FVS algorithm captures the structure of the data by constructing a basis into F. All the samples are projected onto the selected subspace of the FV; therefore, the statistical information is preserved. After the data selection step several algorithms can be applied easily. We investigated the kernel function approximation. We showed on simulated and real data that we can obtain better results than SVM with many fewer feature vectors than the number of support vectors required for SVM. The number of selected vectors depends on the choice of the kernel. For Gaussian kernel, the rank of the matrix *K* depends on the value of sigma. The smaller sigma is, the higher the rank is and the larger the selected basis S eventually reaching M vectors. The capacity of generalization is function of sigma; therefore there exists a relationship between the capacity of generalization and the number of FV forming a basis.

We also investigate results of the LDA [10] and PCA combined with the FVS which provides a new method for getting the GDA [4] and the kernel PCA [12]. The memory requirement for matrix storage in the GDA or kernel PCA drops from the size of the data set to the size of the selected data set. Therefore, the computation time at the utilization phase is reduced.

In our approach, the main part of the adaptation of the classical algorithms to kernel methods is the FVS. We presented an iterative algorithm of FVS, we are currently investigating the improvement and the optimization of this algorithm.

Acknowledgments

The Matlab code of the FVS algorithm and function approximation is available for everyone who wants to test the method. Just contact us by e-mail at fatiha.anouar@effem.com or gaston.baudat@eu.effem.com.

7 References

[1] Aizerman M. A., Braverman E. M., Rozonoér L. I., "Theoretical foundations of the potential function method in pattern recognition learning", Automation and Remote Control, 25:821-837, 1964.

[2] Albert A., “Regression and the Moore-Penrose pseudo-inverse”. Academic Press, New York, NY, 1972.

[3] Anouar F., Badran F., Thiria S., "Probabilistic Self Organizing Map and Radial Basis Function", Journal Neurocomputing 20, 83-96, 1998.

[4] Baudat G., Anouar F., “Generalized discriminant analysis using a kernel approach”, Neural Computation 12, pp 2385-2404, 2000.

[5] Bishop C.M., “Neural Network for Pattern Recognition”, Clarendon Press, Oxford, 1995.

[6] Boser B. E., Guyon I. M., Vapnik V. N., “A training algorithm for optimal margin classifiers”. In D. Haussler , editor, 5th Annual ACM Workshop on COLT, pages 144-152, Pittsburgh, PA, 1992. ACM Press.

[7] Burges C. J.C., “A Tutorial on Support Vector machine for Pattern Recognition ”, support vector web page, http://svm.first.gmd.de, 1998.

[8] Detrano R., Medical center, Long Beach and Cleveland Clinic Foundation.

[9] Duda R. O., Hart P. E., Pattern Classification and Scene Analysis. John Wiley&Sons, 1973.

[10] Fukunaga K., “Introduction to Statistical Pattern Recognition”, Academic Press, INC, 2nd ed, 1990.

[11] Poggio T., “On optimal nonlinear associative recall”, Biological Cybernetics, 19:201-209, 1975.

[12] Schölkopf B., Smola A. J., Müller K. R., “Nonlinear Component Analysis as A Kernel Eigenvalue Problem”, Neural Computation 10, 1299-1319, 1998.

[13] Vapnik V., “The Nature of Statistical Learning Theory”, Springer Verlag, 1995.

[14] Vapnik V., “Three remarks on support vector method of function estimation”, IN Schölkopf B., Burges C. J. C. and Smola A. J., (Eds), “Advances in Kernel Methods”, Cambridge, MA, MIT Press.

[15] Vapnik V., Golowich S. E., Smola A., “Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing” Neural Information Processing Systems, vol 9. MIT Press, Cambridge, MA, 1997.

[16] http://www.ics.uci.edu/~mlearn/MLSummary.html