HYBRID SOM-SVM ALGORITHM FOR REAL TIME SERIES FORECASTING

J.M. Górriz
University of Cádiz
Avda Ramón Puyol E 11202 s/n Algeciras Spain

C.G. Puntonet
University of Granada
Daniel Saucedo E 18071 s/n Granada Spain

E.W: Lang
University of Regensburg
Universitätsstraße D-93040 Regensburg Germany

Keywords: Support vector machines, structural risk minimization, kernel, on-line algorithms, matrix decompositions, resource allocating network.

Abstract: In this paper we show a new on-line parametric model for time series forecasting based on Vapnik-Chervonenkis (VC) theory. Using the strong connection between *support vector machines* (SVM) and *Regularization theory* (RT), we propose a regularization operator in order to obtain a suitable expansion of radial basis functions (RBFs) with the corresponding expressions for updating neural parameters. This operator seeks for the "flattest" function in a feature space, minimizing the risk functional. Finally we mention some modifications and extensions that can be applied to control neural resources and select relevant input space.

1 INTRODUCTION

The purpose of this work is twofold. It introduces the foundations of SVM (Vapnik, 1998) and its connection with RT (Tikhonov and Arsenin, 1997) in order to show the new on-line algorithm for time series forecasting. On the other hand, it attempts to give an overview of Independent Component Analysis, used in this paper to introduce exogenous information to our model.

SVMs are learning algorithms based on the structural risk minimization principle (Vapnik and Chervonenkis, 1974) (SRM) characterized by the use of the expansion of SV "admissible" kernels and the sparsity of the solution. They have been proposed as a technique in time series forecasting (Muller et al., 1999; Muller et al., 1997) and have faced the overfitting problem, presented in classical neural networks, thanks to their high capacity for generalization. The solution for SVM prediction is achieved solving the constrained quadratic programming problem thus SV machines are nonparametric techniques, i.e. the number of basis functions are unknown before hand. The solution of this complex problem in *real-time applications* can be extremely uncomfortable because of high computational time demand.

SVM are essentially Regularization Networks (RN) with the kernels being Green´s function of the corresponding regularization operators (Smola et al.,). Using this connection, with a clever choice of regularization operator (based on SVM philosophy), we should obtain a parametric model being very resistant to the overfitting problem. Our parametric model is a *Resource allocating Network* (Platt, 1991) characterized by the control of neural resources and by the use of matrix decompositions, i.e. Singular Value Decomposition (SVD) and QR Decomposition to input selection and neural pruning (Górriz, 2003).

We organize the essay as follows. In section 2 we give a brief overview of the basic VC theory. SV algorithm and its connection to RT Theory will be presented in section 3. The new on-line algorithm will be compare to a previous version of it and the standard SVM in section 5. Finally we state some conclusions in section 6.

2 FOUNDATIONS ON VC THEORY

A general notion of functional approximation problems[1] can be describe as follows:

[1]Before discussing this problem, we prove the existence of an exact representation for continuous function in terms of simpler functions using Kolmogorov´s Theorem.

Let $\Delta \equiv \{(x_1, y_1), \ldots, (x_\ell, y_\ell)\}$ a set of independent and identically distributed training samples with unknown probability distribution function $F(x, y)$. The problem of learning is that of choosing from the given set of functions $f(x, \alpha_\ell)$, $\alpha \in \Lambda$, where Λ is a set of parameters, the one that best approximates the output y of the system. Thus the selection of the desired function must be based on the training set Δ, i.e. applying empirical risk minimization principle:

$$R(\alpha_\ell) \equiv \int L(y, f(x, \alpha) dF(x, y) \qquad (1)$$

$$\simeq \frac{1}{\ell} \sum_{i=1}^{\ell} (y_i - f(x_i, \alpha))^2 = R_{emp}(\alpha_\ell). \qquad (2)$$

where we substitute the loss function $L(y, f(x, \alpha)$, measuring the discrepancy between the response y to a given input x and the solution $f(x, \alpha)$, for a specific loss which forms the least squares method. Under certain conditions empirical risk functional converges towards the expected risk hence the approximation in equation 1 holds, i.e. $\ell \to \infty$. However under small sample sizes non convergence may occur detecting the overfitting problem (Muller et al., 2001). The way of avoiding it is introducing a regularization term (Tikhonov and Arsenin, 1997) to limit the complexity of the loss function class arising the problem of model selection (Muller et al., 2001).

The theory for controlling the generalization ability of learning machines is devoted to constructing an new inductive principle for minimizing the risk functional and, at once, for controlling the complexity of the loss function class. This is a major task, as we said latter, whenever a small sample of training instances "ℓ" is used (Vapnik, 1998). To construct learning methods we use the bounds found by Vapnik:

$$R(\alpha_\ell^k) \leq R_{emp}(\alpha_\ell^k) + \Re\left(\frac{\ell}{h_k}\right). \qquad (3)$$

where R is the actual risk, R_{emp} is the empirical risk depending on samples, $\Re$ is the confidence interval, α_ℓ^{k}[2] is the set of selected parameters that defines the class of approximation functions and h is the VC dimension[3]. In order to minimize the right-hand side of inequality 3 we apply SRM principle as follows:
Let $\mathcal{L}_1 \subset \mathcal{L}_2 \subset \ldots \subset \mathcal{L}_k \ldots$, a nested "admissible"[4] family of loss functions classes with finite VC dimension denoted by "h_i" with $i = 1 \ldots, k$, for a given set of observations Δ the SRM principle chooses the suitable class $\mathcal{L}_k$ (and the function $L(x, \alpha_\ell^k)$ minimizing the guaranteed risk (right-hand side of inequality 3. In other words, the higher complexity in class function the lower empirical risk with the higher confidence interval (the second term in the bounds of the expected risk).

3 SUPPORT VECTOR MACHINES AND REGULARIZATION THEORY

The SV algorithm is a nonlinear generalization of the generalized portrait developed in the sixties by Vapnik and Lerner in (Vapnik and Lerner, 1963). The basic idea in SVM for regression and function estimation, is to use a mapping function Φ from the input space $\mathcal{X}$ into a high dimensional feature space $\mathcal{F}$ and then to apply a linear regression. Thus the standard linear regression transforms into:

$$f(x) = \langle \omega \cdot \Phi(x) \rangle + b. \qquad (4)$$

where $\Phi : \mathcal{X} \to \mathcal{F}$, b is a bias or threshold and $\omega \in \mathcal{F}$ is a vector defining the function class. The target is to determinate ω, i.e. the set of parameters in the neural network, minimizing the regularizated risk expressed as:

$$R_{reg}[f] = R_{emp}[f] + \lambda ||\omega||^2. \qquad (5)$$

thus we are enforcing "flatness" in feature space, that is we seek small ω. Note that equation 5 is very common in RN with a certain second term.

SVM algorithm is a way of solving the minimization of equation 5, that can be expressed as a quadratic programming problem using the formulation stated in (Vapnik, 1998):

$$minimize \quad \frac{1}{2}||\omega||^2 + C \sum_{i=1}^{\ell} (\xi_i + \xi_i^*). \qquad (6)$$

given a suitable Loss function $L(\cdot)$[5], a constant $C \geq 0$ and with slack variables $\xi_i, \xi_i^* \geq 0$. The optimization problem is solve constructing a Lagrange function by introducing dual variables, using equation 6 and the selected loss function.

Once it is uniquely solved, we can write the vector ω in terms of the data points as follows:

$$\omega = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*) \Phi(x_i). \qquad (7)$$

where α_i, α_i^* are the solutions of the mentioned quadratic problem. Once this problem, with high

[2]The subindex k is related to the structure or subset of loss functions we use in the approximation

[3]Roughly speaking, the VC dimension h measures how many training points can be separated for all possible labelling using functions of the class.

[4]In the strict sense presented in (Vapnik, 1998), that is, they are bounded functions or satisfy a certain inequality.

[5]For example Vapnik´s ϵ insensitive loss function (Vapnik, 1998).

$$L(f(x)-y) = \left\{ \begin{array}{ll} |f(x) - y| - \epsilon & for \quad |f(x) - y| \geq \epsilon \\ 0 & otherwise \end{array} \right\}$$

computational demand [6], is solved we take equation 7 into 4 and obtain the solution in terms of dot products:

$$f(x) = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)\langle \Phi(x_i) \cdot \Phi(x)\rangle + b. \quad (8)$$

At this point we use a trick to avoid computing the dot product in high dimensional feature space in equation 8, replacing it by a kernel function that satisfies Mercer´s condition. Mercer´s Theorem guarantees the existence of this kernel function:

$$f(x) = \sum_{i=1}^{\ell} h_i \cdot k(x_i, x) + b. \quad (9)$$

where $h_i \equiv (\alpha_i - \alpha_i^*)$ and $k(x_i, x) = \langle \Phi(x_i) \cdot \Phi(x)\rangle$.

Finally we note, regarding the sparsity of the SV expansion 8, that only the elements satisfying $|f(x_i) - y_i| \geq \epsilon$, where ϵ is the standard deviation of $f(x_i)$ from y_i (see selected loss function), have nonzero Lagrange multipliers α_i, α_i^*. This can be proved applying Karush-Kuhn-Tucher (KKT) conditions (Kuhn and Tucker, 1951) to the SV dual optimization problem.

RT appeared in the methods for solving *ill posed problems* (Tikhonov and Arsenin, 1997). In RN we minimize a expression similar to equation 5. However, the search criterium is enforcing smoothness (instead of flatness) for the function in input space (instead of feature space). Thus we get:

$$R_{reg}[f] = R_{emp}[f] + \frac{\lambda}{2}||\hat{P}f||^2. \quad (10)$$

where $\hat{P}$ denotes a regularization operator in the sense of (Tikhonov and Arsenin, 1997), mapping from the Hilbert Space H of functions to a dot product Space D such as $\langle f, g\rangle \quad \forall f, g \in H$ is well defined. Applying Fréchet´s differential[7] to equation 10 and the concept of Green´s function of $\hat{P}^*\hat{P}$:

$$\hat{P}^*\hat{P} \cdot G(x_i, x_j) = \delta(x_i - x_j). \quad (11)$$

(here δ denotes the Dirac´s δ, that is $\langle f, \delta(x_i)\rangle = f(x_i)$), we get (Górriz, 2003):

$$f(x) = \lambda \sum_{i=1}^{\ell} [y_i - f(x_i)]_\epsilon \cdot G(x, x_i). \quad (12)$$

The correspondence between SVM and RN (equations 9 and 12) is proved if and only if the Green´s function G is an "admissible" kernel in the terms of Mercer´s theorem,i.e. we can write G as:

$$G(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j)\rangle \quad with \quad (13)$$

$$\Phi : x_i \to (\hat{P}G)(x_i, .). \quad (14)$$

[6]This calculation must be compute several times during the process

[7]Generalizated differentiation of a function: $dR[f] = \left[\frac{d}{d\rho}R[f + \rho h]\right]$, where $h \in H$.

A similar prove of this connection can be found in (Smola et al.,). Hence given a regularization operator, we can find an admissible kernel such that SV machine using it will enforce flatness in feature space and minimize the equation 10. Moreover, given a SV kernel we can find a regularization operator such that the SVM can be seen as a RN.

4 ON-LINE ALGORITHM USING REGULARIZATION OPERATORS

In this section we show a new on-line RN based on "Resource Allocating Network" algorithms (RAN) [8] (Platt, 1991) which consist of a network using RBFs, a strategy for

- Allocating new units (RBFs), using two part novelty condition (Platt, 1991)
- Input space selection and neural pruning using matrix decompositions such as SVD and QR with pivoting (Górriz, 2003).

and a learning rule based on SRM as discuss in the previous sections. Our network has 1 layer as is stated in equation 9. In terms of RBFs the latter equation can be expressed as:

$$f(x) = \sum_{i=1}^{N(t)} h_i \cdot \exp\left(-\frac{||x(t) - x_i(t)||^2}{2\sigma_i^2(t)}\right) + b. \quad (15)$$

where $N(t)$ is the number of neurons, $x_i(t)$ is the center of neurons and $\sigma_i(t)$ the radius of neurons, at time "t". In order to minimize equation 10 we propose a regularization operator based on SVM philosophy. We enforce flatness in feature space, as described in section 3, using the regularization operator $||\hat{P}f||^2 \equiv ||\omega||^2$, thus we get:

$$R_{reg}[f] = R_{emp}[f] + \frac{\lambda}{2}\sum_{i,j=1}^{N(t)} h_i h_j k(x_i, x_j). \quad (16)$$

We assume that $R_{emp} = (y - f(x))^2$ we minimize equation 16 adjusting the centers and radius (gradient descend method $\Delta\chi = -\eta\frac{\partial R[f]}{\partial \chi}$, with simulated annealing):

$$\Delta x_i = -2\frac{\eta}{\sigma_i}(x - x_i)h_i(f(x) - y)k(x, x_i) + ||\omega||^2. \quad (17)$$

where $||\omega||^2 = \alpha \sum_{i,j=1}^{N(t)} h_i h_j k(x_i, x_j)(x_i - x_j)$ and

$$\Delta h_i = \tilde{\alpha}(t)f(x_i) - \eta(f(x) - y)k(x, x_i). \quad (18)$$

[8]The principal feature of these algorithms is sequential adaptation of neural resources.

Table 1: Pseudo-code of SVM-online (t_o denotes current iteration; k denotes prediction horizon)

```
Initialize parameters and variables
Build input Toeplitz matrix A
    using (3W-1) input values
Input space selection: determinate
    Np relevant lags L using SVD and $QR_{wp}$ [8]
Determinate input vector: $x = x(t_o - k - L(1))$
while (true)
  if ($n_{rbfs} > 0$)
    Compute f(x)
    Find nearest RBF:$||x - x_{dmin}||$
  else
    $f(x) = x(t_o - k - 1)$
  Calculate error: $e = ||f(x) - x(t_o)||$
  if ($e > epsilon$ and $||x - x_{dmin}|| > delta$) [7]
    Add RBF with parameters:
    $x_i = x, \sigma_i = kappa \cdot ||x - c_{dmin}||, h = e$
  else
  Execute pruning
(SVD and $QR_wp$ to neural activations)[8]
  Update parameters minimizing actual risk[15][16]
  if ($e > theta \cdot \epsilon$ and $n_{inps} < max_{inps}$)
    $n_{inps} = n_{inps} + 1$
    Determinate new lags: $L = [L_1, L_2, ..., L_{Np}]$
    Add $rbf_{add}$ RBFs
$t_o = t_o + 1$
end
```

where $\alpha(t), \tilde{\alpha}(t)$ are scalar-valued "adaptation gain", related to a similar gain used in the stochastic approximation processes, as in these methods, it should decrease in time. The second summand in equation 17 can be evaluated in several regions inspired by the so called "divide-and-conquer" principle and used in unsupervised learning, i.e. competitive learning in self organizing maps (Kohonen, 1990) or in SVMs experts (Cao, 2003). This is necessary because of volatile nature of time series, i.e. stock returns, switch their dynamics among different regions, leading to gradual changes in the dependency between the input and output variables. Thus the super-index in the latter equation is redefined as:

$$N_c(t) = \{s_i(t) : ||x(t) - x_i(t)|| \leq \rho\}. \qquad (19)$$

the set of neurons close to current input. The structure of the algorithm is shown below as pseudo-code, including the set of initial parameters:

Table 2: Evolution of NRMSE (Normalized Root Mean Square Error). Xth-S detones the Xth-step prediction NRMSE on the test set (Noisy Mackey-Glass with delay changing operation mode.

Method	1st-S	25th-S	50th-S	75-th	100th-S
NAPA_PRED	1.1982	0.98346	0.97866	0.91567	0.90985
Standard SVM	0.7005	0.7134	0.7106	0.7212	0.7216
SVM_online	0.7182	0.71525	0.71522	0.72094	0.7127

5 EXPERIMENTS

The application of our network is to predict complex time series. We choose the high-dimensional chaotic system generated by the Mackey-Glass delay differential equation:

$$\frac{dx(t)}{dt} = -b \cdot x(t) + a \cdot \frac{x(t-\tau)}{1 + x^{10}(t-\tau)} \quad . \qquad (20)$$

with $b = 0.1$, $a = 0.2$ and delay $t_d = 17$. This equation was originally presented as a model of blood regulation and became popular in modelling time series benchmark. We add two modifications to equation 20:

- Zero-mean gaussian noise with standard deviation equal to $1/4$ of the standard deviation of the original series.
- Dynamics changes randomly in terms of delay (between 100-300 time steps) $t_d = 17, 23, 30$.

We integrated the chaotic model using MatLab software on a Pentium III at 850MHZ obtaining 2000 patterns. For our comparison we use 100 prediction results from SVM_online (presented in this paper), standard SVM (with ϵ-insensitive loss) and NAPA_PRED (RAN algorithm using matrix decompositions being one of the best on-line algorithms to date(Górriz, 2003)). Clearly there´s a remarkable difference between previous on-line algorithm and SVM philosophy. Standard SVM and SVM_online achieve similar results for this set of data at the beginning of the process. In addition, there´s is noticeable improvement in the last iterations because of the volatile nature of the series. The change in time delay t_d, leads to gradual changes in the dependency between the input and output variables and, in general, it´s hard for a single model including SVMs to capture such a dynamic input-output relationship inherent in the data. Focussing our attention on the on-line algorithm, we observe the better performance of the new algorithm such as lower number of neurons ("sparsity"), input space dimension and forecasting results.

6 CONCLUSIONS

Based on SRM and the principle of "divide and conquer" , a new online algorithm is developed by combining SVM and SOM using a resource allocating network and matrix decompositions. Minimizing the regularizated risk functional, using an operator the enforce flatness in feature space, we build a hybrid model that achieves high prediction performance, comparing with the previous on-line algorithms for time series forecasting. This performance is similar to the one achieve by SVM but with lower computational time demand, essential feature in real-time systems. The benefits of SVM for regression choice consist in solving a -uniquely solvable- quadratic optimization problem, unlike the general RBF networks, which requires suitable non-linear optimization with danger of getting stuck in local minima. Nevertheless the RBF networks used in this paper, join various techniques obtaining high performance, even under extremely volatile conditions, since the level of noise and the change of delay operation mode applied to the chaotic dynamics was rather high.

REFERENCES

Cao, L. (2003). *Support Vector Machines Experts for Time Series Forecasting*. Neurocomputing, vol 51, 321–339 (2003).

Górriz, J. (2003). *Algoritmos Híbridos para la Modelizacióńd e Series temporales con técnicas AR-ICA*. PhD Thesis, University of Cádiz, Departamento de Electrónica (2003).

Kohonen, T. (1990). *The Self-Organizing Map*. Proceedings of the IEEE, num 9, vol 78, 1464–1480 (1990).

Kuhn, H. and Tucker, A. (1951). *Nonlinear Programming.* In 2^{nd} Symposium on Mathematical Statistics and Probabilistics,University of California Press, 481–492 (1951).

Muller, K., Mika, S., Ratsch, G., Tsuda, K., and Scholkopf, B. (2001). *An Introduction to Kernel-Based Learning Algorithms*. IEEE Transactions on Neural Networks, vol 12, num 2, 181–201 (2001).

Muller, K., Smola, A., Ratsch, G., Scholkopf, B., and Kohlmorgen, J. (1999). *Using Support Vector Machines for time series prediction*. in B. Scholkopf, C.J.C. Burges, A.J. Smola (Eds.) Advances in kernel Methods- Spport Vector Learning, MIT Press, Cambridge, MA. (1999) 243–254.

Muller, K., Smola, A., Ratsch, G., Scholkopf, B., Kohlmorgen, J., and Vapnik, V. (1997). *Predicting time series with Support Vector Machines*. in ICANN'97: Proceedings of the seventh International Conference on Artificial Neural Networks, Lausanne, Switzerland (1997) 999-1004.

Platt, J. (1991). *A resource-allocating network for function interpolation*. Neural Computation, 3, (1991) 213–225.

Smola, A., Scholkopf, B., and Muller, K. *The connection between regularization operators and support vector kernels*. Neural Networks, 11, 637–649.

Tikhonov, A. and Arsenin, V. (1997). *Solutions of Ill-Posed Problemsk*. Winston. Washington D.C., U.S.A. Berlin Heidelberg New York (1997) 415–438.

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley, N.Y. (1998).

Vapnik, V. and Chervonenkis, A. (1974). *Theory of Pattern Recognition*. [in Russian]. Nauka, Moscow (1974).

Vapnik, V. and Lerner, A. (1963). *Pattern Recognition using Generalized Portrait Method.* Automation and Remote Control, vol 24, issue 6,(1963).