

Estratégias neurais para treinamento de *Least Squares Support Vector Machines*

Bernardo Penna Resende de Carvalho e Antônio de Pádua Braga
bpenna@gmail.com, apbraga@cpdee.ufmg.br
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brasil

Abstract

We present in this work strategies for solving the system of linear equations intrinsic to the LS-SVMs, that has difficult solution by the fact that it isn't positive definite. Least Squares Support Machines Vector (LS-SVMs) were created in 1999, corresponding to a modified version of Support Machines Vector (SVMs), developed in 1992. The optimization problem for the LS-SVMs can be solved from a system of linear equations instead of quadratic programming, as it occurs in the SVMs. The main characteristic of the LS-SVMs is the low computational cost comparing to the SVMs, without quality loss in the solution, because the principles that both have been based are the same. In this work we considered the following strategies: OLAM associative memory, Hopfield neural network and Adaline neural network, the last one applied with the algorithms gradient descending, rprop and quickprop.

1. Introdução

Propomos neste trabalho estratégias inéditas para o treinamento das *Least Squares Support Vector Machines* (LS-SVMs) [10]. As estratégias que empregamos são baseadas em modelos de redes neurais artificiais: memória associativa OLAM [7], rede de Hopfield [5] e rede Adaline (gradiente descendente [13], rprop [9] e quickprop [3]). Estas estratégias são utilizadas no treinamento das LS-SVMs, ou seja, na resolução do sistema de equações lineares intrínseco a esta. As redes de Hopfield e Adaline se beneficiam pelo fato de serem iterativas, de modo que mesmo soluções obtidas após a interrupção de suas execuções tenham qualidade satisfatória. Já a OLAM tem como principal característica um custo computacional baixo.

Support Vector Machines (SVMs) [12] são máquinas de aprendizagem com apenas uma camada escondida, submetidas a um treinamento supervisionado. Elas baseiam-se na Teoria de Aprendizagem Estatística, através do Princípio de Minimização do Risco Estrutural [11]. Seu treinamento é realizado através da resolução de um QP (*quadratic pro-*

gramming), que possui um custo computacional elevado.

LS-SVMs [10] correspondem a modificações das SVMs, através do uso de uma função objetivo de mínimos quadrados e da utilização de restrições de igualdades. A principal característica das LS-SVMs é o seu menor custo computacional em relação às SVMs, sem perda na qualidade das soluções, uma vez que os princípios em que ambas se baseiam são os mesmos.

O treinamento das LS-SVMs é realizado resolvendo-se um sistema de equações lineares ao invés de programação quadrática. Este sistema não pode ser resolvido diretamente pelos métodos clássicos de otimização devido ao fato de não ser definido positivo. Em [1] e [6], o sistema de equações das LS-SVMs foi modificado de forma a gerar um outro sistema, agora definido positivo, tornando-se possível a resolução do primeiro de forma indireta, e não diretamente como propomos neste trabalho.

2 *Support Vector Machines*

A aprendizagem das SVMs [11] consiste em se minimizar:

Erro empírico: erro dos dados de treinamento, responsável pelos ajustes do hiperplano aos dados apresentados durante o processo de aprendizagem, comum na maioria das máquinas de aprendizagem.

Erro estrutural: erro de generalização, responsável pela alta capacidade de generalização das SVMs, pois possibilita o desenvolvimento de um limite inferior de generalização.

A obtenção de um equilíbrio entre ambos os erros acima descritos significa a possibilidade de superar tendências de excesso de ajustes (*overfitting*) obtendo, ao mesmo tempo, a capacidade de uma boa generalização. Ambas as características estão presentes nas soluções obtidas pelas SVMs, bem como pelas obtidas pelas LS-SVMs.

Dado o conjunto de treinamento $(x_i, y_i)_{i=1}^N$ com dados de entrada $x_i \in \mathbb{R}^n$ e saída binária correspondente $y_i \in \{-1, +1\}$, a superfície de decisão criada pelas SVMs é representada por:

$$\omega^T \varphi(x) + b = 0 \quad (1)$$

onde ω é o vetor de pesos, b é o termo de polarização e $\varphi(\cdot)$ é o mapeamento realizado em um espaço de dimensão elevada.

A partir do hiperplano acima, a classificação do padrão i é dada por:

$$\begin{cases} \omega^T \varphi(x_i) + b \geq +1 & \text{se } y_i = +1 \\ \omega^T \varphi(x_i) + b \leq -1 & \text{se } y_i = -1 \end{cases} \quad (2)$$

Pode-se expressar a equação acima por:

$$y_i [\omega^T \varphi(x_i) + b] \geq 1 - \xi_i \quad (3)$$

para o padrão de entrada i , através da utilização de variáveis de folga não negativas ξ_i , de modo a classificar corretamente padrões que estão ligeiramente fora da região de sua classe.

O processo de treinamento consiste na obtenção de valores para os pesos e do termo de polarização b de forma a minimizar uma função de custo $J(\omega, \xi)$.

O problema primal das SVMs é o seguinte:

$$\min_{\omega, b, \xi} J(\omega, \xi) = \frac{1}{2} \omega^T \omega + C \sum_{i=1}^N \xi_i \quad (4)$$

sujeito a:

$$\begin{cases} y_i [\omega^T \varphi(x_i) + b] \geq 1 - \xi_i \\ \xi_i \geq 0, \quad i = 1, \dots, N \end{cases}$$

Através da aplicação do Lagrangeano [8], é obtida uma formulação dual que obedece ao Teorema de Mercer [4]:

$$\max_{\alpha, b} L(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad (5)$$

sujeito a:

$$\begin{cases} \sum_{i=1}^N \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C, \quad i = 1, \dots, N \end{cases}$$

onde α_i é o multiplicador de Lagrange correspondente ao padrão i e C um parâmetro de treinamento fornecido pelo usuário, que limita o valor dos multiplicadores de Lagrange.

A solução do problema de otimização dual é a mesma do primal [8]. O problema dual, mais simples de se resolver, é utilizado de forma a se obter os multiplicadores de Lagrange e o termo de polarização, que definem os vetores de suporte.

Pode-se verificar que a saída obtida pelas SVMs é da forma:

$$f(x) = \text{sign} \left[\sum_{i=1}^N \alpha_i y_i K(x, x_i) + b \right] \quad (6)$$

onde a função $K(x, x_i) = \varphi(x)^T \varphi(x_i)$ é chamada de função de Kernel, responsável pela realização de um produto no próprio espaço de entrada, ao invés de se fazer no espaço de características.

A classificação de um determinado vetor como suporte é realizada a partir do valor de cada multiplicador de Lagrange associado ao mesmo no final do processo de otimização. Estes vetores são definidos a partir das condições de KKT (Karush-Kuhn-Tucker) [4] para o problema de classificação:

- $\alpha_i = 0$, $y_i f(x_i) > 1$, Vetor comum, situado no lado correto, não influi na construção do hiperplano ótimo.
- $0 < \alpha_i < C$, $y_i f(x_i) = 1$, Vetor de suporte, situado sobre a margem do lado correto, conhecido como vetor de suporte *non-bound*.
- $\alpha_i = C$, $y_i f(x_i) < 1$, Vetor de suporte, situado no lado errado, conhecido como vetor de suporte *bound*.

Abaixo se encontram algumas funções que podem ser utilizadas como função de kernel:

KERNEL:	Expressão:	Parâmetros:
RBF	$e^{-\ x_i - x_j\ ^2 / 2\sigma^2}$	σ^2
Polinomial	$(x_i^T x_j + a)^b$	a, b
Sigmóide	$\tanh(\beta_0 x_i^T x_j + \beta_1)$	β_0, β_1

3 Least Squares Support Vector Machines

A formulação das LS-SVMs [10] consiste na modificação do problema primal de forma a possibilitar a resolução através de um sistema de equações lineares.

O problema primal agora é formulado da seguinte maneira:

$$\min_{\omega, b, e} J(\omega, e) = \frac{1}{2} \omega^T \omega + C \frac{1}{2} \sum_{i=1}^N e_i^2 \quad (7)$$

sujeito a:

$$y_i [\omega^T \varphi(x_i) + b] = 1 - e_i, \quad i = 1, \dots, N$$

A partir das mudanças efetuadas, a classificação realizada agora é da forma:

$$\begin{cases} \omega^T \varphi(x_i) + b = +1 & \text{se } y_i = +1 \\ \omega^T \varphi(x_i) + b = -1 & \text{se } y_i = -1 \end{cases} \quad (8)$$

que também pode ser expressa por:

$$y_i[\omega^T \varphi(x_i) + b] + e_i = 1 \quad (9)$$

A partir do Lagrangeano [8], tem-se:

$$L(\omega, b, e; \alpha) = J(\omega, b, e) - \sum_{i=1}^N \alpha_i \{y_i[\omega^T \varphi(x_i) + b] - 1 + e_i\} \quad (10)$$

Pelas condições de otimalidade, o sistema KKT (Karush-Kuhn-Tucker) [4] obtido é o seguinte:

$$\begin{cases} \frac{\partial L}{\partial \omega} = 0 \rightarrow \omega = \sum_{i=1}^N \alpha_i y_i \varphi(x_i) \\ \frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^N \alpha_i y_i = 0 \\ \frac{\partial L}{\partial e_i} = 0 \rightarrow e_i = \frac{\alpha_i}{C}, \quad i = 1, \dots, N \\ \frac{\partial L}{\partial \alpha_i} = 0 \rightarrow y_k[\omega^T \varphi(x_i) + b] - 1 + e_i = 0, \quad i = 1, \dots, N \end{cases} \quad (11)$$

Pela condição $\alpha_i = C e_i$, nota-se a perda da escassez do vetor de multiplicadores de Lagrange, que na nova formulação é proporcional aos erros. O sistema de equações lineares acima pode ser escrito na forma matricial:

$$\begin{bmatrix} I & 0 & 0 & -Z^T \\ 0 & 0 & 0 & -Y^T \\ 0 & 0 & CI & -I \\ Z & Y & I & 0 \end{bmatrix} \begin{bmatrix} \omega \\ b \\ e \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vec{1} \end{bmatrix} \quad (12)$$

onde:

$$I = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad \vec{1} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$Z = \begin{bmatrix} \varphi_1(x_1)y_1 & \dots & \varphi_N(x_1)y_1 \\ \vdots & \ddots & \vdots \\ \varphi_1(x_N)y_N & \dots & \varphi_N(x_N)y_N \end{bmatrix}$$

$$\omega = \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_N \end{bmatrix} \quad e = \begin{bmatrix} e_1 \\ \vdots \\ e_N \end{bmatrix} \quad \alpha = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix}$$

Substituindo a primeira e a terceira expressões de (11) na última, tem-se as seguintes equações:

$$\begin{cases} \sum_{i=1}^N \alpha_i y_i = 0 \\ \alpha \sum_{i=1}^N \sum_{j=1}^N (y_i y_j \varphi(x_i)^T \varphi(x_j) + \frac{1}{C}) + yb = 1 \end{cases} \quad (13)$$

que podem ser escritas na forma matricial:

$$\begin{bmatrix} 0 & -Y^T \\ Y & H \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{1} \end{bmatrix} \quad (14)$$

onde:

$$H = ZZ^T + \frac{I}{C}$$

em que a função de Kernel $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$ está inserida na matriz H acima, ou seja, a nova formulação mantém a funcionalidade de se fazer um mapeamento implícito dos dados no espaço de características.

Nota-se que a função implementada pelas LS-SVMs é:

$$f(x) = \text{sign}[\sum_{i=1}^N \alpha_i y_i K(x, x_i) + b] \quad (15)$$

que corresponde à mesma obtida através das SVMs, como indica a equação (6).

4 Estratégias propostas para o treinamento das LS-SVMs

O treinamento das LS-SVMs, expresso em (14), pode ser resolvido por diferentes tipos de redes neurais artificiais, que serão discutidos nesta seção.

4.1 OLAM

A solução do sistema $AX = B$ pode ser dada por:

$$X = A^+ B \quad (16)$$

onde a matriz pseudo-inversa é:

$$A^+ = (A^T A)^{-1} A^T \quad (17)$$

em que:

$$A^+ A = I$$

A primeira proposta deste trabalho é a utilização de uma OLAM (Optimal linear associative memory) [7], que mapeia a matriz pseudo-inversa para a resolução do sistema de equações lineares de (14).

4.2 Rede de Hopfield

Propomos como outra alternativa para a solução de (14), a rede neural de Hopfield [5], que possui um algoritmo de treinamento não-supervisionado, baseado em estados da rede. Ela é um modelo matricial não-linear recorrente, amplamente empregada para problemas com mal condicionamento, como este em que lidamos.

Ao término do processo de treinamento, a seguinte condição de estabilidade é satisfeita:

$$Y = \text{sign}(WY - \theta) \quad (18)$$

onde W é o peso sináptico matricial da rede e θ é o vetor de polarização aplicado externamente à rede.

Cada estado da rede tem um valor de energia associado, de forma que esta energia decresce monotonicamente à medida que uma certa trajetória é descrita no espaço de estados, até que se chegue a a solução final (18) de mínima energia, que também corresponde à solução de (14).

4.3 Rede Adaline

A saída da rede Adaline (Adaptive Linear Neuron) [13] possui a seguinte forma:

$$Y = WX + b \quad (19)$$

Forçando o termo de polarização de Adaline a ser sempre nulo ($b = 0$), tem-se:

$$Y = WX \quad (20)$$

Fazendo com que a entrada da rede seja $X = A^T$, obtém-se a solução de (14) como sendo a saída da rede Adaline:

$$Y = B^T \quad (21)$$

Baseando-se na equação (20), propomos a utilização da rede Adaline para a obtenção da solução de (14).

O principal método utilizado para a obtenção dos pesos da rede Adaline é o gradiente descendente [2]. Entretanto, para problemas mais complexos, técnicas desenvolvidas para diminuir o tempo de processamento também foram empregadas em nosso trabalho:

Rprop (resilient propagation): [9] desenvolvido de forma a não mais o valor do gradiente ser considerado para a atualização dos pesos, mas sim o seu sinal. Deste modo, o método tende a chegar ao mínimo da função custo mais rapidamente.

Quickprop: [3] desenvolvido de modo a se efetuar uma minimização na função de erro muito rápida, baseando-se em uma função de custo quadrática. Possui uma difícil convergência para problemas mais complexos, como pode ser observado no experimento deste trabalho.

5. Resultados

Nesta seção são apresentados os resultados obtidos nos experimentos, utilizando-se os modelos de redes neurais propostos para o treinamento das LS-SVMs e comparando-os com resultados obtidos pela SVM.

O problema utilizado é um problema de classificação de padrões bidimensional, com sobreposição de dados, gerados artificialmente de forma a ser possível uma análise gráfica das soluções obtidas.

Na figura 1 se encontra o resultado obtido por uma SVM, treinada com um método QP (*quadratic programming*), após o ajuste dos parâmetros e a escolha do kernel RBF (*radial basis function*). Estes parâmetros serão também utilizados nos demais experimentos, para que se possa fazer uma comparação consistente das soluções obtidas com o uso de todas as estratégias propostas neste trabalho.

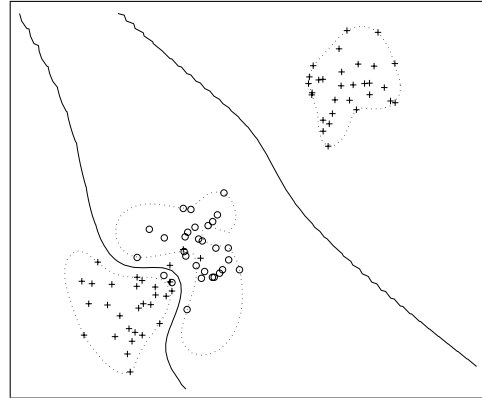


Figura 1: SVM (QP)

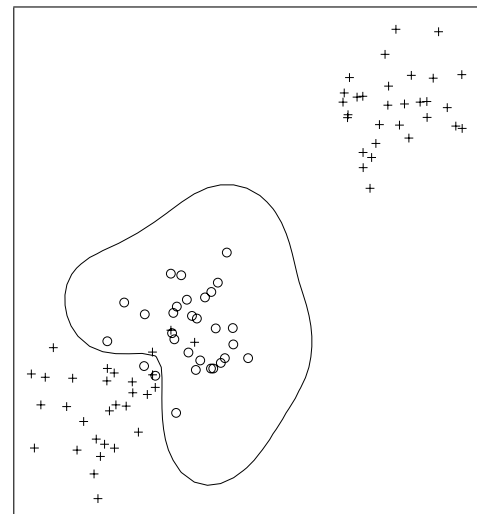


Figura 2: LS-SVM (OLAM)

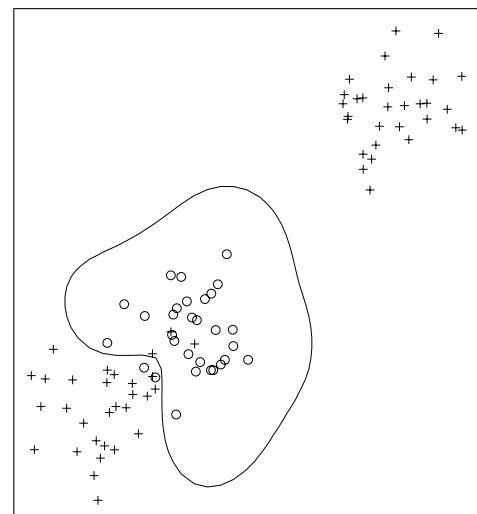


Figura 3: LS-SVM (Hopfield)

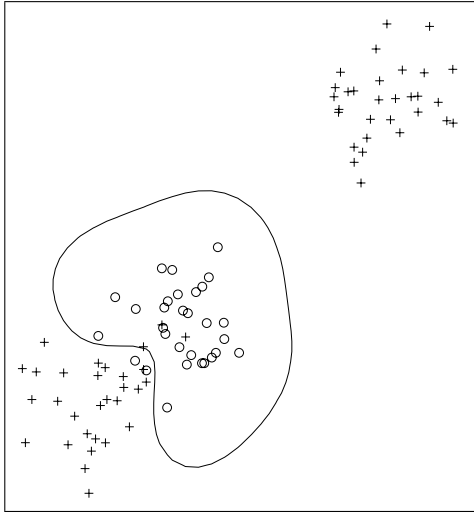


Figura 4: LS-SVM (Adaline - Gradiente)

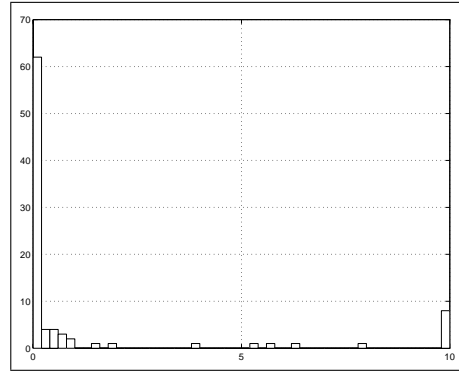


Figura 7: Histograma: SVM

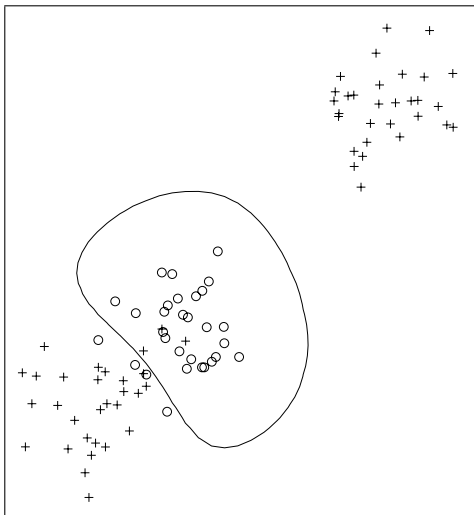


Figura 5: LS-SVM (Adaline - Rprop)

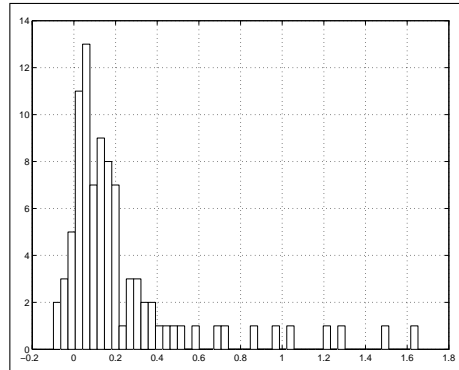


Figura 8: Histograma: LS-SVM (OLAM)

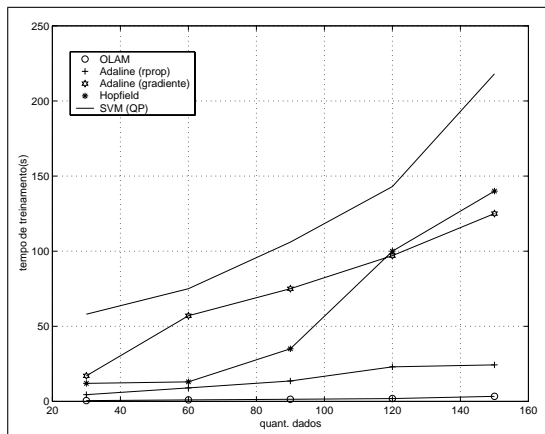


Figura 6: Comparação LS-SVMs

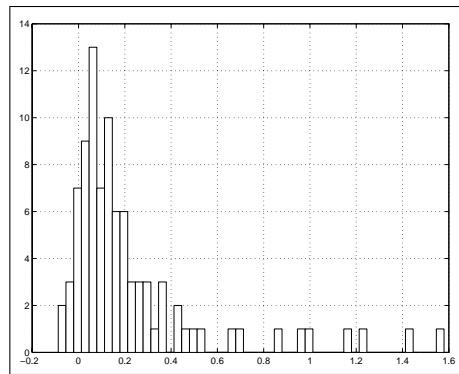


Figura 9: Histograma: LS-SVM (Adaline)

Podemos notar pelas figuras 2 a 5 que nas fronteiras entre as classes (fronteiras de decisão) todas as soluções apresentam comportamentos semelhantes, possuindo notadamente uma capacidade de generalização elevada. A partir da figura 1 entretanto, pode-se perceber grandes diferenças entre a solução encontrada pela SVM e pelas LS-SVMs, independente do modelo de rede neural empregado. Apesar deste fato, as soluções apresentadas nas figuras 1 a 5 apresentam qualidades comparáveis, em termos de generalização.

O treinamento da rede Adaline através do gradiente descendente (figura 4) convergiu igualmente em relação às soluções em que foram empregadas a rede de Hopfield (figura 3) e a OLAM (figura 2). Já quando empregado o método Rprop, a solução da rede Adaline é visivelmente inferior (figura 5). O método Quickprop não obteve convergência para o problema utilizado.

A figura 6 apresenta uma medida de comparação das estratégias propostas neste trabalho: o tempo de treinamento. Pela figura, pode-se observar que todas as estratégias utilizadas para LS-SVMs no trabalho possuem um menor custo computacional, quando comparadas com a SVM. A OLAM quase não eleva seu tempo com o aumento da quantidade de dados de treinamento, enquanto a rede Adaline cresce praticamente de forma linear. Pode-se notar que Rprop, apesar de possuir uma solução ligeiramente inferior, gasta menos tempo que o método gradiente descendente aplicado à rede Adaline. A rede de Hopfield obteve resultados rápidos em conjuntos de dados menos complexos, porém com seu aumento ela se mostrou ineficiente.

As figuras 7, 8 e 9 têm como objetivo demonstrar uma consequência resultante do uso das LS-SVMs. Uma característica marcante das SVMs é o fato da maioria dos multiplicadores de Lagrange associados aos vetores de treinamento serem nulos (figura 7). A denominação de vetores de suporte se aplica a um pequeno número de vetores de treinamento, cujos multiplicadores associados não são nulos. O mesmo não ocorre com as LS-SVMs, como pode ser observado pelas figuras 8 e 9, uma vez que os valores destes multiplicadores se distribuem sem a predominância de valores nulos. Esta é uma característica inerente às LS-SVMs, independente da estratégia usada em seu treinamento.

6 Conclusões

Com a intenção de evitar a necessidade de se resolver problemas QPs, foram criadas as LS-SVMs, cuja solução pode ser obtida com um baixo custo computacional, através da resolução de um problema de otimização por meio de um sistema de equações lineares. As soluções encontradas pelas duas formulações possuem qualidade equivalente, uma vez que ambas são baseadas no Princípio de Minimização do Risco Estrutural.

Neste trabalho, propusemos e comparamos estratégias

neurais para o treinamento das LS-SVMs, pela resolução de um sistema de equações lineares que não pode ser resolvido diretamente por métodos clássicos de otimização. As estratégias propostas foram empregadas com sucesso, obtendo soluções com alta capacidade de generalização.

As redes de Hopfield e Adaline se beneficiam pelo fato de serem iterativas. A memória associativa OLAM, apesar de não ser iterativa, possui baixo custo computacional, mesmo com o aumento da complexidade dos dados de treinamento. Já a rede de Hopfield obteve bons resultados apenas em conjuntos de dados menos complexos.

Referências

- [1] J. A. K. Suykens B. Hamers and B. De Moor. A comparison of iterative methods for least squares support vector machine classifiers. 2001.
- [2] R. J. Williams D. E. Rumelhart, G. E. Hinton. Learning internal representations by error propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1986.
- [3] Scott E. Fahlman. Faster-learning variations on backpropagation: An empirical study. *Connectionist Models Summer School*, 1988.
- [4] R. Fletcher. *Practical Methods of Optimization*. 1987.
- [5] J. Hopfield. Neural networks and physical systems with emergent collective computational properties. In *National Academy of Sciences of the USA*, 1982.
- [6] S.S. Keerthi and S.K. Shevade. Smo algorithm for least-squares svm formulations. 2003.
- [7] T. Kohonen. Correlation matrix memories. *IEEE Trans. Computers*, pages 353–359, 1972.
- [8] D. G. Luenberger. *Introduction to Linear and Nonlinear Programming*. 1973.
- [9] Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Proc. of the IEEE Intl. Conf. on Neural Networks*, 1993.
- [10] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.
- [11] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. 1995.
- [12] Vladimir Vapnik. *Statistical Learning Theory*. 1998.
- [13] B. Widrow and M. E. Hoff. Adaptive switching circuits. *IRE WESCON Convention Record*, 1960.