In conclusion, the investigated $LP_{\tau}NM$ GO method has been applied to NN learning and the results from multiple (100) independent test runs have shown consistent and stable performance (although slower than BP). For all of the reported problems, the proposed method has produced NN with better generalization abilities (compared to BP and DE), demonstrating very competitive results that qualify it as an efficient and reliable technique for training NNs with moderate degrees of dimensionality.

References

[1] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representation by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Cambridge, MA: MIT Press, 1986, pp. 318–362.

[2] M. Bianchini and M. Gori, "Optimal learning in artificial neural networks: A review of theoretical results," *Neurocomput.*, vol. 13, pp. 313–346, 1996.

[3] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Clarendon, 1995.

[4] R. Battiti and G. Tecchiolli, "Training neural nets with the reactive tabu search," *IEEE Trans. Neural Netw.*, vol. 6, no. 5, pp. 1185–1200, Sep. 1995.

[5] J.-T. Tsai, J.-H. Chou, and T.-K. Liu, "Tuning the structure and parameters of a neural network by using hybrid Taguchi-genetic algorithm," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 69–80, Jan. 2006.

[6] M. Rocha, P. Cortez, and J. Neves, "Evolutionary neural network learning," in *Lecture Notes in Artificial Intelligence 2902*. Berlin, Germany: Springer-Verlag, 2003, pp. 740–741.

[7] R. Chelouah and R. Siarry, "Tabu search applied to global optimization," *Eur. J. Oper. Res.*, vol. 123, pp. 256–270, 2000.

[8] R. T. Zheng, N. Q. Ngo, P. Shum, and S. C. Tjin, "A staged continuous tabu search algorithm for global optimization and its applications to the design of fiber Bragg gratings," *Comput. Optim. Appl.*, vol. 30, pp. 319–335, 2005.

[9] I. Sobol', *Uniformly Distributed Points in a Multidimensional Cub* (in Russian), ser. Mathematics and Cybernetics. Moscow, Russia: Znanie, 1985.

[10] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*. Philadelphia, PA: SIAM, 1992.

[11] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer J.*, vol. 7, pp. 308–313, 1965.

[12] H. Niederreiter, "Low-discrepancy and low-dispersion sequences," *J. Number Theory*, vol. 30, pp. 51–70, 1988.

[13] I. Sobol', "On the systematic search in a hypercube," *SIAM J. Numer. Anal.*, vol. 16, pp. 790–793, 1985.

[14] P. Bratley, B. Fox, and H. Niederreiter, "Implementation and tests of low-discrepancy sequences," *ACM Trans. Model. Comput. Simul.*, vol. 2, pp. 195–213, 1992.

[15] A. Georgieva and I. Jordanov, "A Hybrid method for stochastic global optimization using low-discrepancy sequences of points," *J. Global Optim.*, 2007, submitted for publication.

[16] R. Chelouah and R. Siarry, "Genetic and Nelder-Mead algorithms hybridized for a more accurate global optimization of continuous multidimensional functions," *Eur. J. Oper. Res.*, vol. 148, pp. 335–348, 2003.

[17] A. Bortoletti, C. Fiore, S. Fanelli, and P. Zellini, "A new class of quasi-Newtonian methods for optimal learning in MLP-networks," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 263–273, Mar. 2003.

[18] J. R. Quinlan, "Combining instance-based and model-based learning," in *Proc. Int. Mach. Learn. Conf. (ML)*, P. E. Utgoff, Ed., San Mateo, CA, 1993, pp. 236–243.

[19] X. Wang, Z. Tang, H. Tamura, M. Ishii, and W. Sun, "An improved backpropagation algorithm to avoid the local minima problem," *Neurocomput.*, vol. 56, pp. 455–460, 2004.

[20] K. Price and R. Storn, "Differential evolution algorithm," [Online]. Available: http://www.icsi.berkeley.edu/~storn/code.html

[21] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Real-time learning capability of neural networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 863–878, Jul. 2006.

A Geometrical Method to Improve Performance of the Support Vector Machine

Peter Williams, Sheng Li, Jianfeng Feng, and Si Wu

***Abstract*—The performance of a support vector machine (SVM) largely depends on the kernel function used. This letter investigates a geometrical method to optimize the kernel function. The method is a modification of the one proposed by S. Amari and S. Wu. Its concern is the use of the prior knowledge obtained in a primary step training to conformally rescale the kernel function, so that the separation between the two classes of data is enlarged. The result is that the new algorithm works efficiently and overcomes the susceptibility of the original method.**

***Index Terms*—Classification, conformal transformation, kernel function, Riemannian goemetry, support vector machine (SVM).**

I. Introduction

The support vector machine (SVM) is a general method for pattern classification and regression proposed by Vapnik [3]. The essential idea is to use a kernel function to map the original input data into a high-dimensional space so that the two classes of data become, as far as possible, linearly separable [4], [5]. Thus, the kernel is the key that determines the performance of the SVM. From the viewpoint of a regularization theory, the kernel implies a smoothness assumption on the structure of the discriminant function [6]–[8]. In case we have some prior knowledge about the data, we may use it to construct a good kernel (see, e.g., [9]–[11]). Otherwise, the kernel has to be optimized in a data-dependent way (see, e.g., [1], [2], [5], [12], [13], and references therein).

Amari and Wu [1], [2] have proposed a two-stage training process to optimize a kernel function. Their idea is based on the understanding of that the kernel mapping induces a Riemannian metric in the original input space [9], [1] and that a good kernel should enlarge the separation between the two classes. In their method, the first step of the training involves using a primary kernel to find out where the separating boundary is roughly located. In the second step, the primary kernel is conformally scaled, which magnifies the Riemannian metric around the boundary and, hence, the separation between the two classes. In the original algorithm proposed in [1], the kernel is enlarged at the positions of the support vectors (SVs), which takes into account the fact that the SVs are in the vicinity of the boundary. This approach, however, is susceptible to the distribution of the SVs, since the magnification tends to be biased towards the high-density region of the SVs, and the distribution of the SVs is determined by the distribution of data points. Although a modified version was suggested in [2] to meet this difficulty, the algorithm still suffers a certain level of susceptibility. Also, the modified algorithm is hard to apply in high-dimensional cases.

In this letter, we present a new way of scaling the kernel function. The new approach will enlarge the kernel by acting directly on the distance measure to the boundary, instead of the positions of the SVs as used before. Experimental studies based on both artificial and real-world data show that the new algorithm works robustly, has few free

Manuscript received February 9, 2006; revised July 18, 2006; accepted November 12, 2006.

P. Williams, S. Li, and S. Wu are with the Department of Informatics, the University of Sussex, Brighton BN1 9QH, U.K. (e-mail: siwu@sussex.ac.uk).

J. Feng is with the Department of Computer Science, the University of Warwick, Coventry CV4 7AL, U.K.

Digital Object Identifier 10.1109/TNN.2007.891625

1045-9227/$25.00 © 2007 IEEE

parameters, and is easily implemented for any dimension of the input space.

II. Scaling the Kernel Function

The SVM solution to a binary classification problem is given by a discriminant function of the form [4], [5]

$$f(\mathbf{x}) = \sum_{s \in SV} \alpha_s y_s K(\mathbf{x}_s, \mathbf{x}) + b. \tag{1}$$

A new out-of-sample case is classified according to the sign of $f(\mathbf{x})$. The SVs are, by definition, those $\mathbf{x}_i$ for which $\alpha_i > 0$. For separable problems, each SV $\mathbf{x}_s$ satisfies

$$f(\mathbf{x}_s) = y_s = \pm 1.$$

In general, when the problem is not separable or is judged too costly to separate, a solution can always be found by bounding the multipliers α_i by the condition $\alpha_i \leq C$, for some (usually large) positive constant C. There are then two classes of the SVs which satisfy the following distinguishing conditions:

1) $y_s f(\mathbf{x}_s) = 1, \qquad 0 < \alpha_s < C$;
2) $y_s f(\mathbf{x}_s) < 1, \qquad \alpha_s = C$.

The SVs in the first class lie on the appropriate separating margin. Those in the second class lie on the wrong side (though they may be correctly classified in the sense that sign$f(\mathbf{x}_s) = y_s$). We will call the SVs in the first class the *regular* SVs.

A. Kernel Geometry

It has been observed that the kernel $K(\mathbf{x}, \mathbf{x}')$ induces a Riemannian metric in the input space S [10], [1]. The metric tensor induced by K at $\mathbf{x} \in S$ is

$$g_{ij}(\mathbf{x}) = \frac{\partial}{\partial x_i} \frac{\partial}{\partial x'_j} K(\mathbf{x}, \mathbf{x}') \bigg|_{\mathbf{x}'=\mathbf{x}} \tag{2}$$

where x_i represents the ith component of the vector $\mathbf{x}$.

This arises by considering K to correspond to the inner product

$$K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}') \tag{3}$$

in some higher dimensional feature space H, where ϕ is a mapping of S into H. The inner product metric in H then induces the Riemannian metric (2) in S via the mapping ϕ.

The volume element in S with respect to this metric is given by

$$dV = \sqrt{g(\mathbf{x})} dx_1 \ldots dx_n \tag{4}$$

where $g(\mathbf{x})$ is the determinant of the matrix whose (i, j)th element is $g_{ij}(\mathbf{x})$. The factor $\sqrt{g(\mathbf{x})}$, which we call the *magnification* factor, expresses how a local volume is expanded or contracted under the mapping ϕ. Amari and Wu [1] suggest that it may be beneficial to increase the separation between the sample points in S which are close to the separating boundary, by using a kernel $\tilde{K}$, whose corresponding mapping $\tilde{\phi}$ provides increased separation in H between such samples.

The problem is that the location of the boundary is initially unknown. Amari and Wu, therefore, suggest that the problem should first be solved in a standard way using some initial kernel K. It should then be solved a second time using a conformal transformation $\tilde{K}$ of the original kernel given by

$$\tilde{K}(\mathbf{x}, \mathbf{x}') = D(\mathbf{x}) K(\mathbf{x}, \mathbf{x}') D(\mathbf{x}') \tag{5}$$

for a suitably chosen positive function $D(\mathbf{x})$. It is easy to check that $\tilde{K}$ satisfies the Mercer positivity condition. It follows from (2) and (5) that the metric $\tilde{g}_{ij}(\mathbf{x})$ induced by $\tilde{K}$ is related to the original $g_{ij}(\mathbf{x})$ by

$$\begin{aligned}\tilde{g}_{ij}(\mathbf{x}) = {} & D(\mathbf{x})^2 g_{ij}(\mathbf{x}) + D_i(\mathbf{x}) K(\mathbf{x}, \mathbf{x}) D_j(\mathbf{x}) \\ & + D(\mathbf{x}) \{K_i(\mathbf{x}, \mathbf{x}) D_j(\mathbf{x}) + K_j(\mathbf{x}, \mathbf{x}) D_i(\mathbf{x})\}\end{aligned} \tag{6}$$

where $D_i(\mathbf{x}) = \partial D(\mathbf{x}) / \partial x_i$ and $K_i(\mathbf{x}, \mathbf{x}) = \partial K(\mathbf{x}, \mathbf{x}') / \partial x_i \,|_{\mathbf{x}'=\mathbf{x}}$. If $g_{ij}(\mathbf{x})$ is to be enlarged in the region of the initial class boundary, $D(\mathbf{x})$ needs to be the largest in that vicinity and its gradient needs to be small far away. Note that if D is chosen in this way, the resulting kernel $\tilde{K}$ becomes data-dependent.

Amari and Wu consider the function

$$D(\mathbf{x}) = \sum_{i \in SV} e^{-\kappa \|\mathbf{x} - \mathbf{x}_i\|^2} \tag{7}$$

where κ is a positive constant. The idea is that the SVs should normally be found close to the boundary, so that a magnification in the vicinity of the SVs should implement a magnification around the boundary. A possible difficulty of (7) is that $D(\mathbf{x})$ can be rather sensitive to the distribution of the SVs, considering that the magnification will tend to be larger at high-density region of SVs and lower otherwise. A modified version was proposed in [2] which considers a different κ_i for different SVs. κ_i is chosen in a way to accommodate the local density of the SVs, so that the sensitivity with respect to the distribution of the SVs is diminished. By this, the modified algorithm achieves some improvement; however, the cost it brings associated with fixing κ_i is huge. Also, its performance in high-dimensional cases is uncertain.

Here, rather than attempt further refinement of the method embodied in (7), we will describe a more direct way of achieving the desired magnification.

B. New Approach

The idea here is to choose D so that it decays directly with distance, suitably measured, from the boundary determined by the first-pass solution using K. Specifically, we consider

$$D(\mathbf{x}) = e^{-\kappa f(\mathbf{x})^2} \tag{8}$$

where f is given by (1) and κ is a positive constant. This takes its maximum value on the separating surface where $f(\mathbf{x}) = 0$, and decays to $e^{-\kappa}$ at the margins of the separating region where $f(\mathbf{x}) = \pm 1$. This is where the regular SVs lie. In the case where K is the simple inner product in S, the level sets of f, and hence of D, are just hyperplanes parallel to the separating hyperplane. In that case, $|f(\mathbf{x})|$ measures perpendicular distance to the separating hyperplane, taking as the unit the common distance of a regular SVs from the hyperplane. In the general case, the level sets are curved as the nonintersecting hypersurfaces.

III. Geometry and Magnification

To proceed further, we need to consider specific forms for the kernel K.

A. RBF Kernels

Consider the Gaussian radial basis function (RBF) kernel

$$K(\mathbf{x}, \mathbf{x}') = e^{-\|\mathbf{x} - \mathbf{x}'\|^2 / 2\sigma^2}. \tag{9}$$

This is of the general type where $K(\mathbf{x}, \mathbf{x}')$ depends on $\mathbf{x}$ and $\mathbf{x}'$ only through the norm of their separation so that

$$K(\mathbf{x}, \mathbf{x}') = k(\|\mathbf{x} - \mathbf{x}'\|^2). \tag{10}$$

Referring back to (2), it is straightforward to show that the induced metric is Euclidean with

$$g_{ij}(\mathbf{x}) = -2k'(0)\delta_{ij}. \tag{11}$$

In particular, for the Gaussian kernel (9) where $k(\xi) = e^{-\xi/2\sigma^2}$, we have

$$g_{ij}(\mathbf{x}) = \frac{1}{\sigma^2}\delta_{ij} \tag{12}$$

so that $g(\mathbf{x}) = \det\{g_{ij}(\mathbf{x})\} = 1/\sigma^{2n}$ and, hence, the volume magnification is the constant

$$\sqrt{g(\mathbf{x})} = \frac{1}{\sigma^n}. \tag{13}$$

B. Inner Product Kernels

For another class of the kernels, $K(\mathbf{x}, \mathbf{x}')$ depends on $\mathbf{x}$ and $\mathbf{x}'$ only through their inner product so that

$$K(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} \cdot \mathbf{x}'). \tag{14}$$

A well-known example is the inhomogeneous polynomial kernel

$$K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x} \cdot \mathbf{x}')^d \tag{15}$$

for some positive integer d. For the kernels of this type, it follows from (2) that the induced metric is:

$$g_{ij}(\mathbf{x}) = k'(\|\mathbf{x}\|^2)\delta_{ij} + k''(\|\mathbf{x}\|^2)x_i x_j. \tag{16}$$

To evaluate the magnification factor, we need the following.

Lemma 1: Suppose that $\mathbf{a} = (a_1, \ldots, a_n)$ is a vector and that the components A_{ij} of a matrix $\mathbf{A}$ are of the form $A_{ij} = \alpha\delta_{ij} + \beta a_i a_j$. Then, $\det \mathbf{A} = \alpha^{n-1}(\alpha + \beta\|\mathbf{a}\|^2)$.

It follows that, for kernels of the type (14), the magnification factor is

$$\sqrt{g(\mathbf{x})} = \sqrt{k'(\|\mathbf{x}\|^2)^n \left(1 + \frac{k''(\|\mathbf{x}\|^2)}{k'(\|\mathbf{x}\|^2)}\|\mathbf{x}\|^2\right)} \tag{17}$$

so that for the inhomogeneous polynomial kernel (15), where $k(\xi) = (1 + \xi)^d$

$$\sqrt{g(\mathbf{x})} = \sqrt{d^n(1 + \|\mathbf{x}\|^2)^{n(d-1)-1}(1 + d\|\mathbf{x}\|^2)}. \tag{18}$$

For $d > 1$, the magnification factor (18) is a radial function, taking its minimum value at the origin and increasing, for $\|\mathbf{x}\| \gg 1$, as $\|\mathbf{x}\|^{n(d-1)}$. This suggests that it might be most suitable, for binary classification, when one of the classes forms a bounded cluster centered on the origin.

C. Conformal Kernel Transformations

To demonstrate the approach, we consider the case where the initial kernel K in (5) is the Gaussian RBF kernel (9). For illustration, consider the binary classification problem shown in Fig. 1(a), where 100 points have been selected at random in the square as a training set, and classified according to whether they fall above or below the curved boundary, which has been chosen as e^{-4x^2} up to a linear transform. Our approach requires a first-pass solution using conventional methods. Using a Gaussian radial basis kernel with width 0.5 and soft-margin parameter $C = 10$, we obtain the solution shown in Fig. 1(b). This plots contours of the discriminant function f, which is of the form (1). For sufficiently large samples, the zero contour in Fig. 1(b) should coincide with the curve in Fig. 1(a).

To proceed with the second-pass, we need to use the modified kernel given by (5) where K is given by (9) and D is given by (8). It is interesting to first calculate the general metric tensor $\tilde{g}_{ij}(\mathbf{x})$ when K is the Gaussian RBF kernel (9) and $\tilde{K}$ is derived from K by (5). Substituting in (6), and observing that in this case $K(\mathbf{x}, \mathbf{x}) = 1$ while $K_i(\mathbf{x}, \mathbf{x}) = K_j(\mathbf{x}, \mathbf{x}) = 0$, we obtain

$$\tilde{g}_{ij}(\mathbf{x}) = \frac{D(\mathbf{x})^2}{\sigma^2}\delta_{ij} + D_i(\mathbf{x})D_j(\mathbf{x}). \tag{19}$$

The $\tilde{g}_{ij}(\mathbf{x})$ in (19) is of the form considered in Lemma 1. Observing that $D_i(\mathbf{x})$ are the components of $\nabla D(\mathbf{x}) = D(\mathbf{x})\nabla \log D(\mathbf{x})$, it follows that the ratio of the new-to-the-old magnification factors is given by

$$\sqrt{\frac{\tilde{g}(\mathbf{x})}{g(\mathbf{x})}} = D(\mathbf{x})^n \sqrt{1 + \sigma^2\|\nabla \log D(\mathbf{x})\|^2}. \tag{20}$$

This is true for any positive scalar function $D(\mathbf{x})$. Let us now use the function given by (8) for which

$$\log D(\mathbf{x}) = -\kappa f(\mathbf{x})^2 \tag{21}$$

where f is the first-pass solution given by (1) and shown, for example, in Fig. 1(b). This gives

$$\sqrt{\frac{\tilde{g}(\mathbf{x})}{g(\mathbf{x})}} = \exp\{-n\kappa f(\mathbf{x})^2\} \times \sqrt{1 + 4\kappa^2\sigma^2 f(\mathbf{x})^2\|\nabla f(\mathbf{x})\|^2}. \tag{22}$$

The means that the following is true:

1) the magnification is constant on the separating surface $f(\mathbf{x}) = 0$;
2) along contours of constant $f(\mathbf{x}) \neq 0$, the magnification is greatest where the contours are closest.

The latter is because of the occurrence of $\|\nabla f(\mathbf{x})\|^2$ in (22). The gradient points uphill orthogonally to the local contour, hence in the direction of steepest ascent; the larger its magnitude, the steeper is the ascent, and hence the closer are the local contours. This character is illustrated in Fig. 1(c), which shows the magnification factor for the modified kernel based on the solution of Fig. 1(b). Notice that the magnification is low at distances remote from the boundary.

Solving the original problem again, but now using the modified kernel $\tilde{K}$, we obtain the solution shown in Fig. 1(d). Comparing this with the first-pass solution of Fig. 1(b), notice the steeper gradient in the vicinity of the boundary and the relatively flat areas remote from the boundary.

In this instance, the classification provided by the modified solution shows little improvement on the original classification. This is an accident of the choice of the training set shown in Fig. 1(a). We have repeated the experiment 10 000 times, with a different choice of 100 training sites and 1000 test sites on each occasion, and have found

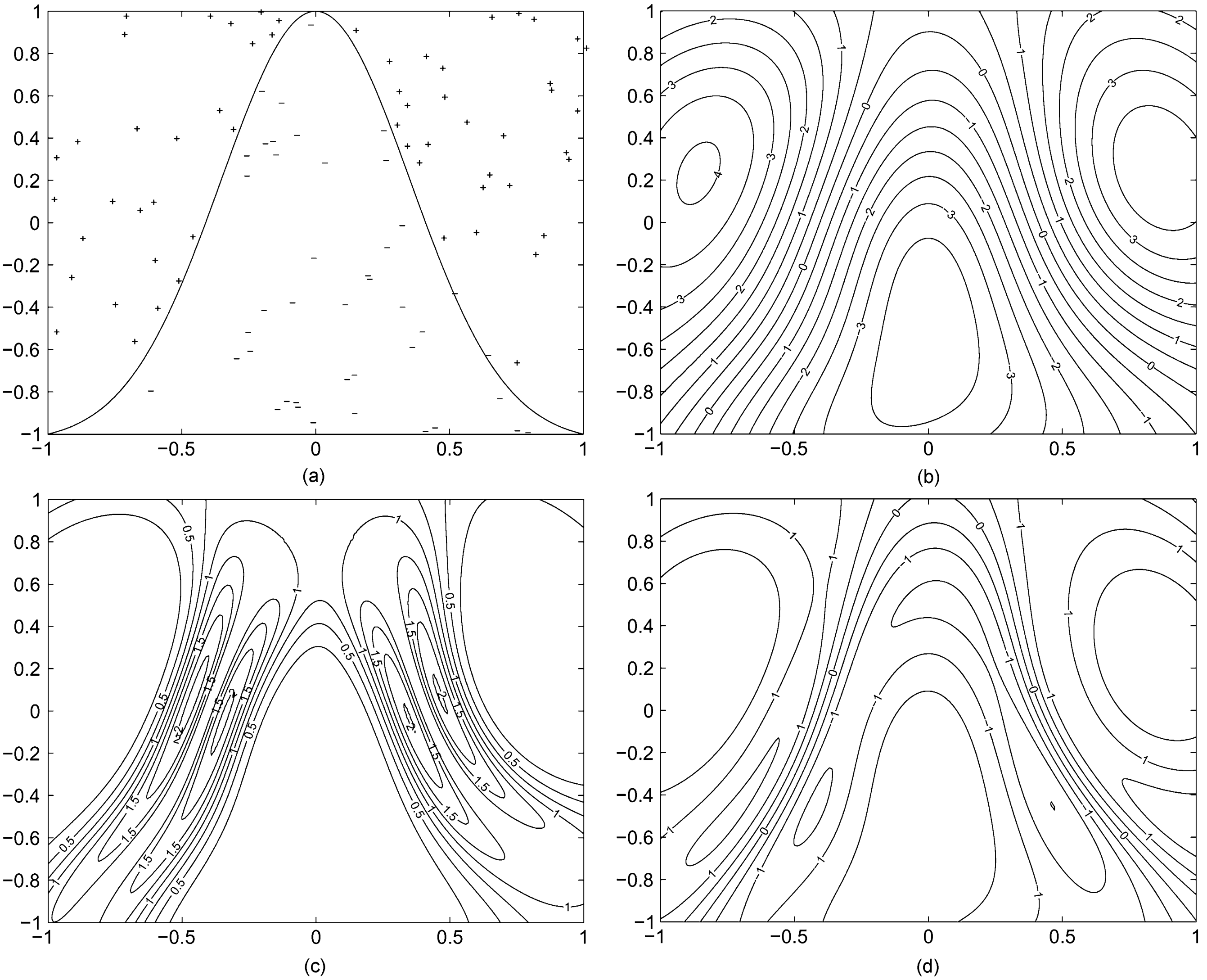

Fig. 1. (a) Training set of 100 random points classified according to whether they lie above $(+)$ or below $(-)$ the Gaussian boundary shown. (b) First-pass SVM solution to the problem in (a) using a Gaussian kernel. The contours show the level sets of the discriminant function f defined by (1). (c) Contours of the magnification factor (22) for the modified kernel using $D(\mathbf{x}) = \exp\{-\kappa f(\mathbf{x})^2\}$ with f defined by the solution of (b). (d) Second-pass solution using the modified kernel.

an average of 14.5% improvement in the classification performance.[1] A histogram of the percentage improvement, over the 10 000 experiments, together with a normal curve with the same mean and standard deviation, is shown in Fig. 2(a). The previous comparison is based on a specific choice of C and σ values. To compare fairly, we also search exhaustively the parameters that achieve the optimal performance in the first-step training, and obtain that the minimum error is 5.8% when $C = 20$ and $\sigma = 0.6$. In this case, our method decreases the error to be 4.6%.

IV. Parameter Setting

A. Choice of κ

The only free parameter in the new approach is κ. It is clear that κ is dimensionless, in the sense of being scale invariant. Suppose all input dimensions in the input space S are multiplied by a positive scalar a. To obtain the same results for the first-pass solution, a new $\sigma_a = a\sigma$ must be used in the Gaussian kernel (9). This leads to the first-pass solution f_a where $f_a(a\mathbf{x}) = f(\mathbf{x})$ with f being the initial solution using σ. It then follows from (5) and (8) that provided κ is left unchanged and the rescaled second-pass solution automatically satisfies the corresponding covariance relation $\tilde{f}_a(a\mathbf{x}) = \tilde{f}(\mathbf{x})$ where $\tilde{f}$ is the original second-pass solution using σ.

It may appear that there is a relationship between κ and σ in (22) for the magnification ratio. Using a corresponding notation, however, it is straightforward to show that the required covariance $\tilde{g}_a(a\mathbf{x})/g_a(a\mathbf{x}) = \tilde{g}(\mathbf{x})/g(\mathbf{x})$ also holds provided that κ is left unchanged. The reason is that $\sigma\|\nabla f(\mathbf{x})\|$ is invariant under rescaling since a multiplies σ and divides $\nabla f(\mathbf{x})$.

For the purposes of magnification, it is understandable that κ should be reasonably large, so that $D(\mathbf{x})$ decays quickly when $|f(\mathbf{x})| \gg 1$ and magnification is focussed on the boundary. On the other hand, κ cannot be too large, otherwise $D(\mathbf{x})$ will only differ significantly from zero in the region where $f(\mathbf{x}) \approx 0$ with the consequence that the second-pass solution will be essentially the same as the first one. This implies that the relative magnification should cover a significant area around the primary boundary, for instance, the region where $-1 < f(\mathbf{x}) < 1$;

[1] If there are 50 errors in 1000 for the original solution and 40 errors for the modified solution, we call this a 20% improvement. If there are 60 errors for the modified solution, we call it a -20% improvement.

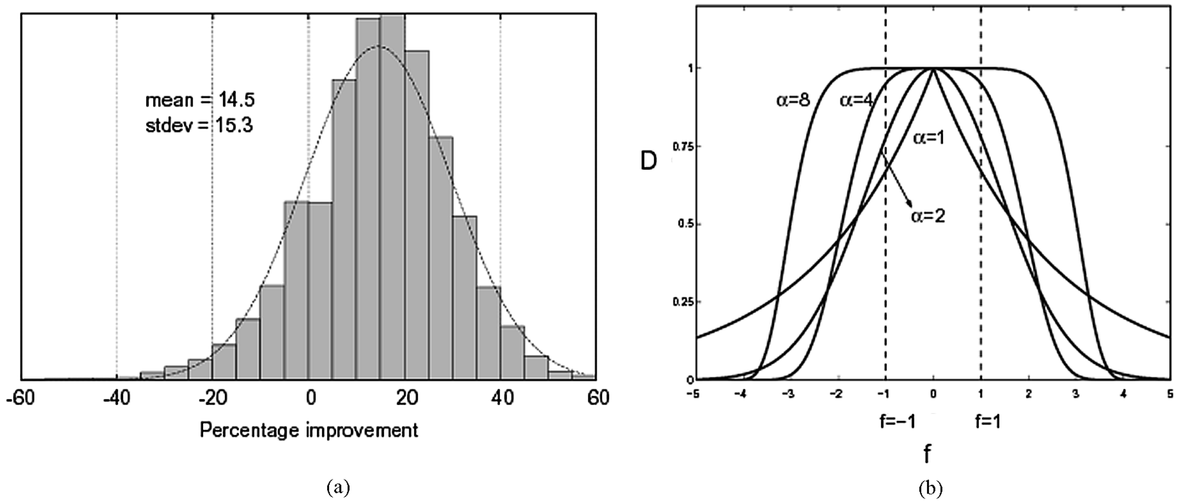

Fig. 2. (a) Histogram of the percentage improvement in the classification, over 10 000 experiments, together with a normal curve with the same mean and standard deviation. (b) Scaling factor D versus the distance measure f:$D = e^{-\kappa|f|^{\alpha}}$ for $\alpha = 1, 2, 4$, and 8.

in other words, we should not fully trust the first-pass solution. In the trials reported previously, a suitable κ is approximately the reciprocal $|f|_{\max}$, the maximum of the absolute value of $f(\mathbf{x})$ in the first-pass solution, for a typical value $|f|_{\max} = 4$, and hence, $D(\mathbf{x})$ is 1, 0.78, or 0.02 for $f(\mathbf{x}) = 0, 1$, or $|f|_{\max}$, respectively.

Furthermore, κ is independent of the dimension of the input space, since its effect is to act on the 1-D distance $|f(\mathbf{x})|$. This point has been confirmed by a simulation experiment with the 3-D data, in which we choose a Gaussian boundary similar to Fig. 1(a) except that it is 2-D. Again, κ is set to be the reciprocal of $|f|_{\max}$. By using 400 training and 2000 testing data points, we observe an average improvement of (the data is being collecting, but preliminary test has shown the improvement) over 10 000 trials.

B. Choice of $D(\mathbf{x})$

Apart from (8), we can also choose other forms for $D(\mathbf{x})$. In principle, provided $D(\mathbf{x})$ decays with respect to the distance from the boundary, this suffices to achieve our goal. For instance, $D(\mathbf{x})$ may be chosen to have a more general form $D(\mathbf{x}) = e^{-\kappa|f(\mathbf{x})|^{\alpha}}$ with $\alpha > 0$. We tested this idea by setting $\alpha = 1, 4$, and 8 and applying these to the data set illustrated in Fig. 1(a). It turns out that the average improvements are 1.3%, 10.2%, and 6.1% for $\alpha = 1, 4$, and 8, respectively. They are all smaller than the case when $\alpha = 2$, as presented previously. The optimal values of κ associated with these three cases are 0.4, 0.05, and 0.0001, respectively. This tendency for optimal κ to decrease with increments in α agrees with the picture underlying the method. The contribution of κ should balance the decay of D when $|f(\mathbf{x})| > 1$ and maintain D to be of order unity when $|f(\mathbf{x})| < 1$ [see Fig. 2(b)].

V. Experimental Study

We also apply the new method for some benchmark real-world problems and obtain encouraging results. Tables I–III show the simulation results for the mushroom database, tic-tac-toe endgame database, and congressional voting records database in the University of California at Irvine (UCI) Machine Learning Repository (the misclassification rates are illustrated). In each trial, 100 training examples are randomly chosen from the database, and the testing error is estimated by tenfold cross validation. The final results are obtained by averaging over 100 trials. We see that the original method (7) does not improve the classification performance (and in many cases, it even degrades the performance) due to its susceptibility to the distribution of input data, whereas the new method works efficiently, confirming our theoretic analysis. It is worth to point out that our new method works well for a broad range of parameters, indicating that our method can correct the general "bad" kernel in the first step training. We should point out that in some databases our method does not achieve notable improvement compared to the first-step training (it, however, never degrades the result if the parameters are properly chosen). We attribute this to that our method is geometry-based, whereas, for some real-world problems, the choice and the measure of input components are rather *ad hoc* and do not have any geometric meaning. How to apply our method to these cases is currently under investigation.

TABLE I
Mushroom Database

Parameters	Before Modification	Old Method	New Method
$C = 10, \sigma = 0.6$	12.89%	12.89%	8.37%
$C = 10, \sigma = 1.0$	5.24%	8.84%	3.89%
$C = 50, \sigma = 0.6$	12.16%	11.76%	8.08%
$C = 100, \sigma = 0.6$	12.54%	14.34%	8.13%

TABLE II
Tic-Tac-Toe Endgame Database

Parameters	Before Modification	Old Method	New Method
$C = 10, \sigma = 0.6$	31.52%	31.52%	14.15%
$C = 10, \sigma = 1.0$	13.09%	20.49%	10.09%
$C = 50, \sigma = 0.6$	31.02%	31.22%	13.29%
$C = 100, \sigma = 0.6$	32.09%	32.69%	13.33%

TABLE III
Congressional Voting Records Database

Parameters	Before Modification	Old Method	New Method
$C = 10, \sigma = 0.6$	24.11%	25.11%	20.08%
$C = 10, \sigma = 1.0$	17.56%	16.76%	11.32%
$C = 50, \sigma = 0.6$	24.01%	25.21%	20.19%
$C = 100, \sigma = 0.6$	24.27%	25.87%	20.76%

VI. Conclusions and Discussions

This letter investigates a data-dependent method for optimizing the kernel function of SVMs. The proposed algorithm is a modification of the one in [1] and [2]. Compared with the original one, the new algorithm achieves a better performance in terms of being robust with respect to the data distribution. This is confirmed by the experimental study.

The algorithm is rather simple, which has only one free parameter κ,[2] consumes very low computational cost,[3] and is applicable for any dimension of input space. It is, therefore, valuable to use this method as a general methodology to top up a normal SVM training to further improve the classification performance.

References

[1] S. Amari and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Netw.*, vol. 12, pp. 783–789, 1999.

[2] S. Wu and S. Amari, "Conformal transformation of kernel functions: A data-dependent way to improve support vector machine classifiers," *Neural Process. Lett.*, vol. 15, pp. 59–67, 2001.

[3] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[4] N. Cristanini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[5] B. Schölkopf and A. Smola, *Learning with Kernels*. Cambridge, MA: MIT Press, 2002.

[6] A. Smola, B. Scholkopf, and K. Muller, "The connection between regularization operators and support vector kernels," *Neural Netw.*, vol. 11, pp. 637–649, 1998.

[7] F. Girosi, "An equivalence between sparse approximation and support vector machines," *Neural Comput.*, vol. 20, pp. 1455–1480, 1998.

[8] T. Poggio and S. Smale, "The mathematics of learning: Dealing with data," *Notices Amer. Math. Soc.*, vol. 50, pp. 537–544, 2003.

[9] B. Schölkopf, P. Simard, A. Smola, and V. Vapnik, "Prior knowledge in support vector kernels," in *Advances in Neural Information Processing Systems 10*. Cambridge, MA: MIT Press, 1998.

[10] C. Burges, , B. Schölkopf, C. Burges, and A. Smola, Eds., "Geometry and invariance in kernel based method," in *Advances in Kernel Methods*. Cambridge, MA: MIT Press, 1999, pp. 89–116.

[11] N. Cristianini, H. Lodhi, and J. Shawe-Taylor, "Latent semantic kernels," *J. Intell. Inf. Syst.*, vol. 18, pp. 127–152, 2002.

[12] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, , S. Thrun, L. Saul, and B. Schölkopf, Eds., "Learning with local and global consistency," in *Advances in Neural Information Processing Systems 16*. Cambridge, MA: MIT Press, 2004.

[13] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 912–919.

[2] The choice of κ is roughly guided.

[3] All information needed to construct the scaling function $D(\mathbf{x})$, i.e., $f(\mathbf{x})$, is available after a normal SVM training. Also, as observed in simulation, the second-step training is often extremely fast. This is understandable, since the modified kernel has been adjusted to "concentrate" around the boundary (the grand matrix only significantly differs from zero in that region).

Global Asymptotic Stability of Delayed Cellular Neural Networks

Huaguang Zhang and Zhanshan Wang

***Abstract*—A new criterion for the global asymptotic stability of the equilibrium point of cellular neural networks with multiple time delays is presented. The obtained result possesses the structure of a linear matrix inequality and can be solved efficiently using the recently developed interior-point algorithm. A numerical example is used to show the effectiveness of the obtained result.**

***Index Terms*—Cellular neural networks, global asymptotic stability, multiple time delays.**

I. Introduction

Cellular neural networks have been successfully applied to signal processing, especially in image processing, solving nonlinear algebraic and transcendental equations and some classes of optimization problems. Some of these applications require that the equilibrium point of the designed cellular neural networks be unique and globally asymptotically stable. A number of criteria to achieve such a design have been proposed; see, for instance, [1]–[13], and the references cited therein.

At present, stability results for the cellular neural network model studied in [7], [8], and [11]–[13] are mainly based on such approaches as M-matrix, algebraic inequalities, and so on. The characteristic of those results is to take absolute value operation on the interconnection matrix, which leads to the ignorance of neuron's inhibitory and excitatory effects on neural networks. Recently, linear matrix inequality (LMI) technique has been used to deal with the stability problem for neural network [3], [4], [6], [8]–[10]. The feature of LMI-based result is that it can consider the neuron's inhibitory and excitatory effects on neural networks. However, few stability results have been obtained for the cellular neural network model studied in [7] and [11]–[13] on the basis of LMI, which is important, as did for neural network model studied in [3], [4], [6], and [8]–[10]. In this letter, the global asymptotic stability of cellular neural networks with multiple time delays is further discussed, inspired by [7] and [8]. A new sufficient condition is given to ascertain the global stability of the cellular neural networks with multiple time delays, which is easy to verify.

II. Problem Description and Main Result

Consider a cellular neural network model with multiple time delays

$$\frac{dx_i(t)}{dt} = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f\left(x_j(t)\right) + \sum_{j=1}^{n} b_{ij} f\left(x_j(t-\tau_{ij})\right) + u_i \qquad (1)$$

where x_i is the state variable of the ith neuron, a_{ij} and b_{ij} are connection weight and delayed connection weight coefficients, respectively,

Manuscript received June 27, 2006; revised October 11, 2006 and December 15, 2006; accepted December 27, 2006. This work was supported by the National Natural Science Foundation of China under Grants 60325311, 60534010, and 60572070, by the Funds for Creative Research Groups of China under Grant 60521003, and by the Program for Cheung Kong Scholars and Innovative Research Team in Universities under Grant IRT0421, the Education Ministry, China.

The authors are with the School of Information Science and Engineering and the Key Laboratory of Integrated Automation of Process Industry, Northeastern University, Shenyang, Liaoning, 110004, P. R. China (e-mail: hg_zhang@21cn.com).

Digital Object Identifier 10.1109/TNN.2007.891628

1045-9227/$25.00 © 2007 IEEE