STRUCTURAL RISK MINIMIZATION USING NEAREST NEIGHBOR RULE

A. Ben Hamza, Hamid Krim, and Bilge Karacali

Department of Electrical and Computer Engineering
North Carolina State University, Raleigh, NC 27695, USA
{abhamza, ahk, bkaraca2}@eos.ncsu.edu

ABSTRACT

We present a novel nearest neighbor rule-based implementation of the structural risk minimization principle to address a generic classification problem. We propose a fast reference set thinning algorithm on the training data set similar to a support vector machine approach. We then show that the nearest neighbor rule based on the reduced set implements the structural risk minimization principle, in a manner which does not involve selection of a convenient feature space. Simulation results on real data indicate that this method significantly reduces the computational cost of the conventional support vector machines, and achieves a nearly comparable test error performance.

1. INTRODUCTION

Structural Risk Minimization (SRM) principle [1–3], is a technique for nonparametric inference which has found application in classification and estimation problems. It is essentially based on an Empirical Risk Minimization principle, in an effort to control the performance on future (yet to be observed) data. The construction of such a performance measure is based on a training data set, and its subsequent minimization over the set of all candidate estimates, ultimately leads to a solution. The drawback of this approach is over-fitting the data set: given samples from a real valued function on the real line, one can always perfectly fit these samples with a polynomial of unlimited high order. The performance of such a fit is too specific to be useful for future data, and hence, unsatisfactory. Similarly, with a sufficient number of nodes, Neural Networks have been shown to approximate any continuous function to any desired degree of precision [4]. Much like model order selection and the related description length [5], and based on the given set of observed samples, the goal is to then select an estimate that would perform well on unseen data. This is called a generalization ability of an estimator [3].

The aim of SRM is to control the generalization ability of an estimator, via the so-called capacity [3] of the set of candidate functions. One should pick as the *best* estimate the function which most agrees with the training data set, from a candidate function set with the lowest capacity.

Support Vector Machines (SVM) have been proposed [3] to implement a SRM principle. In a classification setting, this entails the construction of a linear discriminant between classes in a nonlinear feature space, based on a *small* subset of the training data, called the *support vector*s. When mapped back to the original space, this discriminant corresponds to a nonlinear decision surface. One of the main limitations of this approach is its computational cost: finding a linear discriminant in a feature space ensues a quadrature optimization problem over a number of parameters equal to the number of data in the training set. A number of techniques and studies making use of support vectors separately from the rest of the training set have been proposed [3, 6, 7]. The basic problem, however, remains: the support vectors are only known upon solving the problem, and their number may be comparable to the size of the training set. In addition to the computational cost, a natural question often arises, namely that of selecting a *convenient* nonlinear space in which to construct the linear discriminant.

Towards improving on computational complexity in the implementation of what is widely contemplated as a very promising classification strategy, namely the SRM-based classification, we propose a novel nearest neighbor rule-based [8] approach to implement the SRM principle in a binary classification problem.

The idea of the nearest neighbor (NN) classification rule may be traced back to [9], and perhaps to even earlier alternative views. The convergence properties were established in [10]. In [8], Devijver and Kittler provide a comprehensive overview of the method extending to k nearest neighbors rules with or without rejections. Bhattacharya and Kaller [11] show how to obtain exact thinnings for the k nearest neighbors rule using k-Delaunay graphs of the training set. A step towards using nearest neighbor rule in non-Euclidean spaces and using Hausdorff metrics is provided in [12].

The paper is outlined as follows: the next section is devoted to the classification problem formulation. In Section 3, we establish an equivalence between SRM and reference set thinning of nearest neighbor rule based on order statistics. In Section 4, we propose an efficient and practical algorithm to achieve the desired classification. And finally Section 5 provides substantiating examples to show the much improved performance of the proposed technique for breast cancer detection.

2. PROBLEM STATEMENT

The classification problem can be formulated as follows. Let $\mathcal{T} = \{(\boldsymbol{x}_i, y_i) \in X \times Y : i \in I\}$ a training data set, where X is a Hilbert space with an induced metric ρ which measures the similarity between patterns, $Y = \{y_1, \ldots, y_M\}$ a set of M classes, and $I = \{1, \ldots, \ell\}$. By a classifier we mean a function $f : X \to Y$ that classifies a given feature vector $\boldsymbol{x}$ to the class $y = f(\boldsymbol{x})$.

For the sake of simplicity and without loss of generality, consider a two-class classification problem with $Y = \{-1, 1\}$. The objective is to then construct a classifier that would, as accurately as possible, classify a priori unknown data into one of the classes. One of the concepts of multivariate ordering is the ψ-ordering defined as $\boldsymbol{x} \leq_\psi \boldsymbol{z}$ if $\psi(\boldsymbol{x}) \leq \psi(\boldsymbol{z})$, where ψ is a real-valued function. Denote by $\tilde{\boldsymbol{x}}_{(k)}$ the k-th ψ-order statistic of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell$, that is $\tilde{\boldsymbol{x}}_{(1)} \leq_\psi \ldots \leq_\psi \tilde{\boldsymbol{x}}_{(\ell)}$. One advantage of this type of or-

0-7803-7663-3/03/$17.00 ©2003 IEEE

This paper was originally published in the Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, & Signal Processing, April 6-10, 2003, Hong Kong (cancelled). Reprinted with permission.

dering compared to the marginal ordering, is that $\tilde{\boldsymbol{x}}_{(k)}$ is a point of the original sample. We fix $\boldsymbol{x} \in X$, and consider the function $\psi(\boldsymbol{z}) = \rho(\boldsymbol{x}, \boldsymbol{z})$ defined on X, then $\tilde{\boldsymbol{x}}_{(k)}$ is the k-th nearest neighbor of $\boldsymbol{x}$, and $d_{(k)} = \rho(\boldsymbol{x}, \tilde{\boldsymbol{x}}_{(k)})$ is the k-th order statistic of the ℓ univariate random variables $d_i = \rho(\boldsymbol{x}, \boldsymbol{x}_i)$, $i \in I$, arranged in nondecreasing order. Let $(\tilde{\boldsymbol{x}}_{(k)}, \tilde{y}_{(k)})$ be the k-th order statistic of the samples in $\mathcal{T}$ according to increasing values of d_i. The k-nearest neighbor classifier is then defined as

$$f(\boldsymbol{x}) = \begin{cases} -1 \text{ if } \frac{1}{k}\sum_{i=1}^{k} \mathbb{I}_{\{\tilde{y}_{(i)}=-1\}} \geq \frac{1}{k}\sum_{i=1}^{k} \mathbb{I}_{\{\tilde{y}_{(i)}=1\}} \\ 1 \text{ otherwise,} \end{cases} \quad (1)$$

where $\mathbb{I}_{\{\cdot\}}$ is an indicator function. When $k = 1$, the k-nearest neighbor is exactly the nearest neighbor (NN) classifier. Note that the NN classifier requires computation of $\rho(\boldsymbol{x}, \boldsymbol{x}_i)$ for all $i \in I$, for each datum $\boldsymbol{x}$ to be classified. For a large sample size ℓ, this represents a significant computational cost, and it only makes sense to attempt to alleviate this problem by developing intelligent search algorithms which quickly yield an overall minimal distance. Another effective way of coping with the computational challenge is to reduce the sample size and to approximate the classifier in Eq. (1) by restricting the indices i, j to a subset J of the original index set I. The NN classifier f_J corresponding to the set J is given by

$$f_J(\boldsymbol{x}) = \begin{cases} -1 & \text{if } \mathbb{I}_{\{\tilde{y}_{(1)}=-1\}} \geq \mathbb{I}_{\{\tilde{y}_{(1)}=1\}} \\ 1 & \text{otherwise,} \end{cases} \quad (2)$$

where $\tilde{y}_{(1)}$ is the second component of $(\tilde{\boldsymbol{x}}_{(1)}, \tilde{y}_{(1)})$ minimum order statistic of the samples in the training data set $\mathcal{T}_J = \{(\boldsymbol{x}_j, y_j) \in X \times Y : j \in J\}$ according to increasing values of d_j.

3. SRM PRINCIPLE AND NN CLASSIFIER

Let $\mathcal{F}$ be a set of indicator functions defined on X with values in $Y = \{-1, 1\}$. The risk $R(f)$ of an indicator function $f \in \mathcal{F}$ has the following upper bound [3] with probability $1 - \nu$:

$$R(f) \leq R^{emp}(f) + \varepsilon(f, \mathsf{h}, \nu) \quad (3)$$

where $\nu \in [0, 1]$, $R^{emp}(f)$ is the training error of f (or *empirical risk*), $\varepsilon(f, \mathsf{h}, \nu)$ is the confidence term, and h is the Vapnik-Chervonenkis (VC) dimension of $\mathcal{F}$. Recall that the VC dimension of $\mathcal{F}$ is the size of the largest subset S of X such that $\mathcal{F}_{|S} = \mathcal{F}$. Consider a nondecreasing sequence of subsets $\{\mathcal{F}_n\}_{n\geq 1}$ of $\mathcal{F}$ with corresponding VC dimensions $\{\mathsf{h}_n\}_{n\geq 1}$,

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \subset \mathcal{F}_n \subset \ldots$$

which implies $\mathsf{h}_1 \leq \mathsf{h}_2 \leq \ldots \leq \mathsf{h}_n \leq \ldots$.

Clearly, one can further lower the training error over $\mathcal{F}_n$ as n grows large. The confidence term $\varepsilon(f, \mathsf{h}, \nu)$, however, increases with increasing VC dimension (or increasing n). The SRM principle leads to minimizing the empirical risk by choosing an indicator function from a particular set $\mathcal{F}_{n^\star}$, so that overall, the right hand side of Eq. (3) is minimized. One natural way of minimizing this bound is to keep the first term $R^{emp}(f)$ at zero while minimizing the second one, $\varepsilon(f, \mathsf{h}, \nu)$. In addition, it can be shown that the confidence term $\varepsilon(f, \mathsf{h}_n, \nu)$ monotonically increases with increasing VC dimension h_n [3]. The problem of constructing/training a classifier may be viewed as that of selecting a function f from a set of indicator functions $\mathcal{F}_n$ subject to $R^{emp}(f) = 0$ and $\mathsf{h}_n \leq \mathsf{h}_i \; \forall i \in I^0 = \{j \in \mathbb{N} \mid \exists f_j \in \mathcal{F}_j \text{ such that } R^{emp}(f_j) = 0\}$.

The problem is thus reduced to identifying the minimum VC-dimension set $\mathcal{F}_{n^\star}$ among all possible sets $\mathcal{F}_n$ which have an element f_n achieving zero training error. The NN classifier in Eq. (2) clearly attains zero training error since each point's distance to itself is zero and eventually, each point in the training set is assigned back to its true class. The following result [13] relates the nature of the NN classifier to the VC dimension.

Proposition 1 *The VC dimension* h *associated with a function set that produces the classifier in Eq. (2) is equal to Card(J), the cardinality of the index set* J.

In order to reduce the VC dimension associated with the NN classifier in Eq. (2), we hence need to reduce the cardinality of J.

4. CLASSIFIER CONSTRUCTION

The problem of eliminating data points from a reference set of a NN classifier is referred to as the reference set thinning problem, and is introduced to reduce the cost of computing $\rho(\boldsymbol{x}, \boldsymbol{x}_i)$ for all $i \in I$. The goal is to then obtain an index set $J \subset I$ that results in an exact/approximate classification rule for all $\boldsymbol{x} \in X$. Devijver and Kittler [8] propose editing and condensing algorithms to achieve that. In [11], Bhattacharya and Kaller propose a Delaunay graph-based approach to produce exact thinnings of the initial training set for k-NN rule.

The objective of reference set thinning techniques is to achieve identical or very similar classification rules based on a smaller reference set than the original. If the smaller set produces an identical partition of the observation space, it is called an *exact thinning*. On the other hand, if a significant reduction in the number of data points in the reference set can be achieved by allowing small deviations from the original NN classification, then *inexact thinning*s are also considered admissible [11].

The algorithm proposed in this paper is similar in spirit to an index set thinning algorithm, but is drastically different from the conventional techniques cited above as our algorithm is obtained by explicitly applying the SRM principle to NN classification. The goal of the algorithm is to construct a novel NN classifier by implementing the SRM principle rather than to reduce the size of the reference set in an attempt to alleviate the computational complexity of the original NN classifier. A comparative analysis of the developed algorithm and conventional set thinning techniques may be interesting but inconsequential to our primary goal as it provides no additional insight. Let $\mathcal{T} = \{(\boldsymbol{x}_i, y_i) \in X \times Y : i \in I\}$ be a training data set. Denote by $\ell_1 = \text{Card}\left(\{y_i \in Y : y_i = -1\}\right)$ and $\ell_2 = \text{Card}\left(\{y_i \in Y : y_i = 1\}\right)$. Our problem can be formulated as:

$$\begin{aligned} &\text{minimize} && \text{Card}(J) && (4)\\ &\text{subject to} && J \subset \{1, 2, \ldots, \ell\} && (5)\\ & && R^{emp}(f_J) = 0. && (6) \end{aligned}$$

An exhaustive search is clearly impractical, since one would need to construct classifiers on a total number of $(2^{\ell_1} - 1)(2^{\ell_2} - 1)$ different index sets, test them on the training set, and pick the one that has the smallest cardinality corresponding to a classifier with no training errors. We instead propose the following algorithm: Consider all pairwise distances $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j)$ such that $y_i = -1$ and $y_j = 1$, and let d_k be an enumeration of those distances for $k = 1, \ldots, \ell_1\ell_2$. Let $d_{(k)}$ denote the k-th order statistics in the increasing order.

1. *initialize* $J = \emptyset$, $k = 1$
2. *while* $R^{emp}(f_J) > 0$ *do*
   (a) *find* $\boldsymbol{x}_i$ *and* $\boldsymbol{x}_j$ *so that* $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) = d_{(k)}$, $y_i = -1$, $y_j = 1$
   (b) *if* $\{i, j\} \not\subset J$, *update* $J \leftarrow J \cup \{i, j\}$
   (c) *increment* $k \leftarrow k + 1$

Note that the proposed procedure increases the inclusion of indices of closest pairs of points from the two classes, and keeps updating the set J to a successful and complete classification of the training set. The underlying intuition is that most classification errors occur in the regions where the domains of the two classes are closest to each other. Given a set of training data, these regions are identified by the pairs of data points of opposite classes with the smallest distance, and the separating surface should serve primarily these high-risk points. When constructing the NN rule based classifier $f_J(\boldsymbol{x})$, one therefore needs to include these points in the reference set J. The algorithm clearly converges, since in a worst case scenario all the points in the training set are included in J, and it is guaranteed to classify itself perfectly. While significantly reduced relative to Delaunay graphs and support vector machines, the remaining computational complexity of the proposed technique may be mostly accounted for by the computation of the pairwise distances and the subsequent construction and testing of $f_J(\boldsymbol{x})$ while updating J. It can be shown that the overall complexity of the algorithm is $O(n\ell^3)$ [13]. A graphical illustration of the resulting classifier is shown in Fig. 1. The crosses and dots represent points of two classes. The circled points are included in the final set indexed by J, and the edges in between indicate the smallest distance pairs taken into account in the course of the algorithm. The resulting classifier is based on 21 data points out of a total number of 40, and all the training set is classified correctly.

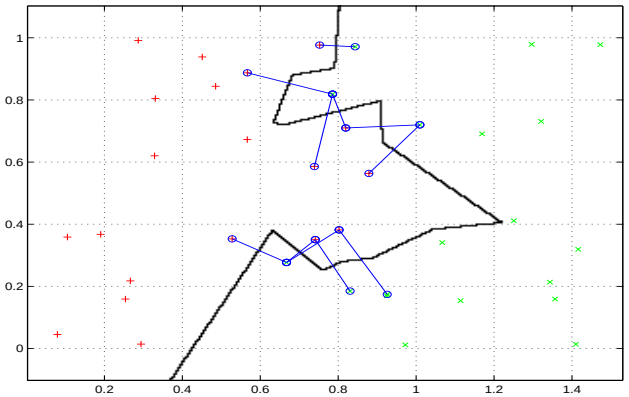

Fig. 1. Illustration of the classifier $f_J(\boldsymbol{x})$

Fig. 2 shows the decision curves constructed by $f_J(\boldsymbol{x})$, $f(\boldsymbol{x})$ in Eq. (1), and the support vector machine (SVM) classifier $f_{SVM}(\boldsymbol{x})$ based on the polynomial kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + 1)^6$.

The curves denoted by f_{3-NN} and f_{5-NN} are the decision boundaries of 3-neighbor and 5-neighbor NN classifiers respectively. The dark curve represents $f_J(\boldsymbol{x})$, the lighter piece-wise linear continuous curve represents $f(\boldsymbol{x})$, and the smooth curve represents $f_{SVM}(\boldsymbol{x})$ decision boundaries. An interesting observation is that the curves of $f_J(\boldsymbol{x})$ and $f(\boldsymbol{x})$ overlap for most of the boundary region between the two classes, and deviate towards the end points. The reason is intuitive: the shape of the NN decision curve in a neighborhood is governed by the closest points in the reference set. As the points between the two classes are mostly common to both reference sets, $f(\boldsymbol{x})$ and $f_J(\boldsymbol{x})$ decision curves are therefore expected to overlap over that region.

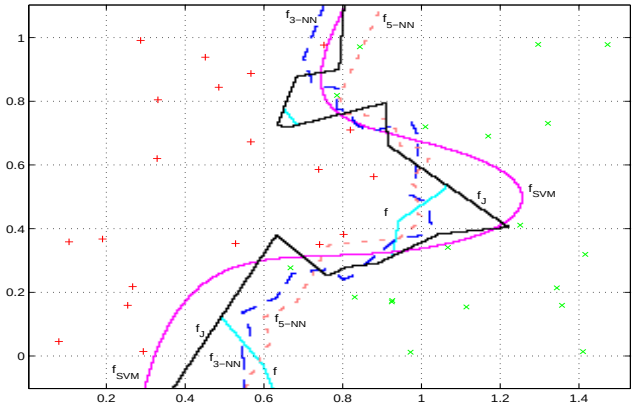

Fig. 2. Decision curves.

It is interesting to note the similarity between the construction of $f_J(\boldsymbol{x})$ and, to a large extent, that of the SVM [3]. In both cases, the resulting classifiers and the corresponding decision surfaces are based on only fractions of the data points in the training set. As the complexity of the classification rules are directly related to the number of points involved, they both seek the *simplest* classifier that agrees with the training data, and hence implement the SRM principle. There is, however, one major difference: while support vector machines may not always be able to achieve perfect classification of the training set, $f_J(\boldsymbol{x})$ is guaranteed to achieve that, provided there are no points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ with $y_i = -1$ and $y_j = 1$ such that $\boldsymbol{x}_i = \boldsymbol{x}_j$. The separation capability of the support vector machines is directly related with the choice of the feature space. In general, a feature space in which the training set is linearly separable may not be found. In that case, one needs to solve a modified optimization problem to construct an SVM classifier [3] with an even yet greater computational complexity.

5. SIMULATION RESULTS

We apply the proposed technique to breast cancer detection, which is a two-class problem, and then compare its computational and classification performances to the conventional SVM classifiers.

Given a nonlinear feature space into which the data are mapped, and a positive definite symmetric function K which defines its inner product measure, the SVM classifier is given by

$$\hat{y} = f_{SVM}(\boldsymbol{x}) = \text{sign}\left\{ \sum_{i=1}^{\ell} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b_0 \right\}, \qquad (7)$$

where α_i and b_0 are found by solving the optimization problem

$$\text{maximize} \qquad \Lambda^T \mathbf{1} - \frac{1}{2} \Lambda^T D \Lambda \qquad (8)$$

$$\text{subject to} \qquad \Lambda \in \mathbb{R}_+^{\ell} \qquad (9)$$

where $\Lambda = (\alpha_1, \ldots, \alpha_\ell)^T$, $\mathbf{1} = (1, \ldots, 1)^T$ an ℓ-dimensional vector of 1's, and $D_{i,j} = y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Solving for the SVM classifier in Eq. (7) therefore involves the computation of the $\ell \times \ell$ matrix D and solving the quadrature optimization problem in Eq. (8). Our method requires the computation of $\ell_1 \ell_2$ pairwise distances which is about $1/4^{th}$ the size of D for $\ell_1 = \ell_2 = \ell/2$ to proceed with the algorithm. Given that the computational complexity of the algorithm is polynomial ($O(n\ell^3)$), it is considered to be an *easy* [8] problem and proves to be in general faster than quadrature optimizations. In the following simulations, we construct both a SVM-based classifier and our proposed classifier on a breast cancer data set [14]. The data set consists of $n = 9$ measurements on $\ell_1 = 444$ benign and $\ell_2 = 239$ malignant cells. We randomly select (inclusive) sets of training data of sizes $\ell = 40, 120, 200, 280, 360, 440$ with equal number of points from the two classes, and use them as the basis for our classifier construction. We record the computation times and test error performances on the whole data set. The SVM based classifier was constructed using the kernel

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\tau^2}} \tag{10}$$

with $\tau = 3$. A general rule of thumb for selecting a τ for a Radial Basis Function kernel is to have $2\tau^2$ at the same order of magnitude as the average of $||\boldsymbol{x}_i - \boldsymbol{x}_j||^2$. As τ goes to 0, the corresponding nonlinear transformation maps the data points to different vertices of the unit cube in the transform space, so that $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ becomes very small. On the other hand, when τ approaches infinity, $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ goes to 1, indicating that the images of all data points are in a close neighborhood of each other. In the former case, linear separation is trivially obtained. In the latter case, however, separating the two classes becomes increasingly difficult, since all data points are aggregated together. This particular value for τ was chosen as a result of a line search over several different values at the same order of magnitude as the average of $||\boldsymbol{x}_i - \boldsymbol{x}_j||$. The constructed classifiers are able to perfectly separate the training sets in all simulations. The computation times are displayed in Fig. 3. As expected, constructing the SVM classifier takes more time than our nearest neighbor classifier for all simulated values of ℓ. This agrees with the fact that our algorithm has a polynomial order of computational complexity, and hence is easier to solve than the optimization in Eq. (8). Fig. 4 shows the classification errors over the whole data set. The number of misclassifications of the two methods are fairly similar and the overall performances are almost indistinguishable.

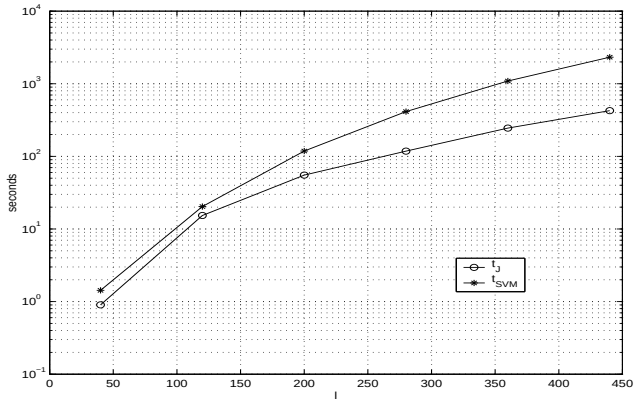

Fig. 3. Computation times of $f_J(\boldsymbol{x})$ and $f_{SVM}(\boldsymbol{x})$

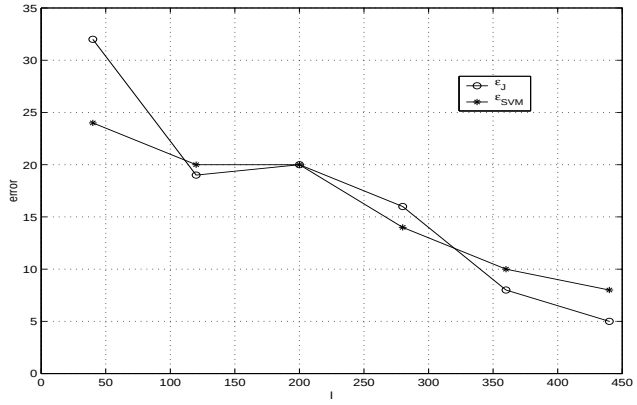

Fig. 4. Test errors of $f_J(\boldsymbol{x})$ and $f_{SVM}(\boldsymbol{x})$

6. REFERENCES

[1] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Lerarning*, vol. 20, pp. 273–297, 1995.

[2] V. Vapnik, S. Golowich, and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," in *Adv. in Neural Information Systems*, vol. 9. MIT Press, Cambridge, 1997.

[3] V. Vapnik, "Statistical learning theory," in *Adap. Learning Sys. Sig. Proc.Comm. and Cont.* John Wiley & Sons, 1998.

[4] S. Haykin, "Neural networks expand sp's horizon," *IEEE Signal Proc. Magazine*, vol. 13, no. 2, pp. 24–49, 1996.

[5] J. Rissanen, "Stochastic complexity in statistical inquiry," in *Computer Science*, vol. 15. World Scientific, 1989.

[6] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proceedings of CVPR'97*, June 1997.

[7] B. Schölkopf, C. Burges, and A. Smola, Eds., *Advances in Kernel Methods: Support Vector Learning*, MIT Press, 1999.

[8] P. A. Devijver and J. Kittler, *Pattern Recognition: A statistical approach*, Prentice Hall, London, 1982.

[9] E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination," Tech. Rep., USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

[10] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. on Information Theory*, vol. 13, 1967.

[11] B. Bhattacharya and D. Kaller, "Reference set thinning for the k-nearest neighbor decision rule," in *Fourteenth international conference on pattern recognition*, 1998, vol. 1.

[12] M. Farach-Colton and P. Indyk, "Approximate nearest neighbor algorithms for hausdorff metrics via embeddings," in *40th Annual Symp. Foundations of Comp. Science*, 1999.

[13] B. Karacali and H. Krim, "Fast minimization of structural risk by nearest neighbor method," *IEEE Trans. on Neural Networks*, submitted.

[14] O.L. Mangasarian and W.H. Wolberg, "Cancer diagnosis via linear programming," *SIAM News*, vol. 23, pp. 1–18, 1990.