Support Vector Learning

Bernhard Schölkopf
Microsoft Research Ltd
1 Guildhall Street
Cambridge CB2 3NH
England

bsc@microsoft.com
http://www.first.gmd.de/∼bs

Acknowledgements

Peter Bartlett
Chris Burges
Sebastian Mika
Klaus-Robert Müller
John Shawe-Taylor
Alexander Smola
Vladimir Vapnik
Bob Williamson

B. Schölkopf, DAGM, 14/9/1999

Roadmap

1. learning from examples
2. the kernel trick
3. support vector classification
4. other kernel algorithms
5. conclusions

... start with a small demo

B. Schölkopf, DAGM, 14/9/1999

1. Learning from Examples *(Vapnik, 1979)*

Learn a function f from examples

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell) \in \mathbb{R}^N \times Y, \quad \text{generated from } P(\mathbf{x}, y),$$

such that the expected number of errors on a test set, also drawn from $P(\mathbf{x}, y)$,

$$R(f) = \int l(f(\mathbf{x}), y)\, dP(\mathbf{x}, y),$$

is minimal. Here, l is called the loss function.

- $Y = \{\pm 1\}$: *pattern recognition*
  Loss: 1 if $f(\mathbf{x}) \neq y$, and 0 otherwise
- $Y = \mathbb{R}$: *regression estimation*
  Loss: e.g. L_2 error $l(f(\mathbf{x}), y) = (f(\mathbf{x}) - y)^2$

Problem: P is unknown. $\longrightarrow$ need an *induction principle*.
...minimize the training error $R_{emp}(f) = \sum_{i=1}^{\ell} l(f(\mathbf{x}_i), y_i)$?

B. Schölkopf, DAGM, 14/9/1999

The Importance of the Set of Functions

What about allowing *all* functions from $\mathbb{R}^N$ to $\{\pm 1\}$?

Training set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell) \in \mathbb{R}^N \times \{\pm 1\}$
Test patterns $\bar{\mathbf{x}}_1, \ldots, \bar{\mathbf{x}}_{\bar{\ell}} \in \mathbb{R}^N$,
such that $\{\bar{\mathbf{x}}_1, \ldots, \bar{\mathbf{x}}_{\bar{\ell}}\} \cap \{\mathbf{x}_1, \ldots, \mathbf{x}_\ell\} = \{\}$.

For any f there exists f^* s.t.:
1. $f^*(\mathbf{x}_i) = f(\mathbf{x}_i)$ for all i
2. $f^*(\bar{\mathbf{x}}_j) \neq f(\bar{\mathbf{x}}_j)$ for all j.

Based on the training set alone, there is *no* means of choosing which one is better. On the test set, however, they give *opposite* results. There is 'no free lunch' (Ibragimov and Has'minskii, 1981; Wolpert, 1996).
$\longrightarrow$ a restriction must be placed on the *functions* that we allow

Vapnik-Chervonenkis(VC)-Theory

Necessary and sufficient conditions for consistency of empirical risk minimization: one-sided convergence, uniformly over all functions that can be implemented by the learning machine.

$$\lim_{\ell \to \infty} P\{\sup_f (R[f] - R_{emp}[f]) > \epsilon\} = 0 \text{ for all } \epsilon > 0.$$

Hard to check for a learning machine $\rightarrow$ Vapnik, Chervonenkis and others give conditions in terms of capacity concepts, e.g.

- the VC-entropy grows sublinearly with ℓ
- the VC-dimension is finite
- the entropy numbers are well-behaved
- the classification margin is large

(e.g. Vapnik and Chervonenkis, 1974; Vapnik, 1998; Williamson et al., 1998; Alon et al., 1997)

Convergence of Means to Expectations

Law of large numbers:

$$R_{emp}[f] \rightarrow R[f]$$

as $\ell \rightarrow \infty$.

Does this imply that for the function f^* minimizing R_{emp}, and the function f^{**} minimizing R, we have

$$R_{emp}[f^*] \rightarrow R[f^{**}], \quad R[f^*] \rightarrow R[f^{**}]$$

as $\ell \rightarrow \infty$ (*"consistency"* of empirical risk minimization)?

No.
Need a *uniform* version of the law of large numbers. Uniform over *all functions that the learning machine can implement.*

Restricting the Class of Functions

Two views:
1. Statistical Learning (VC) Theory: take into account the *capacity* (e.g. VC-dimension, covering numbers, entropy numbers) of the class of functions that the learning machine can implement

2. The Bayesian Way: place *Prior distributions* $P(f)$ over the class of functions

2. The 'Kernel Trick'

A method for constructing nonlinear algorithms from linear ones: computing dot products in feature spaces via kernels

- Implicit in a result of functional analysis (Mercer, 1909)
- Used in a learning algorithm convergence proof (Aizerman et al., 1964)
- Used in a learning algorithm (Boser et al., 1992)
- General tool to construct nonlinear learning algorithms (Schölkopf et al., 1998b)

The "Curse of Dimensionality"

- classical statistics: **harder** problem, as the data are high-dimensional
- kernel-based learning: can be **simpler**:
  if Φ is chosen such that $\{\tilde{f}\}$ allows small training error *and* has low capacity, then good generalization is guaranteed.

VC Bound for Pattern Recognition with Margins

Classifiers $f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w} \cdot \mathbf{x})$ with $\|\mathbf{w}\| \leq \Lambda, \|\mathbf{x}\| \leq R$.
$\nu :=$ fraction of training examples with margin $y_i \cdot (\mathbf{w} \cdot \mathbf{x}_i) < \gamma$.

Theorem (Bartlett and Shawe-Taylor (1999)):

- for all distributions P generating the data
- with probability $\geq 1 - \delta$ over the ℓ training patterns
- for any $\gamma > 0$, and $\delta \in (0, 1)$,

$$\text{Test Error } \leq \nu + \sqrt{\frac{c}{\ell}\left(\frac{R^2\Lambda^2}{\gamma^2}\log^2 \ell + \log(1/\delta)\right)}$$

(c being some embarassingly large constant)

Hence, the margin determines the capacity.

Feature Spaces

Preprocess the data with

$$\begin{aligned} \Phi : \mathbb{R}^N &\to F \\ \mathbf{x} &\mapsto \Phi(\mathbf{x}) \end{aligned}$$
$$\text{where } N \ll \dim(F).$$

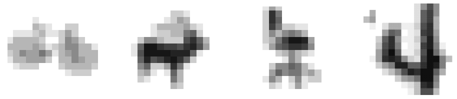
to get data

$$(\Phi(\mathbf{x}_1), y_1), \ldots, (\Phi(\mathbf{x}_\ell), y_\ell) \in F \times \{\pm 1\}.$$

Now learn

$$\begin{aligned} \tilde{f} : F &\to Y \\ \Phi(\mathbf{x}) &\mapsto y. \end{aligned} \tag{1}$$

Product Feature Space



Patterns $\mathbf{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$ where most information is contained in products between entries x_i, e.g. in $x_1 \cdot x_2$.
$\longrightarrow$ work in the space F of all products of d entries (e.g. pixels).

Number of products of d pixels in N-dimensional images:

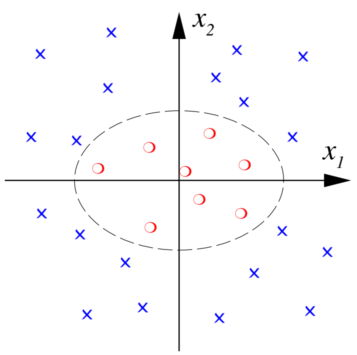
$$\frac{(N+d-1)!}{d!(N-1)!}$$

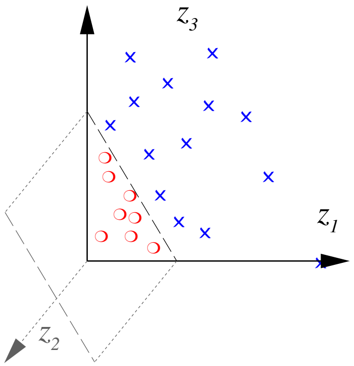
— grows like N^d.

E.g. $N = 16 \times 16$, and $d = 5 \longrightarrow$ dimensionality 10^{10}

Example: All Degree 2 Monomials

$$\begin{aligned} \Phi : \mathbb{R}^2 &\to \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2) \end{aligned}$$





The Kernel Trick *(Boser, Guyon, and Vapnik, 1992)*

$$\begin{aligned} (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})) &= (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)(y_1^2, \sqrt{2}\, y_1 y_2, y_2^2)^\top \\ &= (\mathbf{x} \cdot \mathbf{y})^2 \\ &=: k(\mathbf{x}, \mathbf{y}) \end{aligned}$$

$\longrightarrow$ the dot product in F can be computed in $\mathbb{R}^2$

The Kernel Trick, II

More general: $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$, $d \in \mathbb{N}$:

$$\begin{aligned} (\mathbf{x} \cdot \mathbf{y})^d &= \left(\sum_{j=1}^N x_j \cdot y_j \right)^d \\ &= \sum_{j_1, \ldots, j_d = 1}^N x_{j_1} \cdot \cdots \cdot x_{j_d} \cdot y_{j_1} \cdot \cdots \cdot y_{j_d} = (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})), \end{aligned}$$

where Φ maps into the space spanned by all ordered products of d input directions

The Kernel Trick, IV

- *Interpretation:* the kernel substitutes the dot product and acts as a nonlinear *similarity measure*
- this way, we can apply linear methods to vectorial as well as *non-vectorial data* (e.g. Hofmann and Buhmann, 1997) (dual representation)
- *any* algorithm that only depends on dot products can benefit from the kernel trick
- examples of common kernels:

$$\begin{aligned} \text{Polynomial} \quad k(\mathbf{x},\mathbf{y}) &= (\mathbf{x}\cdot\mathbf{y}+c)^d \\ \text{Sigmoid} \quad k(\mathbf{x},\mathbf{y}) &= \tanh(\kappa(\mathbf{x}\cdot\mathbf{y})+\Theta) \\ \text{RBF} \quad k(\mathbf{x},\mathbf{y}) &= \exp\left(-\|\mathbf{x}-\mathbf{y}\|^2/(2\,\sigma^2)\right) \end{aligned}$$

The Kernel Trick, III (Mercer's Theorem)

Still more general: If k is a continuous kernel of a positive integral operator on $L_2(\mathcal{C})$ (where $\mathcal{C}$ is some compact space),

$$\int f(\mathbf{x})k(\mathbf{x},\mathbf{y})f(\mathbf{y})\,d\mathbf{x}\,d\mathbf{y} \geq 0,$$

it can be expanded as

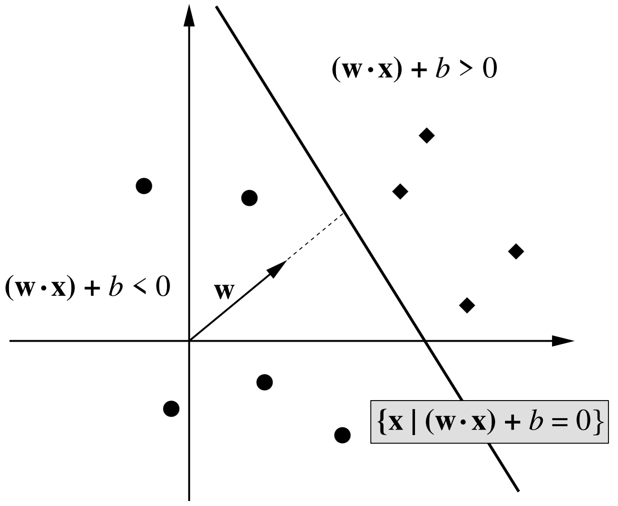
$$k(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{N_F} \lambda_i \psi_i(\mathbf{x})\psi_i(\mathbf{y})$$

using Eigenfunctions ψ_i, $\lambda_i > 0$, and $N_F \in \mathbb{N} \cup \{\infty\}$.
In that case

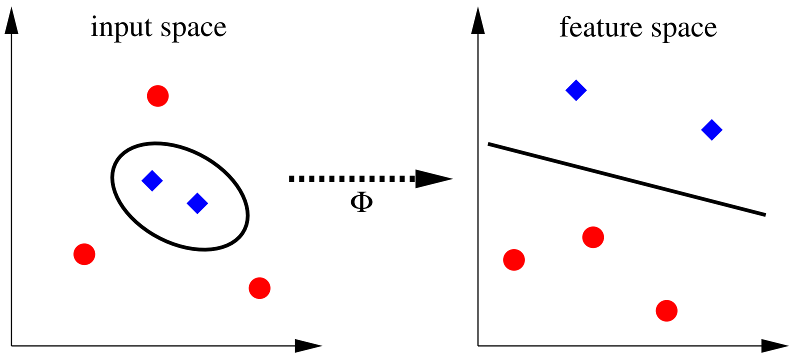
$$\Phi(\mathbf{x}) := \begin{pmatrix} \sqrt{\lambda_1}\psi_1(\mathbf{x}) \\ \sqrt{\lambda_2}\psi_2(\mathbf{x}) \\ \vdots \end{pmatrix}$$

satisfies $(\Phi(\mathbf{x})\cdot\Phi(\mathbf{y})) = k(\mathbf{x},\mathbf{y})$.

Separating Hyperplane



3. Support Vector Classifiers



Boser, Guyon, and Vapnik (1992)

Eliminating the Scaling Freedom *(Vapnik, 1995)*

Note: if $c \neq 0$, then

$$\{\mathbf{x} | (\mathbf{w} \cdot \mathbf{x}) + b = 0\} = \{\mathbf{x} | (c\mathbf{w} \cdot \mathbf{x}) + cb = 0\}.$$

Hence $(c\mathbf{w}, cb)$ describes the same hyperplane as $(\mathbf{w}, b)$.

Definition: The hyperplane is in *canonical* form w.r.t. $X^* = \{\mathbf{x}_1, \ldots, \mathbf{x}_r\}$ if $\min_{\mathbf{x}_i \in X} |(\mathbf{w} \cdot \mathbf{x}_i) + b| = 1$.

Formulation as an Optimization Problem

Hyperplane with maximum margin: minimize

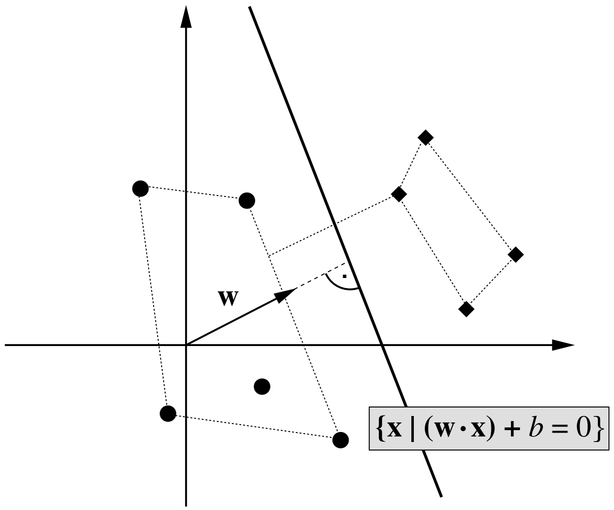
$$\|\mathbf{w}\|^2$$

(recall: margin $\sim 1/\|\mathbf{w}\|$) subject to

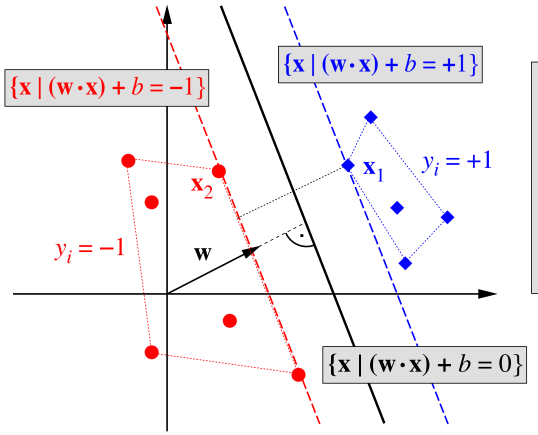
$$y_i \cdot [(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1 \quad \text{for } i = 1 \ldots \ell$$

(i.e. the training data are separated correctly).

Optimal Separating Hyperplane *(Vapnik and Lerner, 1963)*



Canonical Optimal Hyperplane



Note:

$$(\mathbf{w} \cdot \mathbf{x}_1) + b = +1$$
$$(\mathbf{w} \cdot \mathbf{x}_2) + b = -1$$

$$\Rightarrow \quad (\mathbf{w} \cdot (\mathbf{x}_1 - \mathbf{x}_2)) = 2$$

$$\Rightarrow \quad \left(\frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot (\mathbf{x}_1 - \mathbf{x}_2)\right) = \frac{2}{\|\mathbf{w}\|}$$

Derivation of the Dual Problem

At the extremum, we have

$$\frac{\partial}{\partial b} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0, \quad \frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0,$$

i.e.

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0$$

and

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i.$$

Substitute both into L to get the *dual problem*

Lagrange Function

Introduce Lagrange multipliers $\alpha_i \geq 0$ and a Lagrangian

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{\ell} \alpha_i \left(y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) - 1 \right).$$

L has to minimized w.r.t. the *primal variables* $\mathbf{w}$ and b and maximized with respect to the *dual variables* α_i

- if a constraint is violated, then $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) - 1 < 0 \longrightarrow$
  - α_i will grow to increase L — how far?
  - $\mathbf{w}$, b want to decrease L; i.e. they have to change such that the constraint is satisfied. If the problem is separable, this ensures that $\alpha_i < \infty$.
- similarly: if $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) - 1 > 0$, then $\alpha_i = 0$: otherwise, L could be increased by decreasing α_i *(KKT conditions)* (e.g. Bertsekas, 1995)

A Mechanical Interpretation *(Burges and Schölkopf, 1997)*

Assume that each SV $\mathbf{x}_i$ exerts a perpendicular force of size α_i and sign y_i on a solid plane sheet lying along the hyperplane.

Then the solution is mechanically stable:

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0 \quad \text{implies that the forces sum to zero}$$

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i \quad \text{implies that the torques sum to zero,}$$

via

$$\sum_i \mathbf{x}_i \times y_i \alpha_i \cdot \mathbf{w}/\|\mathbf{w}\| = \mathbf{w} \times \mathbf{w}/\|\mathbf{w}\| = 0.$$

The Support Vector Expansion

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i$$

where for all $i = 1, \dots, \ell$ either

$y_i \cdot [(\mathbf{w} \cdot \mathbf{x}_i) + b] > 1 \quad \Longrightarrow \alpha_i = 0 \longrightarrow \mathbf{x}_i$ irrelevant
or
$y_i \cdot [(\mathbf{w} \cdot \mathbf{x}_i) + b] = 1$ (*on* the margin) $\longrightarrow \mathbf{x}_i$ "Support Vector"

The solution is determined by the examples on the margin.

Thus

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sgn}\left((\mathbf{x} \cdot \mathbf{w}) + b\right) \\ &= \operatorname{sgn}\left(\sum_{i=1}^{\ell} \alpha_i y_i (\mathbf{x} \cdot \mathbf{x}_i) + b\right). \end{aligned}$$

Nonseparable Problems *(Cortes and Vapnik, 1995)*

If $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1$ cannot be satisfied, then $\alpha_i \to \infty$.

Modify the constraint to

$$y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - \xi_i$$

with

$$\xi_i \geq 0$$

("*soft margin*") and penalize

$$\sum_{i=1}^{\ell} \xi_i$$

in the objective function.

The ν-Property

SVs: $\alpha_i > 0$
"margin errors:" $\xi_i > 0$

KKT-Conditions $\Longrightarrow$

- All margin errors are SVs.
- Not all SVs need to be margin errors.
  Those which are *not* lie exactly on the edge of the margin.

Proposition:
1. *fraction of Margin Errors* $\leq \nu \leq$ *fraction of SVs.*
2. *asymptotically:* ... $= \nu =$ ...

Dual Problem

Dual: maximize

$$W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

subject to

$$\alpha_i \geq 0, \quad i = 1, \ldots, \ell, \quad \text{and} \quad \sum_{i=1}^{\ell} \alpha_i y_i = 0.$$

Both the final decision function and the function to be maximized are expressed in dot products $\longrightarrow$ can use kernels.

Soft Margin SVMs

C-SVM (Cortes and Vapnik, 1995): for $C > 0$, minimize

$$\tau(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{\ell} \sum_{i=1}^{\ell} \xi_i$$

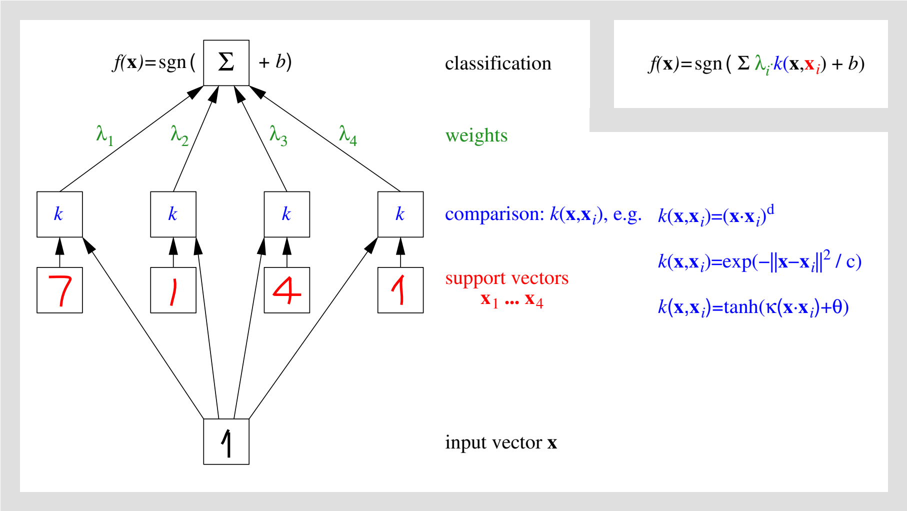
subject to $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$ (margin $2/\|\mathbf{w}\|$)

ν-SVM (Schölkopf et al., 1999c): for $0 \leq \nu < 1$, minimize

$$\tau(\mathbf{w}, \boldsymbol{\xi}, \rho) = \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{\ell} \sum_{i} \xi_i$$

subject to $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq \rho - \xi_i, \quad \xi_i \geq 0$ (margin $2\rho/\|\mathbf{w}\|$)

The SVM Architecture



Results

US Postal Service (7291 examples, 16×16); three SVMs:

Polynomial classifiers:	$k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})^d$
RBF classifiers:	$k(\mathbf{x},\mathbf{y}) = \exp\left(-\Vert\mathbf{x}-\mathbf{y}\Vert^2/(2\sigma^2)\right)$
MLPs:	$k(\mathbf{x},\mathbf{y}) = \tanh(\kappa\cdot(\mathbf{x}\cdot\mathbf{y}) - \Theta)$

- they use almost the same ($\approx$ 90%) small sets ($\approx$ 4% of the data base) of SVs.
- all three systems perform well ($\approx$ 4% error)

C4.5 decision tree	16%	(Cortes and Vapnik, 1995)
LeNet1	5%	(LeCun et al., 1989)
SVM	4%	(Schölkopf et al., 1995)
SVM + invariances	3%	(Schölkopf, 1997)
Humans; Tangent Distance	2.5%	(Simard et al., 1993)

Duals, Using Kernels

C-SVM dual: maximize

$$W(\boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j)$$

subject to $0 \le \alpha_i \le \frac{C}{\ell}$, $\sum_i \alpha_i y_i = 0$.

ν-SVM dual: maximize

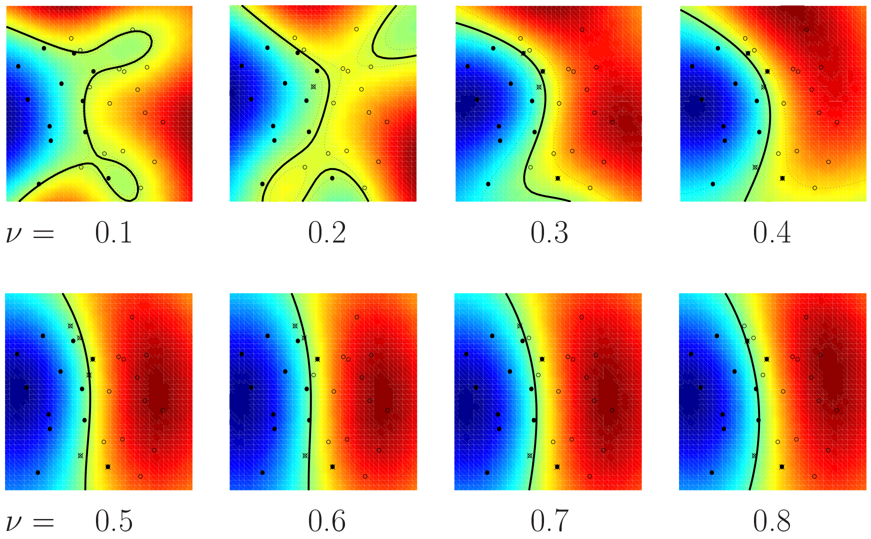
$$W(\boldsymbol{\alpha}) = -\frac{1}{2}\sum_{ij} \alpha_i\alpha_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j)$$

subject to $0 \le \alpha_i \le \frac{1}{\ell}$, $\sum_i \alpha_i y_i = 0$, $\sum_i \alpha_i \ge \nu$

In both cases: *decision function*:

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{\ell} \alpha_i y_i k(\mathbf{x},\mathbf{x}_i) + b\right)$$

ν-SVC Toy Example



Further SV Pattern Recognition Applications

- Text categorization (Joachims, 1997; Dumais, 1998)
- Face detection (Osuna, Freund, and Girosi, 1997b)
- Object recognition (Blanz et al., 1996)
- High-Energy Physics (Barabino et al., 1999; Vannerem et al., 1999)
- DNA analysis (Zien et al., 1999)
- etc.

In all cases, state-of-the-art performances.

Chunk Training

- naive approach: the complexity of maximizing
$$W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$
scales with the third power of the training set size ℓ
- only SVs are relevant $\longrightarrow$ only compute $(k(\mathbf{x}_i, \mathbf{x}_j))_{ij}$ for SVs. Extract them iteratively by cycling through the training set in chunks (Vapnik, 1979)
- in fact, one can use chunks which do not even contain all SVs. Maximize over these sub-problems. In particular, throw out SVs that hit the upper boundary (Osuna et al., 1997a).
- the extreme case: by making the sub-problems very small (just two points), one can solve them analytically (Platt, 1998).

MNIST Results

SVM + invariances is the best single classifier on the MNIST benchmark set of handwritten characters (60000 examples, 28×28) (Schölkopf, 1997).

Better than: *LeNet 5*, *Tangent Distance*. Slightly worse than a *boosted ensemble of Neural Nets:* 0.8% vs. 0.7% error (LeCun et al., 1995).

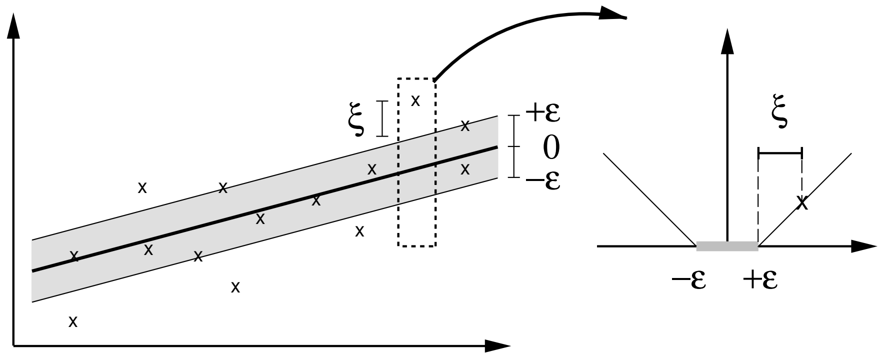
Speeding up the decision rule

Approximate
$$\Psi = \sum_{i=1}^{\ell} \alpha_i y_i \Phi(\mathbf{x}_i)$$
by
$$\Psi' = \sum_{i=1}^{N_z} \gamma_i \Phi(\mathbf{z}_i),$$
with $N_z \ll \ell$: Minimize
$$\rho = \|\Psi - \Psi'\|^2$$
Note that ρ can be expressed in terms of k by using $(\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})) = k(\mathbf{x}, \mathbf{y})$.
(e.g. Burges, 1996; Burges and Schölkopf, 1997; Osuna and Girosi, 1998; Schölkopf et al., 1999b)

4. Further Kernel Algorithms

(a) SV regression
(b) one-class problems
(c) kernel PCA
(d) Fisher kernel algorithms

ε-SV Regression Estimation *(Vapnik, 1995)*



SV Machines: Crucial Aspects

Computational The training of an SV machine comprises a (possibly large) convex quadratic program with simple constraints.

Statistical The kernel method allows to reduce a large class of learning machines to separating hyperplanes in some kernel feature space. For these, capacity can be controlled independent of dimensionality.

(a) SV Regression: ε-Insensitive Loss *(Vapnik, 1995)*

Goal: generalize SV pattern recognition to regression, preserving the following properties:

- formulate the algorithm for the linear case, and then use kernel trick
- sparse representation of the solution in terms of SVs

ε-Insensitive Loss:

$$|y - f(\mathbf{x})|_\varepsilon := \max\{0, |y - f(\mathbf{x})| - \varepsilon\}$$

Estimate a linear regression $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b$ by minimizing

$$\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{\ell}\sum_{i=1}^{\ell} |y_i - f(\mathbf{x}_i)|_\varepsilon .$$

ν-SV Regression *(Schölkopf et al., 1998a)*

Again, use ν to eliminate another parameter:
Estimate ε from the data s.t. the ν-property holds.

Primal problem: for $0 \leq \nu \leq 1$, minimize

$$\tau(\mathbf{w}, \varepsilon) = \frac{1}{2}\|\mathbf{w}\|^2 + C\left(\nu\varepsilon + 1/\ell \sum_{i=1}^{\ell} |y_i - f(\mathbf{x}_i)|_\varepsilon\right)$$

The ν-Property

Proposition 1 *Assume $\varepsilon > 0$. The following statements hold:*

(i) ν is an upper bound on the fraction of errors.

(ii) ν is a lower bound on the fraction of SVs.

(iii) Suppose the data were generated iid from a 'well-behaved' distribution $P(\mathbf{x}, y)$. With probability 1, asymptotically, ν equals both the fraction of SVs and the fraction of errors.*

* Essentially, $P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$ with $P(y|\mathbf{x})$ continuous (some details omitted).

Solving the Regression Problem

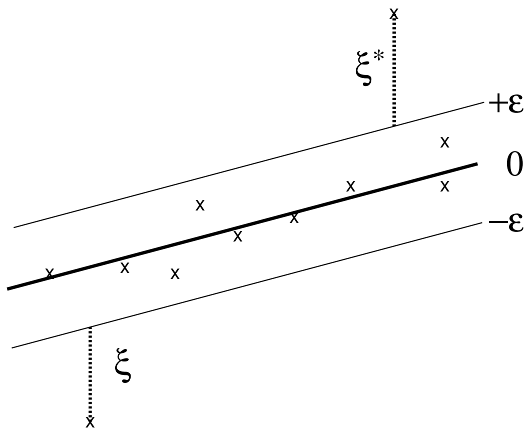
- compute the dual
- introduce kernels
- solve the QP

The regression estimate takes the form

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b,$$

where $\alpha_i \neq 0$ only if the corresponding point $\mathbf{x}_i$ is *not inside* the tube. Call $\mathbf{x}_i$ a *Support Vector*.

A Graphical Proof of the ν-Property



Cost function: $\frac{1}{2C}\|\mathbf{w}\|^2 + \nu\varepsilon + \frac{1}{\ell}\sum_{i=1}^{\ell}(\xi_i + \xi_i^*)$

Boston Housing Benchmark

- 506 examples, 13-dimensional.

Results (MSE):

- Bagging regression trees: 11.7 (Breiman, 1994)
- ε-SV regression: 7.6 (Stitson et al., 1999)

- 100 runs, with 25 randomly selected test points.
- training set is split into actual training set and validation set (80 points) for selecting ε, C, and kernel parameters
- ftp://ftp.ics.uci.com/pub/machine-learning-databases/housing

Robustness of SV Regression

Proposition. Using SVR with $|.|_\varepsilon$, local movements of target values of points outside the tube do not change the estimated regression.

Proof.

1. Shift y_i locally $\longrightarrow$ $(\mathbf{x}_i, y_i)$ still outside the tube $\longrightarrow$ original dual solution $\boldsymbol{\alpha}^{(*)}$ still feasible ($\alpha_i^{(*)} = C$, since *all* points outside the tube are at the upper bound).
2. The primal solution, with ξ_i transformed according to the movement, is also feasible.
3. The KKT conditions are still satisfied, as still $\alpha_i^{(*)} = C$. Thus (Bertsekas, 1995, e.g.), $\boldsymbol{\alpha}^{(*)}$ is still the optimal solution.

ν-SV-Regression: Automatic Tube Tuning

Demo: **sinc** function with different amounts of noise.

Comparison: ν vs. ε

ν-SVR	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
automatic ε	2.6	1.7	1.2	0.8	0.6	0.3	0.0	0.0	0.0	0.0
MSE	9.4	8.7	9.3	9.5	10.0	10.6	11.3	11.3	11.3	11.3
Errors	0.0	0.1	0.2	0.2	0.3	0.4	0.5	0.5	0.5	0.5
SVs	0.3	0.4	0.6	0.7	0.8	0.9	1.0	1.0	1.0	1.0

ε-SVR	0	1	2	3	4	5	6	7	8	9	10
MSE	11.3	9.5	8.8	9.7	11.2	13.1	15.6	18.2	22.1	27.0	34.3
Errors	0.5	0.2	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SVs	1.0	0.6	0.4	0.3	0.2	0.1	0.1	0.1	0.1	0.1	0.1

- RBF kernel, C and σ chosen as in (Schölkopf et al., 1997)

Separating Unlabelled Data from the Origin

Given some $\rho > 0$, call

$$\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_\ell)$$

separable if there exists $\mathbf{w}$ such that $(\mathbf{w} \cdot \Phi(\mathbf{x}_i)) \geq \rho$ for all i.
The distance of the hyperplane $\{\mathbf{z} \in F : (\mathbf{w} \cdot \mathbf{z}) = \rho\}$ to the origin is $\rho / \|\mathbf{w}\|$.
Hence, for any $\rho > 0$, the solution of

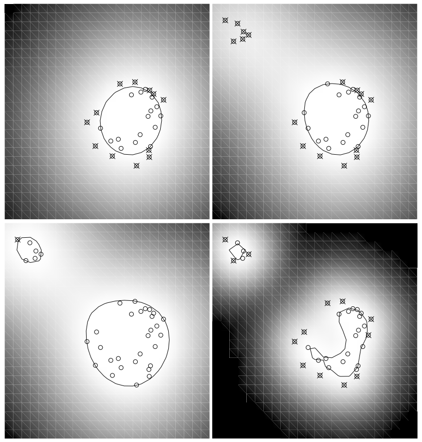
$$\min_{\mathbf{w} \in F} \frac{1}{2}\|\mathbf{w}\|^2 \qquad \text{subject to} \quad (\mathbf{w} \cdot \Phi(\mathbf{x}_i)) \geq \rho$$

gives the unique *supporting hyperplane* with the properties

- it is closer to the origin than all data
- its distance to the origin is maximal among all such hyperplanes

(cf. Vapnik, Chervonenkis, Lerner, 1963/64, *Generalized Portrait*)

Demo using $k(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{c})$



Top: $\nu = c = 0.5$, *bottom left:* $\nu = 0.1, c = 0.5$, *bottom right:* $\nu = 0.5, c = 0.1$

(b) Unsupervised Learning

training points $\mathbf{x}_1, \ldots, \mathbf{x}_\ell \sim P$

- extreme view: unsupervised learning = density estimation. Hardest problem!
- easier problem: for any $\nu \in (0, 1]$, compute a function $f : \mathcal{X} \to \{\pm 1\}$ such that

$$P(f^{-1}(\{1\})) \approx 1 - \nu.$$

$\approx$ estimate the *support* of a distribution, not its density.

ν-Soft Margin Version *(Schölkopf et al., 1999)*

Separate "most of" the data from the origin by a large margin by solving, for $\nu \in (0, 1]$,

$$\min_{\mathbf{w} \in F, \boldsymbol{\xi} \in \mathbb{R}^\ell, \rho \in \mathbb{R}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\ell}\sum_i \xi_i - \nu\rho$$

$$\text{subject to} \quad (\mathbf{w} \cdot \Phi(\mathbf{x}_i)) \geq \rho - \xi_i, \quad \xi_i \geq 0 \quad \text{for all } i.$$

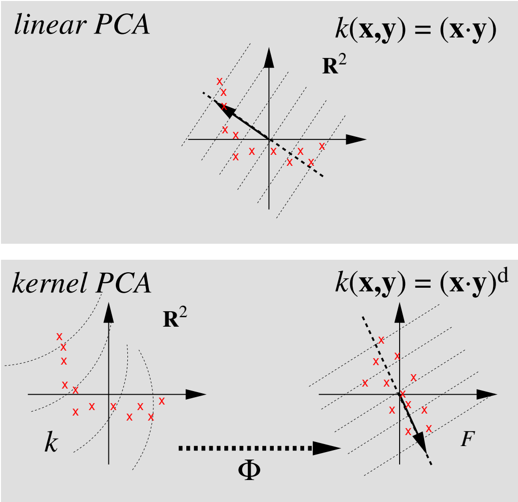
Result of the optimization:

- the decision function $f(\mathbf{x}) = \operatorname{sgn}((\mathbf{w} \cdot \Phi(\mathbf{x})) - \rho)$ will be positive for "most" examples $\mathbf{x}_i$ contained in the training set
- $\|\mathbf{w}\|$ will be small, hence the separation from the origin should be large

Related approaches: enclose data in a sphere (Schölkopf et al., 1995; Tax and Duin, 1999)

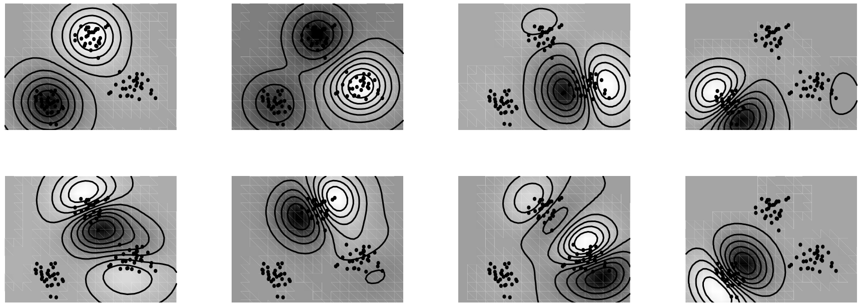
(c) Kernel PCA

(Schölkopf et al., 1998b)



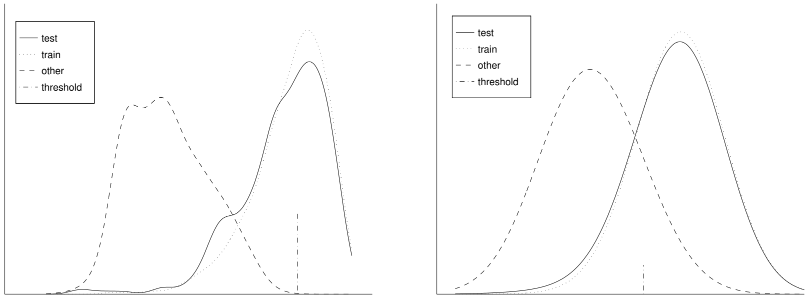
Toy Example: RBF Kernel, 8 Principal Components

$$k(\mathbf{x},\mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{0.1}\right)$$



Matlab code available from http://svm.first.gmd.de

Result on USPS Data



$\nu = 0.5$, 50% SVs, 49% outliers $\quad$ $\nu = 0.05$, 6% SVs, 4% outliers

Recognizer trained only on 0; output histograms for 0 and non-0.

US postal service digit recognition task (7298 training examples, 2007 test examples, size 16×16); Gaussian kernel of width 0.5.

Kernel Principal Component Analysis

To solve the Eigenvalue problem for

$$C = \frac{1}{\ell}\sum_{j=1}^{\ell} \Phi(\mathbf{x}_j)\Phi(\mathbf{x}_j)^\top,$$

expand the solutions as

$$\mathbf{V}^k = \sum_{i=1}^{\ell} \alpha_i^k \Phi(\mathbf{x}_i)$$

and solve the corresponding problem for the $\boldsymbol{\alpha}^k$: diagonalize

$$K_{ij} := (\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)) = k(\mathbf{x}_i,\mathbf{x}_j).$$

Compute projections on the Eigenvectors in F for a test point $\mathbf{x}$:

$$(\Phi(\mathbf{x})\cdot\mathbf{V}^k) = \sum_{i=1}^{\ell} \alpha_i^k\, k(\mathbf{x},\mathbf{x}_i)$$

(d) The Fisher kernel *(Jaakkola and Haussler, 1999)*

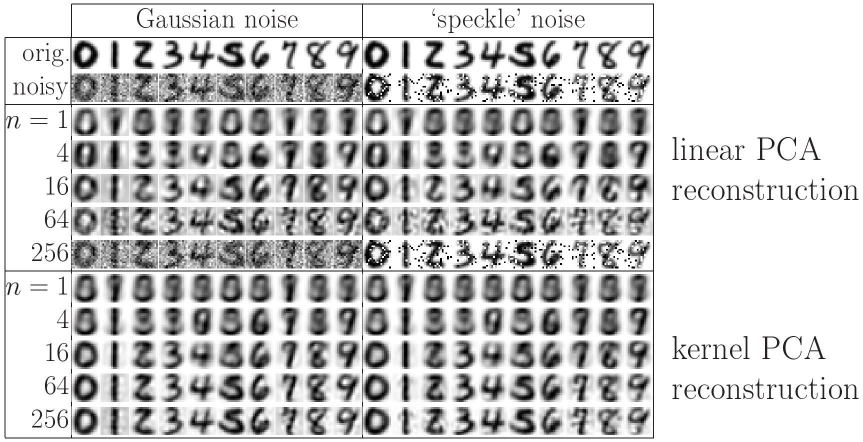
Goal: define a suitable similarity measure for data generated from an underlying probabilistic model $p(X|\theta)$ (e.g. HMM)

- Score vector $U_X = \nabla_\theta \log p(X|\theta)$
- Kernel
$$K(X_i, X_j) \propto U_{X_i}^T I^{-1} U_{X_j},$$
  with I being the Fisher information matrix
- $K(X_i, X_j)$ is invariant to reparameterization of θ
- successfully applied to biosequence analysis problems

5. Conclusions

- carry out linear methods in high-dimensional spaces with a computational cost independent of their dimensionality
- every algorithm that can be formulated in dot products could benefit from this technique
- state-of-the-art results with mathematically well-founded algorithms

Denoising of USPS Handwritten Digits *(Mika et al., 1999)*



Another application: face modelling (Romdhani et al., 1999).

Regularization Properties of the Fisher kernel

Smola and Schölkopf (1998a) showed that an SVM with kernel k minimizes the regularized risk
$$R_{reg}[f] = R_{emp}[f] + \lambda \|Tf\|^2,$$
over the functions
$$f(\mathbf{x}) = \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b$$
if k is Green's function of T^*T.
Applied to the Fisher kernel, we get (Oliver et al., 1999):
$$\|Tf\|^2 = \|f\|^2_{L_2(p)},$$
i.e. the Fisher Kernel induced by a generative probability model with density p corresponds to a regularization term equal to the $L_2(p)$-norm of the estimated function.

References

M. Aizerman, E. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821 – 837, 1964.

N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale–sensitive Dimensions, Uniform Convergence, and Learnability. *Journal of the ACM*, 44(4):615–631, 1997.

N. Barabino, M. Pallavicini, A. Petrolini, M. Pontil, and A. Verri. Support vector machines vs multi-layer perceptrons in particle identification. In M. Verleysen, editor, *Proceedings ESANN*, pages 257 – 262, Brussels, 1999. D Facto.

P. Bartlett and J. Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 43 – 54. MIT Press, Cambridge, MA, 1999.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.

V. Blanz, B. Schölkopf, H. Bülthoff, C. Burges, V. Vapnik, and T. Vetter. Comparison of view-based object recognition algorithms using realistic 3D models. In C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff, editors, *Artificial Neural Networks — ICANN'96*, pages 251 – 256, Berlin, 1996. Springer Lecture Notes in Computer Science, Vol. 1112.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, Pittsburgh, PA, July 1992. ACM Press.

L. Breiman. Bagging predictors. Technical Report 421, Department of Statistics, UC Berkeley, 1994. ftp://ftp.stat.berkeley.edu/pub/tech-reports/421.ps.Z.

C. J. C. Burges. Simplified support vector decision rules. In L. Saitta, editor, *Proceedings, 13th Intl. Conf. on Machine Learning*, pages 71–77, San Mateo, CA, 1996. Morgan Kaufmann.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):1–47, 1998.

C. J. C. Burges and B. Schölkopf. Improving the accuracy and speed of support vector learning machines. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 375–381, Cambridge, MA, 1997. MIT Press.

C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273 – 297, 1995.

S. Dumais. Using SVMs for text categorization. *IEEE Intelligent Systems*, 13(4), 1998. In: M.A. Hearst, B. Schölkopf, S. Dumais, E. Osuna, and J. Platt: Trends and Controversies — Support Vector Machines.

T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):1–25, 1997.

T. S. Jaakkola and D. Haussler. Exploiting Generative Models in Discriminative Classifiers. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*. MIT Press, 1999.

T. Joachims. Text categorization with support vector machines: Learning with many relevant features. Technical Report 23, LS VIII, University of Dortmund, 1997.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. J. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541 – 551, 1989.

Y. LeCun, L. D. Jackel, L. Bottou, A. Brunot, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Müller, E. Säckinger, P. Simard, and V. Vapnik. Comparison of learning algorithms for handwritten digit recognition. In F. Fogelman-Soulié and P. Gallinari, editors, *Proceedings ICANN'95 — International Conference on Artificial Neural Networks*, volume II, pages 53 – 60, Nanterre, France, 1995. EC2. The MNIST benchmark data is available from http://www.research.att.com/~yann/ocr/mnist/.

J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London*, A 209:415–446, 1909.

S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de-noising in feature spaces. In *Advances in Neural Information Processing Systems 11*, 1999.

N. Oliver, B. Schölkopf, and A.J. Smola. Natural regularization in svms. 1999.

E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. AI Memo 1602, Massachusetts Institute of Technology, 1997a.

E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *Proc. Computer Vision and Pattern Recognition '97*, pages 130–136, 1997b.

E. Osuna and F. Girosi. Reducing run-time complexity in SVMs. In *Proceedings of the 14th Int'l Conf. on Pattern Recognition, Brisbane, Australia*, 1998.

J. C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research, 1998.

S. Romdhani, Gong. S., and A. Psarrou. Multi-view nonlinear active shape model using kernel pca. In *British Machine Vision Conference*, 1999.

B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In U. M. Fayyad and R. Uthurusamy, editors, *Proceedings, First International Conference on Knowledge Discovery & Data Mining*. AAAI Press, Menlo Park, CA, 1995.

B. Schölkopf. *Support Vector Learning*. R. Oldenbourg Verlag, München, 1997. Doktorarbeit, TU Berlin.

B. Schölkopf, P. L. Bartlett, A. Smola, and R. C. Williamson. Support vector regression with automatic accuracy control. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of the 8th International Conference on Artificial Neural Networks*, Perspectives in Neural Computing, pages 111 – 116, Berlin, 1998a. Springer Verlag.

9. Further Details...

This tutorial cannot explain all details, or cover all contributions to SV research. For further information, see

- `http://svm.first.gmd.de` *(SVM home page)*
- Vapnik (1995, 1998) *(SVMs and learning theory)*
- Schölkopf, Burges, and Smola (1999a) *(Proceedings of the 1997 NIPS workshop on SV learning)*
- Burges (1998); Smola and Schölkopf (1998b) *(SV tutorials)*
- Smola, Bartlett, Schölkopf, and Schuurmans (1999) *(Proceedings of the 1998 NIPS workshop on large margin classifiers)*

M. Stitson, A. Gammerman, V. Vapnik, V. Vovk, C. Watkins, and J. Weston. Support vector regression with ANOVA decomposition kernels. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 285 – 291. MIT Press, Cambridge, MA, 1999.

D.M.J. Tax and R.P.W. Duin. Data domain description by support vectors. In M. Verleysen, editor, *Proceedings ESANN*, pages 251 – 256, Brussels, 1999. D Facto.

P. Vannerem, K.-R. Müller, A.J. Smola, B. Schölkopf, and S. Söldner-Rembold. Classifying lep data with support vector algorithms. In *Proceedings of AIHENP'99*, 1999.

V. Vapnik. *Estimation of Dependences Based on Empirical Data [in Russian].* Nauka, Moscow, 1979. (English translation: Springer Verlag, New York, 1982).

V. Vapnik. *The Nature of Statistical Learning Theory.* Springer Verlag, New York, 1995.

V. Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition [in Russian].* Nauka, Moscow, 1974. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie-Verlag, Berlin, 1979).

V. Vapnik and A. Lerner. Pattern recognition using generalized portraits. *Avtomatika i Telemekhanika*, 24:774 – 780, 1963.

R. C. Williamson, A. J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. Technical Report 19, NeuroCOLT, http://www.neurocolt.com, 1998.

A. Zien, G. Rätsch, S. Mika, B. Schölkopf, C. Lemmen, A. Smola, T. Lengauer, and K.-R. Müller. Engineering support vector machine kernels that recognize translation initiation sites in DNA, 1999. submitted to GCB'99.

B. Schölkopf, C. J. C. Burges, and A. J. Smola. *Advances in Kernel Methods — Support Vector Learning.* MIT Press, Cambridge, MA, 1999a.

B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 1999b. in press.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299 – 1319, 1998b.

B. Schölkopf, A. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *To appear in: Neural Computation*, 1999c. Also: NeuroColt2-TR 1998-031.

B. Schölkopf, K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with Gaussian kernels to radial basis function classifiers. *IEEE Trans. Sign. Processing*, 45:2758 – 2765, 1997.

B. Schölkopf, R.C. Williamson, A.J. Smola, and J. Shawe-Taylor. SV estimation of a distribution's support. Accepted for NIPS'99, 1999.

P. Simard, Y. LeCun, and J. Denker. Efficient pattern recognition using a new transformation distance. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 50–58, San Mateo, CA, 1993. Morgan Kaufmann.

A. Smola and B. Schölkopf. On a kernel-based method for pattern recognition, regression, approximation and operator inversion. *Algorithmica*, 22:211 – 231, 1998a.

A. Smola and B. Schölkopf. A tutorial on support vector regression. NeuroColt2-TR 1998-030, http://svm.first.gmd.de, 1998b.

A. J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans. *Advances in large margin classifiers.* MIT Press, Cambridge, MA, 1999. in press.