How to teach Support Vector Machine to learn vector outputs

Sandor Szedmak [1 2]

joined work with

John Shawe-Taylor, Craig Saunders [1] and Juho Rousu [2]

the list is open for the future...

[1] ISIS, Electronics and Computer Science
University of Southampton

[2] Department of Computer Science
University of Helsinki

Helsinki 2006

Outline

Previous work

- Micchelli and Pontil(2003): On Learning Vector-valued Functions,
  - Regularized least square approach,
- Taskar, Guestrin and Koller (2003): Max-Margin Markov Networks,
  - SVM based, label vector learning,
- Tsochantaridis, Joachims, Hofmann and Altun (2005): Large Margin Methods for Structured and Interdependent Output Variables,
  - SVM based vector learning, similar to the Taskar's framework.

Common problem is the high computational complexity.

Learning strategy

Embedding where the structures of the input and output objects are represented in properly chosen spaces(Hilbert, Banach, ...).

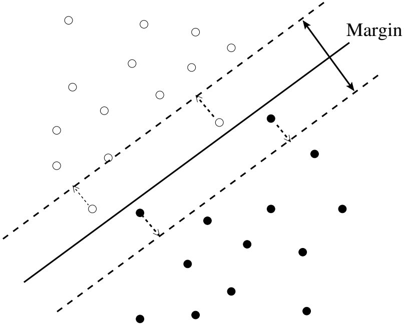
Optimization has to find the similarity based matching between the input and the output representations.

Inversion(Pre-image problem) has to recover the best fitting output structure of its representation.

Embedding

Embedding	$\phi : \overbrace{\text{input space}}^{\mathcal{X}} \rightarrow \overbrace{\text{feature space}}^{\mathcal{H}_\phi}$ $\psi : \overbrace{\text{output space}}^{\mathcal{Y}} \rightarrow \overbrace{\text{label space}}^{\mathcal{H}_\psi}$
Similarity transformation	$\widetilde{\mathbf{W}} = (\mathbf{W}, \mathbf{b}) \Rightarrow \psi(\mathbf{y}) \sim \widetilde{\mathbf{W}}\phi(\mathbf{x})$
Inversion	$\psi^{-1}(\mathbf{Y})$

The “Classical” Support Vector Machine(SVM)



$$\min \quad \frac{1}{2}\boxed{\mathbf{w}^T\mathbf{w}} + C\mathbf{1}^T\xi$$

w.r.t. $\boxed{\mathbf{w} : \mathcal{H}_\phi \to \mathbb{R}}$, normal vec.

$\boxed{b \in \mathbb{R}}$, bias

$\xi \in \mathbb{R}^m$, error vector

s.t. $\boxed{y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b)} \geq 1 - \xi_i$

$\xi \geq \mathbf{0}, \; i = 1, \ldots, m$

Reinterpretation of the normal vector $\mathbf{w}$

Original

- $y_i \in \{-1, +1\}$ binary outputs
- $\mathbf{w}$ is the normal vector of the separating hyperplane.

New

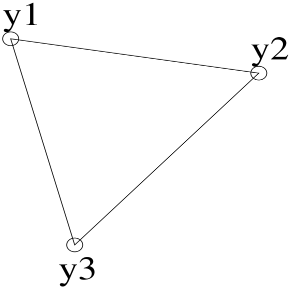
- $y_i \in \mathcal{Y}$ arbitrary outputs
  - $\psi(y_i) \in \mathcal{H}_\psi$ embedded labels in a linear vector space
- $\mathbf{w}^T$ is a linear operator projecting the input space into the output space.
  - The aim to find the highest similarity between the output and the projected input.

The output space is a one dimensional subspace in the SVM.

Affine transformation = Linear transformation +translation

Singular value decomposition of $\mathbf{W} = \mathbf{UDV}^T$

Output



Transformation

$\Longleftarrow (\Longrightarrow)$

$\mathbf{U}$ = Rotation

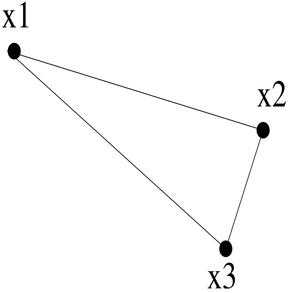
$\mathbf{D} = \begin{cases} \text{Scaling} \\ \text{Projection} \end{cases}$

$\mathbf{V}$ = Rotation

$+$

$\mathbf{b}$ = Translation

Input



Primal problem

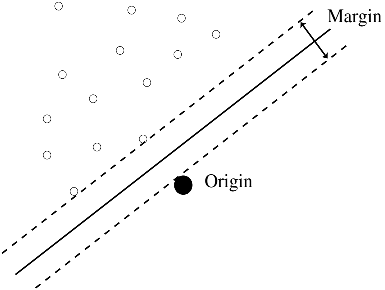
Binary class learning	Vector label learning
Support Vector Machine(SVM)	Maximum Margin Robot(MMR)

min	$\frac{1}{2}\underbrace{\boxed{\mathbf{w}^T\mathbf{w}}}_{\|\mathbf{w}\|_2^2} + C\mathbf{1}^T\boldsymbol{\xi}$	$\frac{1}{2}\underbrace{\boxed{\mathbf{tr}(\mathbf{W}^T\mathbf{W})}}_{\|\mathbf{W}\|^2_{Frobenius}} + C\mathbf{1}^T\boldsymbol{\xi}$
w.r.t.	$\boxed{\mathbf{w}:\mathcal{H}_\phi \to \mathbb{R}}$, normal vec.	$\boxed{\mathbf{W}:\mathcal{H}_\phi \to \mathcal{H}_\psi}$, linear operator
	$\boxed{b \in \mathbb{R}}$, bias	$\boxed{\mathbf{b} \in \mathcal{H}_\psi}$, translation(bias)
	$\boldsymbol{\xi} \in \mathbb{R}^m$, error vector	$\boldsymbol{\xi} \in \mathbb{R}^m$, error vector
s.t.	$\boxed{y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b)} \geq 1 - \xi_i$	$\boxed{\langle \psi(\mathbf{y}_i), \mathbf{W}\phi(\mathbf{x}_i) + \mathbf{b}\rangle_{\mathcal{H}_\psi}} \geq 1 - \xi_i$
	$\boldsymbol{\xi} \geq \mathbf{0},\ i = 1, \dots, m$	$\boldsymbol{\xi} \geq \mathbf{0},\ i = 1, \dots, m$

One-class SVM interpretation
No bias

Let us reformulate the inner-product occurring in the constraints

$$
\begin{aligned}
&\left\langle \psi(\mathbf{y}_i), \mathbf{W}\phi(\mathbf{x}_i)\right\rangle_{\mathcal{H}_\psi} \\
&= \mathbf{tr}\left(\psi(\mathbf{y}_i)^T \mathbf{W}\phi(\mathbf{x}_i)\right) \\
&= \mathbf{tr}\left(\mathbf{W}\phi(\mathbf{x}_i)\psi(\mathbf{y}_i)^T\right) \\
&= \left\langle \mathbf{W}, \left[\psi(\mathbf{y}_i) \otimes \phi(\mathbf{x}_i)\right]\right\rangle_{\mathcal{H}_\psi \otimes \mathcal{H}_\phi}
\end{aligned}
$$



thus, we have a one-class SVM problem living in the tensor product space of the output and the input.
($\otimes$ denotes the tensor product)

One-class SVM interpretation

One step further ...

One can extend the range of applications by using not only tensor product but more general relationship between the output and input, i.e.,

$$\left\langle \mathbf{W}, \mathbf{\Psi}(\mathbf{y}_i, \mathbf{x}_i) \right\rangle_{\mathcal{H}_W}, \ \mathbf{\Psi} : \mathcal{H}_\psi \times \mathcal{H}_\phi \rightarrow \mathcal{H}_W.$$

If $\mathbf{dim}(\mathcal{H}_W) > \mathbf{dim}(\mathcal{H}_\psi) + \mathbf{dim}(\mathcal{H}_\phi)$ then the support of the distribution of one-class sample items is restricted on a manifold in $\mathcal{H}_W$.

Dual problem

$$
\begin{array}{ll}
\min & \sum_{i,j=1}^{m} \alpha_i \alpha_j \overbrace{\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle}^{\kappa_{ij}^{\phi}} \overbrace{\langle \psi(\mathbf{y}_i), \psi(\mathbf{y}_j) \rangle}^{\kappa_{ij}^{\psi}} - \sum_{i=1}^{m} \alpha_i, \\
\text{w.r.t.} & \alpha_i \in \mathbb{R}, \\
\text{s.t.} & \sum_{i=1}^{m} (\psi(\mathbf{y}_i))_t \alpha_i = 0, \; t = 1, \dots, \dim(\mathcal{H}_\psi), \\
& 0 \le \alpha_i \le C, \; i = 1, \dots, m.
\end{array}
$$

- κ_{ij}^{ϕ} input kernel,
- κ_{ij}^{ψ} output kernel
- The objective function is a symmetric function of the input and the output.

To get rid of occurrences of explicit labels ...

The explicit occurrences of the label vectors can be transformed into implicit ones:

$$
\begin{gathered}
\sum_{i=1}^{m} (\psi(\mathbf{y}_i))_t \alpha_i = 0, \; t = 1, \dots, \dim(\mathcal{H}_\psi), \\
\Updownarrow \\
\sum_{i=1}^{m} \kappa_{ij}^{\psi} \alpha_i = 0, \; j = 1, \dots, m
\end{gathered}
$$

This transformation preserves the feasibility domain!

Solution
Quadratic Augmented Lagrangian Form

$$
\begin{array}{ll}
\text{min} & \frac{1}{2}\alpha^T\left[K_{\psi(y)} \bullet K_{\phi(x)}\right]\alpha - \mathbf{1}^T\alpha \\
 & \boxed{+\lambda^T K_{\psi(y)}\alpha + \frac{C_{ALP}}{2}\alpha^T K_{\psi(y)}^T K_{\psi(y)}\alpha} \qquad \Leftarrow \text{biased case} \\
\text{w.r.t.} & \alpha \in \mathbb{R}^m, \ \text{primal variables}, \\
 & \lambda \in \mathbb{R}^m, \ \text{Lagrangian variables}, \\
\text{s.t.} & \boxed{\mathbf{0} \le \alpha \le C}, \Leftarrow \ \text{Simple box constraint}
\end{array}
$$

C_{ALP} Augmented Lagrangian Penalty Parameter

$\bullet$ component-wise(Schur) product

Solution schema

Outer loop

- Fix the Lagrangian variables,
  Inner loop
  - Solve the problem above the box constraint,
- Update the Lagrangian,
- Increase the penalty constant

If there is no bias only the inner loop has to be processed!!!

Prediction
No bias

The linear operator:

$$\mathbf{W} = \sum_{i=1}^{m} \alpha_i \psi(\mathbf{y}_i)\phi(\mathbf{x}_i)^T$$

Prediction in the label space:

$$\begin{aligned}\psi(\mathbf{y}) &= \mathbf{W}\phi(\mathbf{x}) \\ &= \sum_{i=1}^{m} \alpha_i \psi(\mathbf{y}_i) \boxed{\underbrace{\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle}_{\kappa^{\phi}(\mathbf{x}_i, \mathbf{x})}}\end{aligned}$$

Prediction when the labels are implicit
An approach

Assume the set of outcomes is known

$$\begin{aligned} \mathbf{y} &\in \widetilde{\mathcal{Y}} \quad \Leftarrow \text{Set of the possible outputs} \\ \mathbf{y}^* &= \arg\max_{\mathbf{y}\in\widetilde{\mathcal{Y}}} \psi(\mathbf{y})^T \mathbf{W}\phi(\mathbf{x}) \\ &= \arg\max_{\mathbf{y}\in\widetilde{\mathcal{Y}}} \sum_{i=1}^m \alpha_i \overbrace{\langle\psi(\mathbf{y}),\psi(\mathbf{y}_i)\rangle}^{\kappa^\psi(\mathbf{y},\mathbf{y}_i)} \overbrace{\langle\phi(\mathbf{x}_i)^T\phi(\mathbf{x})\rangle}^{\kappa^\phi(\mathbf{x}_i,\mathbf{x})} \end{aligned}$$

Finite outcome

$$\mathbf{y} \in \widetilde{\mathcal{Y}} = \{\mathbf{y}_1, \ldots, \mathbf{y}_K\},\ K \ll \infty$$

The best candidate for $\widetilde{\mathcal{Y}}$ could be the training set!

Prediction when the labels are explicit

Regression type prediction

The task is

$$\mathbf{y} \sim \mathbf{W}\phi(\mathbf{x})$$

Because we implicitly maximize the inner-product instead of minimizing the distance we need to scale the predictor

$$\mathbf{y} \cong \lambda \mathbf{W}\phi(\mathbf{x})$$

A simple, least square estimation of λ based on the training items equals to

$$\lambda = \frac{\mathbf{1}^T(\mathbf{K}_y \bullet \mathbf{K}_\phi)\boldsymbol{\alpha}}{\boldsymbol{\alpha}^T(\mathbf{K}_y \bullet \mathbf{K}_\phi)\boldsymbol{\alpha}},$$

where the denominator is the dual objective value + the sum of the dual variables.

Normalization

- **Preprocessing**

$$
\begin{array}{lcl}
\psi(\mathbf{y}_i) & \Rightarrow & \psi(\mathbf{y}_i)/\|\psi(\mathbf{y}_i)\|, \\
\phi(\mathbf{x}_i) & \Rightarrow & \phi(\mathbf{x}_i)/\|\phi(\mathbf{x}_i)\|,
\end{array}
$$

  - It can happen within the optimization. (no additional cost!)

- **Kernels with implicit normalization**, e.g. Gaussian,

$$
\langle \mathbf{u}, \mathbf{v} \rangle = \exp\big(-d(\mathbf{u}, \mathbf{v})\big), \ d() \geq 0.
$$

- **Spherical embedding**

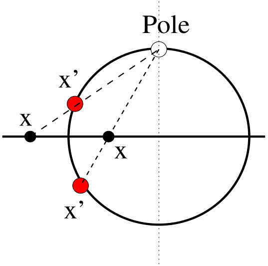
$$
\left.
\begin{array}{l}
\psi : \mathcal{Y} \rightarrow \mathcal{S}_y \subset \mathcal{H}_\psi, \ \mathcal{S}_y : \\
\phi : \mathcal{X} \rightarrow \mathcal{S}_x \subset \mathcal{H}_\phi, \ \mathcal{S}_x :
\end{array}
\right\} \text{Hyper-spheres}
$$

Spherical embedding

- **Spherical embedding**

$$\left.\begin{array}{l}\psi : \mathcal{Y} \rightarrow \mathcal{S}_y \subset \mathcal{H}_\psi,\ \mathcal{S}_y : \\ \phi : \mathcal{X} \rightarrow \mathcal{S}_x \subset \mathcal{H}_\phi,\ \mathcal{S}_x : \end{array}\right\} \text{Hyper-spheres}$$

- **Stereographic projection**

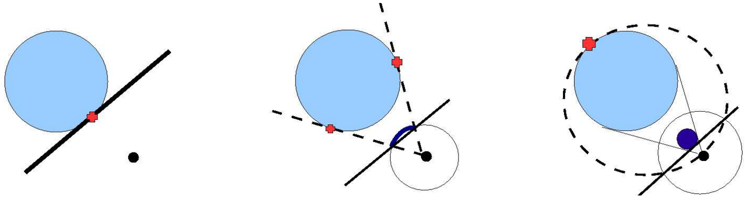


$$\begin{array}{ll}\Phi & : \phi(x) \rightarrow \phi'(x), \\ K'_{ij} & = \langle \phi'(x)_i, \phi'(x)_j \rangle \\ & = R^2 \left(1 - \frac{2R^2 \|\phi(x_i) - \phi(x_j)\|^2}{(\|\phi(x_i)\|^2 + R^2)(\|\phi(x_j)\|^2 + R^2)}\right) \\ & = R^2 \left(1 - \frac{2R^2 (K_{ii} + K_{jj} - 2K_{ij})}{(K_{ii} + R^2)(K_{jj} + R^2)}\right), \\ R & \text{Ball radius.}\end{array}$$

Effect of the normalization

- **Effect of L2 normalization**
  Wandering support vectors

$$\mathbf{x} \rightarrow \mathbf{x} \qquad \mathbf{x} \rightarrow \frac{\mathbf{x}}{\|\mathbf{x}\|_2} \qquad \mathbf{x} \rightarrow \frac{\mathbf{x}}{\|\mathbf{x}\|_2^2}$$



identity — projection onto ball — inversion

Multiview learning
Additive case

We have $\{\psi(\mathbf{y})_i, (\phi^1(\mathbf{x}_i^1), \phi^2(\mathbf{x}_i^2), \ldots)\}$ several sources of inputs taken out of distinct distributions.

$$
\begin{array}{ll}
\min & \frac{1}{2}\boxed{\sum_{k=1}^{n_K} \mathbf{tr}(\mathbf{W}_k^T \mathbf{W}_k)} + C\mathbf{1}^T\boldsymbol{\xi} \\
\text{w.r.t.} & \boxed{\mathbf{W}_k : \mathcal{H}_{\phi^k} \rightarrow \mathcal{H}_{\psi}}, \ \text{linear op.} \\
 & \mathbf{b} \in \mathcal{H}_{\psi}, \ \text{translation(bias)} \\
 & \boldsymbol{\xi} \in \mathbb{R}^m, \ \text{error vector} \\
\text{s.t.} & \boxed{\langle \psi(\mathbf{y}_i), \sum_{k=1}^{n_k} \mathbf{W}_k \phi^k(\mathbf{x}_i^k) + \mathbf{b} \rangle_{\mathcal{H}_{\psi}}} \geq 1 - \xi_i \\
 & \boldsymbol{\xi} \geq \mathbf{0}, \ i = 1, \ldots, m
\end{array}
$$

Kernel: $\mathbf{K}_y \bullet \sum_{k=1}^{n_k} \mathbf{K}_{x^k}$,

$\bullet$ element-wise product

Multiview learning

Product case

$$\min \quad \frac{1}{2}\boxed{\mathbf{tr}(\mathbf{W}^T\mathbf{W})} + C\mathbf{1}^T\boldsymbol{\xi}$$

$$\text{w.r.t.} \quad \boxed{\mathbf{W} : \mathcal{H}_\phi^1 \otimes \mathcal{H}_\phi^2 \rightarrow \mathcal{H}_\psi}, \ \text{linear op.}$$

$$\boxed{\mathbf{b} \in \mathcal{H}_\psi}, \ \text{translation(bias)}$$

$$\boldsymbol{\xi} \in \mathbb{R}^m, \ \text{error vector}$$

$$\text{s.t.} \quad \boxed{\left\langle \psi(\mathbf{y}_i), \mathbf{W}\left(\phi^1(\mathbf{x}_i^1) \otimes \phi^2(\mathbf{x}_i^2)\right) + \mathbf{b} \right\rangle_{\mathcal{H}_\psi}} \geq 1 - \xi_i$$

$$\boldsymbol{\xi} \geq \mathbf{0}, \ i = 1, \ldots, m,$$

Kernel: $\mathbf{K}_y \bullet \mathbf{K}_{x^1} \bullet \mathbf{K}_{x^2}$,

$\bullet$ element-wise product

Representation of multiclass output

- **Indicators**, e.g.: 3 classes $\Rightarrow \ \{(1,0,0),(0,1,0),(0,0,1)\}$,
- **Vectors pointing into the class centers**,
  Class centers can be means or medians,
- **Vertices of hyper-tetrahedron** Vectors with unit length and with minimum pair-wise correlation.

The experiments favour the hyper-tetrahedron, it is the most "symmetric" structure.

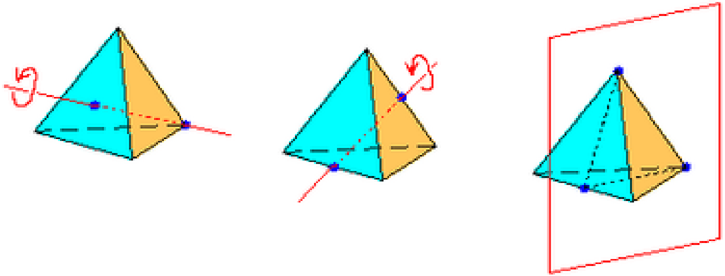
Vertices of hyper-tetrahedron

n-class case:

Consider the matrix **V** with elements:

$$V_{ij} = \begin{cases} 1 & \text{if } i = j, \\ -\frac{1}{n-1} & \text{otherwise.} \end{cases}$$

The labels are rows of the matrix **A** which satisfies $\mathbf{V} = \mathbf{A}\mathbf{A}^T$.
One eigenvalue of **V** is zero, thus **A** has n rows but $n-1$ columns only.



* www.wikipedia.org/wiki/tetrahedron

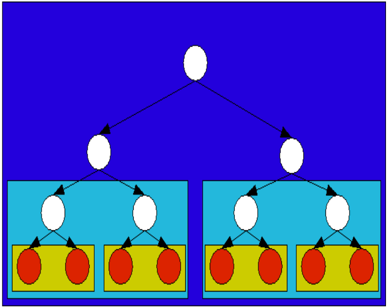
Experiments

	Test error rate (%)							
	SVM		MMR					
	all	one	hyper-tetrahedron			indicator		
	vs. all		Normalized on					
Name			—	item	variable	—	item	variable
abalone *	**72.3**	79.7	73.0	73.0	73.4	73.9	73.0	74.1
glass	30.4	30.8	27.3	27.6	29.2	**26.4**	29.0	29.0
optdigits *	3.8	2.7	2.0	**1.6**	3.3	2.1	1.9	3.3
page-blocks	3.4	3.4	4.4	3.4	3.7	4.5	3.6	**3.3**
satimage *	8.2	**7.8**	8.2	17.5	8.6	8.7	17.7	9.1
spectrometer	42.8	53.7	99.5	**37.5**	53.9	99.6	38.4	53.3
yeast	41.0	**40.3**	41.6	40.6	**40.3**	42.6	41.6	40.9

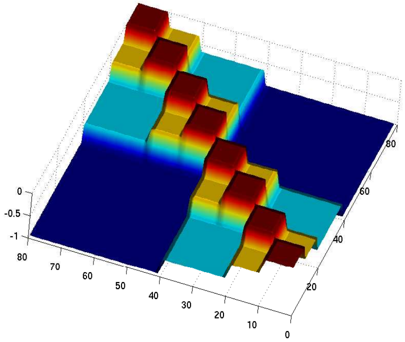
Table: Test error rates (%). If the data set has dedicated training and test subsets, marked with *, then the table shows the accuracy computed on the given test subset otherwise the presented accuracies are averages computed via 5-fold cross-validation.

Embedding Hierarchy
Base idea

Tree



Kernel



Similarities in a hierarchy

- V set of nodes, I is an indexing
- $p(v) \subset V$ connects node v and the root = shortest path, $|p(v)|$ distance between v and the root

Representation of a node(path from the root)

$$\psi(v)_i = \begin{cases} r & \text{if } v_i \notin p(v), \\ sq^k & \text{if } v_i \in v(p) \text{ and } k = |p(v)| - |p(v_i)|, \end{cases}$$

r, s, q parameters control the shape of the label kernel

Performance measures
Correctness of the path

- $\ell_{0/1}$ Zero-one loss
- ℓ_{Δ} Symmetric difference loss
- P Precision
- R Recall
- F1 Combination of the Precision and Recall $\Rightarrow \frac{2PR}{P+R}$

Methods

SVM	Flat SVM
H-SVM	Node-wise SVM
H-RLS	Hierarchical least square (Cesa-Bianchi)
$\text{H-M}^3 - l_\Delta$	H-M^3 trained on ℓ_Δ (Rousu)
$\text{H-M}^3 - l_{\tilde{H}}$	H-M^3 trained on subtree loss (Rousu)
MMR_{lin}	Proposed method with linear input kernel
MMR_{poly}	Proposed method with polynomial(3) kernel

WIPO-alpha dataset

WIPO-alpha	$\ell_{0/1}$	ℓ_{Δ}	P	R	F1
SVM	87.2	1.84	**93.1**	58.2	71.6
H-SVM	76.2	1.74	90.3	63.3	74.4
H-RLS	72.1	1.69	88.5	66.4	75.9
H-M^3-l_{Δ}	70.9	**1.67**	90.3	65.3	75.8
H-M^3-$l_{\tilde{H}}$	65.0	1.73	84.1	70.6	76.7
MMR$_{lin}$	**47.1**	1.77	77.8	**77.8**	**77.8**

Table: Prediction losses $l_{0/1}$ and l_{Δ}, precision, recall and F1 values obtained using different learning algorithms. All figures are given as percentages. Precision and recall are computed in terms of totals of microlabel predictions in the test set.

REUTERS dataset

REUTERS	$l_{0/1}$	l_{Δ}	P	R	F1
SVM	32.9	0.61	**94.6**	58.4	72.2
H-SVM	29.8	0.57	92.3	63.4	75.1
H-RLS	28.1	**0.55**	91.5	65.4	**76.3**
H-M^3-l_{Δ}	27.1	0.58	91.0	64.1	75.2
H-M^3-$l_{\tilde{H}}$	27.9	0.59	85.4	**68.3**	75.9
MMR$_{lin}$	27.8	0.71	82.7	60.4	69.8
MMR$_{poly}$	**26.4**	0.70	85.2	59.1	69.8

Table: Prediction losses $l_{0/1}$ and l_{Δ}, precision, recall and F1 values obtained using different learning algorithms. All figures are given as percentages. Precision and recall are computed in terms of totals of microlabel predictions in the test set.

Computational times

	Reuters	WIPO-alpha
MMR$_{lin}$	2.8	1.9
MMR$_{poly}$	1.8	1.2

Table: The computational times of the optimizer in **seconds** (Intel Pentium 3.5 GHz; interpreted, pure Matlab code)

Future?

Theory and Applications

- Label kernel $\Leftrightarrow$ Generalization theory
- Geometric, algebraic extensions, e.g., representation in Banach space
- Learning convolution operator, e.g., adaptive controls, signal processing, machine vision
- Learning arbitrary graphs, set systems
- Minimax based General, or Generalized, Linear Model
- Minimax based Procrustes Analysis
- ...

This is the End

Thanks!