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

1. Learning strategy
2. Optimization model
3. Multiview learning
4. Embedding multiclass
5. Embedding Hierarchy
6. Future?
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

\[ \phi : \mathcal{X} \rightarrow \mathcal{H}_\phi \]

\[ \psi : \mathcal{Y} \rightarrow \mathcal{H}_\psi \]

Similarity transformation

\[ \tilde{W} = (W, b) \Rightarrow \psi(y) \sim \tilde{W}\phi(x) \]

Inversion

\[ \psi^{-1}(\mathcal{Y}) \]
The "Classical" Support Vector Machine (SVM)

\[
\begin{align*}
\text{min} & \quad \frac{1}{2} w^T w + C \mathbf{1}^T \xi \\
\text{w.r.t.} & \quad \mathbf{w} : \mathcal{H}_\phi \rightarrow \mathbb{R}, \text{ normal vec.} \\
& \quad b \in \mathbb{R}, \text{ bias} \\
& \quad \xi \in \mathbb{R}^m, \text{ error vector} \\
\text{s.t.} & \quad y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \\
& \quad \xi \geq 0, \ i = 1, \ldots, m
\end{align*}
\]
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 \( W = UDV^T \)

\[
\begin{align*}
U & = \text{Rotation} \\
D & = \begin{cases} 
\text{Scaling} \\
\text{Projection}
\end{cases} \\
V & = \text{Rotation} \\
+ \\
b & = \text{Translation}
\end{align*}
\]
<table>
<thead>
<tr>
<th>Primal problem</th>
<th>Vector label learning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Binary class learning</strong></td>
<td><strong>Vector label learning</strong></td>
</tr>
<tr>
<td>Support Vector Machine (SVM)</td>
<td>Maximum Margin Robot (MMR)</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{min} & \quad \frac{1}{2} \left( w^T w \right) + C \mathbf{1}^T \xi \\
\text{w.r.t.} & \quad w : \mathcal{H}_\phi \rightarrow \mathbb{R}, \text{ normal vec.} \\
& \quad b \in \mathbb{R}, \text{ bias} \\
& \quad \xi \in \mathbb{R}^m, \text{ error vector} \\
\text{s.t.} & \quad y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \\
& \quad \xi \geq 0, \quad i = 1, \ldots, m
\end{align*}
\]

\[
\begin{align*}
\text{min} & \quad \frac{1}{2} \text{tr}(W^T W) + C \mathbf{1}^T \xi \\
\text{w.r.t.} & \quad W : \mathcal{H}_\phi \rightarrow \mathcal{H}_\psi, \text{ linear operator} \\
& \quad b \in \mathcal{H}_\psi, \text{ translation(bias)} \\
& \quad \xi \in \mathbb{R}^m, \text{ error vector} \\
\text{s.t.} & \quad \langle \psi(y_i), W \phi(x_i) + b \rangle_{\mathcal{H}_\psi} \geq 1 - \xi_i \\
& \quad \xi \geq 0, \quad i = 1, \ldots, m
\end{align*}
\]

Szedmak (UoS, UoH)  
SVM with vector output  
Helsinki 2006
Let us reformulate the inner-product occurring in the constraints

\[
\langle \psi(y_i), W\phi(x_i) \rangle_{\mathcal{H}_\psi} = \text{tr}(\psi(y_i)^T W\phi(x_i)) \\
= \text{tr}(W\phi(x_i)\psi(y_i)^T) \\
= \langle W, [\psi(y_i) \otimes \phi(x_i)] \rangle_{\mathcal{H}_\psi \otimes \mathcal{H}_\phi}
\]

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.,

\[
\langle W, \psi(y_i, x_i) \rangle_{\mathcal{H}_W}, \quad \psi : \mathcal{H}_\psi \times \mathcal{H}_\phi \to \mathcal{H}_W.
\]

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

\[
\begin{align*}
\min & \quad \sum_{i,j=1}^{m} \alpha_i \alpha_j \langle \phi(x_i), \phi(x_j) \rangle \langle \psi(y_i), \psi(y_j) \rangle - \sum_{i=1}^{m} \alpha_i, \\
\text{w.r.t.} & \quad \alpha_i \in \mathbb{R}, \\
\text{s.t.} & \quad \sum_{i=1}^{m} (\psi(y_i))_t \alpha_i = 0, \ t = 1, \ldots, \dim(\mathcal{H}_\psi), \\
& \quad 0 \leq \alpha_i \leq C, \ i = 1, \ldots, m.
\end{align*}
\]

- \( \kappa_{ij}^\phi \) input kernel,
- \( \kappa_{ij}^\psi \) 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:

\[ \sum_{i=1}^{m} (\psi(y_i))_t \alpha_i = 0, \quad t = 1, \ldots, \dim(\mathcal{H}_\psi), \]

\[ \sum_{i=1}^{m} \kappa_{ij}^\psi \alpha_i = 0, \quad j = 1, \ldots, m \]

This transformation preserves the feasibility domain!
Solution
Quadratic Augmented Lagrangian Form

\[
\begin{align*}
\min & \quad \frac{1}{2} \alpha^T \left[ K_\psi(y) \bullet K_\phi(x) \right] \alpha - 1^T \alpha \\
& \quad + \lambda^T K_\psi(y) \alpha + \frac{C_{ALP}}{2} \alpha^T K_\psi^T(y) K_\psi(y) \alpha \\
\text{w.r.t.} & \quad \alpha \in \mathbb{R}^m, \text{ primal variables,} \\
& \quad \lambda \in \mathbb{R}^m, \text{ Lagrangian variables,} \\
\text{s.t.} & \quad 0 \leq \alpha \leq C, \quad \leftarrow \text{Simple box constraint}
\end{align*}
\]

\( C_{ALP} \) 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!!!
The linear operator:

$$W = \sum_{i=1}^{m} \alpha_i \psi(y_i) \phi(x_i)^T$$

Prediction in the label space:

$$\psi(y) = W \phi(x)$$

$$= \sum_{i=1}^{m} \alpha_i \psi(y_i) \underbrace{\langle \phi(x_i), \phi(x) \rangle}_{\kappa_{\phi}(x_i, x)}$$
Prediction when the labels are implicit

An approach

Assume the set of outcomes is known

\[ y \in \tilde{Y} \iff \text{Set of the possible outputs} \]

\[ y^* = \arg \max_{y \in \tilde{Y}} \psi(y)^T W \phi(x) \]

\[ = \arg \max_{y \in \tilde{Y}} \sum_{i=1}^{m} \alpha_i \left( \kappa_\psi(y, y_i) \left\langle \psi(y), \psi(y_i) \right\rangle \kappa_\phi(x_i, x) \left\langle \phi(x_i), \phi(x) \right\rangle \right) \]

Finite outcome

\[ y \in \tilde{Y} = \{ y_1, \ldots, y_K \}, \quad K \ll \infty \]

The best candidate for \( \tilde{Y} \) could be the training set!
Prediction when the labels are explicit
Regression type prediction

The task is

$$y \sim W\phi(x)$$

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

$$y \equiv \lambda W\phi(x)$$

A simple, least square estimation of $\lambda$ based on the training items equals to

$$\lambda = \frac{1^T(K_y \cdot K_\phi)\alpha}{\alpha^T(K_y \cdot K_\phi)\alpha},$$

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

- **Preprocessing**

  \[ \psi(y_i) \Rightarrow \psi(y_i)/\|\psi(y_i)\|, \]
  \[ \phi(x_i) \Rightarrow \phi(x_i)/\|\phi(x_i)\|, \]

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

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

  \[ \langle u, v \rangle = \exp(-d(u, v)), \quad d() \geq 0. \]

- **Spherical embedding**

  \[ \psi : \mathcal{Y} \rightarrow S_y \subset H_\psi, \quad S_y : \]
  \[ \phi : \mathcal{X} \rightarrow S_x \subset H_\phi, \quad S_x : \]

  \[ \{ \text{Hyper-spheres} \} \]

Szedmak (UoS, UoH)  SVM with vector output  Helsinki 2006
Spherical embedding

\[ \psi : \mathcal{Y} \to S_y \subset \mathcal{H}_\psi, \ S_y : \]
\[ \phi : \mathcal{X} \to S_x \subset \mathcal{H}_\phi, \ S_x : \] \{ Hyper-spheres \}

Stereographic projection

\[ \Phi : \phi(x) \to \phi'(x), \]
\[ K_{ij}' = \langle \phi'(x)_i, \phi'(x)_j \rangle \]
\[ = R^2 \left( 1 - \frac{2R^2 \| \phi(x_i) - \phi(x_j) \|^2}{(\| \phi(x_i) \|^2 + R^2)(\| \phi(x_j) \|^2 + R^2)} \right) \]
\[ = R^2 \left( 1 - \frac{2R^2(K_{ii} + K_{jj} - 2K_{ij})}{(K_{ii} + R^2)(K_{jj} + R^2)} \right), \]

\( R \) Ball radius.
Effect of the normalization

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

\[
x \rightarrow x
\]

- **identity**

\[
x \rightarrow \frac{x}{\|x\|_2}
\]

- **projection onto ball**

\[
x \rightarrow \frac{x}{\|x\|_2^2}
\]

- **inversion**
Multiview learning
Additive case

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

\[
\begin{align*}
\min & \quad \frac{1}{2} \sum_{k=1}^{n_k} \text{tr}(W_k^T W_k) + C \mathbf{1}^T \xi \\
\text{w.r.t.} & \quad W_k : \mathcal{H}_{\phi_k} \rightarrow \mathcal{H}_\psi, \text{ linear op.} \\
\xi & \in \mathbb{R}^m, \text{ error vector} \\
\text{s.t.} & \quad \langle \psi(y)_i, \sum_{k=1}^{n_k} W_k \phi^k(x^k_i) + b \rangle_{\mathcal{H}_\psi} \geq 1 - \xi_i \\
\xi & \geq 0, \ i = 1, \ldots, m
\end{align*}
\]

Kernel:
\[ K_y \cdot \sum_{k=1}^{n_k} K_{x^k}, \]
- element-wise product

---

Szedmak (UoS, UoH)  SVM with vector output  Helsinki 2006  22 / 35
Multiview learning
Product case

\[ \min \quad \frac{1}{2} \text{tr}(W^T W) + C \xi^T \xi \]

w.r.t. \( W : \mathcal{H}_\phi^1 \otimes \mathcal{H}_\phi^2 \rightarrow \mathcal{H}_\psi \), linear op.
\( b \in \mathcal{H}_\psi \), translation(bias)
\( \xi \in \mathbb{R}^m \), error vector

s.t. \( \langle \psi(y_i), W(\phi^1(x_{1i}) \otimes \phi^2(x_{2i})) + b \rangle_{\mathcal{H}_\psi} \geq 1 - \xi_i \)
\( \xi \geq 0, \ i = 1, \ldots, m, \)

Kernel: \( K_y \circ K_{x_1} \circ K_{x_2} \),
\( \circ \) element-wise product

Szedmak (UoS, UoH)
SVM with vector output
Helsinki 2006 23 / 35
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 $\mathbf{V}$ 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 $\mathbf{A}$ which satisfies $\mathbf{V} = \mathbf{A}\mathbf{A}^T$.

One eigenvalue of $\mathbf{V}$ is zero, thus $\mathbf{A}$ has $n$ rows but $n - 1$ columns only.

<table>
<thead>
<tr>
<th>Name</th>
<th>SVM</th>
<th>MMR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>all one vs. all</td>
<td>hyper-tetrahedron normalized on item variable</td>
</tr>
<tr>
<td>abalone *</td>
<td>72.3 79.7</td>
<td>73.0 73.0</td>
</tr>
<tr>
<td>glass</td>
<td>30.4 30.8</td>
<td>27.3 27.6</td>
</tr>
<tr>
<td>optdigits *</td>
<td>3.8 2.7</td>
<td>2.0 1.6</td>
</tr>
<tr>
<td>page-blocks</td>
<td>3.4 3.4</td>
<td>4.4 3.4</td>
</tr>
<tr>
<td>satimage *</td>
<td>8.2 7.8</td>
<td>8.2 17.5</td>
</tr>
<tr>
<td>spectrometer</td>
<td>42.8 53.7</td>
<td>99.5 37.5</td>
</tr>
<tr>
<td>yeast</td>
<td>41.0 40.3</td>
<td>41.6 40.6</td>
</tr>
</tbody>
</table>

**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

Szedmak (UoS, UoH)

SVM with vector output

Helsinki 2006
Similarities in a hierarchy

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

Representation of a node (path from the root)

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

if $v_i \in v(p)$ and $k = |p(v)| - |p(v_i)|$,

$r, s, q$ parameters control the shape of the label kernel
Performance measures

Correctness of the path

\( \ell_{0/1} \) Zero-one loss
\( \ell_\Delta \) Symmetric difference loss
\( P \) Precision
\( R \) Recall
\( F_1 \) Combination of the Precision and Recall

\[ F_1 \Rightarrow \frac{2PR}{P+R} \]
Methods

SVM  Flat SVM
H-SVM Node-wise SVM
H-RLS Hierarchical least square (Cesa-Bianchi)
H-M³ − \( \ell_\Delta \) H-M³ trained on \( \ell_\Delta \) (Rousu)
H-M³ − \( \ell_H \) H-M³ trained on subtree loss (Rousu)
MMR\text{lin} Proposed method with linear input kernel
MMR\text{poly} Proposed method with polynomial(3) kernel

Szedmak (UoS, UoH) SVM with vector output

Helsinki 2006
## WIPO-alpha dataset

<table>
<thead>
<tr>
<th></th>
<th>$\ell_{0/1}$</th>
<th>$\ell_\Delta$</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>87.2</td>
<td>1.84</td>
<td><strong>93.1</strong></td>
<td>58.2</td>
<td>71.6</td>
</tr>
<tr>
<td>H-SVM</td>
<td>76.2</td>
<td>1.74</td>
<td>90.3</td>
<td>63.3</td>
<td>74.4</td>
</tr>
<tr>
<td>H-RLS</td>
<td>72.1</td>
<td>1.69</td>
<td>88.5</td>
<td>66.4</td>
<td>75.9</td>
</tr>
<tr>
<td>H-M(^3)-/(\Delta)</td>
<td>70.9</td>
<td><strong>1.67</strong></td>
<td>90.3</td>
<td>65.3</td>
<td>75.8</td>
</tr>
<tr>
<td>H-M(^3)-/(\bar{H})</td>
<td>65.0</td>
<td>1.73</td>
<td>84.1</td>
<td>70.6</td>
<td>76.7</td>
</tr>
<tr>
<td>MMR(_{lin})</td>
<td><strong>47.1</strong></td>
<td>1.77</td>
<td>77.8</td>
<td><strong>77.8</strong></td>
<td><strong>77.8</strong></td>
</tr>
</tbody>
</table>

**Table:** Prediction losses $l_{0/1}$ and $l_\Delta$, 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.
### Table: Prediction losses $l_{0/1}$ and $l_\Delta$, 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.

<table>
<thead>
<tr>
<th></th>
<th>$l_{0/1}$</th>
<th>$l_\Delta$</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>32.9</td>
<td>0.61</td>
<td>94.6</td>
<td>58.4</td>
<td>72.2</td>
</tr>
<tr>
<td>H-SVM</td>
<td>29.8</td>
<td>0.57</td>
<td>92.3</td>
<td>63.4</td>
<td>75.1</td>
</tr>
<tr>
<td>H-RLS</td>
<td>28.1</td>
<td><strong>0.55</strong></td>
<td>91.5</td>
<td>65.4</td>
<td><strong>76.3</strong></td>
</tr>
<tr>
<td>H-M$^3$-*$l_\Delta$</td>
<td>27.1</td>
<td>0.58</td>
<td>91.0</td>
<td>64.1</td>
<td>75.2</td>
</tr>
<tr>
<td>H-M$^3$-*$l_H$</td>
<td>27.9</td>
<td>0.59</td>
<td>85.4</td>
<td><strong>68.3</strong></td>
<td>75.9</td>
</tr>
<tr>
<td>MMR$_{lin}$</td>
<td>27.8</td>
<td>0.71</td>
<td>82.7</td>
<td>60.4</td>
<td>69.8</td>
</tr>
<tr>
<td>MMR$_{poly}$</td>
<td><strong>26.4</strong></td>
<td>0.70</td>
<td>85.2</td>
<td>59.1</td>
<td>69.8</td>
</tr>
</tbody>
</table>
Computational times

<table>
<thead>
<tr>
<th></th>
<th>Reuters</th>
<th>WIPO-alpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{MMR}_{\text{lin}}$</td>
<td>2.8</td>
<td>1.9</td>
</tr>
<tr>
<td>$\text{MMR}_{\text{poly}}$</td>
<td>1.8</td>
<td>1.2</td>
</tr>
</tbody>
</table>

**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!