How to teach the Support Vector Machine to learn arbitrary outputs

Sandor Szedmak

ISIS, Electronics and Computer Science
University of Southampton

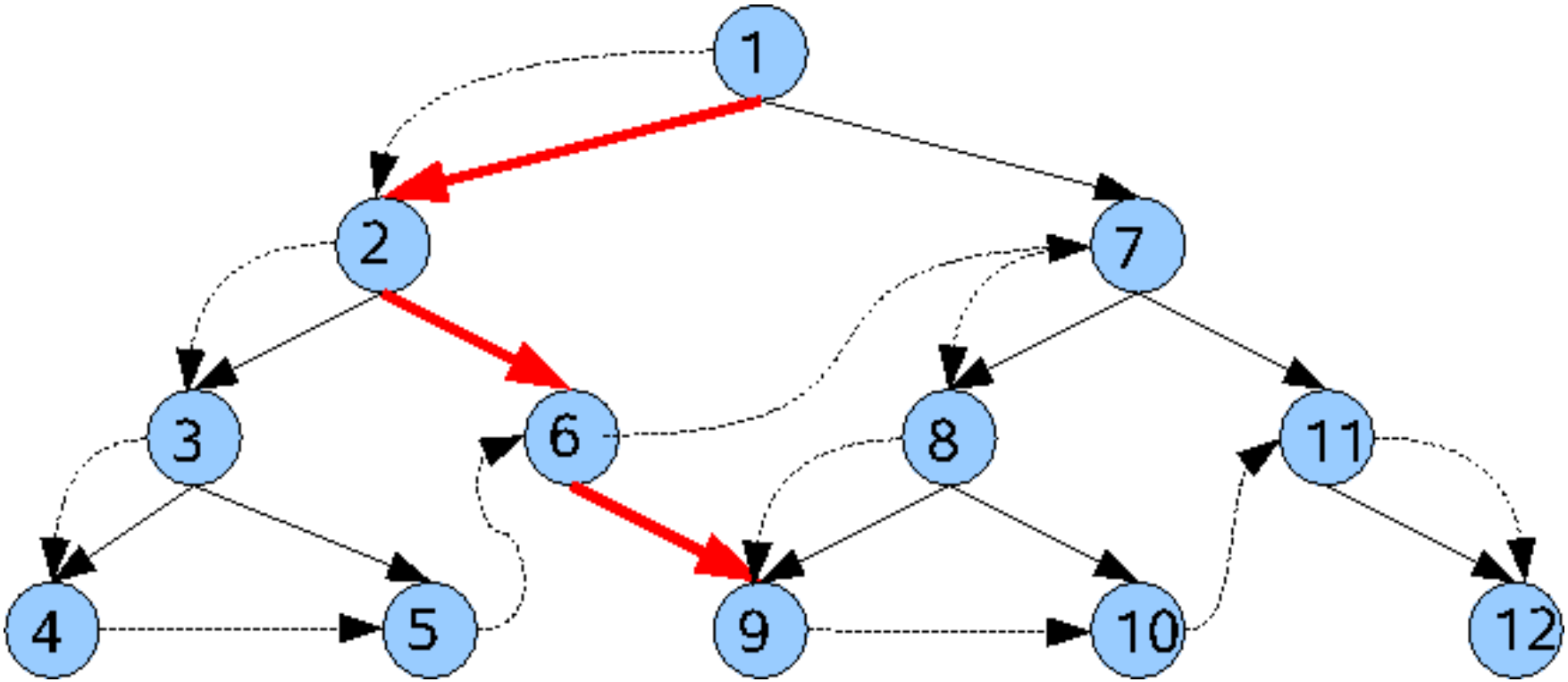
Southampton 04/2008

Contributors

- Katja Astikainen
- Juho Rousu
  - University of Helsinki
- Yizhao Ni
- Craig J. Saunders
  - University of Southampton
- John Shawe-Taylor
- Zhuoran Wang
  - University College London
- Tijl de Bie
  - University of Bristol
- others ...

How one can predict something like this ...

What about SVM?



Prologue

"Young man,

in mathematics you don't understand things. You just get used to them."

John von Neumann, one of the greatest mathematician of the Twenty Century.

Outline

Motivation II

Our tasks are:

- Extend the capability of the Support Vector Machine and the Boosting towards non-binary complex structural output objects.
- Find a structural learning model to be independent from the dimensionality of the predicted output items.

Examples of known approaches:

- Cut the problem into several parts,
  - Apply plenty of binary classifiers ...
- Max-Margin Markov Networks,
  - Taskar(2003)
- Least-square approaches,
  - Cortes(2005)

Main 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

$$\phi: \overbrace{\text{input space}}^{\mathcal{X}} \rightarrow \overbrace{\text{feature space}}^{\mathcal{H}_\phi}$$

$$\psi: \overbrace{\text{output space}}^{\mathcal{Y}} \rightarrow \overbrace{\text{label space}}^{\mathcal{H}_\psi}$$

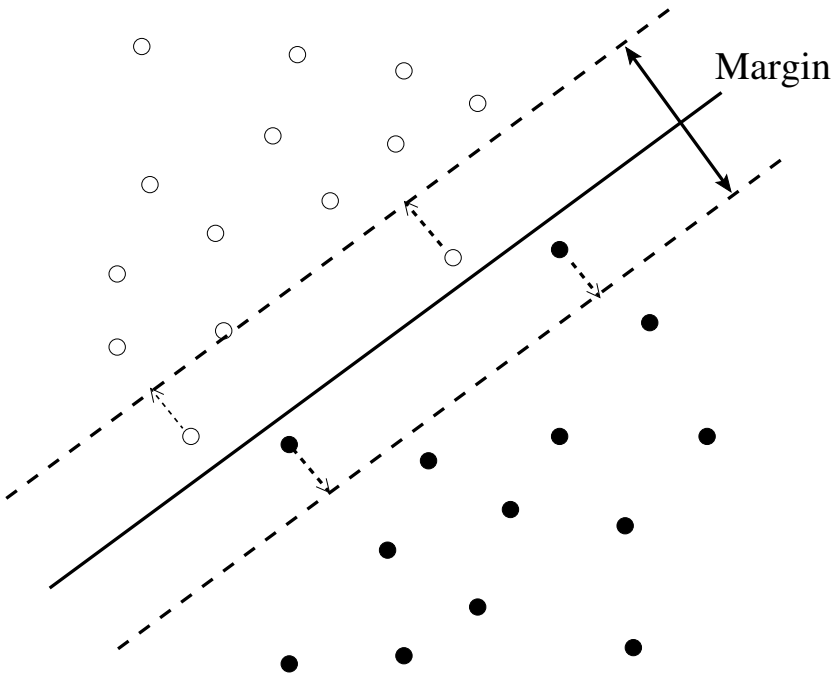
Similarity transformation

$$\widetilde{\mathbf{W}} = (\mathbf{W}, \mathbf{b}) \Rightarrow \psi(\mathbf{y}) \sim \widetilde{\mathbf{W}}\phi(\mathbf{x})$$

Inversion

$$\psi^{-1}(\mathbf{Y})$$

The “Classical” Support Vector Machine(SVM)



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

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

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

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

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

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

Dual problem

$$\begin{aligned}
\text{min} \quad & \sum_{i,j=1}^{m} \alpha_i \alpha_j \underbrace{\boxed{\overbrace{y_i y_j}^{\kappa_{ij}^{Y}} \overbrace{\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle}^{\kappa_{ij}^{\phi}}}}_{K_{YX}} - \sum_{i=1}^{m} \alpha_i, \\
\text{w.r.t.} \quad & \alpha_i \in \mathbb{R}, \\
\text{s.t.} \quad & \sum_{i=1}^{m} y_i \alpha_i = 0, \\
& 0 \leq \alpha_i \leq C, \ i = 1, \ldots, m.
\end{aligned}$$

- κ_{ij}^{ϕ} input kernel,
- κ_{ij}^{Y} **output kernel!**
- $K_{YX} = K_Y \bullet K_X$ joined kernel by element-wise product
- The objective function is a symmetric function of the input and the output.

The extended primal

$$\begin{aligned}
\text{min} \quad & \frac{1}{2}\boxed{\|\mathbf{W}\|_2^2} + C\mathbf{1}^T\boldsymbol{\xi} \\
\text{w.r.t.} \quad & \boxed{\mathbf{W}} \\
& \boxed{b \in \mathbb{R}}, \text{ bias} \\
& \boldsymbol{\xi} \in \mathbb{R}^m, \text{ error vector} \\
\text{s.t.} \quad & \boxed{F(W; \Phi(\mathbf{x}_i, \mathbf{y}_i))} + b \geq 1 - \xi_i \\
& \boldsymbol{\xi} \geq \mathbf{0}, \ i = 1, \ldots, m
\end{aligned}$$

- $F(W; \Phi(\mathbf{x}_i, \mathbf{y}_i))$ linear function of $\mathbf{W}$, parametrized by a function of the input and the output. It has to be monotonic, increasing function of $\|\mathbf{W}\|_2$.

Reinterpretation of the normal vector **w**

Original

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

New

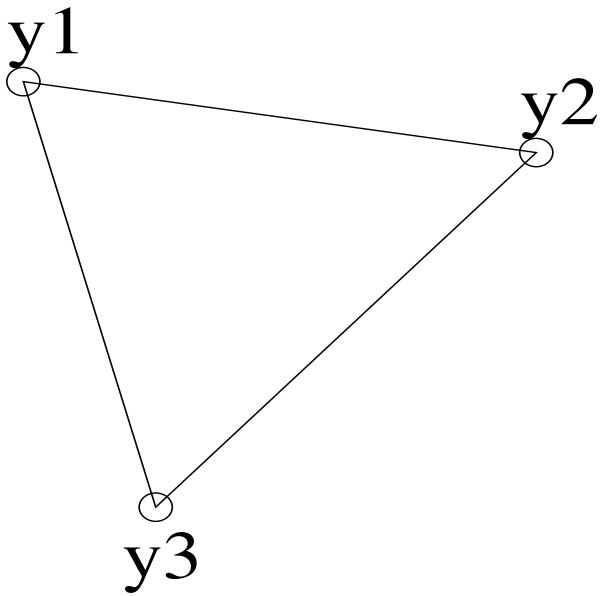
- $y_i \in \mathcal{Y}$ arbitrary outputs
  - $\psi(y_i) \in \mathcal{H}_\psi$ embedded labels in a linear vector space
- $\mathbf{w}^T$ is a linear operator projecting the input space into the output space.
  - The aim to find the highest similarity between the output and the projected input.

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

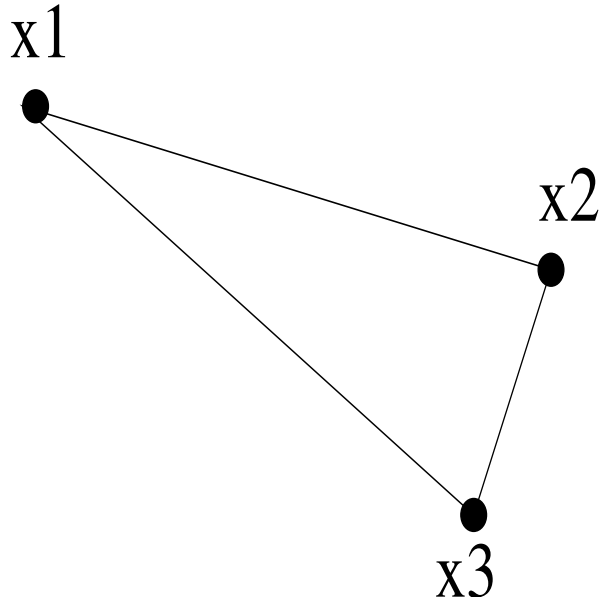
$\mathbf{D} = \begin{cases} \text{Scaling} \\ \text{Projection} \end{cases}$

$\mathbf{V} =$ Rotation

$+$

$\mathbf{b} =$ Translation

Input



Primal problems

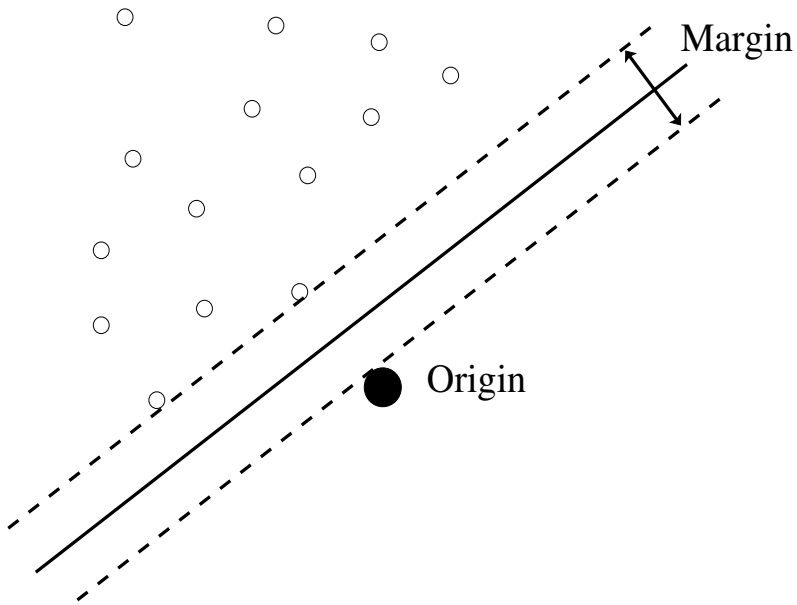
	Binary class learning	Vector label learning
	Support Vector Machine(SVM)	Maximum Margin Robot(MMR)
min	$\frac{1}{2}\underbrace{\boxed{\mathbf{w}^T\mathbf{w}}}_{\|\mathbf{w}\|_2^2} + C\mathbf{1}^T\xi$	$\frac{1}{2}\underbrace{\boxed{\mathbf{tr}(\mathbf{W}^T\mathbf{W})}}_{\|\mathbf{W}\|^2_{Frobenius}} + C\mathbf{1}^T\xi$
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	$\boxed{\mathbf{W} : \mathcal{H}_\phi \to \mathcal{H}_\psi}$, linear operator $\boxed{\mathbf{b} \in \mathcal{H}_\psi}$, translation(bias) $\xi \in \mathbb{R}^m$, error vector
s.t.	$\boxed{y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b)} \geq 1 - \xi_i$ $\xi \geq \mathbf{0},\ i = 1, \ldots, m$	$\boxed{\langle \psi(\mathbf{y}_i), \mathbf{W}\phi(\mathbf{x}_i) + \mathbf{b} \rangle_{\mathcal{H}_\psi}} \geq 1 - \xi_i$ $\xi \geq \mathbf{0},\ i = 1, \ldots, m$

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.,

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

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 $\mathcal{H}_W$.

Alternative linear functions of $\mathbf{W}$

They are subversions of the general case, but they can better express some kind of relationship between the input and output.

- $\mathbf{x_i}, \mathbf{y_i}$ matrices, Ψ covers the operation of matrix product. It allows to use sample items with different dimensionality.
- $\mathbf{x_i}, \mathbf{y_i}$ matrices of same size, Ψ covers the operation of point-wise product.
- $\mathbf{x_i}, \mathbf{y_i}$ are taken from an algebra with special properties, e.g. Clifford, Jordan, Ψ expresses the product operation of the algebra. They can represent complex structures.

Advantage of the tenzor product

- The identity

$$\langle \mathbf{x}_i \otimes \mathbf{y}_i, \mathbf{x}_j \otimes \mathbf{y}_j \rangle = \langle \mathbf{x}_i, \mathbf{x}_j \rangle \langle \mathbf{y}_i, \mathbf{y}_j \rangle$$

  allows us
  - to separate the input and output kernels,
  - to work with vectors which may have infinite number of components, they can be functions, e.g. probability densities, generalized functions - Dirac δ -, etc..

Dual problem

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

- 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[1]:

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

This transformation preserves the feasibility domain!

[1] Tijl De Bie, Private conversation

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

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

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

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

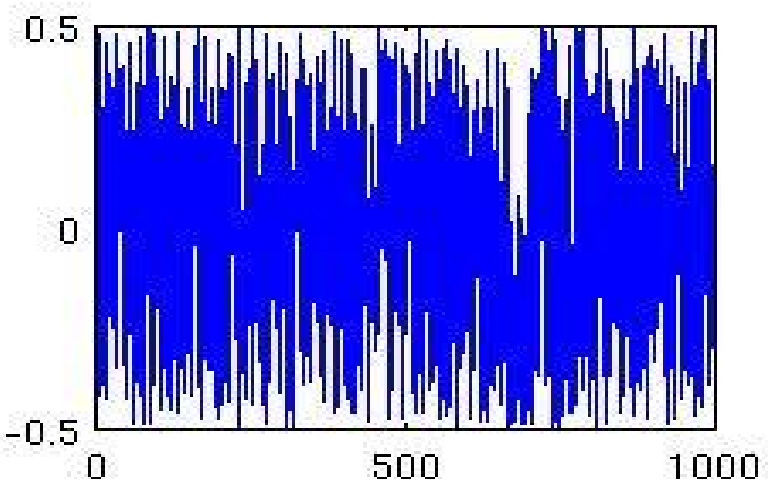
A simple, least square estimation of λ based on the training items equals to

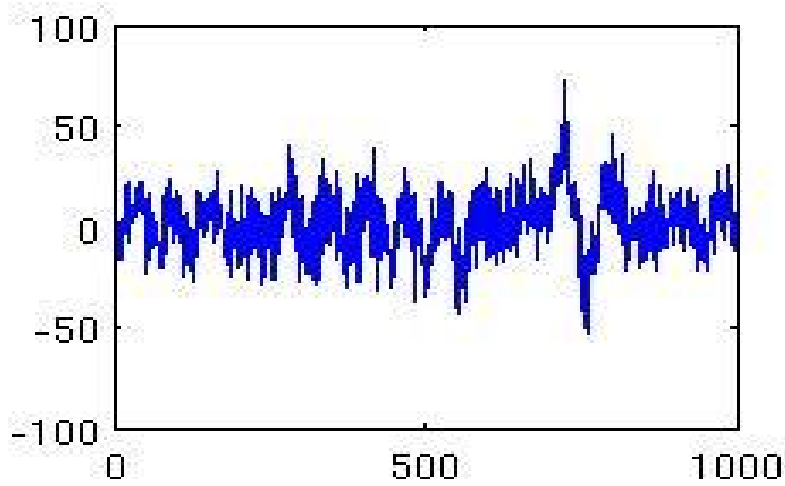
$$\lambda = \frac{\mathbf{1}^T(\mathbf{K}_y \bullet \mathbf{K}_\phi)\boldsymbol{\alpha}}{\boldsymbol{\alpha}^T(\mathbf{K}_y \bullet \mathbf{K}_\phi)\boldsymbol{\alpha}},$$

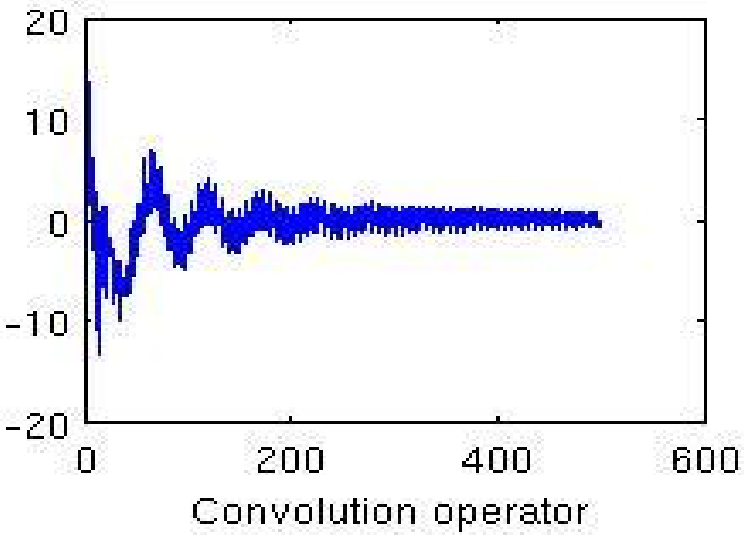
where the denominator is the dual objective value + the sum of the dual variables.

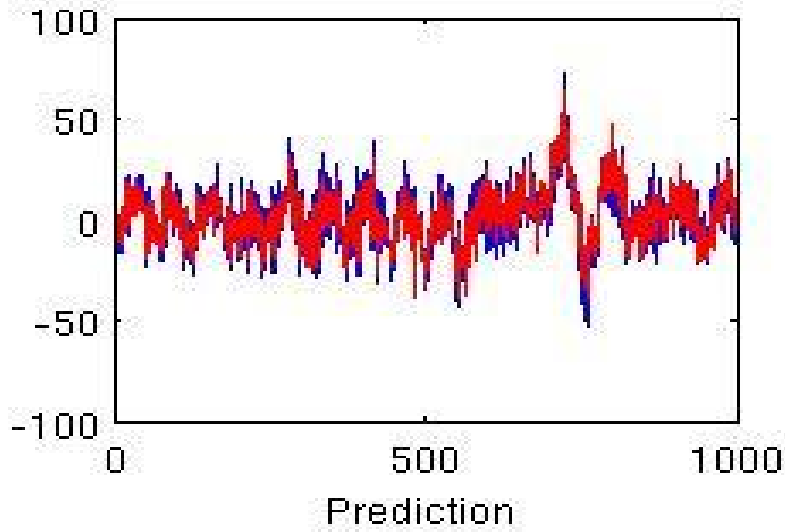
Learning convolution operator

input windows $\Rightarrow$ output windows









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.

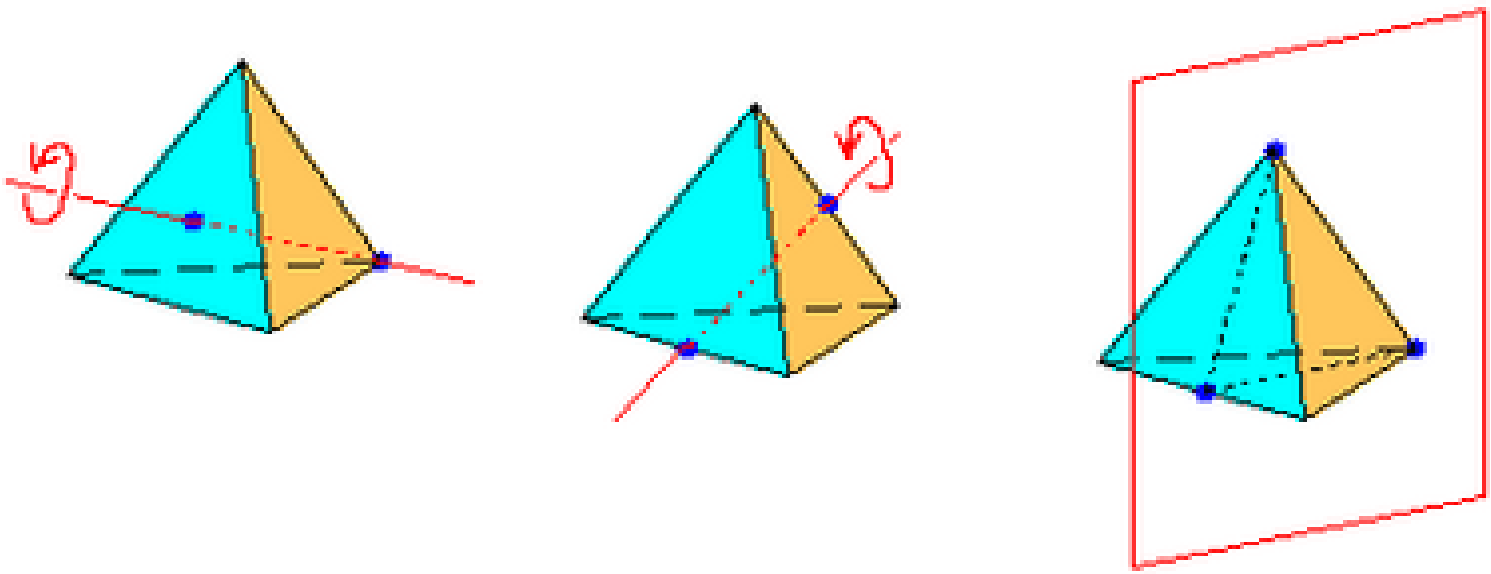
Vertices of hyper-tetrahedron

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

How to use it?

Cook Book

Multiclass classification

Skeleton of the procedure

- Centralize and normalize data and choose input kernel,
- Choose vector labels to the classes, centralize and normalize them as well,
- Solve the MMR problem,
- Find the best fitting class to the predicted label!

Multiclass classification

Centralize and normalize the input and choose input kernel

- Centralize the data!

$$\mathbf{x}_i = \mathbf{x}_i - \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i$$

- To normalize the feature vectors divide them by their Euclidean length(ℓ_2 norm).

$$\mathbf{x}_i = \mathbf{x}_i / \|\mathbf{x}\|_2$$

- In case of large number of classes(>10) the Gaussian kernel might be the best first choice.

Multiclass classification

Choose vector labels to the categories of multiclasses

- Assume the number of classes is 5.
- Labels are indicators to the classes $\in \{0,1\}^5$.
- Example:
Classes	Labels
1	$(1,0,0,0,0)$,
2	$(0,1,0,0,0)$,
3	$(0,0,1,0,0)$,
4	$(0,0,0,1,0)$,
5	$(0,0,0,0,1)$,

    for class k component k of the label vector is set to 1 others are 0.
- The inner products of the labels, *the elements of the output kernel*, equal to 1 if the sample items come from the same class and 0 otherwise.

Multiclass classification

Find the best fitting class to the predicted label

- MMR gives as raw prediction a real valued, not a zero-one, vector $\psi(y)$.
- Solve for the best candidate of classes

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

- In our example the candidates are

$$\begin{aligned}
\widetilde{\mathcal{Y}} = \{&(1,0,0,0,0), \\
&(0,1,0,0,0), \\
&(0,0,1,0,0), \\
&(0,0,0,1,0), \\
&(0,0,0,0,1)\}
\end{aligned}$$

Experiments

Multiclass classification

	Test error rate (%)							
	SVM		MMR					
	all vs. all	one vs. all	hyper-tetrahedron			indicator		
			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.

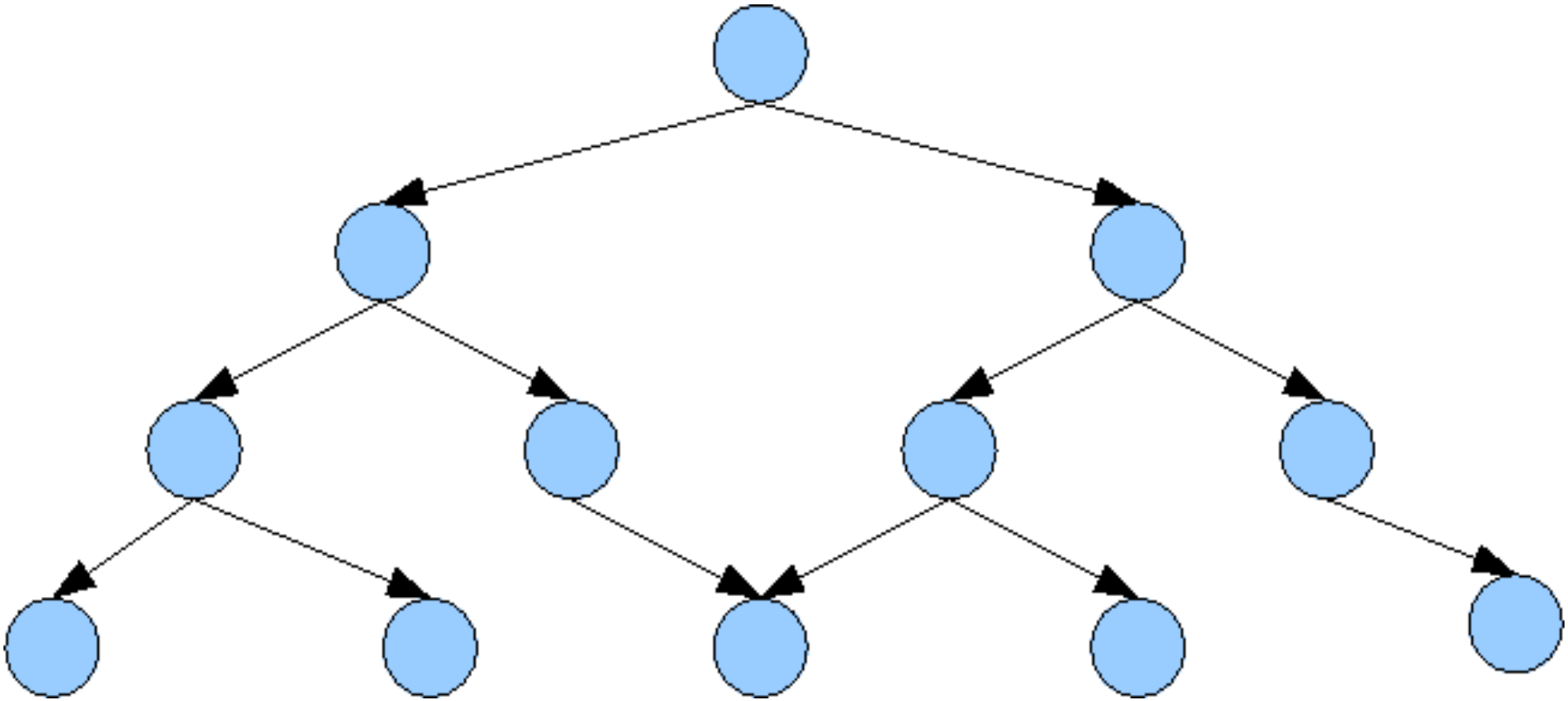
Rooted DAG kernel

Feature and(or) label vectors

- Take a directed, acyclic, rooted graph(rDAG).
- The nodes are indexed by topological order which means if there is a directed edge from node A to node B then $i_A < i_B$ holds for the labels .
- Feature vector $\phi(N)$ to a node N is taken out of $\{0,1\}^n$, where n equals to the number of nodes. The components of $\phi(N)$ corresponds to the nodes and indexed by their topological order.
- A component of the feature vector $\phi(N)$ is 1 if the corresponding node is on the shortest path from the root to N, otherwise it is set to 0.
- Use the centralization and normalization if necessary, e.g. the lengths of the shortest paths have high variance!

Rooted DAG kernel

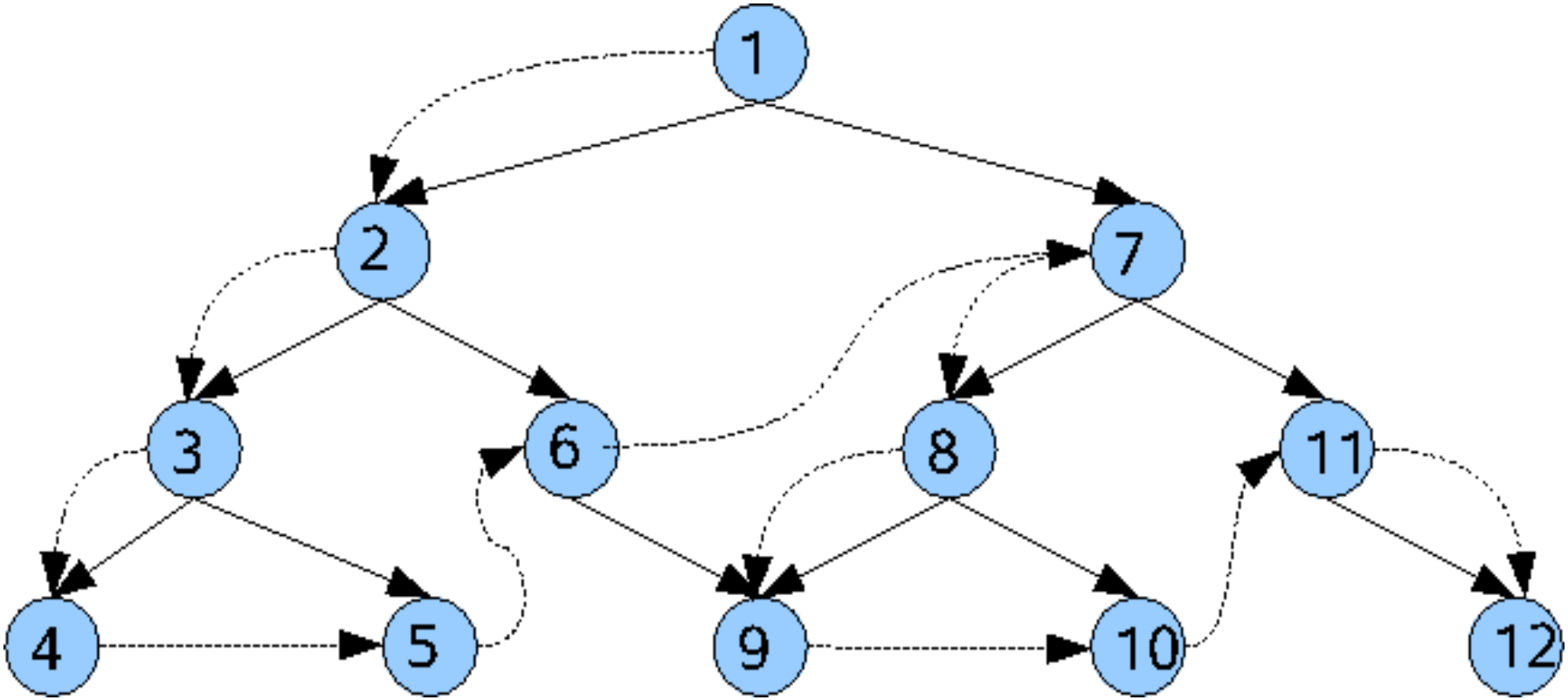
Base graph



Rooted DAG kernel

Numbering by topological order

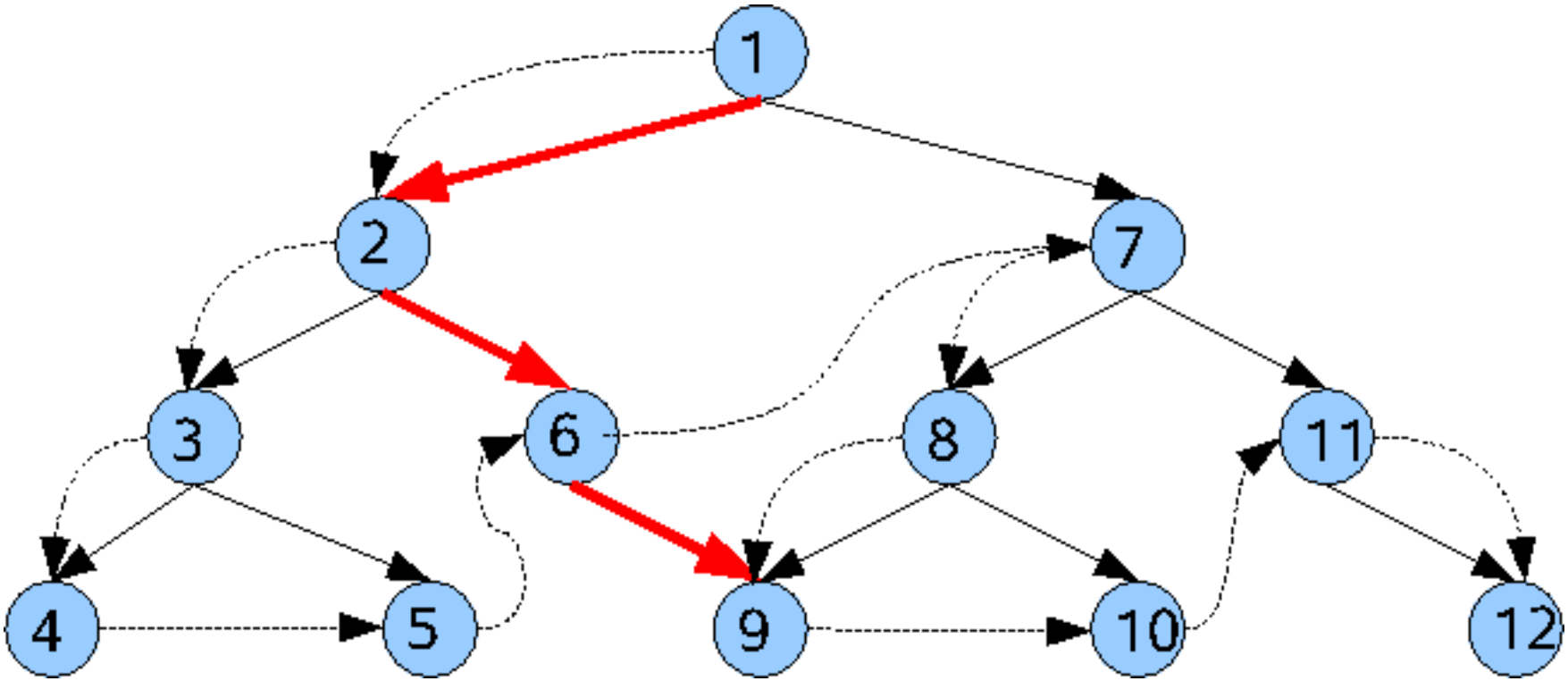
If there is edge from node a to node b then $i_a < i_b$ holds for the labels.



Rooted DAG kernel

Shortest path from root to nodes

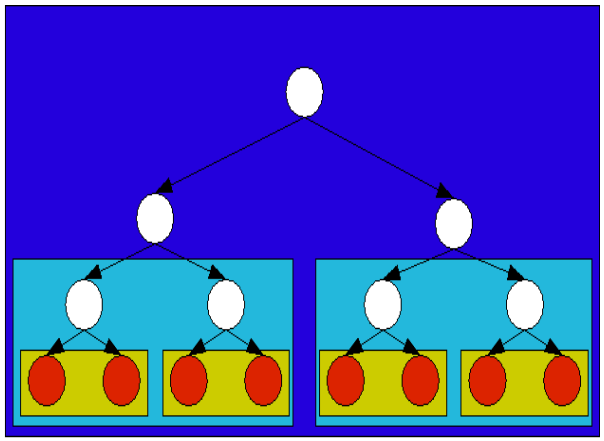
$$n = 12,\ \phi(N_9) = \begin{array}{|c|c|c|c|c|c|c|c|c|c|c|c|}\hline 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ \hline 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline \end{array}$$



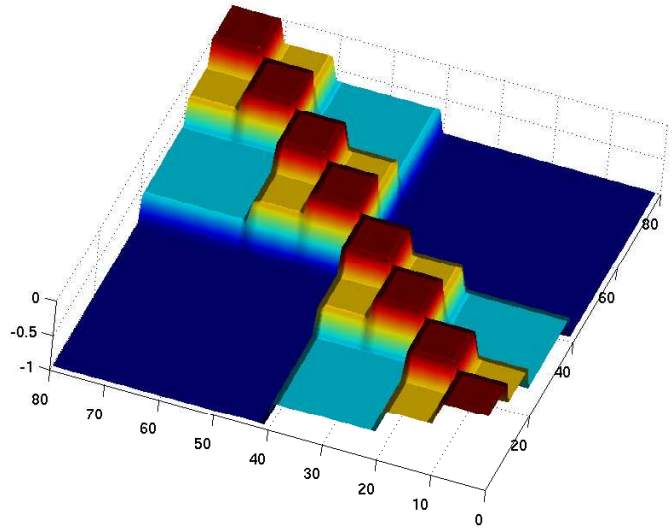
Embedding Hierarchy

Via similarity(dissimilarity)

Tree



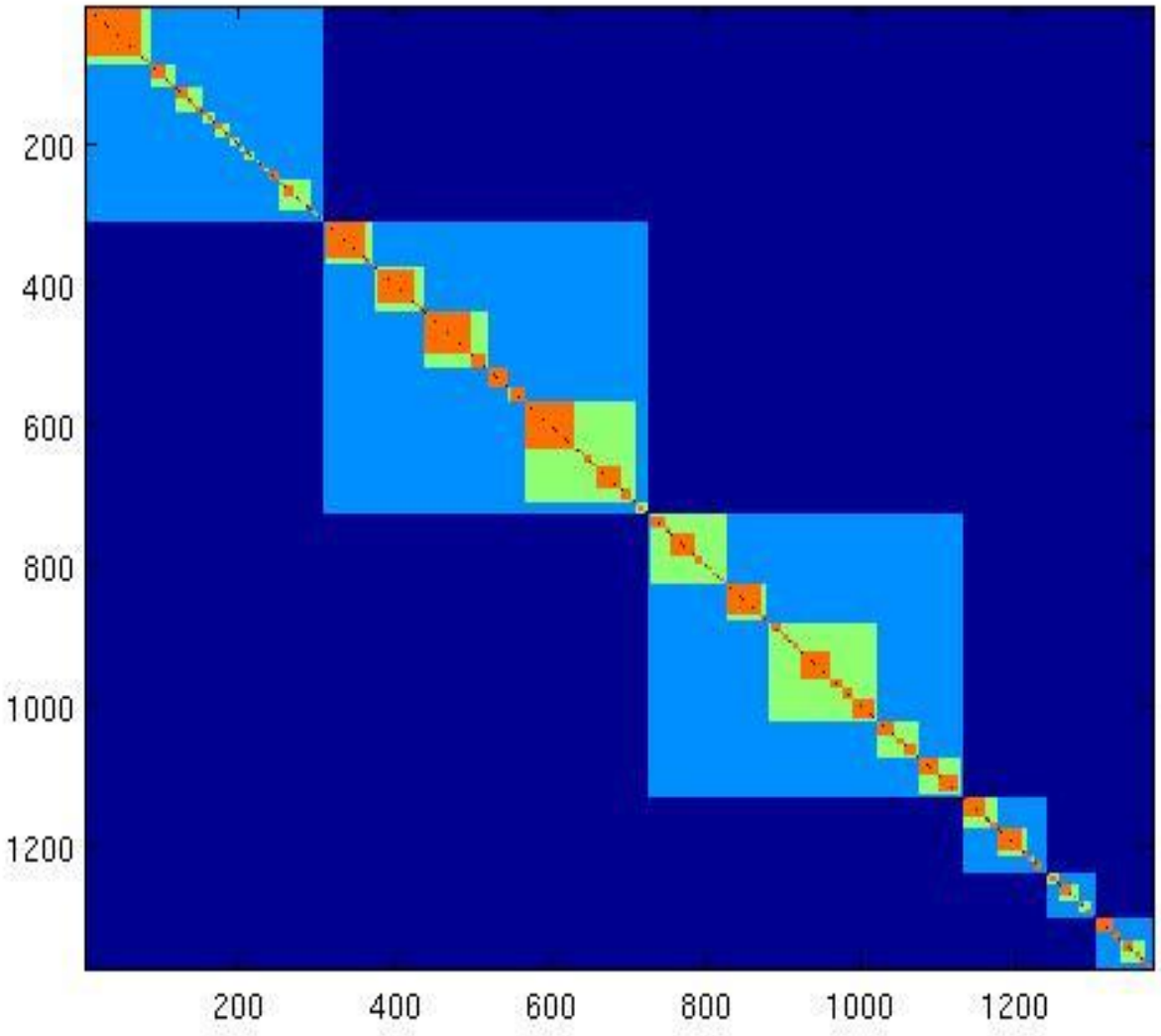
Kernel



$$\begin{bmatrix} 1 & 0.5 & 0 & 0 & \cdots \\ 0.5 & 1 & 0 & 0 & \\ 0 & 0 & 1 & 0.5 & \\ 0 & 0 & 0.5 & 1 & \\ \vdots & & & & \ddots \end{bmatrix}$$

Possible shape of an output kernel

EC(enzyme chemical reaction)



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)
H-M^3 $-$ l_Δ	H-M^3 trained on ℓ_Δ (Rousu)
H-M^3 $-$ $l_{\bar{H}}$	H-M^3 trained on subtree loss (Rousu)
MMR$_{lin}$	Proposed method with linear input kernel
MMR$_{poly}$	Proposed method with polynomial kernel

Enzyme EC-feature dataset

Enzyme-EC	$\ell_{0/1}$	ℓ_{Δ}	P	R	F1
3-levels, 236 nodes					
SVM	99.7	1.3	**99.6**	41.1	58.2
H-SVM	98.5	**1.2**	98.9	41.7	58.7
H-RLS	95.6	2.0	51.9	54.7	53.3
H-M^3-l_{Δ}	95.7	**1.2**	87.0	49.8	63.3
H-M^3-$l_{\bar{H}}$	85.5	2.5	44.5	66.7	53.4
4-levels, 1345 nodes					
MMR$_{Poly(4)}$	**33.0**	2.2(1.6)	72.4(77.0)	**72.4(77.0)**	**72.4(77.0)**

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

Enzyme prediction with special kernels and methods

- Accuracies on test:

Sequence Kernels	Nearest neighbour	MMR linear	MMR poly-51	HM^3 linear	HM^3 poly-51
GTG,STR4,GAP	91.3	85.7	90.4	76.9	93.7

- Kernels are

  GTG protein 3D structure,
  STR4 string kernel, maximum length 4,
  GAP string kernel with gaps

- Presented on
  Machine Learning in Systems Biology (MLSB-2007)
  Evry, France, [1]

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-$l_{\bar{H}}$	65.0	1.73	84.1	70.6	76.7
MMR$_{lin}$	**46.9**	1.77	77.9	**77.9**	**77.9**

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

Computational times

	EC	WIPO-alpha
MMR_{lin}	48	1.9
MMR_{poly}	38	1.2

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

Reformulation of the primal problem

The equality

$$\langle \psi(\mathbf{y}), \mathbf{W}\phi(\mathbf{x}) \rangle_{\psi} = \langle \mathbf{W}, \phi(\mathbf{x})\psi(\mathbf{y})^T \rangle_F$$

and

$$\mathbf{W} = \sum_{k=1}^{m_k} \alpha_k \psi(\mathbf{y}_k)\phi(\mathbf{x}_k)^T$$

give the constraints, where α the dual variable replaced with $\mathbf{u}$ to distinct the primal.

$$\sum_{k=1}^{m_k} u_k \overbrace{\langle \psi(\mathbf{y}_i), \psi(\mathbf{y}_k) \rangle}^{\kappa^{\psi}(\mathbf{y}_i,\mathbf{y}_k)} \overbrace{\langle \phi(\mathbf{x}_k), \phi(\mathbf{x}_i) \rangle}^{\kappa^{\phi}(\mathbf{x}_k,\mathbf{x}_i)} \geq 1 - \xi_i, \ i = 1, \ldots, m$$
$$C \geq u_k \geq 0, \ k = 1, \ldots, m_k$$

Reparametrization of the primal problem

Unbiased case

$\min \ \frac{1}{2}\boxed{\langle \mathbf{W}'\mathbf{W} \rangle_F} + C\mathbf{1}'\boldsymbol{\xi}$	$\frac{1}{2}\boxed{\mathbf{u}'\mathbf{Q}\mathbf{u}} + C\mathbf{1}'\boldsymbol{\xi}$
w.r.t. $\boxed{\mathbf{W} : \mathcal{H}_\phi \to \mathcal{H}_\psi}$, $\boldsymbol{\xi} \in \mathbb{R}^m$,	$\boxed{\mathbf{u} \in \mathbb{R}^{n_r}}$ $\boldsymbol{\xi} \in \mathbb{R}^m$,
s.t. $\boxed{\langle \psi(\mathbf{y}_i), \mathbf{W}\phi(\mathbf{x}_i) \rangle_{\mathcal{H}_\psi}} \geq 1 - \xi_i$ $\boldsymbol{\xi} \geq \mathbf{0}, \ i = 1, \ldots, m$	$\boxed{\sum_{r=1}^{n_r} u_r \kappa_{ir}^{\psi} \kappa_{ri}^{\phi}} \geq 1 - \xi_i$ $\boldsymbol{\xi} \geq \mathbf{0}, \ i = 1, \ldots, m$
$\kappa_{ir}^{H_\psi} = \langle \psi(y_i), \psi(y_r) \rangle_{\mathbf{H}_\psi}$ $\kappa_{ri}^{H_\phi} = \langle \phi(x_r), \phi(x_i) \rangle_{\mathbf{H}_\phi}$	$[\mathbf{K}_{H_\psi}]_{pq} = \kappa_{pq}^{H_\psi}$ $[\mathbf{K}_{H_\phi}]_{rs} = \kappa_{rs}^{H_\phi}$ $\mathbf{Q} = \mathbf{K}_{H_\psi} \bullet \mathbf{K}_{H_\phi}$ or $\mathbf{I}$

Similar reparametrization proposed by Mangasarian for the binary SVM [4].

One class form

- Let **G** be a matrix such that

$$G_{ij} = (\mathbf{\Psi}(x_i, y_i))_j = \kappa_{ij}^{\psi} \kappa_{ji}^{\phi} = \langle \psi(y_i), \psi(y_j) \rangle \langle \phi(x_i), \phi(x_j) \rangle$$

and we have the base problem:

	Primal		Dual
min	$\frac{1}{2}\|\mathbf{u}\|_2^2 + C\mathbf{1}'\xi$	min	$\frac{1}{2}\alpha'\mathbf{G}\mathbf{G}'\alpha - \mathbf{1}'\alpha$
w.r.t.	$\mathbf{u},\ \xi$	w.r.t.	$\alpha \in \mathbb{R}^m,$
s.t.	$\mathbf{G}\mathbf{u} \geq \mathbf{1} - \xi,$	s.t.	$\mathbf{0} \leq \alpha \leq C\mathbf{1}$
	$\xi \geq \mathbf{0}$		

!A consequence!

- **All the information known about the data incorporated into the matrix G**

  $\Rightarrow$ **similar to the kernel trick!**

- **But what kind of matrices are the proper ones to choose them as data descriptors?**

!A consequence!

- **All the information known about the data incorporated into the matrix G**

  $\Rightarrow$ **similar to the kernel trick!**

- **But what kind of matrices are the proper ones to choose them as data descriptors?**

Properties of **G**

G is not restricted to be

Positive (Semi)Definite It can contain non-definite inner products, e.g. Minkowski or Hyperbolic geometry,

Symmetric It can contain anti-symmetric inner products, i.e. $\langle a, b\rangle = -\langle b, a\rangle$

Square matrix

The structures processed in a learning task might have very irregular geometrical properties[2]

- they are not vectors of a Hilbert space, or
- they can not be approximated by this kind of objects.

[2] See Pekalska (2005) [9]

Extending the scope, other kind of regularizations

- Let us change the regularization term

$$\begin{array}{ll} \min & \frac{1}{2}\mathscr{R}(\mathbf{w}) + C\mathbf{1}'\boldsymbol{\xi} \\ \text{w.r.t.} & \mathbf{w},\ \boldsymbol{\xi} \\ \text{s.t.} & \mathbf{G}\mathbf{w} \geq \mathbf{1} - \boldsymbol{\xi} \\ & \boldsymbol{\xi} \geq \mathbf{0}, \end{array}$$

where $\mathscr{R}(.)$ might be $\|.\|_1$, $\|.\|_2$, $\|.\|_2^2$, $\|.\|_\infty$ and any reasonable measures of regularisation.

Examples for the matrix **G**

Similarity case: large values of G_{ij} mean high similarity

- e.g. inner product[3]:

$$G_{ij} = \overbrace{\langle \psi(\mathbf{y}_i), \psi(\mathbf{y}_j) \rangle}^{s^{\psi}(\mathbf{y}_i, \mathbf{y}_j)} \overbrace{\langle \phi(\mathbf{x}_j), \phi(\mathbf{x}_i) \rangle}^{s^{\phi}(\mathbf{x}_j, \mathbf{x}_i)}$$

- Using inverse distances, potential functions

$$G_{ij} = \frac{1}{1 + d^2(\psi(\mathbf{y}_i), \psi(\mathbf{y}_j)) d^2(\phi(\mathbf{x}_j), \phi(\mathbf{x}_i))}$$

[3] Mangasarian (1998): Generalized SVM [4]

Can we use distances or any dissimilarity measures?

Dissimilarity case: small values of G_{ij} mean high similarity

- e.g. distances:

$$G_{ij} = d(\psi(\mathbf{y}_i), \psi(\mathbf{y}_j)) d(\phi(\mathbf{x}_j), \phi(\mathbf{x}_i))$$

We need to change the regularization strategy!

Predictions, a plausible approach

- Conjecture that maximizing(minimizing) the margin gives the best answer.
- Assume that the set of the possible outputs is $\widetilde{\mathcal{Y}}$.
- Similarity case:

$$\mathbf{y}_* = \arg\max_{\mathbf{y}\in\widetilde{\mathcal{Y}}} \sum_{k=1}^{m_k} w_k \overbrace{\langle \psi(\mathbf{y}), \psi(\mathbf{y}_k)\rangle}^{s^{\psi}(\mathbf{y},\mathbf{y}_k)} \overbrace{\langle \phi(\mathbf{x}_k), \phi(\mathbf{x})\rangle}^{s^{\phi}(\mathbf{x}_k,\mathbf{x})}$$

- Dissimilarity case:

$$\mathbf{y}_* = \arg\min_{\mathbf{y}\in\widetilde{\mathcal{Y}}} \sum_{i=1}^{m} \alpha_i \overbrace{d(\psi(\mathbf{y}), \psi(\mathbf{y}_i))}^{d^{\psi}(\mathbf{y},\mathbf{y}_i)} \overbrace{d(\phi(\mathbf{x}_i), \phi(\mathbf{x}))}^{d^{\phi}(\mathbf{x}_i,\mathbf{x})}$$

Let's play!

The game

We are given

- two players,
- a payoff matrix **G**.

Player 2 \ Player 1			
	−2	−1	0
	−1	0	1
	0	1	2

$= \mathbf{G}$

- Player 1 chooses a column index j and
- Player 2 chooses a row index i then
- Player 1 gains G_{ij} and
- Player 2 loses the same.

It is called: two players, zero-sum game.
See von Neumann (1928) [8].

Let's play!

Repeat the game

- Players have to be unpredictable otherwise they can lose
- They change their choice of indeces

The strategies:
Choose column or row with a certain probabilities.

- Player 1 chooses j with probability a_j and
- Player 2 chooses i with probability d_i.

They are called mixed strategies.

The learning game

- Choose g_{ij} as $y_i h_j(\mathbf{x_i})$!
  - $g_{ij} > 0$ if y_i and $h_j(\mathbf{x_i})$ agree in sign and
  - $g_{ij} < 0$ if y_i and $h_j(\mathbf{x_i})$ are distinct.

  G is a real payoff for Player 1.
- The expected payoff for Player 1 equals to

$$\sum_{ij} G_{ij} \textbf{Prob}(\text{Player 1} = j, \text{Player 2} = i)$$
$$= \sum_{ij} G_{ij} a_j d_i,$$

  since the player choices are independent by definition.
- Player 1 tries to maximize, player 2 tries to minimize this value.

L_1 norm regularization, $\Rightarrow$linear programming

The learning game

Players

Learner(1) Nature(2)

Strategies

Find the best weights for weak learners!

Find the worst distribution, the weights to the data!

$$\max_w \min_\alpha \sum_{ij} \alpha_i \mathbf{G}_{ij} w_j \quad = \quad \min_\alpha \max_w \sum_{ij} \alpha_i \mathbf{G}_{ij} w_j$$

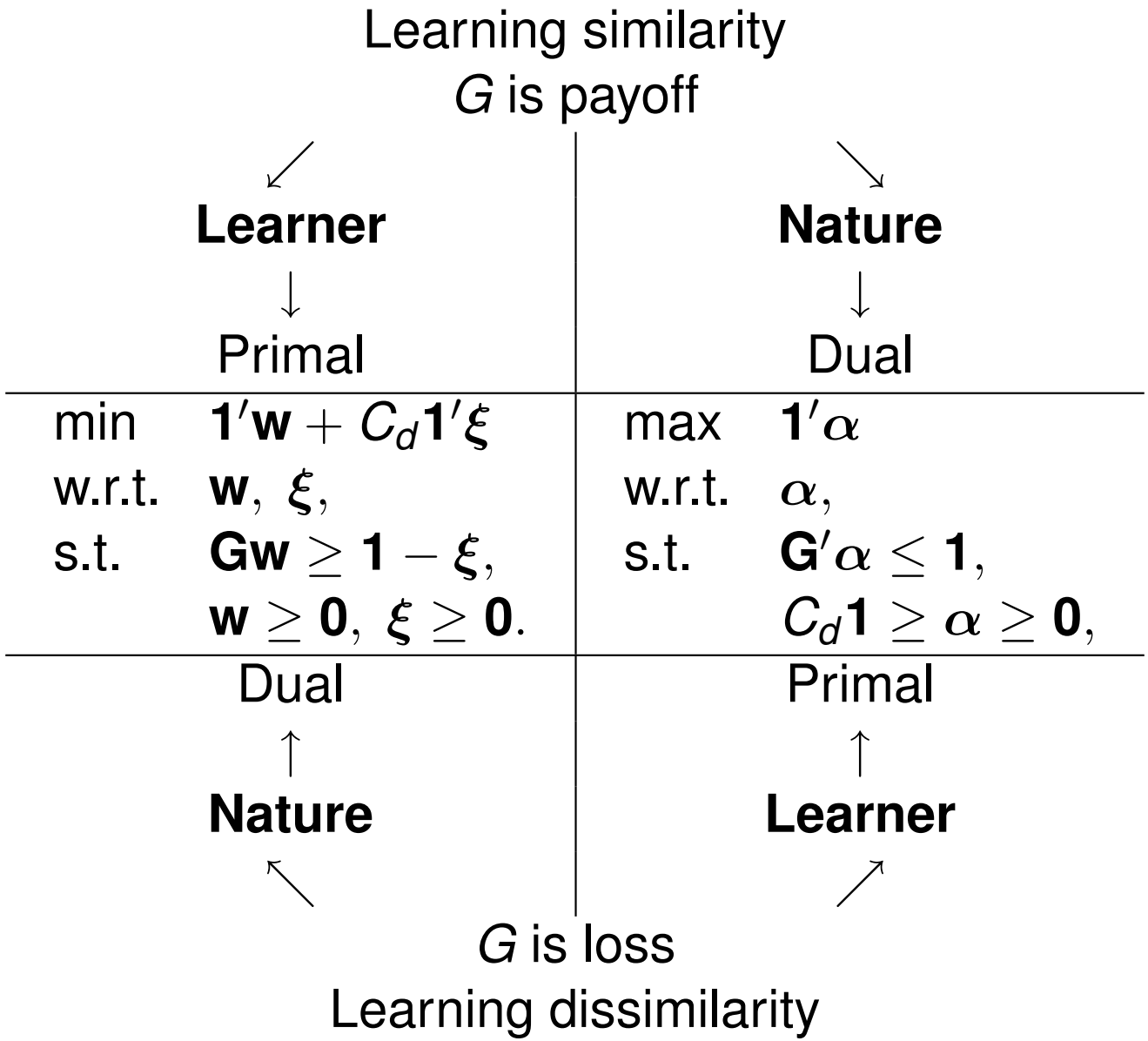
$$\sum_j w_j = 1,\ w_j \geq 0,\ j = 1, \ldots, n,$$
$$\sum_i \alpha_i = 1,\ \alpha_i \geq 0,\ i = 1, \ldots, m.$$

$\mathbf{w}$ Learner strategy

$\boldsymbol{\alpha}$ Nature strategy

J. Neumann, 1928

Learning scenarios, linear example



Boosting

Given

- a sample $\mathcal{S} = \{y_i, x_i\},\ i = 1, \ldots, m$, where
  $y_i \in \{-1, 1\}$ are the labels that we are going to predict,
  $x_i \in \mathbb{R}^{n_x}$ are input vectors,
- a set of so called weak learners
  $\mathcal{H} = \{h_j : x \to \{-1, 1\},\ j = 1, \ldots, n\}$,
  assume if $h_j \in \mathcal{H}$ then $-h_j \in \mathcal{H}$.

Let $h_{ij} \doteq h_j(\mathbf{x_i})$.
We are looking for a predictor, a decision function, as a convex combination of the weak learners

$$f(x) = \sum_j a_j h_j(x),\ \sum_j a_j = 1,\ a_j \geq 0,$$

which can outperform the prediction capability of the weak learners.
See Schapire (2002) [10].

Linear Programming Boosting

How to solve

The point of view of the learner, Player 1, is:

$$\begin{array}{ll} \max_a \min_d & \sum_{ij} g_{ij} a_j d_i \\ \text{s.t.} & \sum_j a_j = 1,\ a_j \geq 0,\ j = 1, \ldots, m, \\ & \sum_i d_i = 1,\ d_i \geq 0,\ i = 1, \ldots, n. \end{array}$$

It boils down into a primal, point of view of the learner, and a dual problem, point view of the nature, where $g_{ij} = y_i h_j(x_i)$

Primal

$$\begin{array}{ll} \max_{\rho,a} & \rho \\ \text{s.t.} & \sum_{j=1}^{n} g_{ij} a_j \geq \rho, \\ & \qquad i = 1, \ldots, m, \\ & \sum_j a_j = 1,\ a_j \geq 0, \end{array}$$

Learner

Dual

$$\begin{array}{ll} \min_{\beta,d} & \beta \\ \text{s.t.} & \sum_{i=1}^{m} g_{ij} d_i \leq \beta, \\ & \qquad j = 1, \ldots, n, \\ & \sum_i d_i = 1,\ d_i \geq 0. \end{array}$$

Nature

Normalization

- **Preprocessing**

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

  - 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**

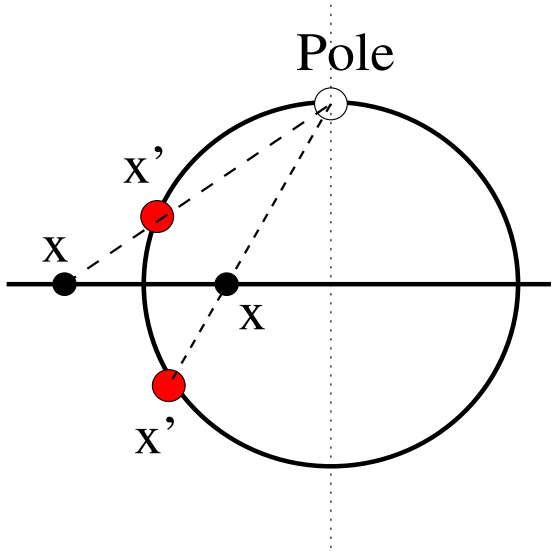
$$\left.\begin{array}{l} \psi : \mathcal{Y} \rightarrow \mathcal{S}_y \subset \mathcal{H}_\psi, \; \mathcal{S}_y : \\ \phi : \mathcal{X} \rightarrow \mathcal{S}_x \subset \mathcal{H}_\phi, \; \mathcal{S}_x : \end{array}\right\} \text{Hyper-spheres}$$

Spherical embedding

- **Spherical embedding**

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

- **Stereographic projection**



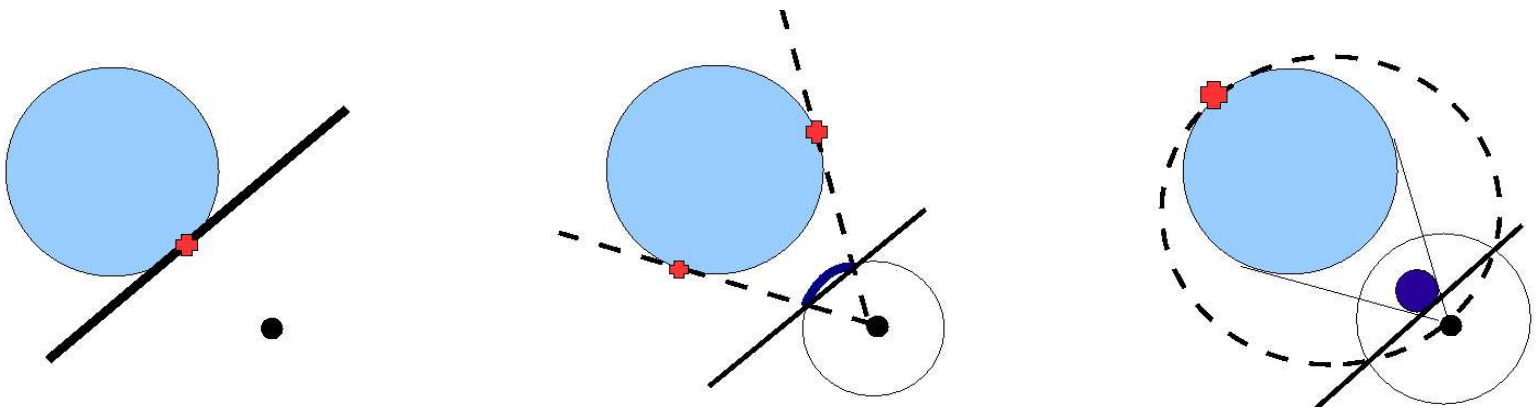
$$\begin{aligned}\Phi &: \phi(x) \to \phi'(x),\\ K'_{ij} &= \langle \phi'(x)_i, \phi'(x)_j \rangle \\ &= R^2\left(1 - \frac{2R^2\|\phi(x_i)-\phi(x_j)\|^2}{(\|\phi(x_i)\|^2+R^2)(\|\phi(x_j)\|^2+R^2)}\right) \\ &= R^2\left(1 - \frac{2R^2(K_{ii}+K_{jj}-2K_{ij})}{(K_{ii}+R^2)(K_{jj}+R^2)}\right),\end{aligned}$$

R Ball radius.

Effect of the normalization

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

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



identity | projection onto ball | inversion

Solution

Quadratic Augmented Lagrangian Form

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

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,
  - The update formula

$$\alpha_i^{k+1} = P_{[0,C]}(-1 - \langle \alpha^k, K_i \rangle / K_{ii})$$
$$i = 1, \ldots, m$$

- Update the Lagrangian,
- Increase the penalty constant

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

Alternative solution approaches

- Extragradient based methods for variational inequalities
  - Korpelevich [3]
  - Nesterov [7], [6]
  - Nemirovski [5]
- Cutting plane methods(e.g. column generation, decomposition)
  - Joachims [2]
- Active set methods

Multiview learning

Additive case

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

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

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

$\bullet$ element-wise product

Multiview learning

Product case

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

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

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

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

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

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

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

$\bullet$ element-wise product

Epilogue

“Grey is, young friend, all theory:
And green of life the golden tree.”

Johann Wolfgang von Göthe: Faust

This is the End

Thanks!


K. Astikainen, J. Rousu, L. Holm, E. Pitkanen, and S. Szedmak.
Towards structured prediction of enzyme function.
In *Machine Learning in Systems Biology (MLSB-2007), Evry, France, September 2007*. 2007.


T. Joachims.
Training linear svms in linear time.
In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), ACM*. 2006.


G. Korpelevich.
The extragrdient method for finding saddle points and other problems.
Ekonomika i Matematicheskie Metody, 12:747–756, 1976.
In Russian; English translation in Matekon.


O. L. Mangasarian.
Generalized support vector machines.
In *Advances in Large Margin Classifiers*, pages 135–146. MIT Press, 2000.

A. Nemirovski.
Prox-method with rate of convergence o(1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems.
SIAM Journal on Optimization, 15:229–251, 2004.

Y. Nesterov.
Dual extrapolation and its application for solving variational inequalities and related problems.
In *CORE Discussion Paper/68, September 2003*. 2003.

Y. Nesterov.
Smooth minimization of nonsmooth functions.
In *CORE Discussion Paper/12, February 2003*. 2003.

John V. Neumann.
Zur theorie der gesellschaftsspiele.
Mathematische Annalen, 100:295–320, 1928.

English Translation Fin, Tucker, A.W. and R.D. Luce, ed., Contributions to the Theory of Games IV, Annals of Mathematics Studies 40, 1959.


E. Pekalska and R.P.W. Duin.
The Dissimilarity Representation for Pattern Recognition. Foundations and Applications.
World Scientific, Singapore, 2005.


R. Schapire.
The boosting approach to machine learning: an overview.
In *MRSI Workshop on Nonlinear Estimation and Control*. 2002.