Learning interpretable SVMs for biological sequence classification

Sören Sonnenburg[†], Gunnar Rätsch[†,♮] Christin Schäfer[†]

Fraunhofer Institut Rechnerarchitektur und Softwaretechnik FIRST

MAX-PLANCK-GESELLSCHAFT

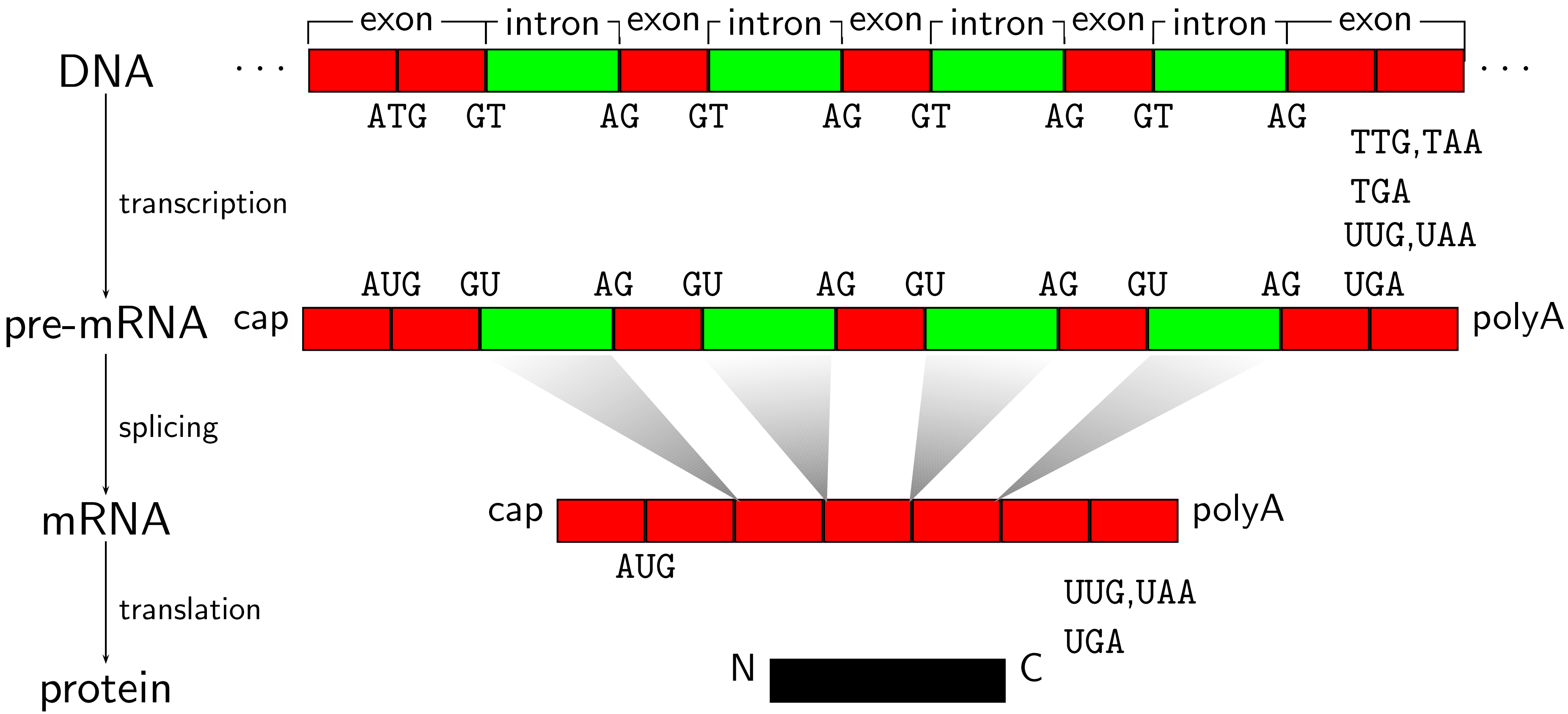
† Fraunhofer FIRST.IDA, Kekuléstr. 7, 12489 Berlin, Germany
♮ Friedrich Miescher Laboratory of the Max Planck Society,
Spemannstr. 39, 72076 Tübingen, Germany
{Soeren.Sonnenburg, Christin.Schaefer}@first.fraunhofer.de,
Gunnar.Raetsch@tuebingen.mpg.de

Roadmap:

- The Motivating Application - Splice Site Prediction
- Multiple Kernel Learning (MKL)
- Derivation of the MKL Optimization Problem
- Algorithms
- Significance Analysis
- Results
- Outlook and Conclusion

The Motivating Application

Splice sites are locations on DNA at boundaries of
- exons (which code for proteins)
- introns (which do not)



Detection of Splice Sites

FIRST

Focus on Acceptor Splice Site Prediction

```
                    Intron Exon
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
TTGTTTTAATATTCAATTTTTTACAGTAAGTTGCCAATTCAATGTTCCAC
TACCTAATTATGAAATTAAAATTCAGTGTGCTGATGGAAACGGAGAAGTC
```

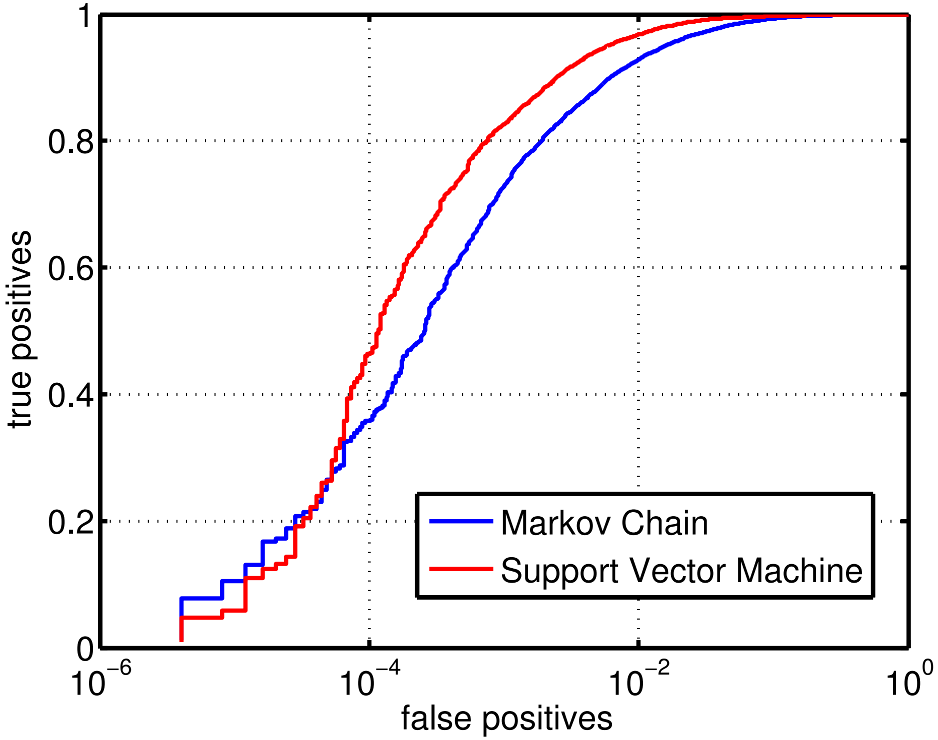
- aligned sequences of fixed length (AG always at same position)
- Task: distinguish *C. elegans* splice sites from fake splice sites

⇒ 2-class classification problem

Standard Approach: Markov Chain

Standard Approach: Use two higher order Markov Chains

- one model for fake splice sites and one for true splice sites trained on $\approx 600,000$ examples, of them only $\approx 36,000$ true splice sites

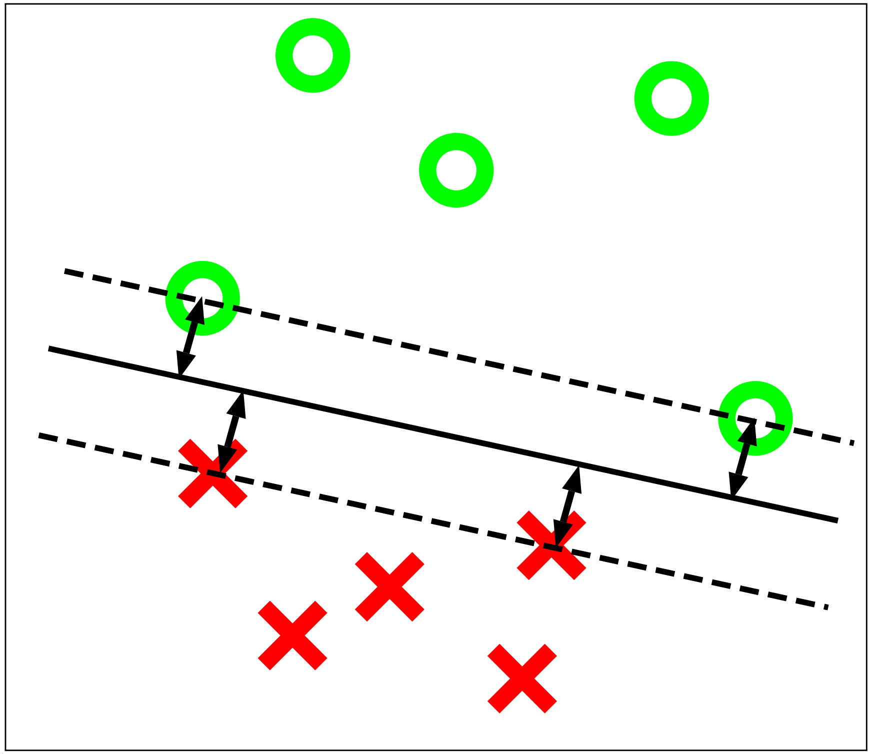


We can do better using SVMs!

Drawback: We loose interpretability of the result!

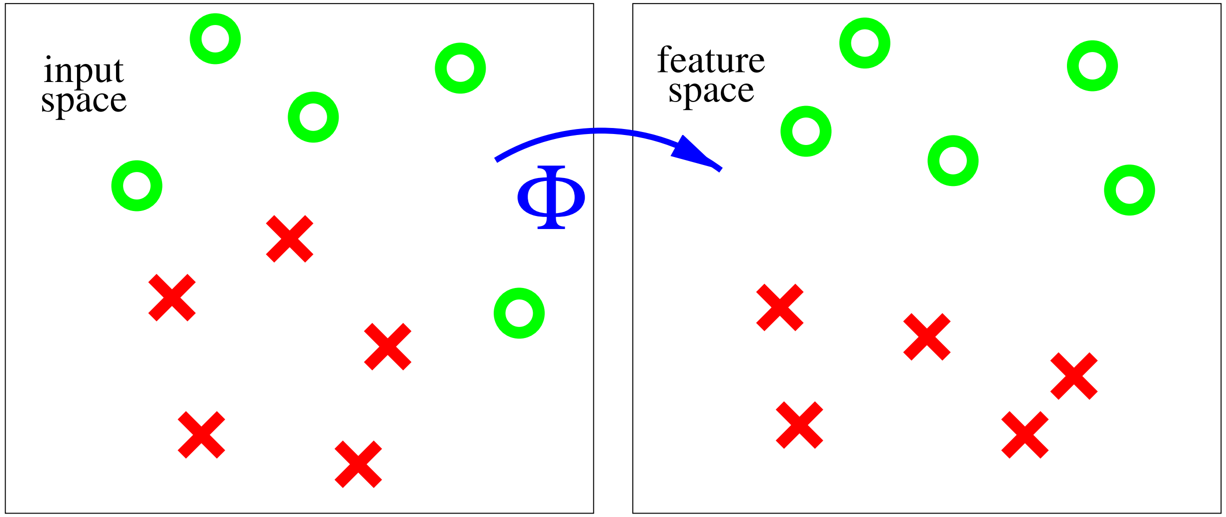
Support Vector Machine

- given: points (e.g. sequences) $\boldsymbol{s}_i \in \mathbb{S}$ $(i = 1, \ldots, N)$ with respective labels $y_i \in \{-1, +1\}$
- in training hyperplane that maximizes margin is chosen

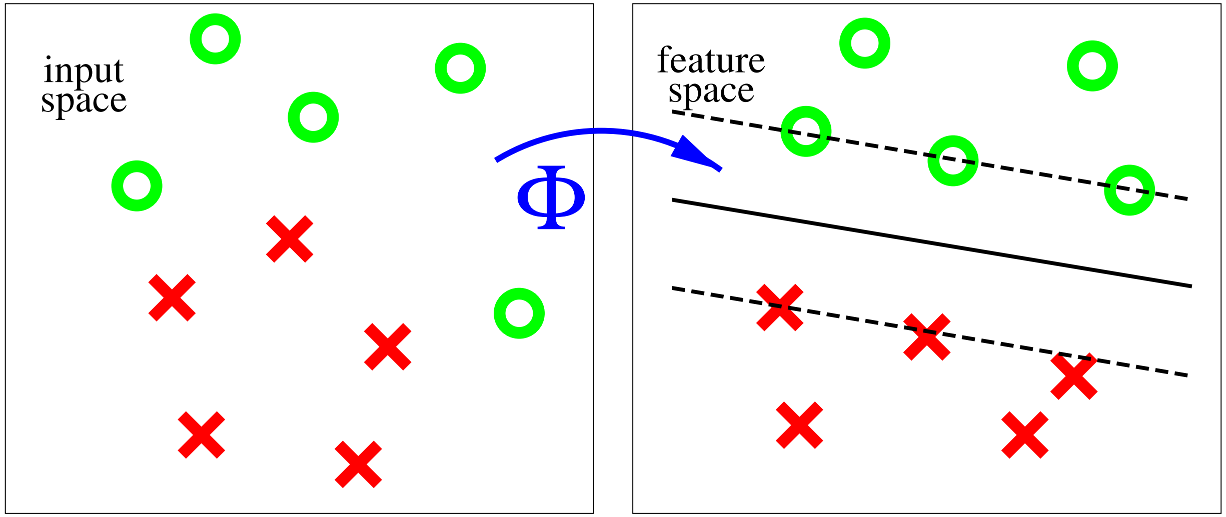


Decision function $f(\boldsymbol{s}) = \boldsymbol{w} \cdot \boldsymbol{s} + b$

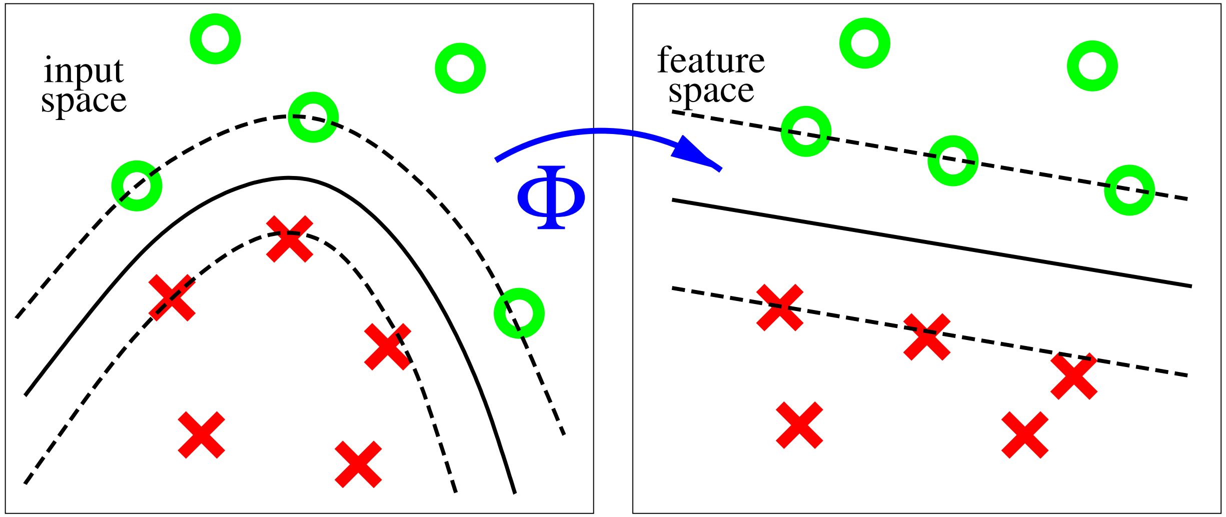
SVM with Kernels



SVM with Kernels



SVM with Kernels



- SVM decision function in kernel feature space:

$$f(\boldsymbol{s}) = \sum_{i=1}^{N} y_i \alpha_i \underbrace{\Phi(\boldsymbol{s}) \cdot \Phi(\boldsymbol{s}_i)}_{=\mathrm{k}(\boldsymbol{s},\boldsymbol{s}_i)} + b \quad (1)$$

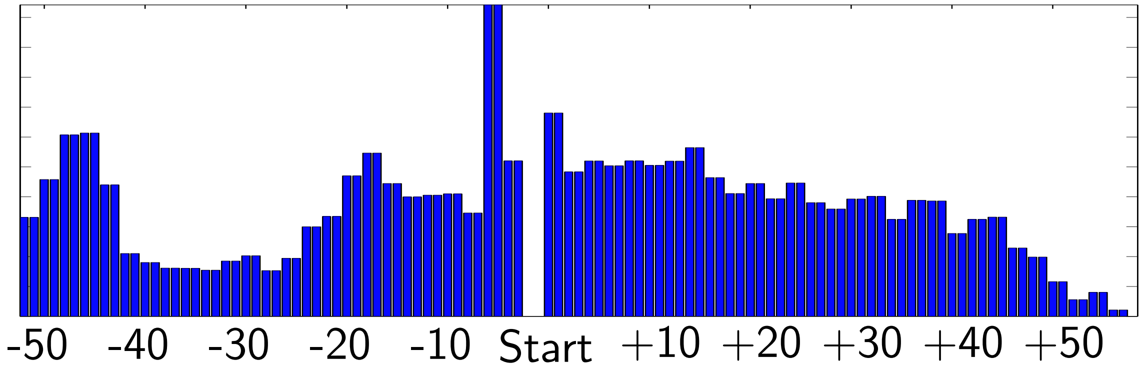
- find parameters $\boldsymbol{\alpha}$ by solving quadratic optimization problem

Problem: Decision function (1) is hard to interpret

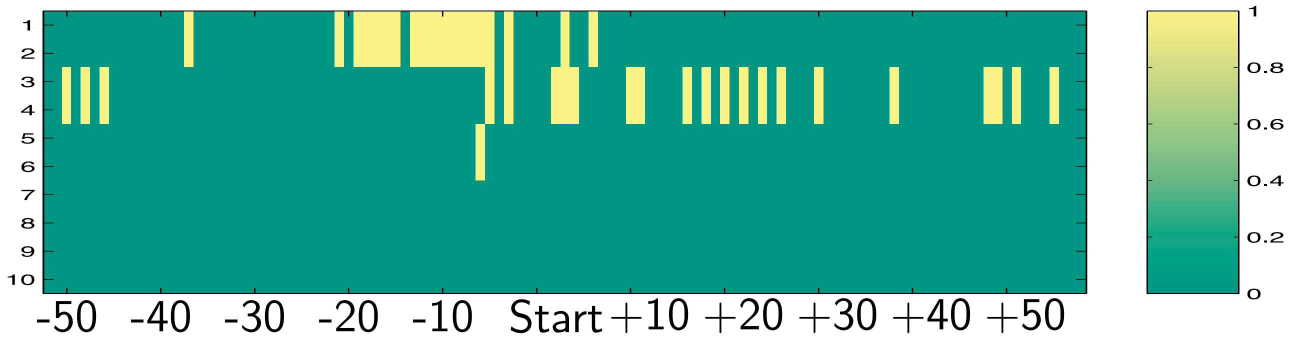
Understanding the SVM Decision

Splice Sites

1. Which positions in the sequence are important for discrimination?



2. What characterizes those positions?



3. Which motifs at which position are important?

Approach: Optimize Combination of Kernels

- Define Kernel as Convex Combination of Subkernels:

$$\mathrm{k}(\boldsymbol{s}, \boldsymbol{s'}) = \sum_{l=1}^{L} \beta_l \, \mathrm{k}_l(\boldsymbol{s}, \boldsymbol{s'})$$

e.g. Weighted Degree Kernel

$$\mathrm{k}(\boldsymbol{s}, \boldsymbol{s}') = \sum_{l=1}^{L} \beta_l \sum_{k=1}^{d} \mathrm{I}(\boldsymbol{u}_{k,l}(\boldsymbol{s}) = \boldsymbol{u}_{k,l}(\boldsymbol{s}'))$$

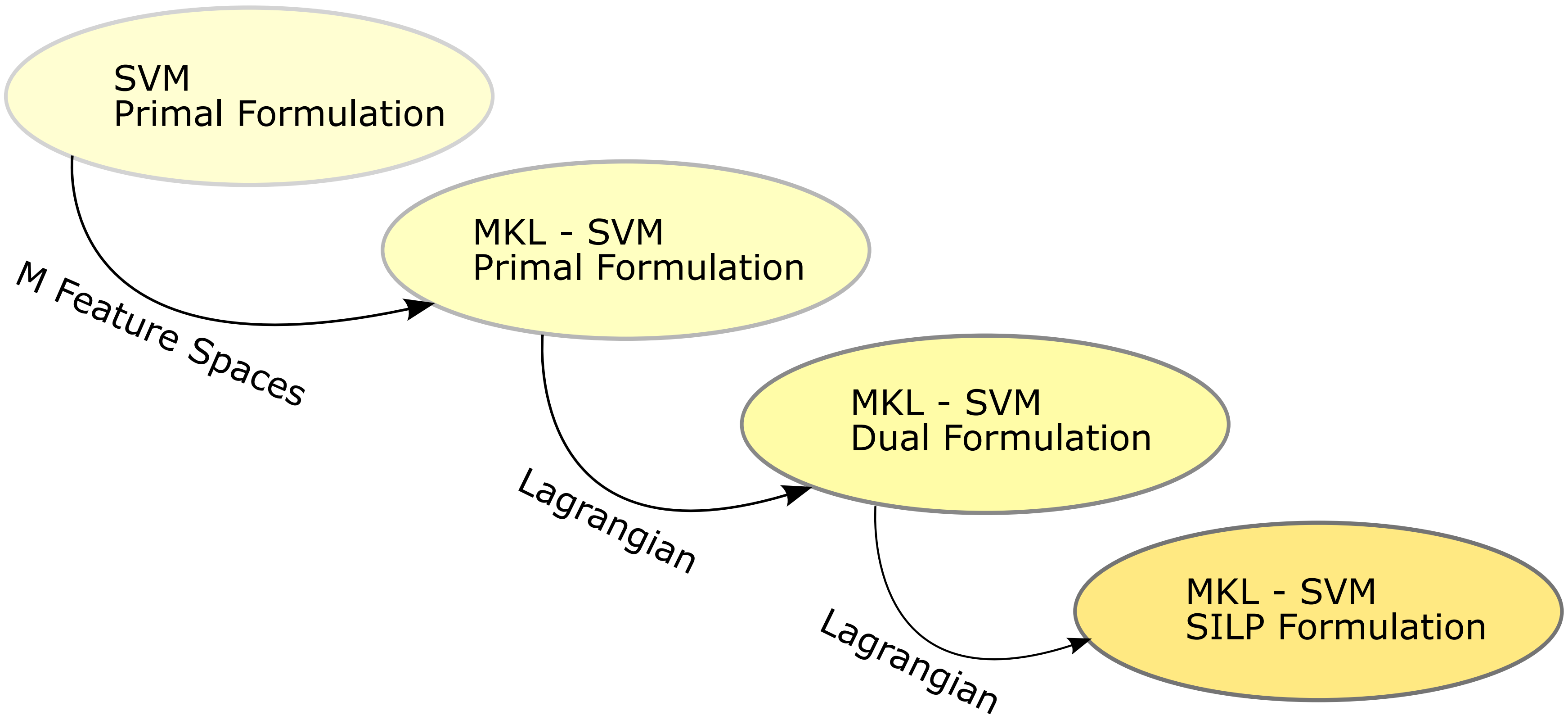
- optimize weights $\boldsymbol{\beta}$ such that margin is maximized

$\Rightarrow$ determine $(\boldsymbol{\beta}, \boldsymbol{\alpha}, b)$ simultaneously

$\Rightarrow$ **Multiple Kernel Learning** (Bach, Lanckriet and Jordan 2004)
$\Rightarrow$ leads to Second Order Cone Problem - *very slow*!
$\Rightarrow$ we propose new *efficient* algorithm

Rewriting the MKL formulation

From the Standard SVM Primal to the Semi-Infinite Linear Program:



MKL Optimization Problem

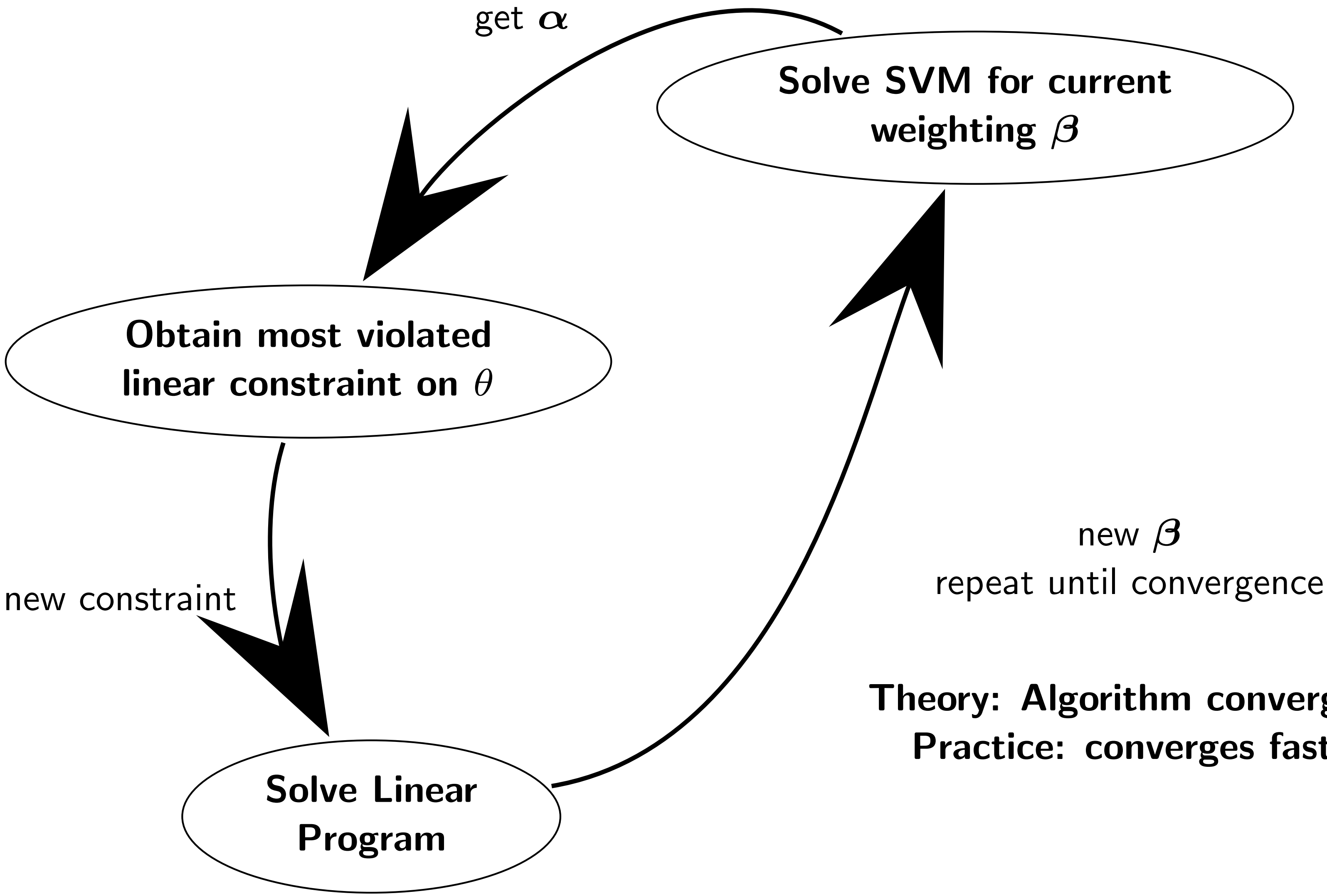
Semi-Infinite Linear Program Formulation:

$$
\begin{array}{ll}
\max & \theta \\
\text{w.r.t.} & \theta \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}_+^M \text{ with } \sum_{j=1}^{M} \beta_j = 1 \\
\text{s.t.} & \sum_{j=1}^{M} \beta_j \left(\frac{1}{2} \sum_{r=1}^{N} \sum_{s=1}^{N} \alpha_r \alpha_s y_r y_s K_j(\boldsymbol{s}_r, \boldsymbol{s}_s) - \sum_{i=1}^{N} \alpha_i \right) \geq \theta \\
 & \text{for all } \boldsymbol{\alpha} \text{ with } 0 \leq \boldsymbol{\alpha} \leq C \text{ and } \sum_{i=1}^{N} y_i \alpha_i = 0
\end{array}
$$

⇒ Linear Problem, but infinitely many constraints

- one constraint for each $\boldsymbol{\alpha}$
- can be solved using Column Generation
- most violated constraints can be identified by solving SVM

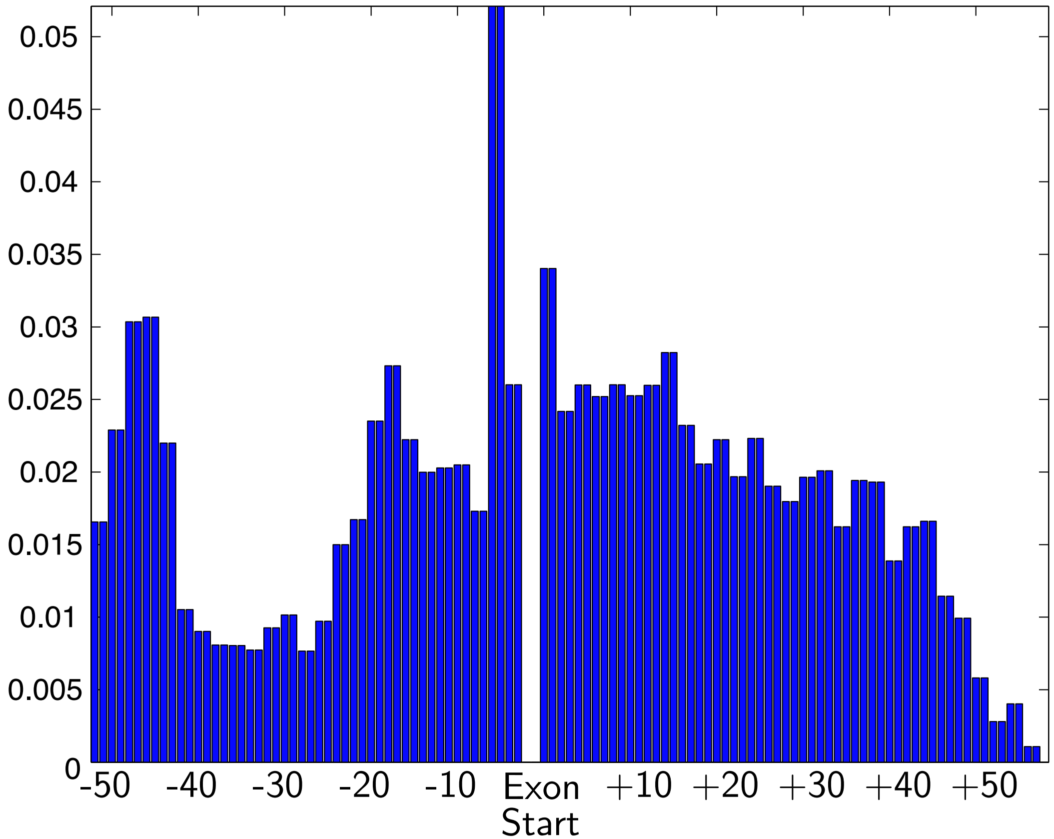
Column Generation



Theory: Algorithm converges!
Practice: converges fast!

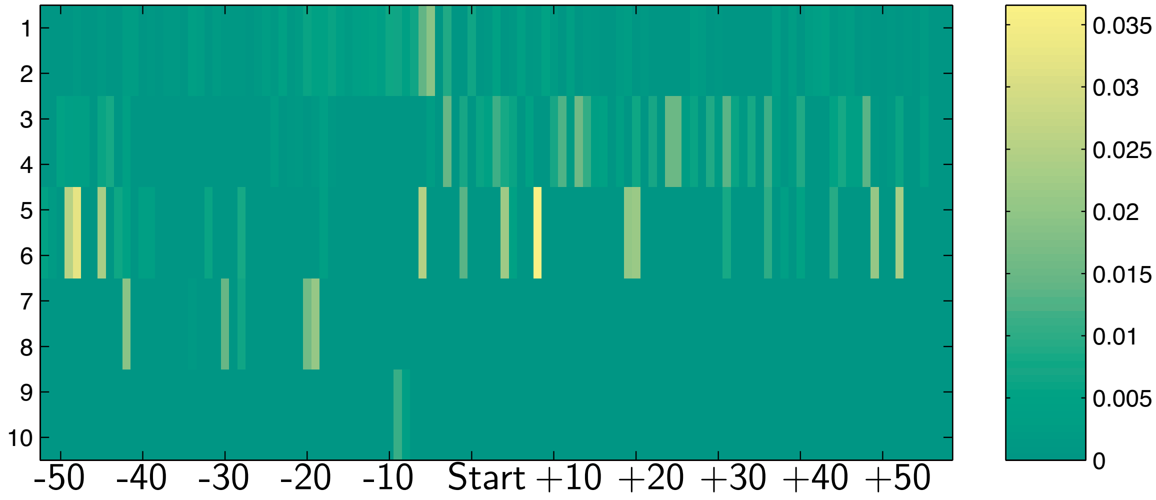
Results on Splice Sites

- Which positions in the sequence are important for discrimination:



Results on Splice Sites

What characterizes the positions:

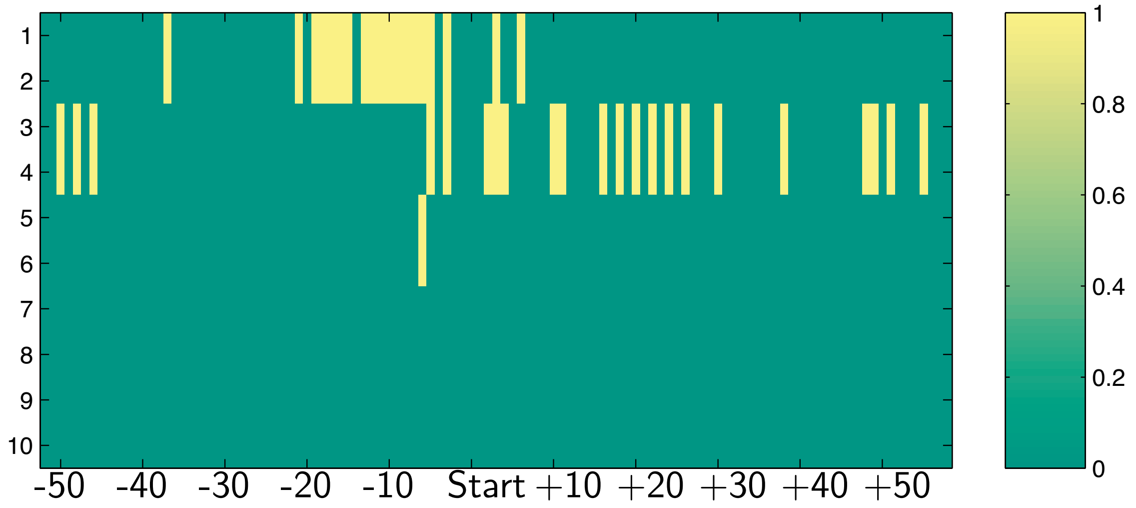


- Can we use that weighting to interpret the SVM solution? ⇒ Not yet!
  - Stability of the weighting β?
  - Which weights are significant?

Use a statistical test to investigate significance of weights

RESULTS ON SPLICE SITES

What characterizes the positions:



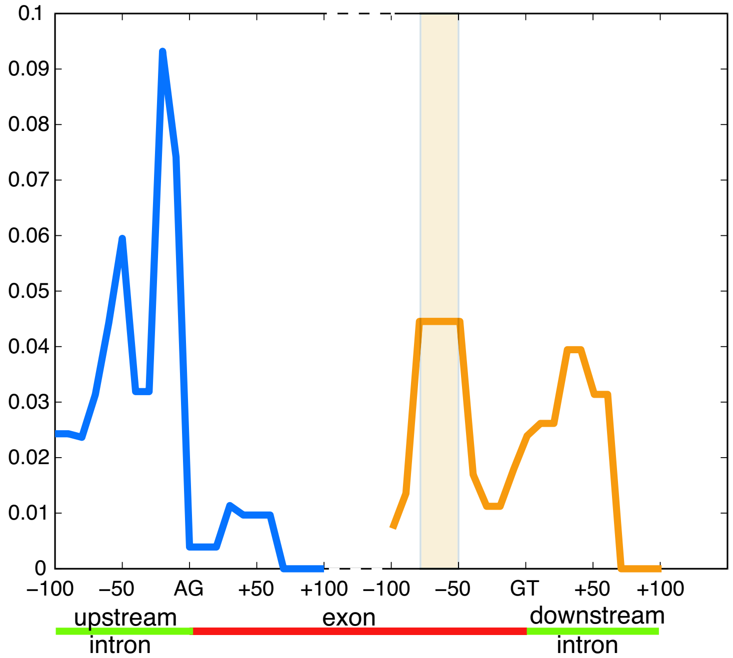
- Can we use that weighting to interpret the SVM solution? $\Rightarrow$ NOW !
  - Stability of the weighting β?
  - Which weights are significant?

Use a statistical test to investigate significance of weights

MKL to understand Alternative Splicing

- Distinguish alternatively from constitutively spliced exons (ISMB 2005).

- Which positions are important?
- Which motifs are important?



Hexamer	E-value
TTTAAA	1.8e-12
AATTTT	2.2e-10
ATTTTA	2.9e-09
CAGCAG	1.2e-08
TAATTT	8.3e-08
TTCCCC	2.1e-07

Conclusion

Conclusion:

- MKL learns convex combination of kernels
  $\Rightarrow$ allows for interpreting SVM result
  $\Rightarrow$ matches prior knowledge about splice sites

- simple iterative algorithm

- suitable for large scale problems ($> 100,000$ examples)

More information:

http://ida.first.fhg/~sonne/mkl_splice

Acknowledgements: Partial support from the PASCAL Network of Excellence (EU #506778), DFG grants JA379/13-2 and MU987/2-1. Thanks to Alexander Zien and K.-R. Müller.

Significance Analysis

- T bootstrap replicates of dataset
- train T times and obtain $\boldsymbol{\beta}_{1\dots T}$ kernel weightings
- Bernoulli variable

$$X_{k,i}^t = \begin{cases} 1, & \beta_{k,i}^t > \tau \\ 0, & \text{else} \end{cases}.$$

- Testhypothesis $\mathcal{H}_0$: The weight $\beta_{k,i}$ is not significant!

$$0.05 = \alpha \geq P_{\mathcal{H}_0}(\text{reject } \mathcal{H}_0) = P_{\mathcal{H}_0}(Z_{k,i} > v) = \sum_{j=v}^{T} \binom{T}{j} \hat{p}_0(1 - \hat{p}_0)$$

(where $Z_{k,i} = \sum_{t=1}^{T} X_{k,i}^t$ and $\hat{p}_0 = \#(\beta_{k,i}^t > \tau)/T \cdot M$)

Standard SVM Optimization Problem

This we all know

SVM Primal formulation:

$$\begin{array}{rl} \min & \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\xi_n \\ \text{w.r.t.} & \boldsymbol{w}\in\mathbb{R}^k, \boldsymbol{\xi}\in\mathbb{R}_+^N, b\in\mathbb{R} \\ \text{s.t.} & y_i\left(\boldsymbol{w}^\top\Phi(\boldsymbol{s}_i)+b\right)\geq 1-\xi_i, \forall i=1,\dots,N \end{array}$$

MKL Optimization Problem I

MKL Primal formulation:

$$\min \quad \frac{1}{2}\left(\sum_{j=1}^{M} \beta_j \left\|\boldsymbol{w}_j\right\|_2\right)^2 + C\sum_{i=1}^{N} \xi_n$$

$$\text{w.r.t.} \quad \boldsymbol{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M), \boldsymbol{w}_j \in \mathbb{R}^{k_j}, \boldsymbol{\xi} \in \mathbb{R}_+^N, \boldsymbol{\beta} \in \mathbb{R}_+^M, b \in \mathbb{R}$$

$$\text{s.t.} \quad y_i\left(\sum_{j=1}^{M} \beta_j \boldsymbol{w}_j^\top \Phi_j(\boldsymbol{s}_i) + b\right) \geq 1 - \xi_i, \forall i = 1, \ldots, N$$

$$\sum_{j=1}^{M} \beta_j = 1$$

Properties: equivalent to SVM for $M = 1$; solution sparse in "blocks"; each block j corresponds to one kernel

MKL Optimization Problem II

Dual Formulation (Bach, Lanckriet, Jordan 2004):

$$\begin{aligned}
\min \quad & \frac{1}{2}\gamma^2 - \sum_{i=1}^{N} \alpha_i \\
\text{w.r.t.} \quad & \gamma \in \mathbb{R}, \boldsymbol{\alpha} \in \mathbb{R}^N \\
\text{s.t.} \quad & 0 \leq \boldsymbol{\alpha} \leq C, \sum_{i=1}^{N} \alpha_i y_i = 0 \\
& \underbrace{\sum_{r=1}^{N}\sum_{s=1}^{N} \alpha_r \alpha_s y_r y_s K_j(\boldsymbol{s}_r, \boldsymbol{s}_s)}_{=:S_j(\boldsymbol{\alpha})} - \gamma^2 \leq 0, \ \forall j = 1, \ldots, M
\end{aligned}$$

"partial Lagrangian:"

$$L := \frac{1}{2}\gamma^2 - \sum_{i=1}^{N} \alpha_i + \sum_{j=1}^{M} \beta_j (S_j(\boldsymbol{\alpha}) - \gamma^2)$$

MKL Optimization Problem II

Reformulation as Semi-Infinite Linear Program:

$$\max_{\boldsymbol{\beta}} \min_{\boldsymbol{\alpha}} \sum_{j=1}^{M} \beta_j \left(\frac{1}{2} S_j(\boldsymbol{\alpha}) - \sum_{i=1}^{N} \alpha_i \right)$$

$$\text{s.t. } 0 \leq \boldsymbol{\alpha} \leq C, \sum_{i=1}^{N} \alpha_i y_i = 0, \sum_{j=1}^{M} \beta_j = 1$$

max	θ
w.r.t.	$\theta \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}_{+}^{M}$ with $\sum_{j=1}^{M} \beta_j = 1$
s.t.	$\sum_{j=1}^{M} \beta_j \left(\frac{1}{2} S_j(\boldsymbol{\alpha}) - \sum_{i=1}^{N} \alpha_i \right) \geq \theta$
	for all $\boldsymbol{\alpha}$ with $0 \leq \boldsymbol{\alpha} \leq C$ and $\sum_{i=1}^{N} y_i \alpha_i = 0$

$\Rightarrow$ **Linear, but infinitely many constraints**

The Semi-Infinite Linear Program I

$$
\begin{aligned}
\max \quad & \theta \\
\text{w.r.t.} \quad & \theta \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}_+^M \text{ with } \sum_{j=1}^{M} \beta_j = 1 \\
\text{s.t.} \quad & \sum_{j=1}^{M} \beta_j \left(\frac{1}{2} S_j(\boldsymbol{\alpha}) - \sum_{i=1}^{N} \alpha_i \right) \geq \theta \\
& \text{for all } \boldsymbol{\alpha} \text{ with } 0 \leq \boldsymbol{\alpha} \leq C \text{ and } \sum_{i=1}^{N} y_i \alpha_i = 0
\end{aligned}
$$

Properties:

- optimize a convex combination
- infinitely many constraints
- quite easy to identify violated constraints

The Semi-Infinite Linear Program II

$$
\begin{aligned}
\max \quad & \theta \\
\text{w.r.t.} \quad & \theta \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^M_+ \text{ with } \sum_{j=1}^{M} \beta_j = 1 \\
\text{s.t.} \quad & \sum_{j=1}^{M} \beta_j \left(\frac{1}{2} S_j(\boldsymbol{\alpha}) - \sum_{i=1}^{N} \alpha_i \right) \geq \theta \\
& \text{for all } \boldsymbol{\alpha} \text{ with } 0 \leq \boldsymbol{\alpha} \leq C \text{ and } \sum_{i=1}^{N} y_i \alpha_i = 0
\end{aligned}
$$

Solving the SILP:

- Column Generation
  - fast, but no convergence rate
- Use Boosting like techniques: Arc-GV or AdaBoost*
  - known convergence rate $\mathcal{O}(\log(M)/\varepsilon^2)$
- SMO like algorithm
  - consider suboptimal SVM solutions: empirically 3-5 times faster

Solving the SILP: Column Generation I

$$
\begin{aligned}
\max \quad & \theta \\
\text{w.r.t.} \quad & \theta \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^M_+ \text{ with } \sum_{j=1}^{M} \beta_j = 1 \\
\text{s.t.} \quad & \sum_{j=1}^{M} \beta_j \left(\frac{1}{2} S_j(\boldsymbol{\alpha}) - \sum_{i=1}^{N} \alpha_i \right) \geq \theta \\
& \text{for all } \boldsymbol{\alpha} \text{ with } 0 \leq \boldsymbol{\alpha} \leq C \text{ and } \sum_{i=1}^{N} y_i \alpha_i = 0
\end{aligned}
$$

- solved by taking set of most violated constraints into account
- most violated constraints given by SVM solution for fixed $\boldsymbol{\beta}$

$$
\sum_{j=1}^{M} \beta_j \left(\frac{1}{2} S_j(\boldsymbol{\alpha}) - \sum_{i=1}^{N} \alpha_i \right) = \frac{1}{2} \sum_{r=1}^{N} \sum_{s=1}^{N} \alpha_r \alpha_s y_r y_s \sum_{j=1}^{M} \beta_j k_j(\boldsymbol{s}_r, \boldsymbol{s}_s) - \sum_{i=1}^{N} \alpha_i,
$$

- iteratively find most violated constraints, solve linear program with current constraints, ..., till convergence to the global optimum

Solving the SILP: Column Generation II

$D^0 = 1$, $\theta^1 = 0$, $\beta_k^1 = \frac{1}{M}$ for $k = 1, \ldots, M$
for $t = 1, 2, \ldots$ **do**
  obtain SVM's $\boldsymbol{\alpha}^k$ with kernel

$$\mathrm{k}^t(\boldsymbol{s}_i, \boldsymbol{s}_j) := \sum_{k=1}^{M} \beta_k^t \, \mathrm{k}_k(\boldsymbol{s}_i, \boldsymbol{s}_j)$$

  for $k = 1, \ldots, M$ **do**
    $D_k^t = \frac{1}{2} \sum_{r,s} \alpha_r^t \alpha_s^t y_r y_s \, \mathrm{k}_k(\boldsymbol{s}_r, \boldsymbol{s}_s) - \sum_r \alpha_r^t$
  end for
  $D^t = \sum_{k=1}^{M} \beta_k^t D_k^t$
  $(\boldsymbol{\beta}^{t+1}, \theta^{t+1}) = \mathrm{argmax} \;\; \theta$
    w.r.t. $\boldsymbol{\beta} \in \mathbb{R}_+^M, \theta \in \mathbb{R}$ with $\sum_k \beta_k = 1$
    s.t. $\sum_{k=1}^{M} \beta_k D_k^r \geq \theta$ for $r = 1, \ldots, t$
  if $|1 - \frac{\theta^{t+1}}{D^t}| \leq \epsilon$ **then break**
end for

Solving the SILP: Boosting I

Boosting

Primal

$$\max_{\boldsymbol{\alpha},\rho} \quad \rho$$
$$\text{subject to} \quad y_n \sum_{j=1}^{J} \alpha_j h_j(\boldsymbol{s}_n) \geq \rho$$
$$\alpha_j \geq 0$$
$$\sum_j \alpha_j = 1$$

Dual

$$\min_{\mathbf{d},\delta} \quad \delta$$
$$\text{subject to} \quad \sum_{n=1}^{N} d_n y_n h_j(\boldsymbol{s}_n) \leq \delta$$
$$d_n \geq 0$$
$$\sum_n d_n = 1$$

- max w.r.t. $-\delta$; d_n corresponds to β_n; h_j corresponds to expression using a certain $\boldsymbol{\alpha}$
- Arc-GV and AdaBoost*: ϱ and δ are optimized

The number of hypotheses can be very large or infinite!

Solving the SILP: Boosting I

An Arc-GV like Algorithm

$D^0 = 1$, $\rho^1 = \tau_k^1 = 0$, $\beta_k^1 = \frac{1}{M}$ for $k = 1, \ldots, M$

for $t = 1, 2, \ldots$ **do**

obtain SVM's $\boldsymbol{\alpha}^k$ with kernel

$$\mathrm{k}^t(\boldsymbol{s}_i, \boldsymbol{s}_j) := \sum_{k=1}^{M} \beta_k^t \, \mathrm{k}_k(\boldsymbol{s}_i, \boldsymbol{s}_j)$$

for $k = 1, \ldots, M$ **do**

$D_k^t = \frac{1}{2} \sum_{r,s} \alpha_r^t \alpha_s^t y_r y_s \, \mathrm{k}_k(\boldsymbol{s}_r, \boldsymbol{s}_s) - \sum_r \alpha_r^t$

end for

$D^t = \sum_{k=1}^{M} \beta_k^t D_k^t$

$\gamma_t = \mathrm{argmin}_{\gamma \in [0,1]} \sum_{k=1}^{M} \beta_k^t \exp\left\{\gamma(D_k - \rho^t)\right\}$

for $k = 1, \ldots, M$ **do**

$\tau_k^{t+1} = \tau_k^t + \gamma_t D_k^t$

$\beta_k^{t+1} = \beta_k^t \exp(\gamma_t D_k^t) / \left(\sum_{k'} \beta_{k'}^t \exp(\gamma_t D_{k'}^t)\right)$

end for

$\rho^{t+1} = \max_k \tau_k^{t+1} / \sum_{r=1}^{t} \gamma_r$

if $|1 - \frac{\rho^{t+1}}{D^t}| \leq \epsilon$ **then break**

end for