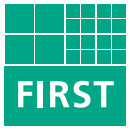
Topics I am working on V0.01-2004-04-14

Soeren.Sonnenburg@first.fraunhofer.de



Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

Roadmap

- how to spend time in the addnet project
  - getting any info out of the $9 + 2 + 2$ microarrays
  - proceeding on the sequence level of ≈ 17000 genes
  - combine expression with sequence level information

- Kernels for splice site recognition
  - TOP/Fisher Kernel
  - Histogram Kernel
  - Salzberg Kernel
  - Polynomial Kernel
  - Locality Improved (2 versions)
  - Weighted Degree
  - Weighted Degree Position
  - Spectrum Kernel

ADDNET

- getting any info out of the $9 + 2 + 2$ microarrays
  - Combining Mouse and Human Microarrays
    - find Mouse sequences in Human Microarrays
    - then use matching dimensions
  - Data normalization (what about BLANKs)

$$\min_{\alpha} \sum_{i,j} (\log(\cosh(\alpha_i \mathsf{array}(i) - \alpha_j \mathsf{array}(j))))$$

- proceeding on the sequence level of ≈ 17000 genes
  - find promotors and then TFBS in promotors ?
  - compute alignment scores ?

- combine expression level information with dna

Kernel Machines

Kernel Machine, e.g. SVM

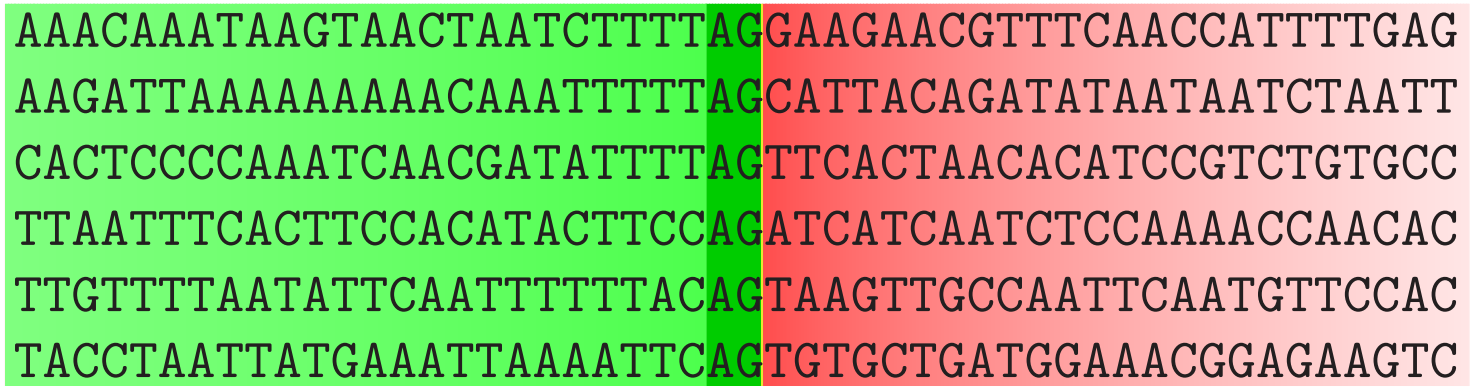
- sequences $\boldsymbol{x}_i \in \mathbb{X}$ $(i = 1, \ldots, \ell)$ with respective labels y_i
- SVM classifier:

$$f(\boldsymbol{x}) = \text{sign}\left(\sum_{i=1}^{\ell} y_i \alpha_i \text{k}(\boldsymbol{x}, \boldsymbol{x}_i) + b\right)$$

Kernel is key ingredient

- tune to incorporate prior knowledge
- determines feature space

Splice Data



- conserved sequence, position dependent
- mutation
- insertion
- deletion
- moved

TOP/Fisher/Histogram Kernel

- Kernels from generative models
  - compare objects using a generative model $\Pr(\boldsymbol{x}|\Theta)$
  - exploit probabilistic model for discriminative training

Fisher Kernel (Jaakkola and Haussler, 1998)

$$\begin{aligned} \mathrm{k_{FK}}(\boldsymbol{x},\boldsymbol{x}') &= \boldsymbol{s}(\boldsymbol{x},\hat{\boldsymbol{\theta}})^\top Z^{-1}(\hat{\boldsymbol{\theta}})\boldsymbol{s}(\boldsymbol{x}',\hat{\boldsymbol{\theta}}) \\ \boldsymbol{s}(\boldsymbol{x},\hat{\boldsymbol{\theta}}) &= \nabla_{\boldsymbol{\theta}} \log \Pr(\boldsymbol{x}|\hat{\boldsymbol{\theta}}) \qquad \text{Fisher score vector} \\ Z(\hat{\boldsymbol{\theta}}) &= \mathrm{E}_{\boldsymbol{x}}\left[\boldsymbol{s}(\boldsymbol{x},\hat{\boldsymbol{\theta}})\boldsymbol{s}(\boldsymbol{x},\hat{\boldsymbol{\theta}})^\top \middle| \hat{\boldsymbol{\theta}}\right] \qquad \text{Fisher information matrix} \end{aligned}$$

Kernel (Tsuda et. al, 2002)

$$\begin{aligned} \mathrm{k_{TOP}}(\boldsymbol{x},\boldsymbol{x}') &= \boldsymbol{f}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x})^\top \boldsymbol{f}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}') \\ \boldsymbol{f}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}) &= (v(\boldsymbol{x},\hat{\boldsymbol{\theta}}), \nabla_\theta\, v(\boldsymbol{x},\hat{\boldsymbol{\theta}}))^\top \\ v(\boldsymbol{x},\hat{\boldsymbol{\theta}}) &= \log(\Pr(y=+1|\boldsymbol{x},\hat{\boldsymbol{\theta}})) - \log(\Pr(y=-1|\boldsymbol{x},\hat{\boldsymbol{\theta}})) \end{aligned}$$

FIRST

Histogram Kernel

- it is just a TOP Kernel from a Markov chain
- very fast compared with TOP from HMM

Computing partial derivatives becomes counting matches weighted with a constant

$$\partial_{\theta_{i,j}} v(\boldsymbol{x}, \boldsymbol{\theta}) = \mathrm{I}(x_i = j)/\theta_{i,j}$$

since the probabilistic model is just

$$\begin{aligned} P(\boldsymbol{x}|\boldsymbol{\theta}^{\pm}) &= P(x_1, \ldots, x_N|\boldsymbol{\theta}^{\pm}) \\ &= P(x_1, \ldots, x_\omega|\boldsymbol{\theta}^{\pm}) \prod_{i=\omega+1}^{N} P(x_i|x_{i-1}, \ldots, x_{i-\omega}, \boldsymbol{\theta}^{\pm}). \end{aligned}$$

Kernels - polynomial kernel

Polynomial Kernel of degree d:

$$\mathrm{k}_{\mathrm{POLY}}(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}, \boldsymbol{x}')^d$$

- not directly applicable
- common approach convert input into binary space, or equivalently count matches

$$\mathrm{k}_{\mathrm{POLY}}(\boldsymbol{x}, \boldsymbol{x}') = \left(\sum_{p=1}^{N} \mathsf{I}_p(\boldsymbol{x}, \boldsymbol{x}') \right)^d$$

$\Rightarrow$ Feature space all d-th order monomials, i.e.

$$x_{i_1} \cdot x_{i_2} \cdot \ldots \cdot x_{i_d}, \; i_{1\ldots d} \in 1 \ldots N$$

$\Rightarrow$ Ultra high dimensional, *not robust*

Kernels - locality improved kernel

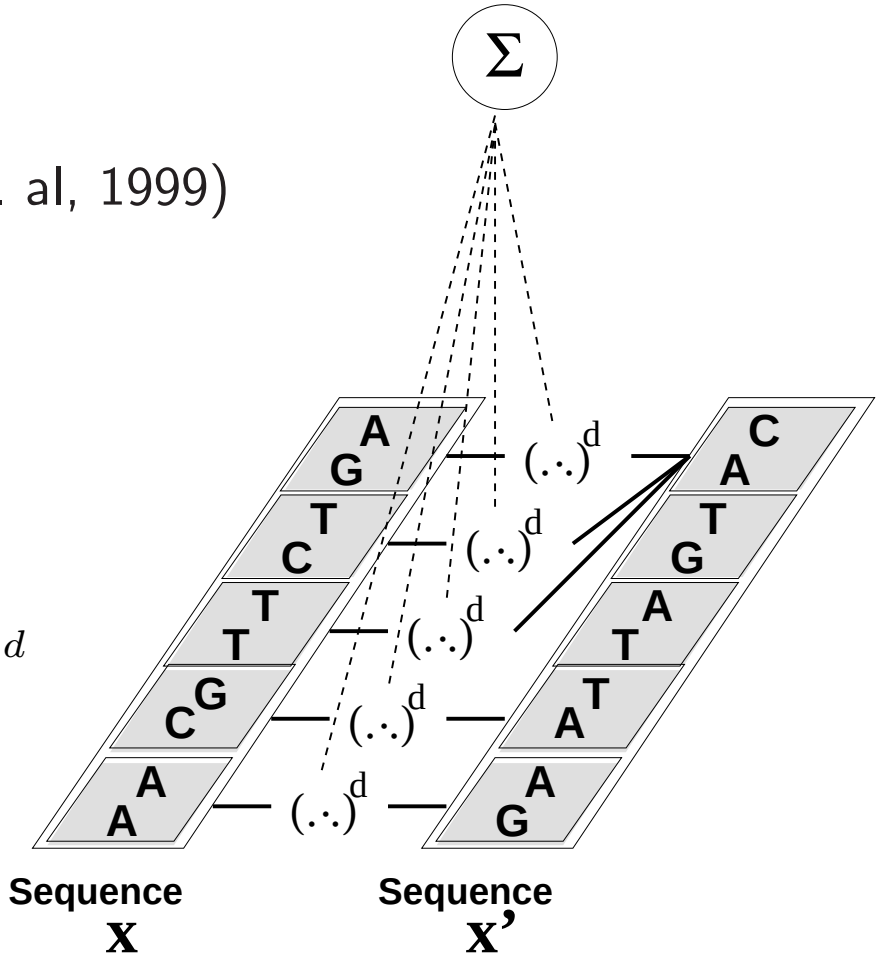
Locality Improved Kernel (Zien et. al, 1999)

- essentially polynomial kernel wich tries to emphasize local information

$$\mathrm{k_{LI}}(\boldsymbol{x}, \boldsymbol{x'}) = \sum_{p=1}^{N} \mathrm{win}_p(\boldsymbol{x}, \boldsymbol{x'})$$

$$\mathrm{win}_p(\boldsymbol{x}, \boldsymbol{x'}) = \left(\sum_{j=-l}^{+l} p_j \mathrm{I}_{p+j}(\boldsymbol{x}, \boldsymbol{x'}) \right)^d$$

$$\mathrm{I}_i(x, x') = \begin{cases} 1, & x_i = x'_i \\ 0, & \text{otherwise} \end{cases}$$



$\Rightarrow$ in the end, counts matches of $2l+1-$ tupels

Kernels - Simple locality improved kernel

Simple Locality Improved Kernel

- different weighting scheme, weight per window

$$\mathrm{k}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{p=l+1}^{N-l} w_p \mathrm{win}_p(\boldsymbol{x}, \boldsymbol{x}')$$

with

$$\mathrm{win}_p(\boldsymbol{x}, \boldsymbol{x}') = \left(\frac{1}{2l+1} \sum_{j=-l}^{+l} \mathrm{I}(x_{p+j} = x'_{p+j}) \right)^d$$

Weighted Degree Kernel

- count matches between two sequences $\boldsymbol{x}$ and $\boldsymbol{x}'$ between the words $\boldsymbol{u}_{\omega,i}(\boldsymbol{x})$ and $\boldsymbol{u}_{\omega,i}(\boldsymbol{x}')$
- $\boldsymbol{u}_{\omega,i}(\boldsymbol{x}) = x_i x_{i+1} \dots x_{i+\omega-1}$ for all i and $1 \leq \omega \leq d$.
- ω denotes the order (length of the word) to be compared.

Weighted Degree Kernel

$$\mathrm{k}_{WD}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{\omega=1}^{d} w_\omega \sum_{i=1}^{N-d} \mathrm{I}(\boldsymbol{u}_{\omega,i}(\boldsymbol{x}) = \boldsymbol{u}_{\omega,i}(\boldsymbol{x}'))$$

- w_ω denotes the weight for a match of length ω
- mismatches are also incorporated, resulting in lower weighting of the imperfect match

FIRST

Weighted Degree Position Kernel

$$\mathrm{k}_{WDpos}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{\omega=1}^{d} w_\omega \sum_{i=1}^{N-d-\mathsf{shift}_i} \sum_{j=1}^{\mathsf{shift}_i} \mathrm{I}(\boldsymbol{u}_{\omega,i+j}(\boldsymbol{x}) = \boldsymbol{u}_{\omega,i+j}(\boldsymbol{x}'))$$

Can cope with

- conserved sequence, position dependent
- mutation, insertion, deletion
- moved

Spectrum Kernel

Spectrum Kernel (Leslie et. al, 2002):

$$\mathrm{k}_{spec} = f(\boldsymbol{x})^\top f(\boldsymbol{x})$$

Inner product of

$$f(\boldsymbol{x}) = (\#A(\boldsymbol{x}))_{A \in \Sigma^k}$$

where $\#A(\boldsymbol{x}) =$ number of occurrences of A in $\boldsymbol{x}$.

- counts matches of k-mers
- no position dependend information