

# Positional Oligomer Importance Matrices

*(Feature Extraction & Interpretable SVMs)*

Sören Sonnenburg  
Fraunhofer FIRST.IDA, Berlin

joint work with  
*Alexander Zien, Petra Philips and Gunnar Rätsch*



**Fraunhofer** Institut  
Rechnerarchitektur  
und Softwaretechnik

# Outline

- 1 Introduction and Motivation
- 2 The Positional Oligomer Scoring System
- 3 POIMs
- 4 Applications
- 5 Discussion

# The Motivating Application - Sequence Classification

```

AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAATAAAAAAAAAACAAATTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
TTGTTTTAATATTCAATTTTTTACAGTAAGTTGCCAATTCAATGTTCCAC
TACCTAATTATGAAATTAATAATTCAGTGTGCTGATGGAAACGGAGAAGTC
  
```

## SVM+String kernel(s) state of the art in detecting

- Gene Start/End
- Splice Sites
- Trans-splicing, Alternative Splicing etc. etc.

**SVM sensitivity  $\approx 2$  times larger at same specificity**

**Drawback: We loose interpretability of the result!**

# Why are SVMs hard to interpret?

**Problem:** Learned  $\alpha$  **weighting of training points**

**But:** One is interested in **discriminating features**

$$\begin{aligned}
 f(\mathbf{x}) &= \sum_{i=1}^N \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b \\
 &= \underbrace{\sum_{i=1}^N y_i \alpha_i \Phi(\mathbf{x}_i)}_{\mathbf{w}} \cdot \Phi(\mathbf{x}) + b = \mathbf{w} \cdot \Phi(\mathbf{x}) + b
 \end{aligned}$$

**Idea:** Use SVMs  $\mathbf{w}$  vector to interpret features

# SVMs are interpretable - depending on feature space

## Condition:

### Feature space enumerable/meaningful/w storable

- linear SVMs
- most of string kernels ( $k$ -mer based)
  - spectrum kernel
  - WD kernels
  - ...

## Problems:

- Feature space may be very high dimensional
- Features not independent

## Idea:

- Compute expected SVM output given a certain feature



# The Scoring System - Definition

<b>k-mer</b>	<b>pos. 1</b>	<b>pos. 2</b>	<b>pos. 3</b>	<b>pos. 4</b>	<b>...</b>
<b>A</b>	+0.1	-0.3	-0.2	+0.2	...
<b>C</b>	0.0	-0.1	+2.4	-0.2	...
<b>G</b>	+0.1	-0.7	0.0	-0.5	...
<b>T</b>	-0.2	-0.2	0.1	+0.5	...
<b>AA</b>	+0.1	-0.3	+0.1	0.0	...
<b>AC</b>	+0.2	0.0	-0.2	+0.2	...
⋮	⋮	⋮	⋮	⋮	⋮
<b>TT</b>	0.0	-0.1	+1.7	-0.2	...
<b>AAA</b>	+0.1	0.0	0.0	+0.1	...
<b>AAC</b>	0.0	-0.1	+1.2	-0.2	...
⋮	⋮	⋮	⋮	⋮	⋮
<b>TTT</b>	+0.2	-0.7	0.0	0.0	...

$$s(\mathbf{x}) := \sum_{k=1}^K \sum_{i=1}^{n-k+1} w(\mathbf{x}[i]^k, i) + b$$

# The Scoring System - Examples

$$s(\mathbf{x}) := \sum_{k=1}^K \sum_{i=1}^{n-k+1} w(\mathbf{x}[i]^k, i) + b$$

## Examples:

- WD-kernel
- WD-kernel with shifts
- Spectrum kernel
- Oligo Kernel

## Not limited to SVMs:

- markov chains (higher order/inhomogeneous/mixed order)



# POIMs

## Idea:

- Compute expected score given that  $k$ -mer  $\mathbf{z}$  appears at position  $j$  in the sequence
- normalized with expected score over all sequences

$$C(\mathbf{z}, j) := \mathbb{E}[s(\mathbf{x}) \mid \mathbf{x}[j] = \mathbf{z}] - \mathbb{E}[s(\mathbf{x})]. \quad (1)$$

## Problem:

- Choosing a background distribution (uniform, 0-th order MC)
- Naive approach already for short sequences and small alphabets infeasible
- $\mathbf{w}$  may be stored in some sparse data structure (like a tree/forest)

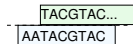
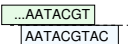
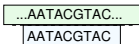
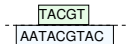
**Needs efficient algorithm for computation**

# Observations

$$\begin{aligned}
 C(\mathbf{z}, j) &:= \mathbb{E}[s(\mathbf{x}) \mid \mathbf{x}[j] = \mathbf{z}] - \mathbb{E}[s(\mathbf{x})] \\
 &= \sum_{(\mathbf{y}, i) \in \mathcal{I}} w(\mathbf{y}, i) [Pr(\mathbf{x}[i] = \mathbf{y} \mid \mathbf{x}[j] = \mathbf{z}) - Pr(\mathbf{x}[i] = \mathbf{y})] \\
 &= u(\mathbf{z}, j) + \sum_{\mathbf{y} \in \Sigma^{|\mathbf{z}|}} u(\mathbf{z}, j)
 \end{aligned}$$

- number of  $k$ -mers grows only linear with data
- all features which are independent of  $(\mathbf{z}, j)$  vanish
- computation can be split in computing contributions from 4 cases:

$$\begin{aligned}
 u(\mathbf{z}, j) &:= \sum_{(\mathbf{y}, i) \in \mathcal{I}(\mathbf{z}, j)} Pr(\mathbf{x}[i] = \mathbf{y} \mid \mathbf{x}[j] = \mathbf{z}) w(\mathbf{y}, i) \\
 &= u^{\vee}(\mathbf{z}, j) + u^{\wedge}(\mathbf{z}, j) + u^{<}(\mathbf{z}, j) + u^{>}(\mathbf{z}, j) - w(\mathbf{z}, j) ,
 \end{aligned}$$



For AATACGTAC: substring, superstring, left and right partial overlap

# Efficient Computation

$$u^{\vee}(\mathbf{z}, j) = w(\mathbf{z}, j) + u^{\vee}(\tau\mathbf{z}', j) + u^{\vee}(\mathbf{z}'\tau', j+1) - u^{\vee}(\mathbf{z}', j+1)$$

$$u^{\wedge}(\mathbf{z}, j) = w(\mathbf{z}, j) - \sum_{(\sigma, \sigma') \in \Sigma^2} Pr(\mathbf{x}[j + |\mathbf{z}|] = \sigma') Pr(\mathbf{x}[j - 1] = \sigma) u^{\wedge}(\sigma\mathbf{z}\sigma', j - 1) \\ + \sum_{\sigma \in \Sigma} Pr(\mathbf{x}[j - 1] = \sigma) u^{\wedge}(\sigma\mathbf{z}, j - 1) + \sum_{\sigma' \in \Sigma} Pr(\mathbf{x}[j + |\mathbf{z}|] = \sigma') u^{\wedge}(\mathbf{z}\sigma', j)$$

$$u^{<}(\mathbf{z}, j) = \sum_{\sigma \in \Sigma} Pr(\mathbf{x}[j - 1] = \sigma) \sum_{k=1}^{|\mathbf{z}|-1} L(\sigma\mathbf{z}[1]^k, j - 1)$$

$$u^{>}(\mathbf{z}, j) = \sum_{\sigma \in \Sigma} Pr(\mathbf{x}[j + |\mathbf{z}|] = \sigma) \sum_{k=1}^{|\mathbf{z}|-1} R(\mathbf{z}[|\mathbf{z}| - k + 1]^k \sigma, j + |\mathbf{z}| - k) ,$$

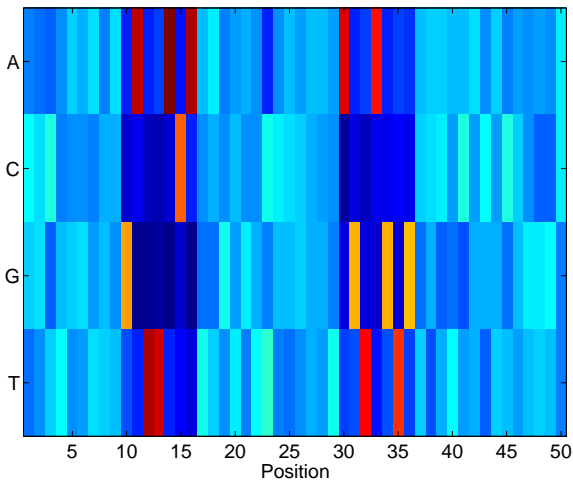
where

$$L(\mathbf{t}, j) := \sum_{(\mathbf{y}, i) \in \mathcal{L}(\mathbf{t}, j)} Pr(\mathbf{x}[i] = \mathbf{y} \mid \mathbf{x}[j] = \mathbf{t}) w(\mathbf{y}, i)$$

$$R(\mathbf{t}, j) := \sum_{(\mathbf{y}, i) \in \mathcal{R}(\mathbf{t}, j)} Pr(\mathbf{x}[i] = \mathbf{y} \mid \mathbf{x}[j] = \mathbf{t}) w(\mathbf{y}, i) .$$

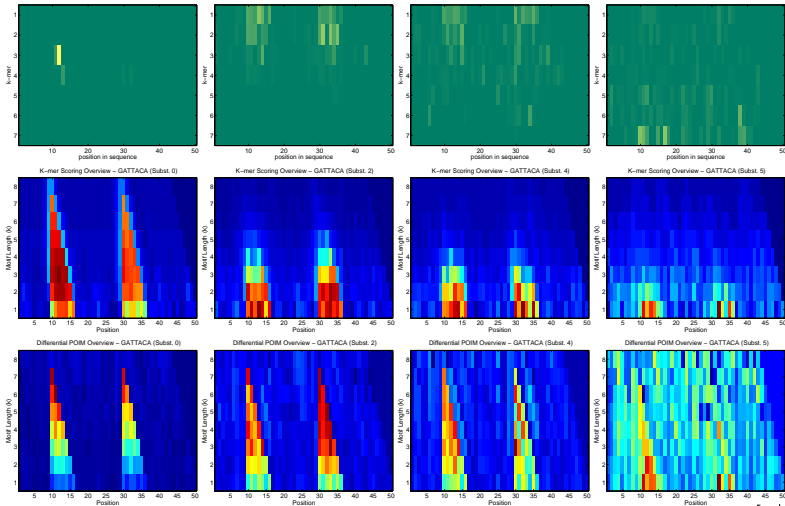
# GATTACA and AGTAGTG at fixed positions 10 and 30

POIM – GATTACA (Subst. 0) Order 1



Comparison with MKL and w

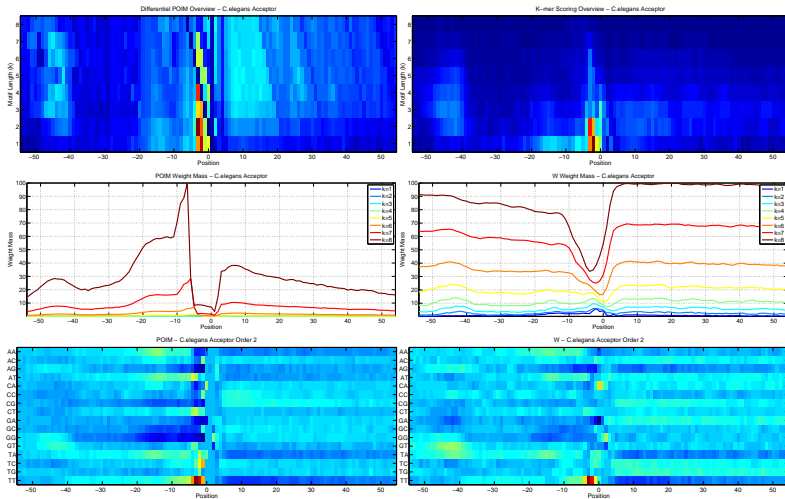
# GATTACA and AGTAGTG at fixed positions 10 and 30





## Real World Problem

# Acceptor Splice Site Recognition



# Open Problems

## Need your help

- **Motif detection**
  - GCGCG vs CGCGC - how to merge?
  - strong motifs dominate weaker ones - how to cluster?
- **Correction necessary for repetitive motifs?**
  - z scores high **zz** will score higher
- **Consensus sequence  $\mathbf{x}^* := \operatorname{argmax}_{\mathbf{x}} s(\mathbf{x})$**  - Is it meaningful?



# Conclusions

## Positional Oligomer Importance Matrices

- developed a method which systematically computes the importances of positional motifs for the expected decision score
  - useful to rank motifs and for visualization
  - applicable for a large class of popular scores (SVM+spec/WD/oligo kernel; markov chain)
  - efficiently implemented for spectrum and WD kernels in <http://www.shogun-toolbox.org>
- nice results on toy and real world data