Positional Oligomer Importance Matrices

(Feature Extraction & Interpretable SVM's)

Sören Sonnenburg
Fraunhofer FIRST.IDA, Berlin

joint work with
Alexander Zien, Petra Philips and Gunnar Rätsch

FIRST

Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

Outline

1. Introduction and Motivation
2. Methods
3. Applications
4. Discussion

The Motivating Application - Sequence Classification

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
TTGTTTTAATATTCAATTTTTTACAGTAAGTTGCCAATTCAATGTTCCAC
TACCTAATTATGAAATTAAAATTCAGTGTGCTGATGGAAACGGAGAAGTC
```

SVM with string-kernels: State-of-the-art in detecting

- Gene Start/End (Sonnenburg, Zien, Rätsch, 2005)
- Splice Sites (Sonnenburg, Schweikert, Philips, et.al. 2007)
- Trans-splicing, Alternative Splicing (Rätsch, Sonnenburg, Schölkopf 2005)

SVM sensitivity ≈ 2 **times larger at same specificity**

Drawback: We loose interpretability of the result!

Why are SVM's hard to interpret?

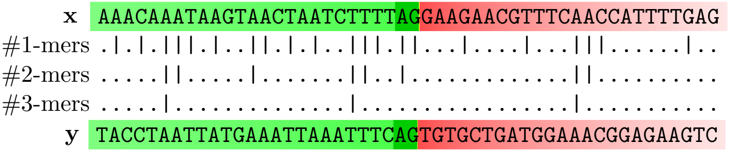
Problem: Learned α weighting of training points
But: One is interested in discriminating features

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

Idea: Use SVM's w vector to interpret features

The Weighted Degree Kernel

$$\mathrm{k}(\mathbf{x},\mathbf{x}') = \sum_{k=1}^{K} \beta_k \sum_{i=1}^{N-k+1} \mathbb{I}\left\{\mathbf{x}[i]^k = \mathbf{x}[i]^k\right\}.$$



```
x        AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
#1-mers  .|.|.|||.|..||.|.|..|||.||...|....|...|||......|..
#2-mers  .....||.....|.......||..|.............||..........
#3-mers  .....|..............|.................|...........
y        TACCTAATTATGAAATTAAATTTCAGTGTGCTGATGGAAACGGAGAAGTC
```

Example: $K = 3: \mathrm{k}(\mathbf{x},\mathbf{x}') = \beta_1 \cdot 21 + \beta_2 \cdot 8 + \beta_3 \cdot 3$

The SVM normal vector $\mathbf{w}$

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

SVM's are interpretable - depending on feature space

Condition:

Feature space enumerable & meaningful, explicitly store w

- linear SVM's
- most of string kernels ($k-$mer based)
  - Spectrum kernel (Leslie, Eskin, Noble, 2002)
  - WD kernels (Rätsch, Sonnenburg, 2005)
  - ...

Problems:

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

Idea:

- Compute expected SVM output given a certain feature

The Scoring System - Definition

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

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

The Scoring System - Examples

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

Examples:

- WD-kernel (Rätsch, Sonnenburg, 2005)
- WD-kernel with shifts (Rätsch, Sonnenburg, 2005)
- Spectrum kernel (Leslie, Eskin, Noble, 2002)
- Oligo Kernel (Meinicke et.al., 2004)

Not limited to SVM's:

- Markov Chains (higher order/inhomogeneous/mixed order)

POIMs

Idea:

- Compute expected score $C(\mathbf{z}, j)$ given that $k-$mer $\mathbf{z}$ appears at position j in the sequence for small k
- Normalize with *expected score* over all sequences

$$C(\mathbf{z}, j) \ := \ \mathbb{E}\left[\, s(\mathbf{x}) \,|\, \mathbf{x}\,[j] = \mathbf{z} \,\right] - \mathbb{E}\left[\, s(\mathbf{x}) \,\right] \qquad (1)$$

Problem:

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

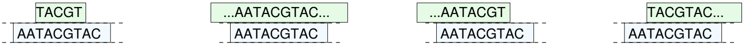
Needs efficient algorithm for computation

Observations

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

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

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



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

Efficient Computation

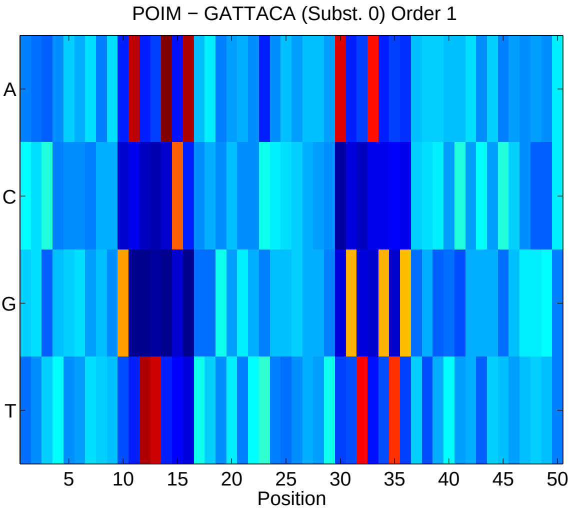
(Zien, Philips, Sonnenburg, 2007)

$$
\begin{aligned}
u^{\vee}(\mathbf{z},j) &= w(\mathbf{z},j) + u^{\vee}(\tau\mathbf{z}',j) + u^{\vee}(\mathbf{z}'\tau',j+1) - u^{\vee}(\mathbf{z}',j+1) \\
u^{\wedge}(\mathbf{z},j) &= w(\mathbf{z},j) - \sum_{(\sigma,\sigma')\in\Sigma^2} Pr\left(\mathbf{x}[j+|\mathbf{z}|]=\sigma'\right) Pr\left(\mathbf{x}[j-1]=\sigma\right) u^{\wedge}(\sigma\mathbf{z}\sigma',j-1) \\
&+ \sum_{\sigma\in\Sigma} Pr\left(\mathbf{x}[j-1]=\sigma\right) u^{\wedge}(\sigma\mathbf{z},j-1) + \sum_{\sigma'\in\Sigma} Pr\left(\mathbf{x}[j+|\mathbf{z}|]=\sigma'\right) u^{\wedge}(\mathbf{z}\sigma',j) \\
u^{<}(\mathbf{z},j) &= \sum_{\sigma\in\Sigma} Pr\left(\mathbf{x}[j-1]=\sigma\right) \sum_{k=1}^{|\mathbf{z}|-1} L(\sigma\mathbf{z}[1]^k,j-1) \\
u^{>}(\mathbf{z},j) &= \sum_{\sigma\in\Sigma} Pr\left(\mathbf{x}[j+|\mathbf{z}|]=\sigma\right) \sum_{k=1}^{|\mathbf{z}|-1} R(\mathbf{z}[|\mathbf{z}|-k+1]^k\sigma,j+|\mathbf{z}|-k) \ ,
\end{aligned}
$$

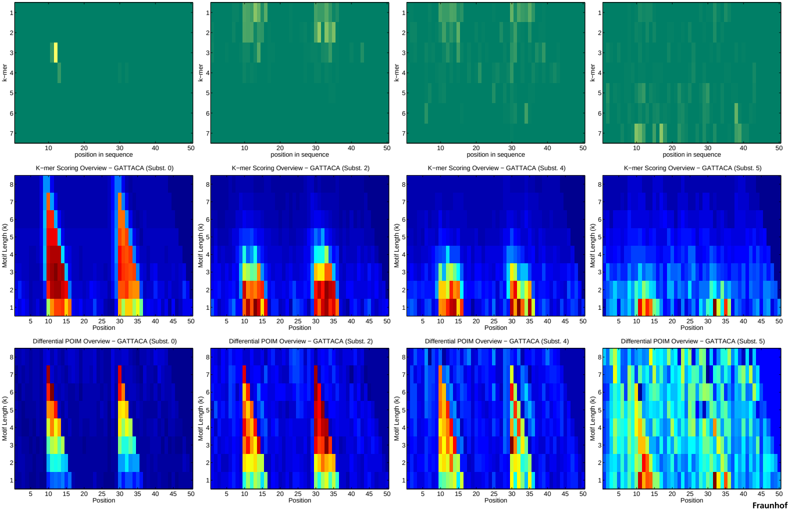
where

$$
\begin{aligned}
L(\mathbf{t},j) &:= \sum_{(\mathbf{y},i)\in\mathcal{L}(\mathbf{t},j)} Pr\left(\mathbf{x}\,[i]=\mathbf{y}\mid\mathbf{x}\,[j]=\mathbf{t}\right) w(\mathbf{y},i) \\
R(\mathbf{t},j) &:= \sum_{(\mathbf{y},i)\in\mathcal{R}(\mathbf{t},j)} Pr\left(\mathbf{x}\,[i]=\mathbf{y}\mid\mathbf{x}\,[j]=\mathbf{t}\right) w(\mathbf{y},i) \ .
\end{aligned}
$$

GATTACA and AGTAGTG at fixed positions 10 and 30



GATTACA and AGTAGTG at fixed positions 10 and 30



K−mer Scoring Overview − GATTACA (Subst. 0)

K−mer Scoring Overview − GATTACA (Subst. 2)

K−mer Scoring Overview − GATTACA (Subst. 4)

K−mer Scoring Overview − GATTACA (Subst. 5)

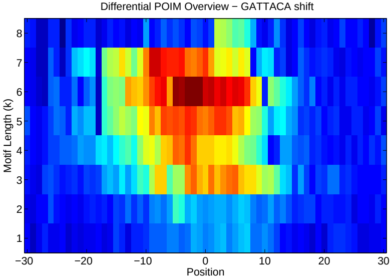
Differential POIM Overview − GATTACA (Subst. 0)

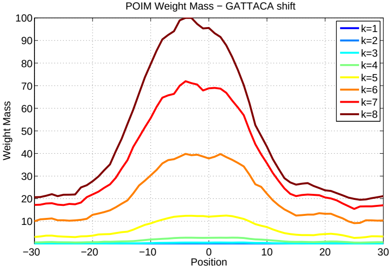
Differential POIM Overview − GATTACA (Subst. 2)

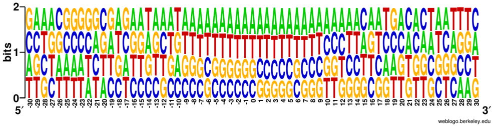
Differential POIM Overview − GATTACA (Subst. 4)

Differential POIM Overview − GATTACA (Subst. 5)

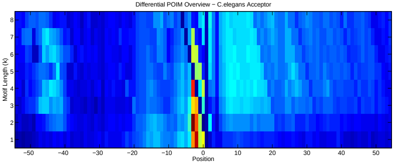
GATTACA at variable positions

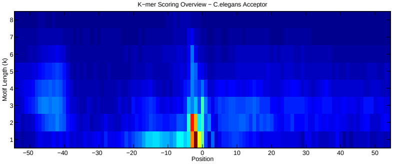


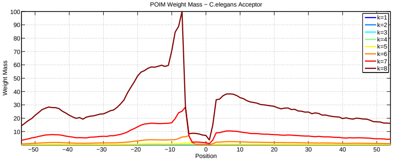


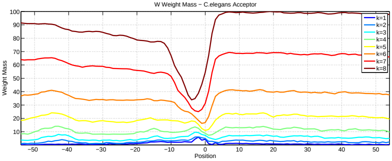


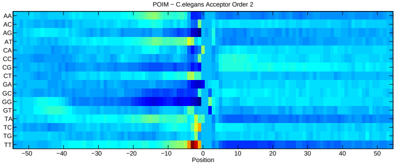
C.elegans Acceptor Splice Site Recognition

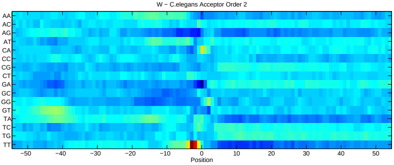

Differential POIM Overview − C.elegans Acceptor


K−mer Scoring Overview − C.elegans Acceptor

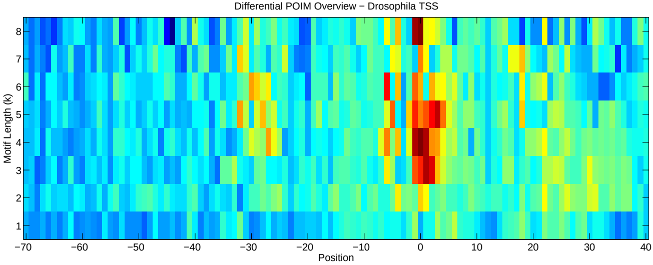

POIM Weight Mass − C.elegans Acceptor

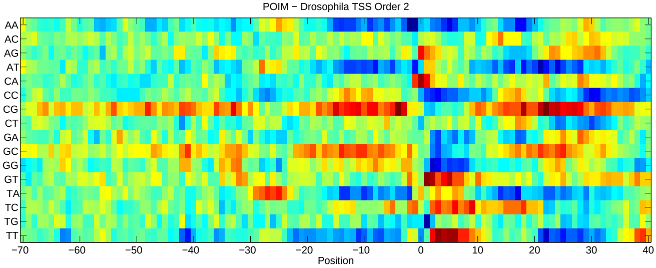

W Weight Mass − C.elegans Acceptor


POIM − C.elegans Acceptor Order 2


W − C.elegans Acceptor Order 2

Drosophila Transcription Starts





Conclusions

Positional Oligomer Importance Matrices

- Developed a method which systematically computes the importances of positional motifs for the expected decision score
    - Feature Ranking in **Feature space**
    - 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

Machine Learning Open Source Software

To support the open source movement JMLR is proud to announce a new track on machine learning open source software.

Contributions to http://jmlr.org/mloss/ should be related to

- Implementations of machine learning algorithms,
- Toolboxes,
- Languages for scientific computing

and should include

- A 4 page description,
- The code,
- A recognised open source license.

Contribute to http://mloss.org the mloss repository!