Kernel methods for integrating biological data

Dick de Ridder, Marc Hulsman & Bastiaan van den Berg
The Delft Bioinformatics Lab, Delft University of Technology
Biotechnology

*Saccharomyces cerevisiae*
- alcohol

*Aspergillus niger*
- citric acid

*Penicillium chrysogenum*
- penicillin

*Lactococcus lactis*
- cheese
Industrial protein production

- Micro-organisms can be used as “cell factories”, genetically modified to produce e.g. specific proteins

- For industrial application, proteins should be:
  - (highly) expressed
    - introduce gene in genome
    - place a strong promoter sequence in front of gene
  - secreted
Industrial protein production (2)

- Billion-dollar industry
- Continuous search for new products: from lab to plant

- Test phase is tedious and costly
- Can we predict what proteins can be successfully expressed?
A bioinformatics problem

• Expression is relatively easy, secretion is hard to get right

• Basic machinery is known, but...
  • signal sequences are not unique
  • many alternative routes
  • lack of knowledge on *heterologous* protein expression
A pattern recognition problem

- When all else (model) fails, turn to pattern recognition
A pattern recognition problem (2)

• Learn from experience:
  dataset of 683 proteins for which secretion was attempted

• Required elements for a pattern recognition approach:
  • Objects: proteins
  • Labels: detected secretion at relatively high level (gel)
  • Target: ...
  • Features: ...

Bastiaan van den Berg
TB1, Thu 11.15
What is the target?

- Predictions need to be experimentally tested: *prioritize*

- Criterion: (partial) area under the ROC curve
What is a good representation?

• Available: DNA sequences

ACTGACCTATAAGCG...

M P L I V ...

• Sequence characteristics

  • Composition:
    • length
    • nucleotide/amino acid composition
    • amino acid subset composition (basic, charged, …)
  
  • Derived:
    • codon adaptation index (DNA)
    • hydrophobicity/philicity (protein)
What is a good representation (2)?

- Protein characteristics (predicted)
  - presence of signal sequence
  - subcellular localization
  - protein function
  - isoelectric point
- Heterologous proteins
  - relation to host organism
- … (whatever works)

- Heterogeneous sources of information (prior knowledge) and data (measurements) need to be integrated
INTEGRATIVE BIOINFORMATICS

KERNEL-BASED ALGORITHMS

KERNELS

KERNEL COMBINATION

EXAMPLE APPLICATIONS
Integrative bioinformatics

- Construct and interpret networks of biochemical interactions in a living cell, making use of all available data and prior knowledge
Molecular interaction networks

Gene A → mRNA A → Protein A
Gene B → mRNA B → Protein B
Gene C → mRNA C → Protein C
Gene D → mRNA D → Protein D
Gene E → miRNA E

Transcription regulation
RNA interference
Metabolic control
Complex formation
Protein interaction
Protein activation
Signaling
Other cells

Enzyme A → Metabolite 1
Enzyme B → Metabolite 2

Complex CD

Protein F
Focus

• Genetic interactions:
  • Transcription regulation etc. (genes cannot interact)
  • Catch-all term used in functional genomics

• Protein-protein interaction:
  • Signalling
  • Transport
  • Complex formation

Tong et al.
Science 2001
Measurements

- **-omics:**
  - sequences
  - transcripts (mRNA)
  - metabolic fluxes
  - protein/metabolite levels
  - protein location
  - protein-protein interaction
  - protein-DNA interaction
  - synthetic sick-or-lethal
  - phenome (conditions)

- Much in (curated) databases

Prior knowledge

- Names, annotations, pathways, reactions, literature…
Pattern recognition for integration

- Predicting interactions is really a classification problem
  - input: measurements/data \( x = (x_1, x_2, \ldots, x_n) \)
  - output: presence of interaction/function \( y \in \{0, 1\} \)

![Diagram showing pattern recognition process](attachment:image.png)
Early integration

- Feature fusion: the standard approach

\[
x_1 \quad x_2 \quad x_3 \quad \cdots \quad x_n \quad \text{Classifier} \quad y
\]

- Usually weighted (nonlinear) combination, optimised w.r.t. target
Late integration

- Classifier combination

- By fixed rule (max, min, …) or by trained combiner
Pitfalls

• Early integration:
  • need to convert all features to single representation, e.g. by binning
    (but: how to do this for sequences, graphs, …?)

• Late integration:
  • choosing combination mechanism
  • integration of different classifiers not straightforward
  • knowledge of data heterogeneity unused
Intermediate integration

- Transform characteristics into a “common language”, an intermediate representation suitable for integration

Pavlidis et al.
*J Comp Biol* 2002

- Probabilities
- (Dis)similarities
- Kernels
Integrating probabilities

- For example, through Bayesian networks...

Troyanskaya et al. PNAS 2003

Jansen et al. Science 2004
Integrating probabilities

• ...or by combining $p$-values or test scores

$$F_w = -2 \sum_{i=1}^{n} w_i \ln(p_i)$$

$$H_0 : F_w \sim \chi^2(2)$$

Hwang et al. 
*PNAS* 2005
Integrating (dis)similarities

• For example, by adding distance matrices
Integrating kernels

- For example, by adding kernels – well-founded similarities

- amino acid comp.
- codon adaptation
- signal sequence
- physiochemical

linear kernel
RBF kernel
weighted degree kernel
polynomial kernel

SVM

y
INTEGRATIVE BIOINFORMATICS

KERNEL-BASED ALGORITHMS

KERNELS

KERNEL COMBINATION

EXAMPLE APPLICATIONS
Support vector machines

\[ f(z) = w^T z + b \]

\[
\begin{align*}
\min & \|w\|^2 \\
\text{s.t.} & \quad w^T x_i + b \geq +1, \quad y = +1 \\
& \quad w^T x_i + b \leq -1, \quad y = -1
\end{align*}
\]
Support vector machines

- Slack variables: allow misclassifications on training set

\[
f(z) = w^T z + b
\]

\[
\min \|w\|^2 + C \sum_{i=1}^{X} \xi_i
\]

\[
f(z) \geq 1 - \xi_i \quad w^T x_i + b \geq +(1 - \xi_i), \quad y = +1
\]

\[
w^T x_i + b \leq -(1 - \xi_i), \quad y = -1
\]

\[
\xi_i \geq 0, \quad \forall i
\]
Support vector machines (2)

- Rewrite (original) optimisation problem (dual):

\[ f(z) = w^T z + b \]

\[ = \sum_{i=1}^{|X|} \alpha_i y_i x_i^T z + b \]

\[ \max_{\alpha} \sum_{i=1}^{|X|} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{|X|} y_i y_j \alpha_i \alpha_j x_i^T x_j \]

\[ 0 \leq \alpha_i \leq C, \quad \forall i \]

\[ \sum_{i=1}^{X} \alpha_i y_i = 0 \]
Support vector machines (3)

- Map input space into feature space using $\Phi$:

$$f(z) = w^T z + b$$

$$= \sum_{i=1}^{X} \alpha_i y_i \Phi(x_i)^T \Phi(z) + b$$

$$\max_{\alpha} \sum_{i=1}^{X} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{X} y_i y_j \alpha_i \alpha_j \Phi(x_i)^T \Phi(x_j)$$

$$0 \leq \alpha_i \leq C, \quad \forall i$$

$$\sum_{i=1}^{X} \alpha_i y_i = 0$$
Support vector machines (4)

- Replace inner product by kernel (similarity) function:

\[
\begin{align*}
  f(z) &= w^T z + b \\
  &= \sum_{i=1}^{X} \alpha_i y_i K(x_i, z) + b \\
  \max_{\alpha} \sum_{i=1}^{X} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{X} y_i y_j \alpha_i \alpha_j K(x_i, x_j) \\
  0 \leq \alpha_i \leq C, \quad \forall i \\
  \sum_{i=1}^{X} \alpha_i y_i &= 0
\end{align*}
\]
Feature space

- Function $\Phi$ maps data into space in which classification may be easier.

$$x = \begin{bmatrix} x_1, x_2 \end{bmatrix}$$

$$\Phi(x) = \begin{bmatrix} x_1^2, x_2^2, \sqrt{2}x_1x_2 \end{bmatrix}$$
Kernels

- Kernels $K(a, b) = \Phi(a)^T \Phi(b)$: using the same algorithm, obtain a nonlinear classifier in original space.

$$K(a, b) = (a^T b + 1)^d$$

$$K(a, b) = \exp \left( -\frac{\|a - b\|^2}{\sigma^2} \right)$$
Kernels (2)

- Not necessary to actually *know* $\Phi(.)$ to construct $K(x,y)$! Any kernel function is valid if it is *positive definite*, i.e. if for any input the resulting kernel matrix $K$ is positive definite ($z^T K z > 0$, $\forall z \in \mathbb{R}^n \neq 0$)

- If $K$ is not positive definite: empirical kernel map (later)

- Other classifiers can be written in terms of inner products and similarly be “kernelised”: kernel nearest mean classifier, kernel $k$-nearest neighbour, kernel LDA (Fisher), …
Training

- Optimisation of $w, b$ for SVM is a convex problem: can use standard solvers

- Find other parameters by grid search, using cross-validation error estimate
  - Trade-off parameter $C$ (at least for SVC)
  - Kernel parameters: $d, \sigma, \ldots$
Support vector regression

- Regressor: \( y = w^T x + b \)

- Loss function: \( \varepsilon \)-insensitive loss,
  \[
  \xi = \begin{cases} 
  0 & |\xi| \leq \varepsilon \\
  |\xi| - \varepsilon & \text{otherwise}
  \end{cases}
  \]

- Optimization problem:
  \[
  \min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{l} (\xi_i + \xi_i^*)
  \]
  \[
  y_i - w^T x_i - b \leq \varepsilon + \xi_i
  \]
  \[
  w^T x_i + b - y_i \leq \varepsilon + \xi_i^* \quad \forall i
  \]
  \[
  \xi_i, \xi_i^* \geq 0
  \]
Support vector regression (2)

Smola & Schölkopf, 1998

ε = 0.1

ε = 0.2

ε = 0.5
Kernel clustering

- Hierarchical clustering using kernel matrices
- Kernel $k$-means
- Kernel MDS
  (= kernel PCA)
- ...

Schölkopf et al., 1996
Kernel dimensionality reduction

- Example: principal component analysis, polynomial kernel

- Similarly:
  - kernel LDA
  - kernel CCA
  - ...

Schölkopf et al., 1996
INTEGRATIVE BIOINFORMATICS

KERNEL-BASED ALGORITHMS

KERNELS

KERNEL COMBINATION

EXAMPLE APPLICATIONS
Vector kernels

- Linear
  \[ K(a, b) = \langle a, b \rangle^d \]

- Polynomial
  \[ K(a, b) = \left( \langle a, b \rangle + 1 \right)^d \]

- Radial basis function
  \[ K(a, b) = \exp \left( - \frac{\|a - b\|^2}{\sigma^2} \right) \]
Protein secretion prediction

- Support vector classifier, 10-fold cross-validation error estimate

- Feature subsets
  - Sequence: amino-acid composition
  - Derived: hydrophobic & hydrophilic peaks
  - Other: length, CAI, pI, ...

[Graph showing ROC curves for different feature sets and kernels, including 89.73 All, linear, 90.55 All, RBF, 90.90 Sequence, RBF, 84.74 Derived, RBF, and 85.36 Other, RBF]
Empirical kernel map

• For almost any other data type: empirical kernel map
  • Any distance measure (not necessarily positive definite) can be used to construct a vector with distances to a number of other objects (the “template set”), e.g. BLAST -log(E)-values to all proteins
  • This vector can then be used in a vector kernel:

\[
\begin{align*}
\text{Template set} \\
&x_1 \quad x_2 \quad x_3 \quad x_4 \quad \ldots \quad x_n \\
\begin{bmatrix}
D_{a1} \\
D_{a2} \\
D_{a3} \\
\vdots \\
D_{an}
\end{bmatrix} &= a' \\
\begin{bmatrix}
D_{b1} \\
D_{b2} \\
D_{b3} \\
\vdots \\
D_{bn}
\end{bmatrix} &= b',
\end{align*}
\]

\[
K(a, b) = \langle a', b' \rangle
\]
Kernel kernels

- Kernel addition
  \[ K(a, b) = \sum_{i=1}^{k} w_i K_i(a, b), \quad w_i > 0 \quad \forall i \]

- Kernel pointwise multiplication
  \[ K(a, b) = \prod_{i=1}^{k} K_i(a, b) \]

- Generalized RBF kernel
  \[ K(a, b) = 1 + \exp\left(-\frac{D(a, b)}{2\sigma^2} - \frac{K'(a, a) - 2K'(a, b) + K'(b, b)}{2\sigma^2}\right) \]

- Kernel normalization
  \[ K(a, b) = \frac{K'(a, b)}{\sqrt{K'(a, a)K'(b, b)}} \]
Kernel kernels (2)

• Convolution kernel:
  • When subkernels operate on subparts, but it is not clear which subparts
  • Try all possible decompositions into subparts:

\[ K_1 \otimes K_2 \otimes \ldots \otimes K_n (a, b) = \sum_{a=a_1a_2\ldots a_n}^{a_n} K_1 (a_1, b_1)K_2 (a_2, b_2)\ldots K_n (a_n, b_n) \]
Local alignment kernel

\[ K_{la}(a, b) = \sum_{n=0}^{\infty} K_{la(n)}(a, b) \]

\[ K_{la(n)}(a, b) = K_t \otimes \left( K_a \otimes K_g \right)^{(n-1)} \otimes K_a \otimes K_t(a, b) \]

Trivial kernel:
\[ K_t(a, b) = 1 \]

Gap kernel
\[ K_g(a, b) = \exp(\beta(|a| + |b|)) \]

Letter alignment kernel:
\[ K_a(a, b) = \begin{cases} 0 & |a| > 1 \lor |b| > 1 \\ \exp(\beta S(a, b)) & \text{otherwise} \end{cases} \]

with \( S \) the substitution cost, e.g. BLOSUM
Kernel kernels (3)

- **Pairwise kernel:**
  - Kernel between pairs of objects rather than individual ones
  - Alternative to linear vector kernel on pair kernels

![Diagram of pairwise kernel](image)

**Protein similarity kernel:**
\[
k_s(q_{ij}, q_{km}) = \langle k_{ps}(p_i, p_j), k_{ps}(p_k, p_m) \rangle
\]

**Pairwise kernel (similarity between pairs):**
\[
k_{pv}(q_{ij}, q_{km}) = k_{ps}(p_i, p_k)k_{ps}(p_j, p_m) + k_{ps}(p_i, p_m)k_{ps}(p_j, p_k)
\]
Set kernels

- Let $\mu(A)$ be a probability distribution over sets $A$ on a domain $D$ (for example $\mu(A) = I_A$, the indicator function)

- Intersection kernel:
  \[
  K \cap (A, B) = \int_D I_A(a)I_B(a)d\mu(a)
  \]

- Union complement kernel:
  \[
  \tilde{K} (A, B) = \int_D I_{D \setminus A}(a)I_{D \setminus B}(a)d\mu(a)
  \]

- Agreement kernel:
  \[
  K (A, B) = \tilde{K} (A, B) + K \cap (A, B)
  \]

- Example: documents represented as sets of words; $\mu(A)$ can be measure of “uniqueness” of word
String kernels

• Spectrum kernel:
  • create a dictionary of all $k$-mers
  • construct vector with #occurrences of each $k$-mer
  • use this in a linear kernel

• Similar:
  • versions with gaps, mismatches
  • mixed spectrum kernel, sum over all $k = 1, \ldots, d$
  • motif kernel, look for specific set(s) of $k$-mers

• Example, $k = 4$: $a = \text{aabbababa}$ $b = \text{abbaabbbab}$

\[
\begin{array}{cccccc}
\text{aabb} & \text{abba} & \text{bbab} & \text{baba} & \text{abab} & \text{baa}\n
a & 1 & 1 & 1 & 2 & 1 & 0 & 0

b & 1 & 2 & 1 & 0 & 0 & 1 & 1
\end{array}
\]

\[K(a, b) = 4\]
String kernels (2)

- Weighted degree kernel: take position into account
  - count number of matching $k$-mers at identical position, for $k = 1, \ldots, d$
  - discount by length of match, i.e. $w = d - k + 1$

- Example, $d = 4$:

\[
K(a, b) = (4 - 1 + 1) \cdot 5 \\
\quad + (4 - 2 + 1) \cdot 2 \\
\quad + (4 - 3 + 1) \cdot 1 \\
= 20 + 4 + 1 = 25
\]
Protein secretion prediction

- String kernel on protein sequences slightly better than kernels on original feature vectors
Advanced kernels

- **Graph kernels**
  - encode graph as string
  - compare random walks

- **Generative model kernels:**
  - \( K(a, b) = P(a, b \mid M) \) : joint probability of \( a \) and \( b \) given a model \( M \)
    (for example, a hidden Markov model)
  - Fisher kernel

- Etc. etc.
Kernel combination

- Zoo of kernels, applicable for different data sources...

- Combine, for example by simply adding kernel matrices
Integrative Bioinformatics

Kernel-based algorithms

Kernels

Kernel combination

Example applications
Weighted kernel combination

- Combination: weighted sum of (normalised) kernel matrices

\[ K'_k(a, b) = \frac{K_k(a, b)}{\sqrt{K_k(a, a)K_k(b, b)}} \]

\[ K_{combined}(a, b) = \sum_{k=1}^{n} \mu_k K'_k(a, b) \]

- Goals:
  - Improve performance
  - Determine important features
  - Sparser model

- Simplest approach: \( \mu_k = 1 \ \forall k \)
  - Note: sum of linear kernels is equal to feature space concatenation
Protein secretion prediction

- Kernels normalised and summed: slightly better than best individual kernel
- Can we optimize the weights $\mu_k$?
Weight optimisation

1. Filter approach: optimize a derived criterion

- Example: maximise the kernel alignment w.r.t. $\mu$

$$A(K_{combined}) = \frac{\langle K_{combined}, K_{ideal} \rangle_F}{\sqrt{\langle K_{combined}, K_{combined} \rangle_F \langle K_{ideal}, K_{ideal} \rangle_F}}$$

where $K_{ideal} = y y^T$ is the ideal kernel matrix,
and $\langle K_1, K_2 \rangle_F = \sum_i \sum_j K_{1ij} K_{2ij}$ is the Frobenius norm and

$$K_{combined}(a,b) = \sum_{k=1}^n \mu_k K_k'(a,b)$$

Cristianini et al., NIPS 2001
Weight optimisation (2)

2. Wrapper approach: optimise SVM performance
   - grid search
   - evolutionary algorithms
   - gradient descent, using estimated derivative of the generalisation error, $E$
     1. set initial guess for $\mu$, use to combine kernels
     2. train SVM on combined kernel
     3. update $\mu$ to minimise $E$, recombine kernels
     4. go to 2

Chapelle et al., Machine Learning 2002
Protein secretion prediction

- Three RBF kernels combined, weights set by grid search:
  - Sequence: $\mu_1 = 1.0$
  - Derived: $\mu_2 = 0.0$
  - Other: $\mu_3 = 0.5$
Protein secretion prediction (2)

- Adding specific kernels does not help here
Weight optimisation (3)

3. Embedded approach: directly optimize SVM margin

- Multiple kernel learning (MKL):

\[
\max \sum_{i=1}^{\left| X \right|} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\left| X \right|} y_i y_j \alpha_i \alpha_j K(x_i, x_j)
\]

\[
\alpha_i \geq 0, \quad \forall i
\]

\[
\sum_{i=1}^{\left| X \right|} \alpha_i y_i = 0
\]

\[
f(z) \geq 1
\]

\[
f(z) \leq -1 \quad f(z) = 0
\]

Lanckriet et al., ICML 2002
Weight optimisation (3)

3. Embedded approach: directly optimize SVM margin

- Multiple kernel learning (MKL):

\[
\begin{align*}
\min_{\mathbf{\mu}} & \quad \max_{\mathbf{\alpha}} \sum_{i=1}^{\left|X\right|} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\left|X\right|} y_i y_j \alpha_i \alpha_j \sum_{k=1}^{n} \mu_k K_k \left( x_i, x_j \right) \\
\text{subject to} & \quad \alpha_i \geq 0, \quad \forall i \\
& \quad \sum_{i=1}^{\left|X\right|} \alpha_i y_i = 0 \\
& \quad \mu_k \geq 0, \quad \forall k \\
& \quad \sum_{k=1}^{n} \mu_k \text{tr} \left( K_k \right) = c
\end{align*}
\]

\[f \left( \mathbf{z} \right) \geq 1\]

\[f \left( \mathbf{z} \right) \leq -1\]

\[f \left( \mathbf{z} \right) = 0\]

Lanckriet et al., ICML 2002
Multiple kernel learning

• Original $L_2$-SVM primal:
  
  $\min \|w\|^2_2 + C \sum_{i=1}^{|X|} \xi_i$
  
  $w^T x_i + b \geq + (1 - \xi_i), \quad y = +1$
  
  $w^T x_i + b \leq -(1 - \xi_i), \quad y = -1$
  
  $\xi_i \geq 0, \quad \forall i$

• Corresponding MKL primal:
  
  $\min \left( \sum_{k=1}^n \|w_k\|_2 \right)^2 + C \sum_{i=1}^{|X|} \xi_i$

• $L_1$-norm over per-kernel $L_2$ norm
• Promotes sparsity at kernel level, similar to group-LASSO regression; kernels with non-zero weight are “support kernels”

Bach et al., ICML 2004
Multiple kernel learning (2)

- Example: protein localisation prediction, with 69 kernels in six groups
Multiple kernel learning (3)

• Sparsity does not always bring better performance
  • Move from $L_1$ norm to $L_p$ norm

• Other extensions:
  • Localized multiple kernel learning (Gönen et al., ICML 2008)
  • Nonlinear kernel combinations (Cortes et al., NIPS 2009)
  • More complex sparsity structures (Szafranski et al., M. Learning 2010)
INTEGRATIVE BIOINFORMATICS
KERNEL-BASED ALGORITHMS
KERNELS
KERNEL COMBINATION
EXAMPLE APPLICATIONS
A. Protein-protein interaction

- **Input $x$**: homology, co-expression, co-localization, etc. (49)
- **Output $y$**: protein interaction (0/1)

- **EA** = evolutionary algorithm

---

Hulsman et al., *IEEE TCBB* 2009
A. Protein-protein interaction (2)
A. Protein-protein interaction (3)

- Alignment and MKL give sparse solutions
  (columns are runs)
B. PPI from literature

• Input $x$: sentence with two identified proteins, e.g.
  "Raf-1 was activated by JAK2 in the presence of p21ras"

• Output $y$: protein interaction (0/1)

• Two parsers result in trees:

Miwa et al., Int J Med Inf 2009
B. PPI from literature (2)

- Three kernels applied to each parse:
  - Bag-of-words (set kernel)
  - Subset tree kernel (#common subtrees)
  - Graph kernel (random walks)

<table>
<thead>
<tr>
<th>L</th>
<th>F</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>30.2</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>52.7</td>
<td>0.822</td>
</tr>
<tr>
<td>T</td>
<td>55.1</td>
<td>0.799</td>
</tr>
<tr>
<td>G</td>
<td>59.1</td>
<td>0.854</td>
</tr>
<tr>
<td>T + B</td>
<td>58.5</td>
<td>0.849</td>
</tr>
<tr>
<td>G + B</td>
<td>57.0</td>
<td>0.847</td>
</tr>
<tr>
<td>T + G</td>
<td>62.0</td>
<td>0.873</td>
</tr>
<tr>
<td>T + G + B</td>
<td>59.9</td>
<td>0.863</td>
</tr>
</tbody>
</table>
C. Enzyme function prediction

- **Input** $x$: sequence representation of enzyme $E$
- **Output** $y$: representation of metabolic reaction $R$

(structured output prediction: density over $R$'s)

$$K(R^a, R^b) = K(S^a, S^b)K(P^a, P^b)$$
$$K(S^a, S^b) = \sum_{i,j} K_{molecule}(S^a_i, S^b_j)$$

$K_{molecule}$ counts number of common small subgraphs

**Diagrams**:

- $S_1$: D-glucose
- $S_2$: ATP
- $P_1$: D-glucose 6-phosphate
- $P_2$: ADP

Astikainen et al., ICB 2010

D-glucose + ATP $\rightarrow$ D-glucose 6-phosphate + ADP

Juho Rousu, TA2, Thu 10.15
CONCLUSIONS
Conclusions

• Integrative bioinformatics:
  • combining prior knowledge & measurements to infer and annotate molecular interaction networks
  • heterogeneous data calls for intermediate integration

• Kernels are ideal vehicles for this
  • many standard algorithms have been “kernelised”
  • a wide variety of applicable kernels is available
  • theory of kernel algorithms and combination is well-developed, and still ongoing research

• Try it yourself!

   Good start: http://www.shogun-toolbox.org/
But...

- Kernel combination is still no free lunch:
  - kernel normalisation is essential
  - kernel combinations can overtrain as well
  - kernel and classifier parameters ($C, \sigma$) have large impact, but require intensive procedures to set
  - computationally intensive, particularly for genome-wide datasets

- In practice:
  - measurement coverage & bias are problematic
  - choosing the right kernel(s) to combine is still an art
  - as often in bioinformatics, the KISS principle applies: simple summation often already works quite well
Thank you!

http://bioinformatics.tudelft.nl/