Input output kernel regression for protein-protein interaction prediction and metabolite identification

Céline Brouard

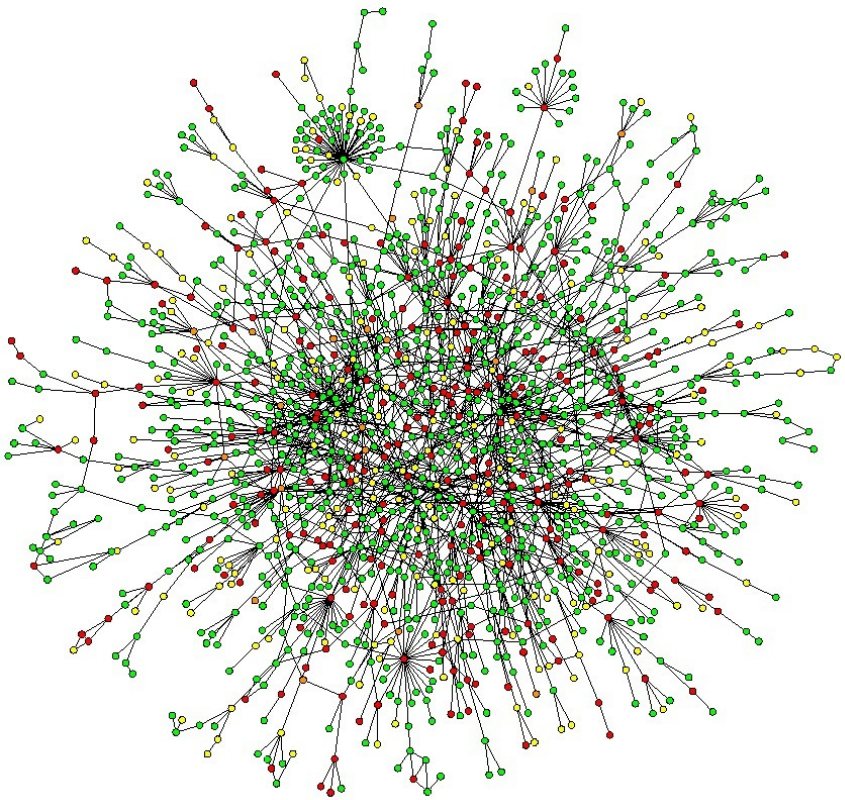
MIA Toulouse

5 Octobre 2018

Outline

Protein-protein interactions

- Most proteins perform their functions by interacting with other proteins.

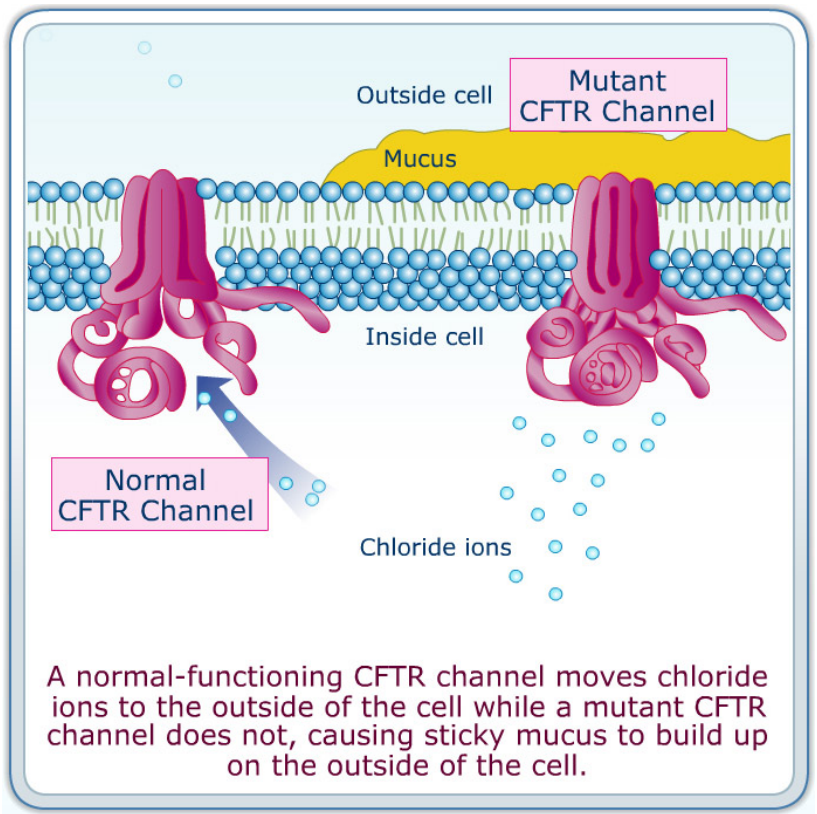


Yeast PPI network

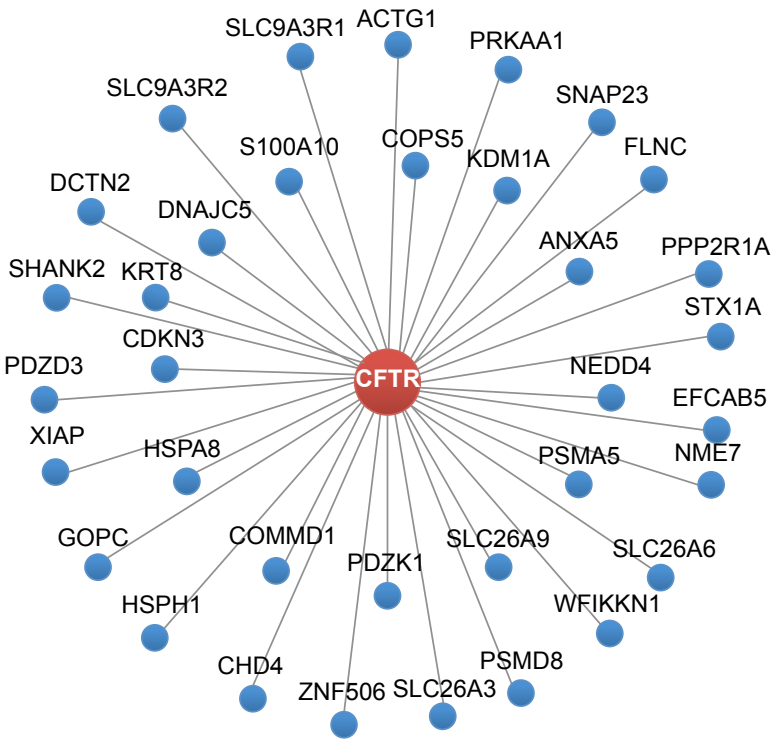
- Nodes $\leftrightarrow$ proteins
- An edge between two nodes means a physical interaction between the corresponding proteins
- We omit that interactions take place in time and space

Cystic fibrosis and the CFTR protein (1)

- **Cystic fibrosis** :
  - lethal, genetic disease
  - related to mutations of the gene *CFTR*, causing an alteration of the protein encoded by this gene
- **CFTR protein**
  - main function : regulates the ion transport through the cellular membrane



Cystic fibrosis and the CFTR protein (2)



- CFTR interacts with many proteins
  $\Rightarrow$ Impact on the stability, the localization and the function of CFTR
- The identification of these interactions is important for understanding the function and the regulation of CFTR

Motivation

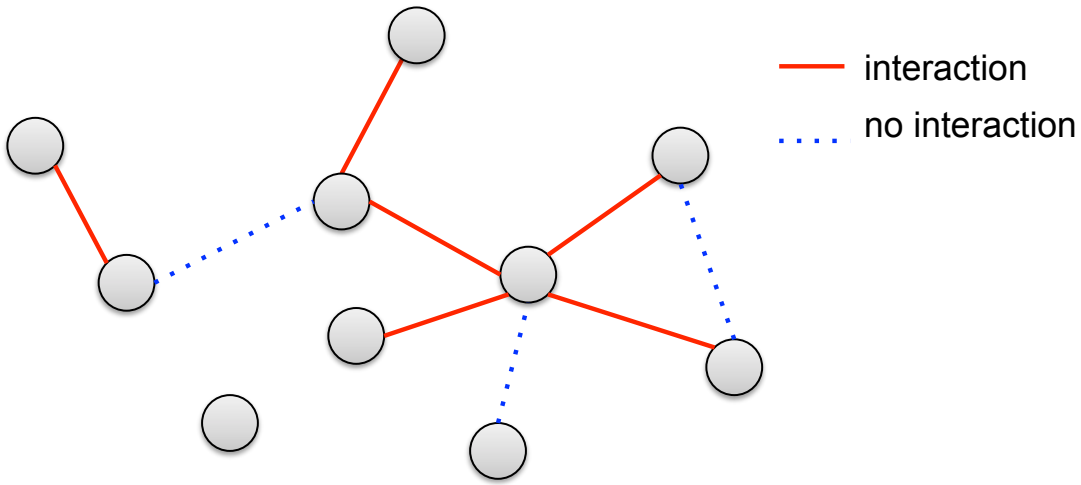
- Limitations of existing experimental methods for PPI detection
  - **Small-scale methods** : very precise but time consuming (determine one pair of proteins at a time)
  - **Large-scale techniques** : allow identifying a large number of interactions in a single experiment but are known to be more error-prone

Goals

- develop *in silico* prediction methods of protein-protein interactions which can be applied in human
- suggest new interactions to biologists for experimental validation
- propose a general framework to solve this problem

Protein-protein interaction prediction

The problem of protein-protein interaction prediction can be seen as a link prediction problem in a graph.



Goal : learning a prediction function

$$f : (u, u') \longrightarrow \begin{cases} 1 & \text{if there exists an interaction between the nodes } u \text{ and } u' \\ 0 & \text{otherwise} \end{cases}$$

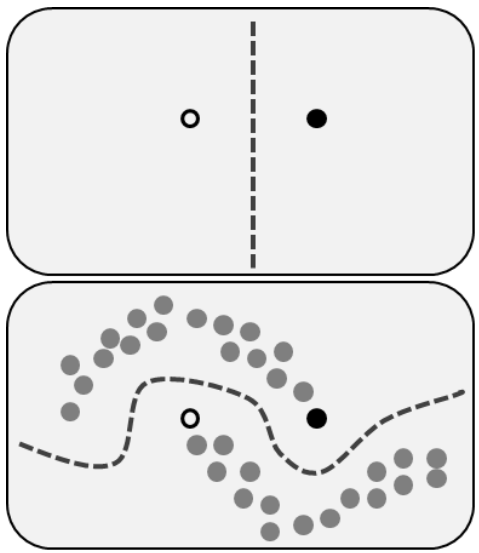
from

- **labeled data**, i.e. a set of known interactions and absences of interactions
- **information on the nodes**

Protein-protein interaction prediction

- few protein-protein interactions are known
- however a lot of properties are known on the proteins

$\Rightarrow$ machine learning approach in the semi-supervised setting in order to benefit form the information of unlabeled data.



- $\mathcal{U}_\ell = \{u_1, \ldots, u_\ell\}$: set of ℓ labeled nodes (for which the presences and absences of links are assumed to be known)
- A_ℓ : adjacency matrix of the known sub-network
- $\{u_{\ell+1}, \ldots, u_{\ell+n}\}$: set of n unlabeled nodes

Output Kernel Regression framework for link prediction

joint work with F. d'Alché-Buc and M. Szafranski

We consider an output kernel $k_y : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ that encodes the information of the proximity between objects as nodes in the unknown graph.

Diffusion kernel [Kondor & Lafferty, 2002]

The labeled Gram matrix K_Y is defined as :

$$K_Y = \exp(-\beta L),$$

where $L = D_\ell - A_\ell$, D_ℓ being the diagonal matrix containing the degrees.

- defines a global and smooth similarity measure
- the kernel value between 2 nodes takes into account all paths in the graph (even non direct) between them

Output Kernel Regression framework for link prediction

The binary classification problem is converted into a kernel learning problem.

Building a classification function from $\hat{k}_y$:

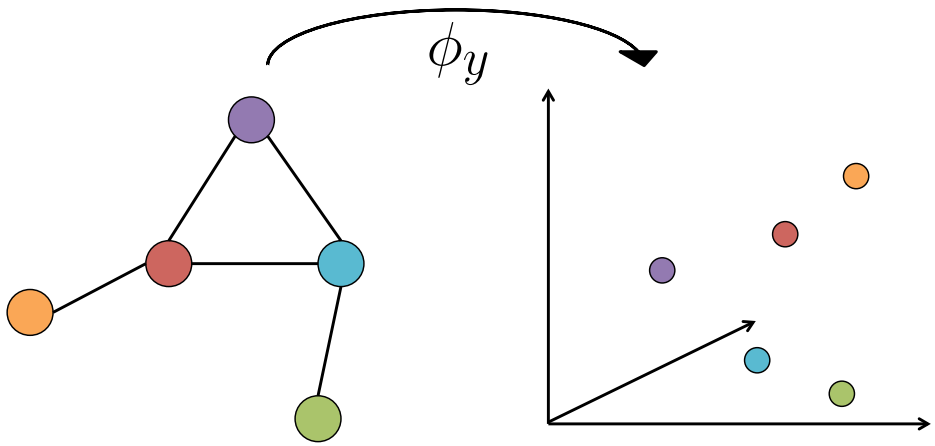
Given an approximation $\hat{k}_y$ of k_y, a classification function f_θ is defined by thresholding its output values :

$$\forall (u, u') \in \mathcal{U} \times \mathcal{U},\ f_\theta(u, u') = \mathrm{sgn}(\hat{k}_y(u, u') - \theta).$$

An interaction is predicted between 2 proteins u and u' when the kernel prediction for this pair is above some threshold.

Output Kernel Regression framework for link prediction

Evaluation of kernel values as a scalar product : $k_y(u, u') = \langle \phi_y(u), \phi_y(u') \rangle_{\mathcal{F}_y}$ where $\mathcal{F}_y$ is a Hilbert space and $\phi_y : \mathcal{U} \to \mathcal{F}_y$ a mapping.



- $\phi_y(u)$ is close to $\phi_y(u')$ in $\mathcal{F}_y$ if u and u' are connected
- Depending on the kernel, $\phi_y(u)$ is not always explicitly known

Output Kernel Regression framework for link prediction

Given an approximation of the output feature map ϕ_y with a vector-valued function h, an approximation of k_y is built from the following scalar product :

$$\hat{k_y}(u, u') = \langle h(u), h(u') \rangle_{\mathcal{F}_{\boldsymbol{y}}}.$$

Using the kernel trick in the output space reduces the problem of learning a pairwise classifier to the problem of learning a single variable function with values in a Hilbert space.

Task of learning the function h : **Output Kernel Regression** (OKR)

Output Kernel Regression framework for link prediction

Previous works : *Output Kernel Regression Trees* [Geurts et al., 2006, 2007].

Proposed approach : *Input Output Kernel Regression* [Brouard et al., 2011, 2016] :

- able to take into account structure in input data
- uses the framework of penalized regression, that allows to use smoothness penalties for semi-supervised learning

We use kernels both in input and output spaces.

Learning functions with values in a Hilbert space

Prediction problem : approximation of the function h whose values are vectors belonging to the output feature space $\mathcal{F}_y$.

RKHS theory devoted to vector-valued functions [Senkene & Tempel'man, 1973 ; Michelli & Pontil, 2005]

- Operator-valued kernels : extension of scalar kernels for vector-valued functions
- Existing applications :
  - Multi-task learning [Michelli & Pontil, 2005 ; Argyriou & Pontil, 2008]
  - Prediction of functional data [Kadri et al., 2010]
  - Structured classification [Dinuzzo et al., 2011]

Operator-valued kernel

Let $\mathcal{X}$ be some input space and $\widetilde{\mathcal{Y}}$ an Hilbert space.

An operator-valued kernel $\mathcal{K}_x$ is a function whose values are operators from $\widetilde{\mathcal{Y}}$ to $\widetilde{\mathcal{Y}}$: $\mathcal{K}_x : \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\widetilde{\mathcal{Y}})$.

$\mathcal{K}_x$ is an operator-valued kernel if :

1. $\forall x, x' \in \mathcal{X}$, $\mathcal{K}_x(x, x') = \mathcal{K}_x(x', x)^*$,
2. $\forall m \in \mathbb{N}, \forall \{(x_i, \tilde{\mathbf{y}}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \widetilde{\mathcal{Y}}$, $\sum_{i,j=1}^m \langle \tilde{\mathbf{y}}_i, \mathcal{K}_x(x_i, x_j)\tilde{\mathbf{y}}_j \rangle_{\widetilde{\mathcal{Y}}} \geq 0$.

Example : decomposable operator-valued kernel

$$\mathcal{K}_x(x, x') = k_x(x, x')A,$$

where $k_x : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a scalar-valued kernel and A is an operator from $\widetilde{\mathcal{Y}}$ to $\widetilde{\mathcal{Y}}$.

Representer theorem in the supervised setting

- Given an OVK $\mathcal{K}_x : \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\widetilde{\mathcal{Y}})$, there exists a unique vector-valued RKHS $\mathcal{H}$ which admits $\mathcal{K}_x$ as the reproducing kernel.

Representer theorem (Micchelli & Pontil 2005)

Given a training set $\{(x_i, \tilde{\mathbf{y}}_i)\}_{i=1}^{\ell} \subseteq \mathcal{X} \times \widetilde{\mathcal{Y}}$, the minimizer of the following optimization problem

$$\operatorname*{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{\ell} \mathcal{L}(h(x_i), \tilde{\mathbf{y}}_i) + \lambda \|h\|_{\mathcal{H}}^2 \ , \quad \lambda > 0$$

admits an expansion of the form

$$\hat{h}(\cdot) = \sum_{j=1}^{\ell} \mathcal{K}_x(\cdot, x_j)\mathbf{c}_j, \quad \mathbf{c}_j \in \widetilde{\mathcal{Y}},\ j = 1, \cdots, \ell.$$

Approximation of the output feature map for link prediction

We use the IOKR framework with a least-squares loss function to learn an approximating of the output feature map ϕ_y :

$$\underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{\ell} \|h(u_i) - \phi_y(u_i)\|^2_{\mathcal{F}_y} + \lambda \|h\|^2_{\mathcal{H}} \ , \quad \lambda > 0$$

- Considered operator-valued kernel : $\mathcal{K}_x(u, u') = k_x(u, u')I$, where k_x is a scalar-valued kernel and I the identity operator.

Solution of the optimization problem :

$$\hat{h}(u) = \sum_{i=1}^{\ell} \alpha_i(u)\phi_y(u_i), \text{ with } \alpha(u) = (\lambda I + K_X)^{-1} k_X^u$$

- $K_X \in \mathbb{R}^{\ell \times \ell}$: kernel matrix of k_x
- $k_X^u = [k_x(u_1, u), \ldots, k_x(u_\ell, u)]^T$

Extension of the representer theorem to the semi-supervised setting

Addition of a regularization term, that forces the target function h to be smooth with respect to the underlying manifold.

- $\{u_i\}_{i=1}^{\ell+n}$: additional set of unlabeled examples
- W : matrix measuring the local similarities between objects in the input space

Optimization problem

$$\underset{h\in\mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{\ell} \|h(u_i) - \phi_y(u_i)\|^2_{\mathcal{F}_y} + \lambda_1 \|h\|^2_{\mathcal{H}} + \lambda_2 \sum_{i,j=1}^{\ell+n} W_{ij} \|h(u_i) - h(u_j)\|^2_{\mathcal{F}_y},$$

where λ_1 and $\lambda_2 > 0$.

Extension of the representer theorem to the semi-supervised setting

Theorem [Brouard et al, 2011 ; Minh & Sindwhani, 2011]

The function h minimizing this optimization problem admits the following form :

$$h(\cdot) = \sum_{j=1}^{\ell+n} \mathcal{K}_x(\cdot, u_j)\mathbf{c}_j, \ \mathbf{c}_j \in \mathcal{F}_y.$$

Solution of the optimization problem :

$$\hat{h}(u) = \sum_{i=1}^{\ell} \alpha_i(u)\phi_y(u_i), \text{ with } \alpha(u) = J(\lambda I_{\ell+n} + K_X(J^T J + 2\lambda_2 L))^{-1} k_X^u$$

- $K_X \in \mathbb{R}^{(\ell+n)\times(\ell+n)}$: kernel matrix of k_x
- $k_X^u = [k_x(u_1, u), \ldots, k_x(u_{\ell+n}, u)]^T$
- $J = [I_\ell, 0] \in \mathbb{R}^{\ell\times(\ell+n)}$
- L : Graph Laplacian of W

Input Output Kernel Regression framework for link prediction

In the supervised setting the approximation of the output kernel can be written as follows :

$$\begin{aligned}\hat{k_y}(u,u') &= \langle h(u), h(u')\rangle_{\mathcal{F}_\mathbf{y}} \\ &= \sum_{i,j=1}^{\ell} \alpha_i(u)\alpha_j(u')\langle \phi_y(u), \phi_y(u')\rangle_{\mathcal{F}_\mathbf{y}} \\ &= \sum_{i,j=1}^{\ell} \alpha_i(u)\alpha_j(u) k_y(u,u')\end{aligned}$$

We can notice that we do not need to know the explicit expressions of the output $\phi_y(u)$ to compute this scalar product
(it is the same in the semi-supervised setting).

Application to yeast PPI network

Build a yeast protein-protein interaction network based on the DIP database (Database of Interacting Proteins)

- Taking into account the proteins annotated for each input kernel and involved in at least one interaction

  $\Rightarrow$ obtaining a network containing 815 nodes with a link density of 0.0054

Experimental protocol :

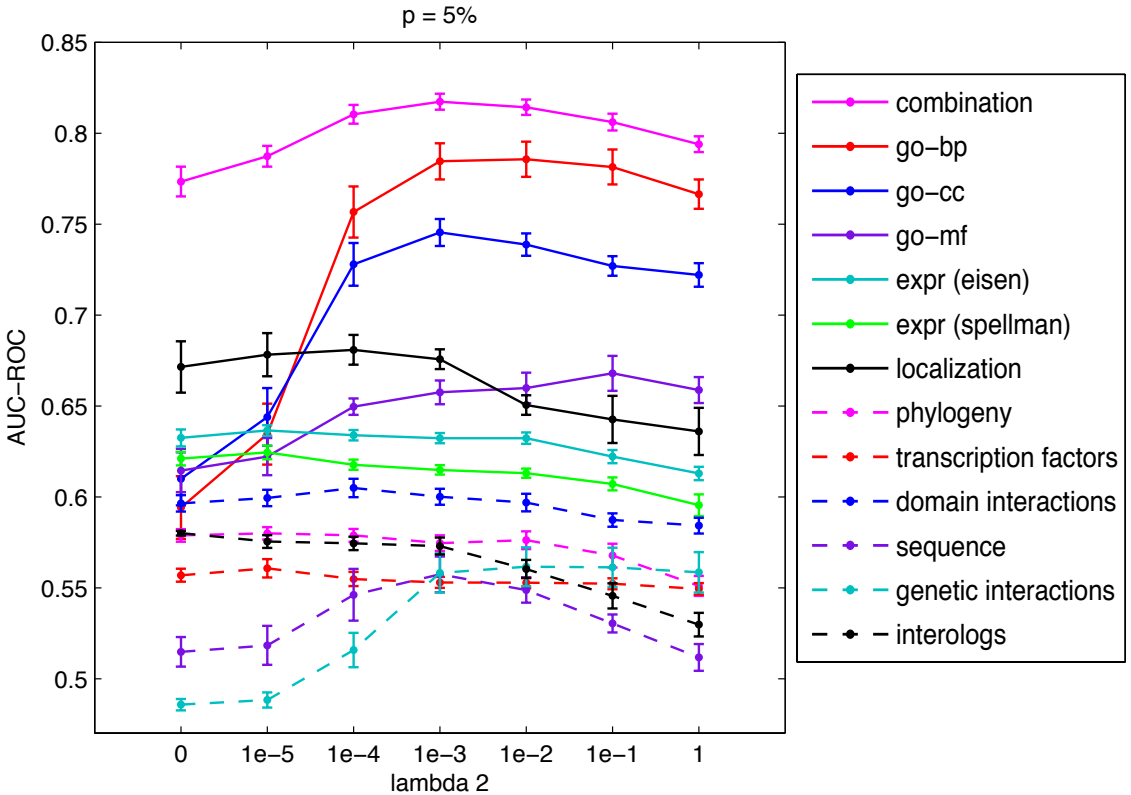
- p% of the nodes are subsampled as labeled nodes
- the performances are averaged over ten random choices of the labeled set

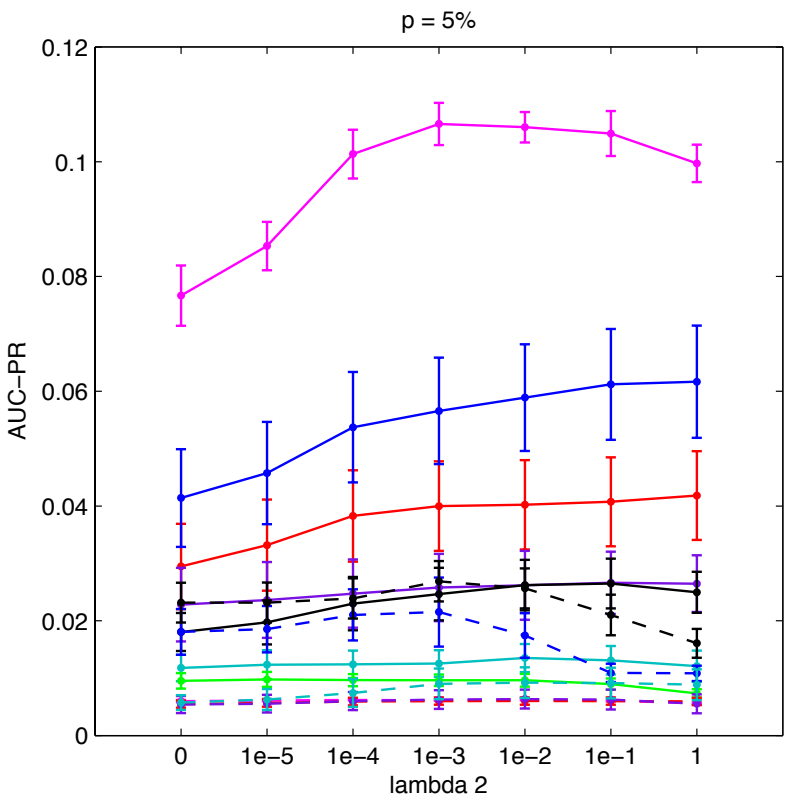
Application to yeast PPI network

Input features	kernel
Gene expressions [Eisen et al., 1998]	gaussian
Gene expressions [Spellman et al., 1998]	gaussian
Subcellular localizations [MIPS]	gaussian
Genetic interactions [BioGRID]	gaussian
Sequence [NCBI Protein]	k-spectrum
Domain-domain interactions [Pfam, DOMINE]	diffusion
Transcription factors [YEASTRACT]	gaussian
Biological processes [Gene Ontology]	gaussian
Molecular functions [Gene Ontology]	gaussian
Cellular components [Gene Ontology]	gaussian
Interologs [Inparanoid, DIP, MINT, BioGRID]	diffusion
Phylogenetic profiles [Phylopro]	gaussian

Contribution of semi-supervised learning

For 5% of labeled nodes





Extern combination : $\forall u, u' \in \mathcal{U}$, $\hat{\kappa_y}(u, u') = \frac{1}{p} \sum_{j=1}^{p} \hat{\kappa_y}^{(j)}(u, u')$,
where $\hat{\kappa_y}^{(j)}$ corresponds to the approximation of the output kernel obtained when the j-th input kernel is used, and p to the number of considered kernels.

Comparison in the supervised setting

5-cv experiment

a) AUC-ROC :

Method	GO-BP	GO-CC	GO-MF	int
Naive	60.8 ± 0.8	64.4 ± 2.5	64.2 ± 0.8	67.7 ± 1.5
kCCA	82.4 ± 3.6	77.0 ± 1.7	75.0 ± 0.6	85.7 ± 1.6
kML	83.2 ± 2.4	77.8 ± 1.1	76.6 ± 1.9	84.5 ± 1.5
Local	79.5 ± 1.6	73.1 ± 1.3	66.8 ± 1.2	83.0 ± 0.5
OK3+ET	84.3 ± 2.4	81.5 ± 1.6	79.3 ± 1.8	86.9 ± 1.6
IOKR	$\mathbf{88.8 \pm 1.9}$	$\mathbf{87.1 \pm 1.3}$	$\mathbf{84.0 \pm 0.6}$	$\mathbf{91.2 \pm 1.2}$

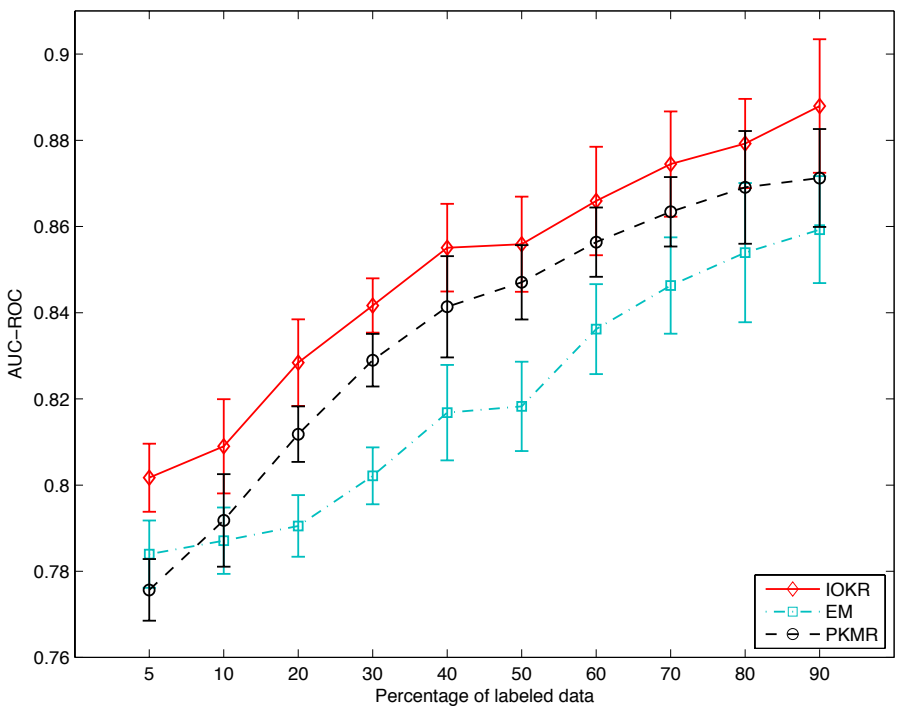
b) AUC-PR :

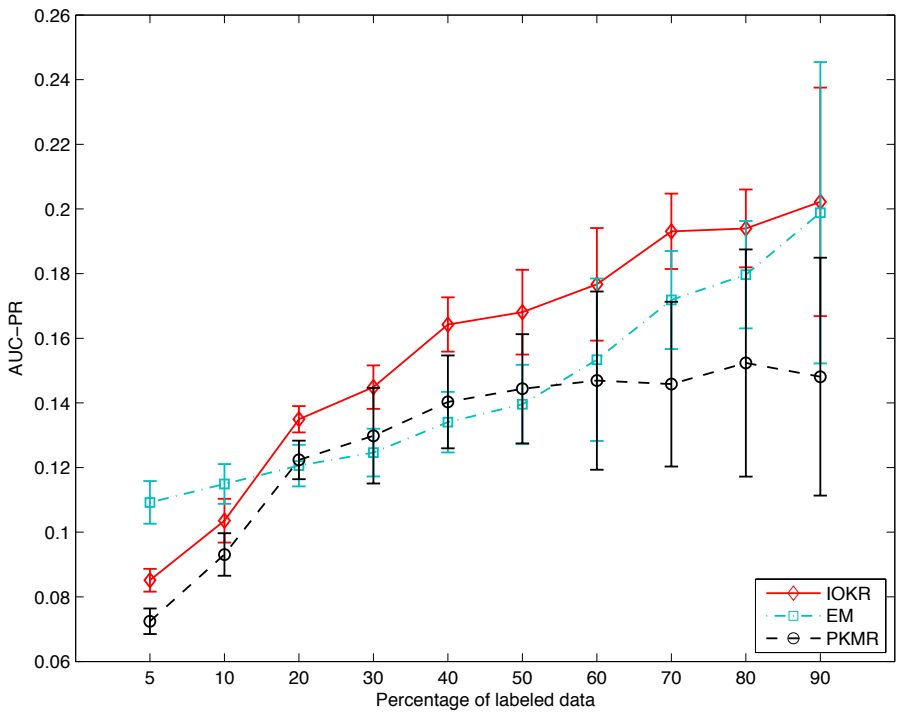
Method	GO-BP	GO-CC	GO-MF	int
Naive	4.8 ± 1.0	2.1 ± 0.6	2.4 ± 0.4	8.0 ± 1.7
kCCA	7.1 ± 1.5	7.7 ± 1.4	4.2 ± 0.5	9.9 ± 0.4
kML	7.1 ± 1.3	3.1 ± 0.6	3.5 ± 0.4	7.8 ± 1.6
Local	6.0 ± 1.1	1.1 ± 0.3	0.7 ± 0.0	22.6 ± 6.6
OK3+ET	$\mathbf{19.0 \pm 1.8}$	$\mathbf{21.8 \pm 2.5}$	$\mathbf{10.5 \pm 2.0}$	$\mathbf{26.8 \pm 2.4}$
IOKR	15.3 ± 1.2	20.9 ± 2.1	8.6 ± 0.3	22.2 ± 1.6

Comparison with transductive approaches

- EM [Kato et al., 2005]
- PKMR (Penalized Kernel Matrix Regression) [Yamanishi & Vert, 2007]

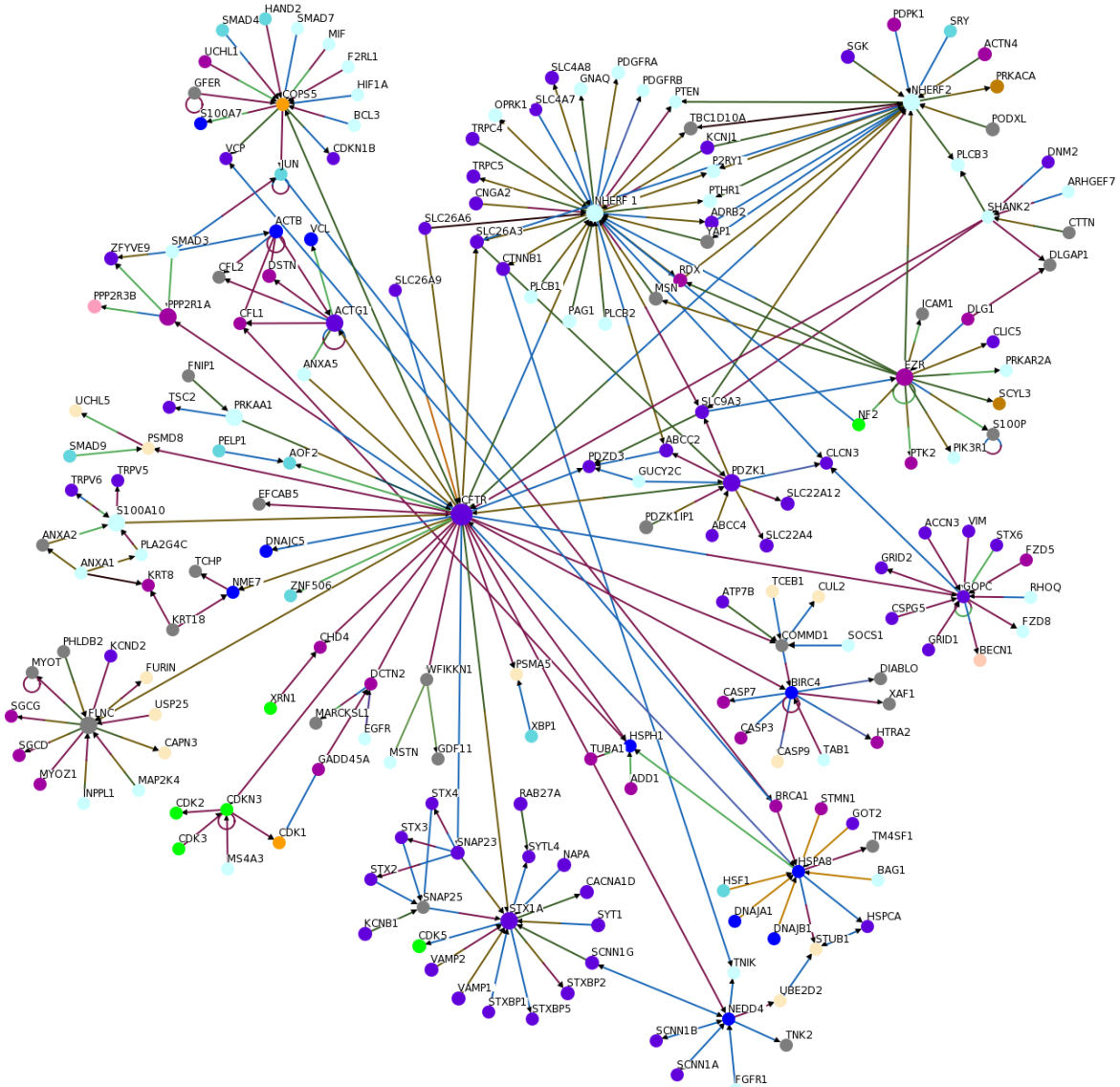
Results corresponding to the combination of predictions obtained for each input kernel :





Inference of the PPI network around CFTR (1)

joint work with A. Edelman



- Network of 198 proteins
- Manual curation of interactions in the literature (BioGRID, DIP, MINT, Intact, NextProt)

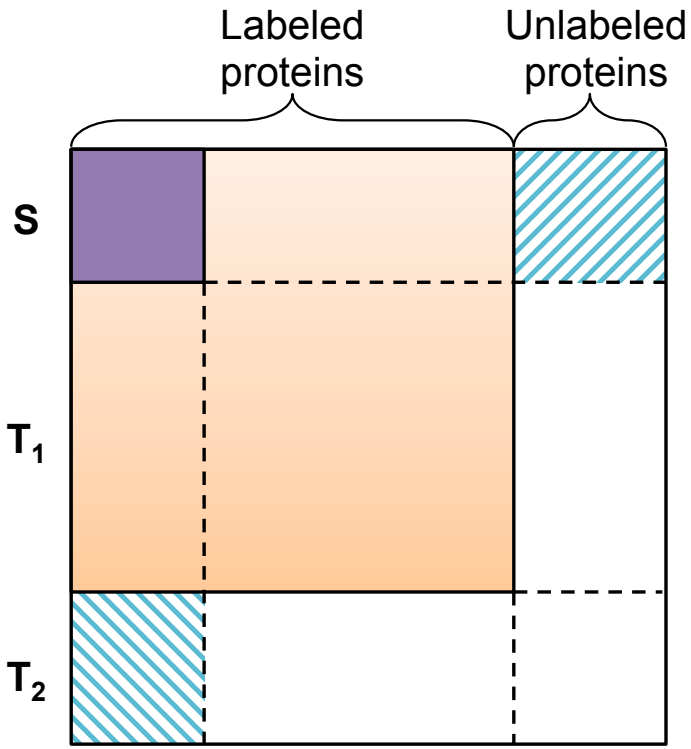
Inference of the PPI network around CFTR (2)

Input features	Kernel type
Gene expressions [Su et al., 2004]	gaussian
Protein expressions [The Human Protein Atlas]	gaussian
Subcellular localizations [The Human Protein Atlas]	gaussian
Sequence [NCBI Protein]	k-spectrum
Domain-domain interactions [Pfam, DOMINE]	diffusion
Biological processes [Gene Ontology]	gaussian
Molecular functions [Gene Ontology]	gaussian
Cellular components [Gene Ontology]	gaussian
Interologs [Inparanoid, DIP, MINT, BioGRID, Intact]	diffusion
Phylogenetic profiles [BLASTP]	gaussian

Protocol

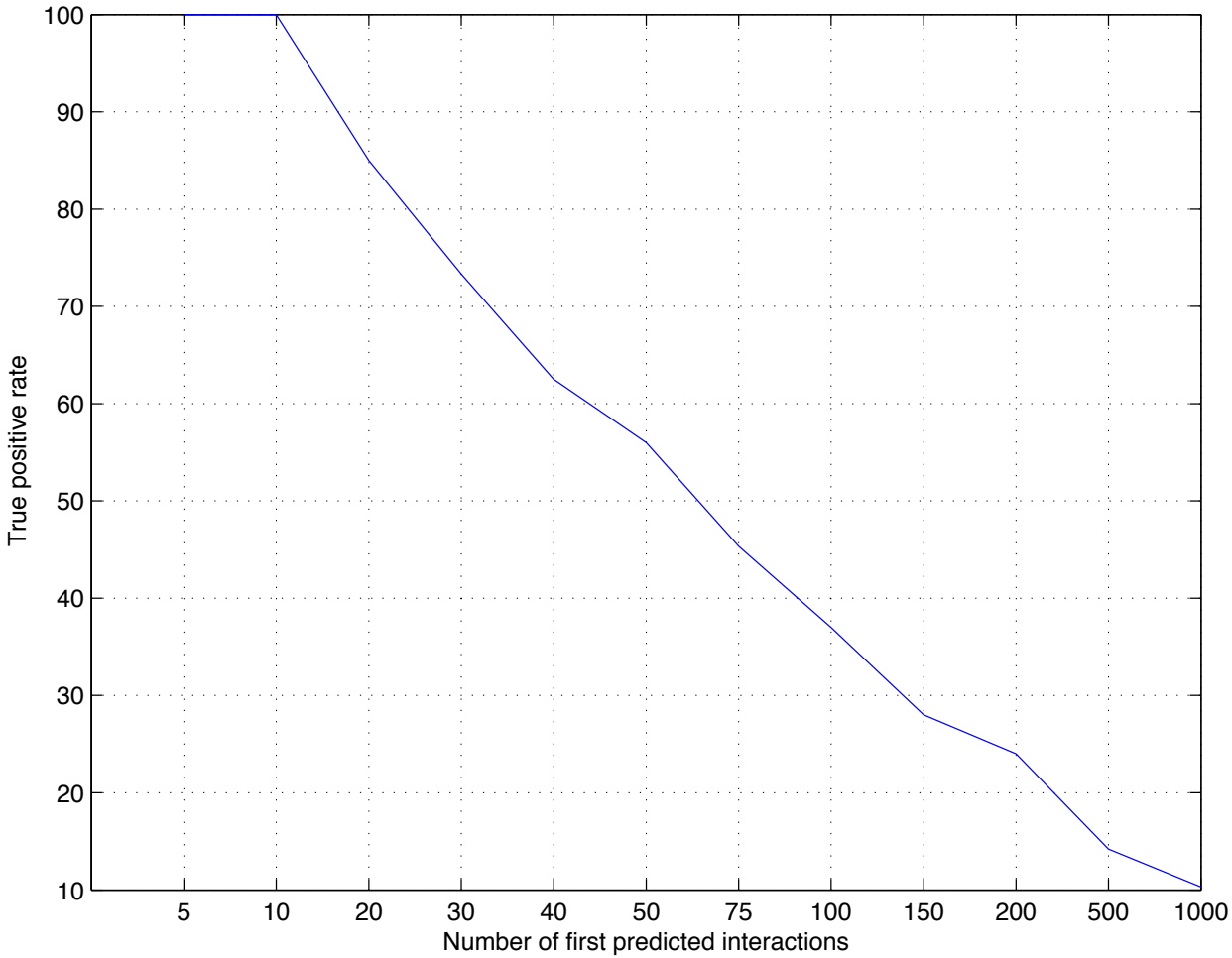
Missing annotations are taken into account

- S : CFTR + proteins interacting directly with CFTR (34)
- T : set of proteins interacting directly with the proteins in the set S (163)
- Prediction of interactions between proteins in S and proteins in T
- Several iterations :
  - ith iteration : T is randomly splited into two subsets $T_{1,i}$ and $T_{2,i}$
  - At the end, the predictions are combined



Prediction of known interactions

- The interactions predicted between two proteins u and u' are sorted according to the value taken by $\hat{\kappa_y}(u, u')$
- True positive rate obtained for the n first predictions :



Prediction of new interactions

- List of interactions obtained from a study of the literature for the first 100 predictions obtained :

Prot 1	Prot 2	Method	Reference
XIAP	PTEN	enzymatic study	Van Themsche C (2009)
NEDD4	PTEN	pull down	Wang X (2008)
NEDD4	PTEN	enzymatic study	Wang X (2008)
SNAP23	VAMP2	pull down	Kawanishi M (2000)
SNAP23	VAMP1	two hybrid	Ravichandran V (1996)
SNAP23	STX6	pull down	Martin-Martin B (2000)
SNAP23	STXBP2	in vivo	Schraw TD (2003)
DNAJC5	STUB1	affinity capture western	Schmidt BZ (2009)
ANXA5	ANXA1	co-localization	Arur S (2003)

Outline

Structured output learning

- Many real world applications involve objects with an explicit or implicit structure
- Examples of structured data that we may want to use as inputs or outputs : graphs, trees ...
- Structured output prediction can also concerns multiple outputs linked by some relationship

Input Output Kernel Regression (IOKR)

[Brouard et al., 2011, 2016]

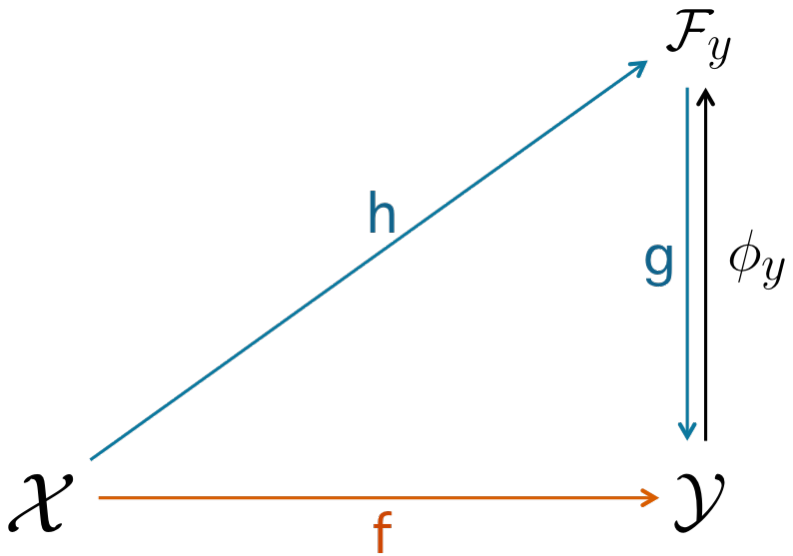
Extension of Input Output Kernel Regression (IOKR) to the general case of structured output prediction :

can be used to learn mappings between two structured spaces $\mathcal{X}$ and $\mathcal{Y}$.

- The internal structure of the outputs is encoded using an output kernel function $k_y : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$
- k_y is associated with a feature space $\mathcal{F}_y$ and a feature mapping function $\phi_y : \mathcal{Y} \to \mathcal{F}_y$:

$$\forall (y, y') \in \mathcal{Y} \times \mathcal{Y}, \;\; k_y(y, y') = \langle \phi_y(y), \phi_y(y') \rangle_{\mathcal{F}_y}$$

Input Output Kernel Regression



Decomposition of the regression problem in two tasks :

1. **Output Kernel Regression** : learn a function $h : \mathcal{X} \to \mathcal{F}_y$ that approximates the output feature map ϕ_y
2. **Computation of the pre-image** : define or learn a function $g : \mathcal{F}_y \to \mathcal{Y}$ to provide an output in the set $\mathcal{Y}$.

Pre-image step

To determine the output $f(x_i)$ in $\mathcal{Y}$ associated with the input $x_i \in \mathcal{X}$, we must determine the pre-image of $h(x_i)$ by ϕ_y :

$$\hat{f}(x_i) = \underset{y \in \mathcal{Y}}{\operatorname{argmin}} \|\hat{h}(x_i) - \phi_y(y)\|^2_{\mathcal{F}_y}$$

Using the kernel trick in the output space, it can be rewritten as :

$$\hat{f}(x_i) = \underset{y \in \mathcal{Y}}{\operatorname{argmin}}\, k_y(y,y) - 2(k_Y^y)^T(\lambda I + K_X)^{-1} k_X^{x_i}.$$

Outline

Introduction

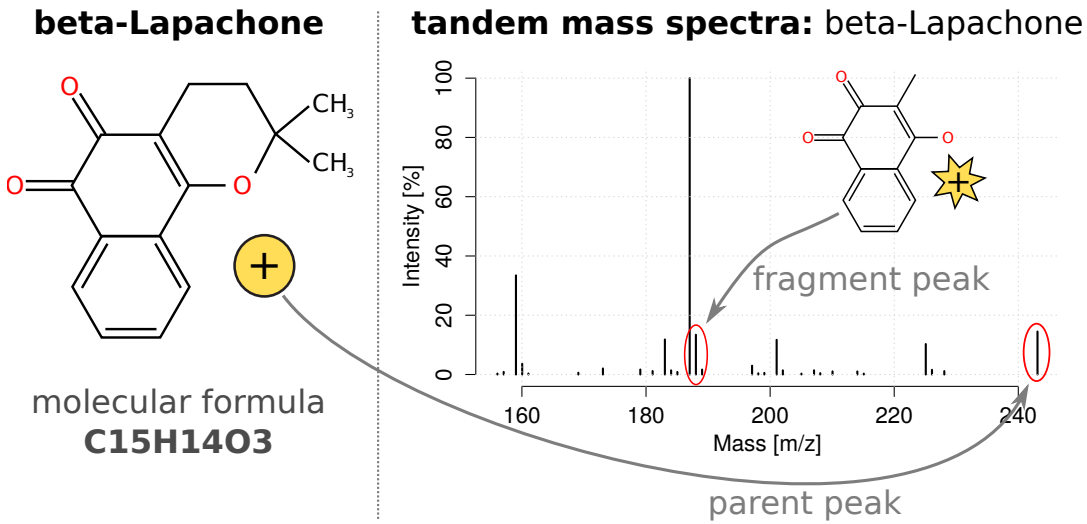
- Metabolites : small molecules inside biological cells
- Key problem in metabolomics : identify the metabolites that are present in a biological sample
- Diverse applications :
  - medical diagnostic
  - pharmaceutical drug development
  - screening for traces of explosives in airport
  - screening of environmental contaminants
  - assessing food and drink quality

Metabolite identification

Metabolite identification relies on tandem mass spectrometry (MS/MS) data, produced by :

- fragmenting the metabolite,
- recoding the masses and relative abundances (intensities) of the molecular fragments

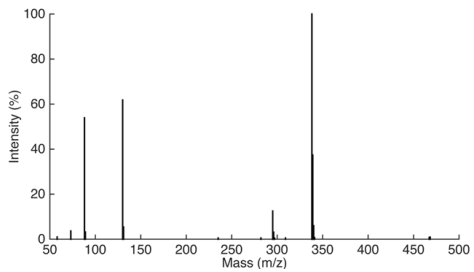
A measurement results in an MS/MS spectrum with peaks representing the intensities as a function of the masses for the different fragments.



Machine learning for metabolite identification

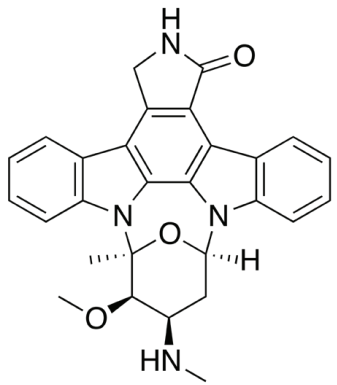
joint work with J. Rousu, H. Shen, S. Böcker and K. Dührkop

MS/MS spectra



f

Metabolite



x

y

Metabolite identification can be seen as a structured prediction problem :

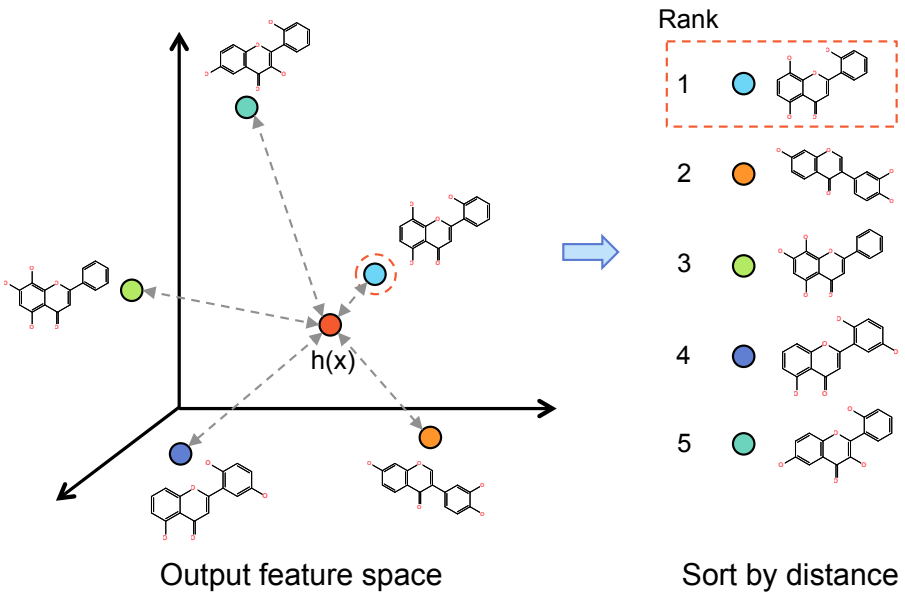
- $\mathcal{X}$: set of MS/MS spectra
- $\mathcal{Y}$: set of molecules
- Learn a function $f : \mathcal{X} \to \mathcal{Y}$ that maps a MS^2 spectrum to a molecule

We use IOKR in the supervised setting to learn this mapping.

Pre-image

$$\hat{f}(x) = \underset{y \in \mathcal{Y}^*}{\operatorname{argmin}} \|\hat{h}(x) - \phi_y(y)\|^2_{\mathcal{F}_y}.$$

- $\mathcal{Y}^*$: set of candidate molecules from molecular databases such as PubChem or KEGG.
- $\mathcal{Y}^*$ can be filtered using the mass of the unknown molecule or its molecular formula if already known
- Search the space of molecules for one with image nearest to $h(x)$:



Input kernels

We considered 24 scalar input kernels and combined them using multiple kernel learning :

$$k_x(x, x') = \sum_{j=1}^{m} \mu_j k_j(x, x').$$

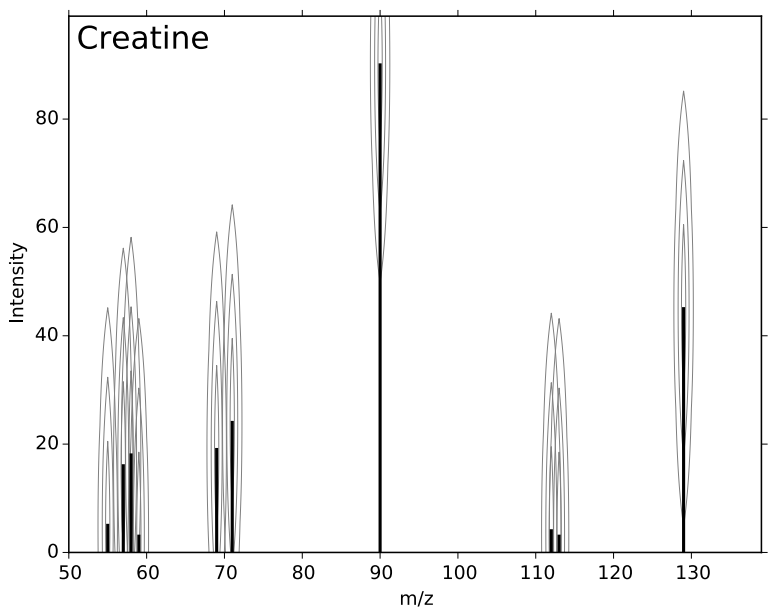
Two MKL approaches :

- Uniform MKL (UNIMKL) : $\mu_j = 1/m, \quad \text{for } j = 1, \ldots, m.$
- ALIGNF [Cortes et al., 2012] :
  search the weights that maximize the centered alignment between $K_x = \sum_j \mu_j K_j$ and K_y.

Input kernels : probability product kernel

[Heinonen et al., 2012]

- A mass spectrum is defined as a set of peaks : $x = \{x(\ell)\}_{\ell=1}^{n_x}$.
- Each peak is modelled as a 2D normal distribution centered around the observed position : $p_{x(\ell)} \sim \mathcal{N}(x(\ell), \Sigma)$.
- The covariance is shared with all peaks : $\Sigma = \begin{bmatrix} \sigma_m^2 & 0 \\ 0 & \sigma_i^2 \end{bmatrix}$.



Input kernel : probability product kernel

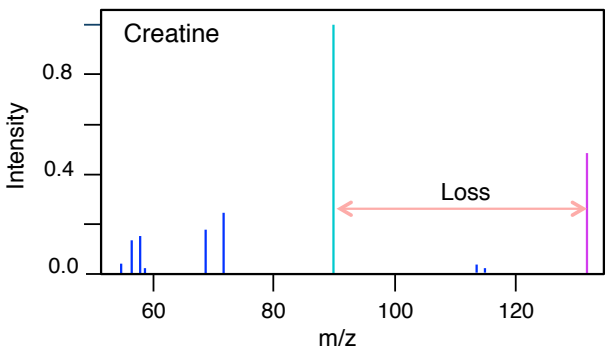
- A spectrum is represented as a mixture of its peak distributions :

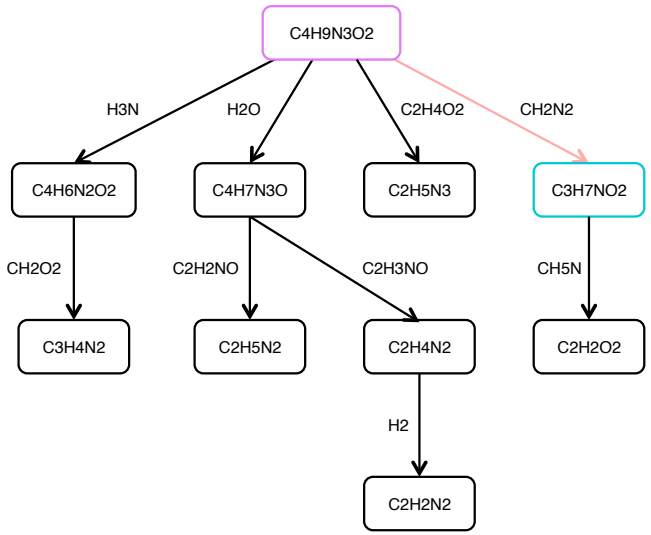
$$p_x = \frac{1}{n_x} \sum_{\ell=1}^{n_x} p_{x(\ell)}.$$

- Probability product kernel [Jebara et al., 2004] between the peaks of two spectra x and x' :

$$\begin{aligned} k(x, x') &= \int_{\mathbb{R}^2} p_x(\mathbf{z}) p_{x'}(\mathbf{z}) d\mathbf{z} \\ &= \frac{1}{n_x n_{x'}} \frac{1}{4\pi\sigma_m\sigma_i} \sum_{\ell,\ell'=1}^{n_x, n_{x'}} \exp\left(-\frac{1}{4} \left(x(\ell) - x'(\ell')\right)^T \Sigma^{-1} \left(x(\ell) - x'(\ell')\right) \right) \end{aligned}$$

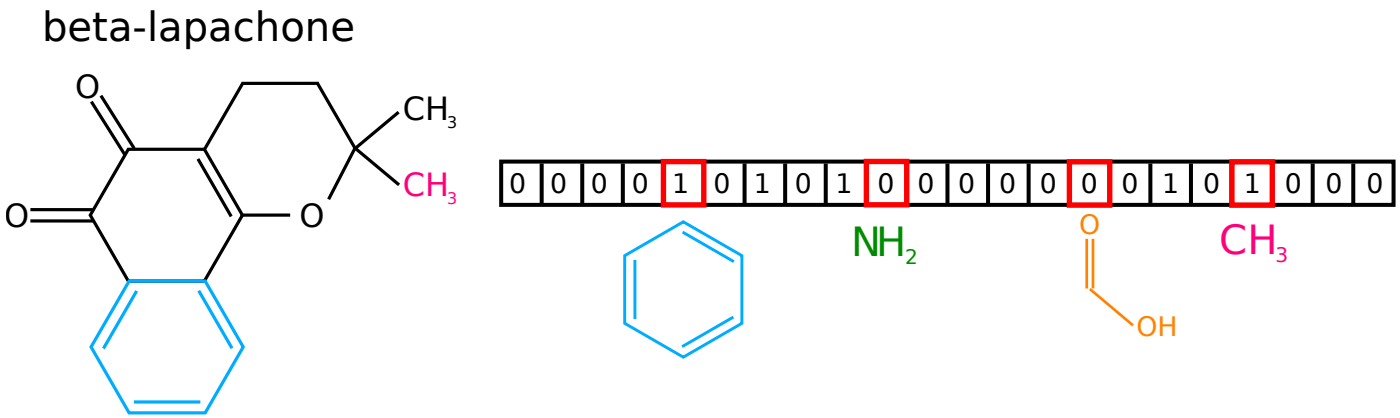
Input kernels : fragmentation trees





- Model of the fragmentation process in a tree shape :
  - Nodes $\approx$ peaks $\approx$ molecular formula of fragments
  - Edges $\approx$ losses
- Fragmentation trees can be predicted from spectra
- We use 23 different kernels based on these trees
  - *Edge-based kernels*
  - *Node-based kernels*
  - *Path-based kernels*
  - *Alignment-based kernels*

Output kernels : molecular fingerprints



- Molecular fingerprint : encodes the structure of a molecule using a bit (or count) vector.
- Each entry indicates the existence or the frequency of a certain molecular property :
  - atom or bond type,
  - substructure (e.g. aromatic ring).

Fingerprint kernels :
linear, polynomial and gaussian kernels over fingerprint vectors

Output kernels

We also considered different graph kernels :

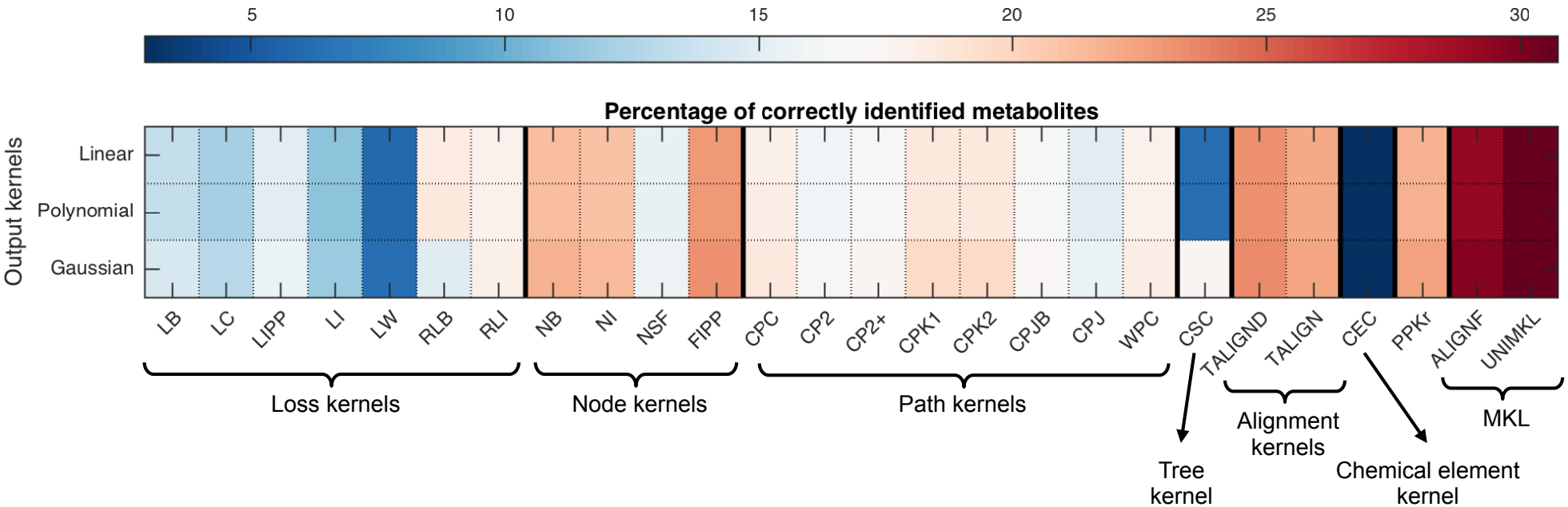
- Path kernel
- Shortest-path kernel
- Graphlet kernel

But we obtained better performances with fingerprint kernels. In the next part, we will therefore show the results obtained with IOKR for fingerprint kernels.

Protocol

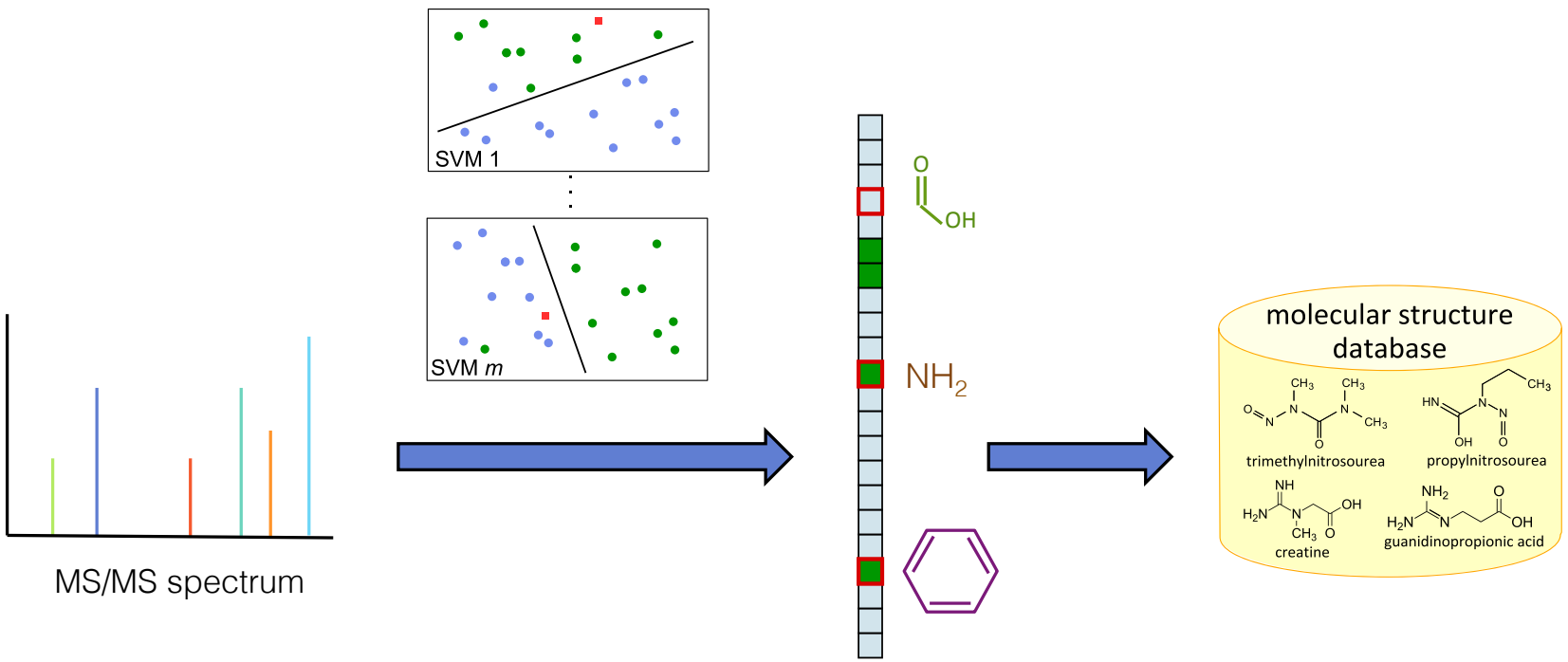
- 4138 tandem mass spectra from the GNPS spectral library.
- 10-fold cross-validation (data with the same structure are contained in the same fold).
- **Pre-image step** : search among PubChem structures having the same molecular formula as the target compound.
- **Evaluation** :
  1. For each test example : evaluate the rank of the true molecular structure among the candidates.
  2. Compute the percentage of structures that have been ranked lower than k for $1 \leq k \leq 20$.

Kernel performances



Comparison with the state of the art : CSI-FingerID

[Shen et al. (2014), Dührkop et al. (2015)]



1. Use SVM to predict fingerprint

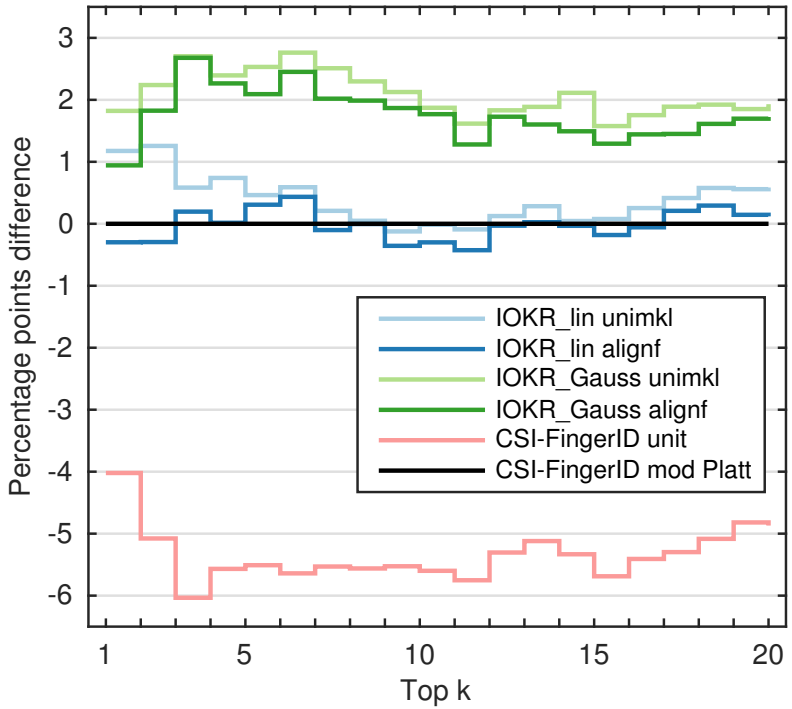
2. Compare with molecular database

Comparison with CSI :FingerID

Scoring function for comparing the candidate fingerprints with the predicted fingerprint :

- **Unit** :
  - counts the number of common molecular properties.
- **Modified Platt** :
  - combines maximum likelihood and Platt scores (posterior probability estimates of the fingerprint) for defining the scoring function.

Comparison with CSI :FingerID



Method	MKL	Top 1	Top 10	Top 20
CSI :FingerID unit	ALIGNF	24.82	60.47	68.2
CSI :FingerID mod Platt	ALIGNF	28.84	66.07	73.07
IOKR linear	ALIGNF	28.54	65.77	73.19
	UNIMKL	30.02	66.05	73.66
IOKR Gaussian	ALIGNF	29.78	67.84	74.79
	UNIMKL	**30.66**	**67.94**	**75.00**

Comparison with CSI :FingerID : running times

- 4138 training compounds (GNPS) / 625 test compounds (MassBank)
- Fix the values of the parameters
- The computation of the fragmentation trees, input kernels and fingerprints was not taken into account.

	Training time	Test time
CSI :FingerID	82 h 28 min 23 s	1 h 11 min 31 s
IOKR linear	**42 s**	**1 min 15 s**
IOKR polynomial	**38 s**	21 min 58 s
IOKR Gaussian	**41 s**	33 min 15 s

- IOKR is ≈7000 times faster to train that CSI :FingerID because CSI :FingerID needs to train 2765 SVMs (one for each molecular property).
- **IOKR linear** : avoid kernel computations in the pre-image step by computing explicitly the output feature vectors.

CASMI challenge 2016

Schymanski et al., 2017

CASMI (Critical Assessment of Small Molecule Identification) : contest on the identification of small molecules from mass spectrometry data.

Three categories :

1. manual methods
2. automatic methods
3. automatic methods using metadata

IOKR performed best in category 2 with 78 molecules identified among 208 challenges (37.5%).

Conclusions and perspectives

- Introduction of a new setting for solving structured prediction problems
- Application on two different problems : link prediction and metabolite identification
- Extensions of IOKR for metabolite identification have been developed

Perspectives :

- Learning better output representations :
  - taking into account dependencies between molecular properties using a probabilistic graphical model
  - learning the output kernel
  - combination of multiple output kernels

Thank you for your attention