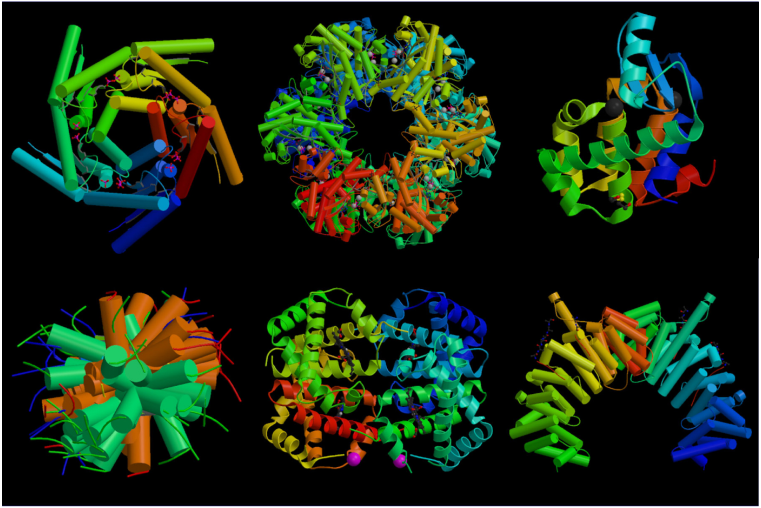
Inference of missing edges in biological networks

Jean-Philippe Vert
Jean-Philippe.Vert@mines-paristech.fr

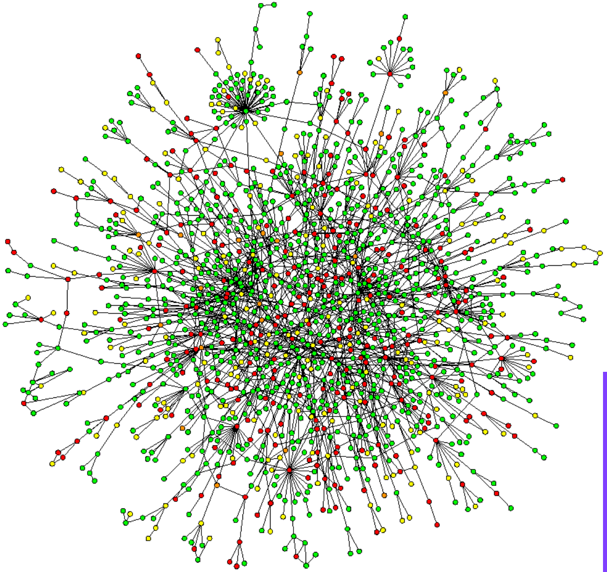
Mines ParisTech, Institut Curie, INSERM U900

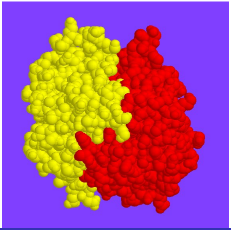
University of Bristol, UK, October 13, 2008.

Proteins

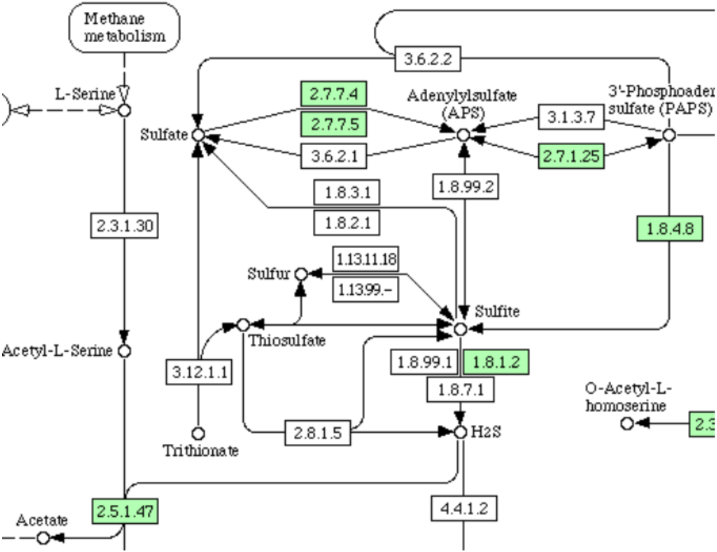


Network 1: protein-protein interaction

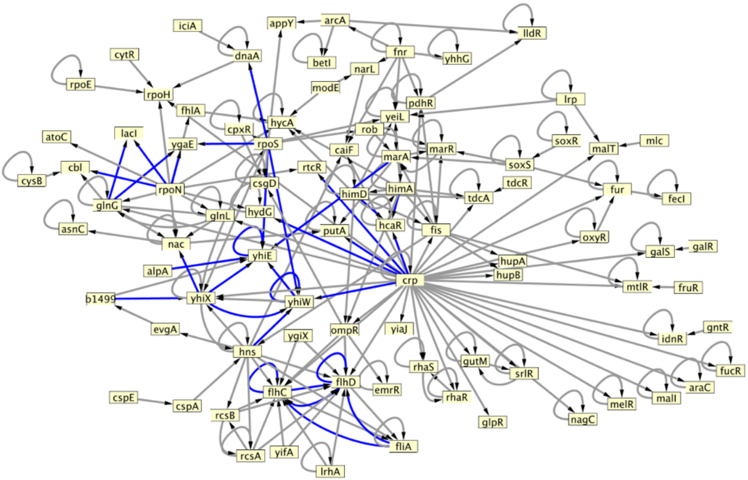




Network 2: metabolic network



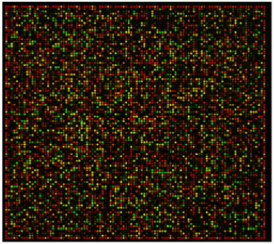
Network 3: gene regulatory network

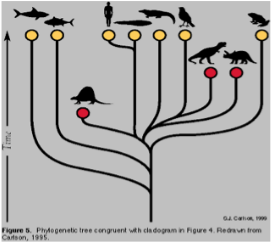


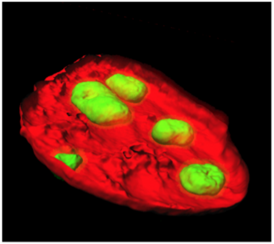
Data available

Biologists have collected a lot of data about proteins. e.g.,

- Gene expression measurements
- Phylogenetic profiles
- Location of proteins/enzymes in the cell

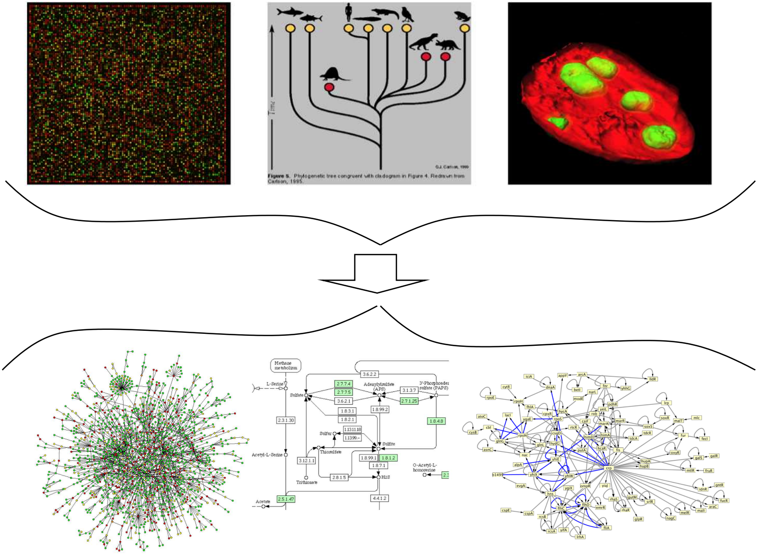






How to use this information "intelligently" to find a good function that predicts edges between nodes.

Our goal



Time
G.J. Carlson, 1999
Figure 5. Phylogenetic tree congruent with cladogram in Figure 4. Redrawn from Carlson, 1995.

More precisely

Formalization

- $\mathcal{V} = \{1, \ldots, N\}$ vertices (*e.g., genes, proteins*)
- $\mathcal{D} = (x_1, \ldots, x_N) \in \mathcal{H}^N$ data about the vertices ($\mathcal{H}$ *Hilbert space*)
- Goal: predict edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$.

"De novo" inference

- Given data about individual genes and proteins $\mathcal{D}$, ...
- ... Infer the edges between genes and proteins $\mathcal{E}$

"Supervised" inference

- Given data about individual genes and proteins $\mathcal{D}$, ...
- ... and given some known interactions $\mathcal{E}_{train} \subset \mathcal{E}$, ...
- ... infer unknown interactions $\mathcal{E}_{test} = \mathcal{E} \backslash \mathcal{E}_{train}$

More precisely

Formalization

- $\mathcal{V} = \{1, \ldots, N\}$ vertices (*e.g., genes, proteins*)
- $\mathcal{D} = (x_1, \ldots, x_N) \in \mathcal{H}^N$ data about the vertices ($\mathcal{H}$ *Hilbert space*)
- Goal: predict edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$.

"De novo" inference

- Given data about individual genes and proteins $\mathcal{D}$, ...
- ... Infer the edges between genes and proteins $\mathcal{E}$

"Supervised" inference

- Given data about individual genes and proteins $\mathcal{D}$, ...
- ... and given some known interactions $\mathcal{E}_{train} \subset \mathcal{E}$, ...
- ... infer unknown interactions $\mathcal{E}_{test} = \mathcal{E} \backslash \mathcal{E}_{train}$

More precisely

Formalization

- $\mathcal{V} = \{1, \ldots, N\}$ vertices (*e.g., genes, proteins*)
- $\mathcal{D} = (x_1, \ldots, x_N) \in \mathcal{H}^N$ data about the vertices ($\mathcal{H}$ *Hilbert space*)
- Goal: predict edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$.

"De novo" inference

- Given data about individual genes and proteins $\mathcal{D}$, ...
- ... Infer the edges between genes and proteins $\mathcal{E}$

"Supervised" inference

- Given data about individual genes and proteins $\mathcal{D}$, ...
- ... and given some known interactions $\mathcal{E}_{train} \subset \mathcal{E}$, ...
- ... infer unknown interactions $\mathcal{E}_{test} = \mathcal{E} \backslash \mathcal{E}_{train}$

Outline

1. De novo methods
2. Supervised methods
3. Conclusion

De novo methods

Typical strategies

- Fit a dynamical system to time series (e.g., PDE, boolean networks, state-space models)
- Detect statistical conditional independence or dependency (Bayesian netwok, mutual information networks, co-expression)

Pros

- Excellent approach if the model is correct and enough data are available
- Interpretability of the model
- Inclusion of prior knowledge

Cons

- Specific to particular data and networks
- Needs a correct model!
- Difficult integration of heterogeneous data
- Often needs a lot of data and long computation time

De novo methods

Typical strategies

- Fit a dynamical system to time series (e.g., PDE, boolean networks, state-space models)
- Detect statistical conditional independence or dependency (Bayesian netwok, mutual information networks, co-expression)

Pros

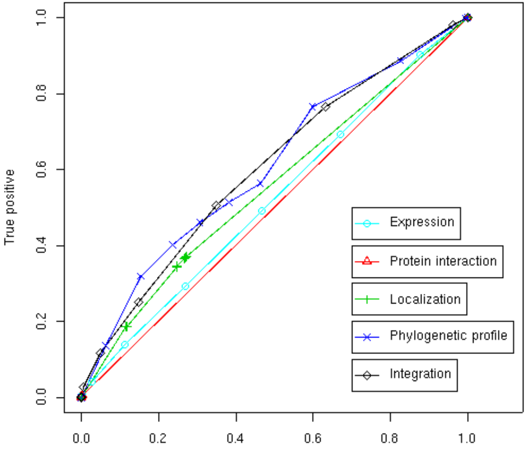
- Excellent approach if the model is correct and enough data are available
- Interpretability of the model
- Inclusion of prior knowledge

Cons

- Specific to particular data and networks
- Needs a correct model!
- Difficult integration of heterogeneous data
- Often needs a lot of data and long computation time

Evaluation on metabolic network reconstruction

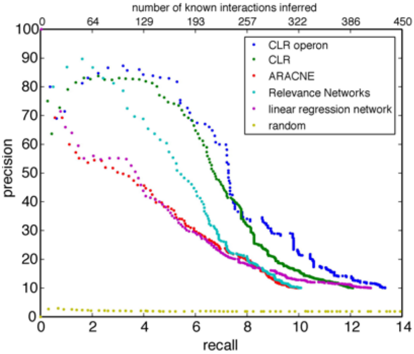
- The known metabolic network of the yeast involves 769 proteins.
- Predict edges from distances between a variety of genomic data (expression, localization, phylogenetic profiles, interactions).



OPEN ACCESS Freely available online PLOS BIOLOGY

Large-Scale Mapping and Validation of *Escherichia coli* Transcriptional Regulation from a Compendium of Expression Profiles

Jeremiah J. Faith[1], Boris Hayete[1], Joshua T. Thaden[2,3], Ilaria Mogno[2,4], Jamey Wierzbowski[2,5], Guillaume Cottarel[2,5], Simon Kasif[1,2], James J. Collins[1,2], Timothy S. Gardner[1,2*]



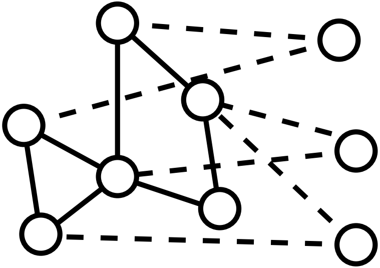
Outline

Supervised methods

Motivation

In actual applications,

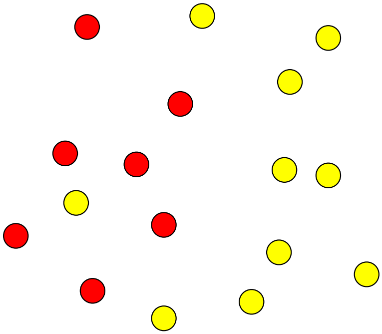
- we know in advance parts of the network to be inferred
- the problem is to add/remove nodes and edges using genomic data as side information



Supervised method

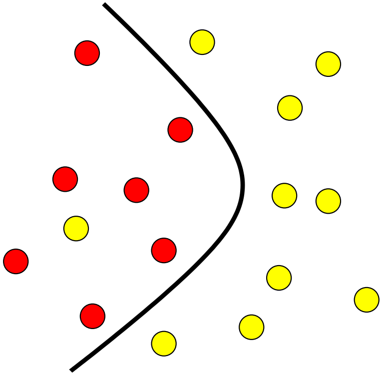
- Given genomic data and the currently known network...
- Infer missing edges between current nodes and additional nodes.

Pattern recognition



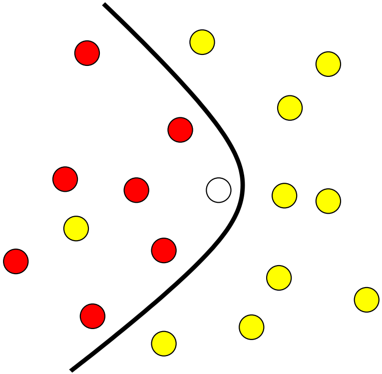
- Given a training set of patterns in two classes, learn to discriminate them
- Many algorithms (ANN, SVM, Decision tress, ...)

Pattern recognition



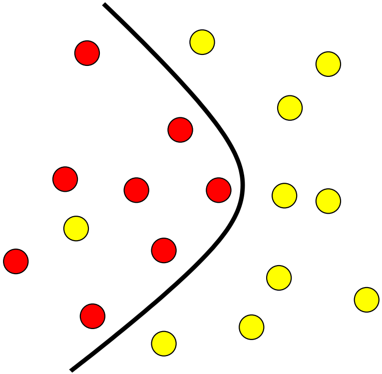
- Given a training set of patterns in two classes, learn to discriminate them
- Many algorithms (ANN, SVM, Decision tress, ...)

Pattern recognition



- Given a training set of patterns in two classes, learn to discriminate them
- Many algorithms (ANN, SVM, Decision tress, ...)

Pattern recognition



- Given a training set of patterns in two classes, learn to discriminate them
- Many algorithms (ANN, SVM, Decision tress, ...)

Pattern recognition and graph inference

Pattern recognition

Associate a binary label Y to each data X

Graph inference

Associate a binary label Y to each pair of data (X_1, X_2)

Two solutions

- Consider each pair (X_1, X_2) as a single data -> learning over pairs
- Reformulate the graph inference problem as a pattern recognition problem at the level of individual vertices -> local models

Pattern recognition and graph inference

Pattern recognition

Associate a binary label Y to each data X

Graph inference

Associate a binary label Y to each pair of data (X_1, X_2)

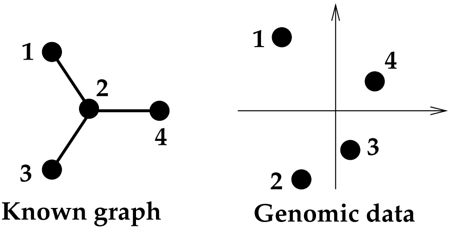
Two solutions

- Consider each pair (X_1, X_2) as a single data -> learning over pairs
- Reformulate the graph inference problem as a pattern recognition problem at the level of individual vertices -> local models

Pattern recognition for pairs

Formulation and basic issue

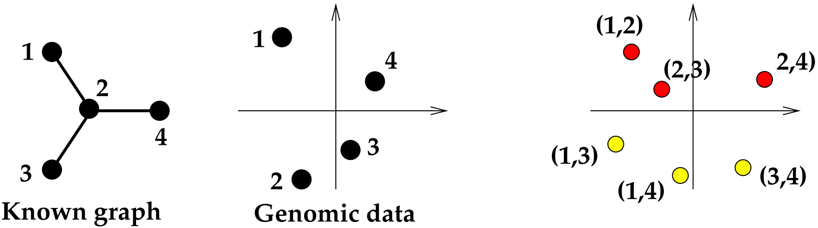
- A pair can be connected (1) or not connected (-1)
- From the known subgraph we can extract examples of connected and non-connected pairs
- However the genomic data characterize individual proteins; we need to work with pairs of proteins instead!



Pattern recognition for pairs

Formulation and basic issue

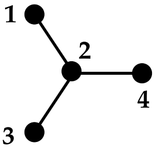
- A pair can be connected (1) or not connected (-1)
- From the known subgraph we can extract examples of connected and non-connected pairs
- However the genomic data characterize individual proteins; we need to work with pairs of proteins instead!



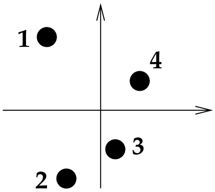
Pattern recognition for pairs

Formulation and basic issue

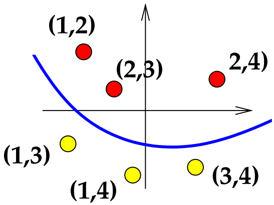
- A pair can be connected (1) or not connected (-1)
- From the known subgraph we can extract examples of connected and non-connected pairs
- However the genomic data characterize individual proteins; we need to work with pairs of proteins instead!



Known graph



Genomic data



Pattern recognition for pairs

Representing a pair as a vector

- Each individual protein is represented by a vector $v \in \mathbb{R}^p$
- We must represent a pair of proteins (u, v) by a vector $\psi(u, v) \in \mathbb{R}^q$ in order to estimate a linear classifier
- Question: how build $\psi(u, v)$ from u and v?

Representing a pair

Direct sum

- A simple idea is to concatenate the vectors u and v to obtain a $2p$-dimensional vector of (u, v):

$$\psi(u, v) = u \oplus v = \begin{pmatrix} u \\ v \end{pmatrix}.$$

- Problem: a linear function then becomes additive...

$$f(u, v) = w^\top \psi(u, v) = w_1^\top u + w^\top v.$$

Representing a pair

Direct sum

- A simple idea is to concatenate the vectors u and v to obtain a $2p$-dimensional vector of (u, v):

$$\psi(u,v) = u \oplus v = \begin{pmatrix} u \\ v \end{pmatrix} .$$

- Problem: a linear function then becomes additive...

$$f(u,v) = w^\top \psi(u,v) = w_1^\top u + w^\top v .$$

Representing a pair

Direct product

- Alternatively, make the direct product, i.e., the p^2-dimensional vector whose entries are all products of entries of u by entries of v:

$$\psi(u, v) = u \otimes v$$

- Problem: can get really large-dimensional...
- Good news: inner product factorizes:

$$(u_1 \otimes v_1)^\top (u_2 \otimes v_2) = \left(u_1^\top u_2\right) \times \left(v_1^\top v_2\right) ,$$

which is good for algorithms that use only inner products (SVM...)

Representing a pair

Direct product

- Alternatively, make the direct product, i.e., the p^2-dimensional vector whose entries are all products of entries of u by entries of v:

$$\psi(u, v) = u \otimes v$$

- Problem: can get really large-dimensional...
- Good news: inner product factorizes:

$$(u_1 \otimes v_1)^\top (u_2 \otimes v_2) = \left(u_1^\top u_2\right) \times \left(v_1^\top v_2\right) ,$$

which is good for algorithms that use only inner products (SVM...)

Representing a pair

Direct product

- Alternatively, make the direct product, i.e., the p^2-dimensional vector whose entries are all products of entries of u by entries of v:

$$\psi(u, v) = u \otimes v$$

- Problem: can get really large-dimensional...
- Good news: inner product factorizes:

$$(u_1 \otimes v_1)^\top (u_2 \otimes v_2) = \left(u_1^\top u_2\right) \times \left(v_1^\top v_2\right) ,$$

which is good for algorithms that use only inner products (SVM...)

Other representations for pairs

Symmetric tensor product (Ben-Hur and Noble, 2006)

$$\psi(u, v) = (u \otimes v) + (v \otimes u) \; .$$

Intuition: a pair (A, B) is similar to a pair (C, D) if:

- A is similar to C and B is similar to D, or...
- A is similar to D and B is similar to C

Metric learning (V. et al, 2007)

$$\psi(u, v) = (u - v)^{\otimes 2} \; .$$

Intuition: a pair (A, B) is similar to a pair (C, D) if:

- $A - B$ is similar to $C - D$, or...
- $A - B$ is similar to $D - C$.

Other representations for pairs

Symmetric tensor product (Ben-Hur and Noble, 2006)

$$\psi(u, v) = (u \otimes v) + (v \otimes u) .$$

Intuition: a pair (A, B) is similar to a pair (C, D) if:

- A is similar to C and B is similar to D, or...
- A is similar to D and B is similar to C

Metric learning (V. et al, 2007)

$$\psi(u, v) = (u - v)^{\otimes 2} .$$

Intuition: a pair (A, B) is similar to a pair (C, D) if:

- $A - B$ is similar to $C - D$, or...
- $A - B$ is similar to $D - C$.

Link with metric learning

Metric learning

For two vectors $u, v \in \mathcal{H}$ let the metric:

$$d_M(u,v) = (u-v)^\top M(u-v)\,.$$

Consider the problem:

$$\min_{M \geq 0} \sum_i l(u_i, v_i, y_i) + \lambda ||M||^2_{Frobenius}\,,$$

where l is a *hinge loss* to enforce:

$$d_M(u_i, v_i) \begin{cases} \leq 1-\gamma & \text{if}(u_i, v_i)\text{is connected}\,, \\ \geq 1+\gamma & \text{otherwise.} \end{cases}$$

Link with metric learning

Metric learning

For two vectors $u, v \in \mathcal{H}$ let the metric:

$$d_M(u,v) = (u-v)^\top M(u-v)\,.$$

Consider the problem:

$$\min_{M \geq 0} \sum_i l(u_i, v_i, y_i) + \lambda ||M||^2_{Frobenius}\,,$$

where l is a *hinge loss* to enforce:

$$d_M(u_i, v_i) \begin{cases} \leq 1-\gamma & \text{if}(u_i, v_i)\text{is connected}\,, \\ \geq 1+\gamma & \text{otherwise.} \end{cases}$$

Link with metric learning

Theorem (V. et al., 2007)

- A SVM with the representation

$$\psi(u, v) = (u - v)^{\otimes 2}$$

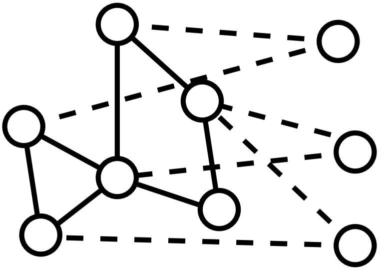
  solves this metric learning problem without the constraint $M \geq 0$.
- Equivalently, train the SVM over pairs with the metric learning pairwise kernel:

$$K_{MLPK}\left((u_1, v_1), (u_2, v_2)\right) = \psi(u_1, v_1)^\top \psi(u_2, v_2)$$
$$= \left[K(u_1, u_2) - K(u_1, v_2) - K(v_1, u_2) + K(u_2, v_2)\right]^2 .$$

Supervised inference with local models

The idea (Bleakley et al., 2007)

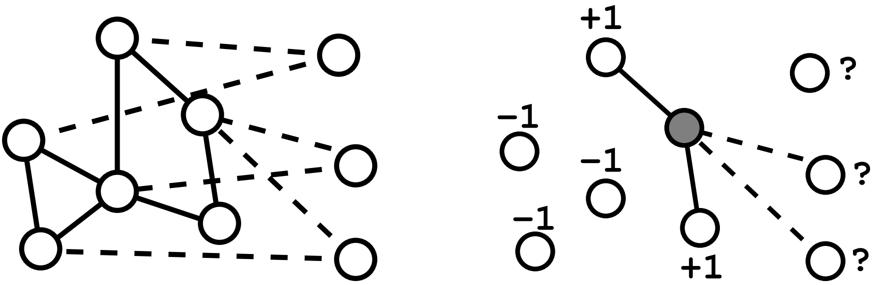
- Motivation: define specific models for each target node to discriminate between its neighbors and the others
- Treat each node independently from the other. Then combine predictions for ranking candidate edges.



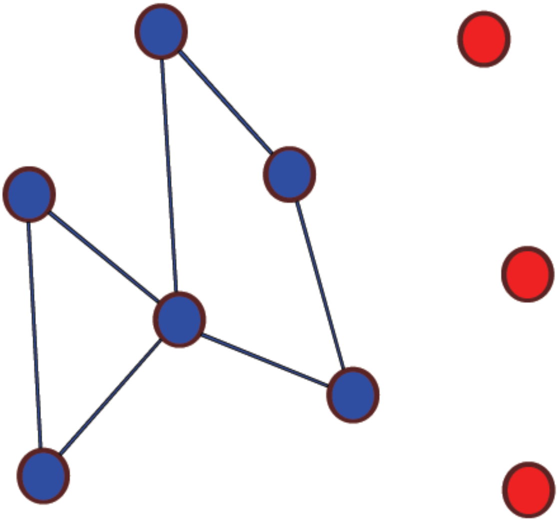
Supervised inference with local models

The idea (Bleakley et al., 2007)

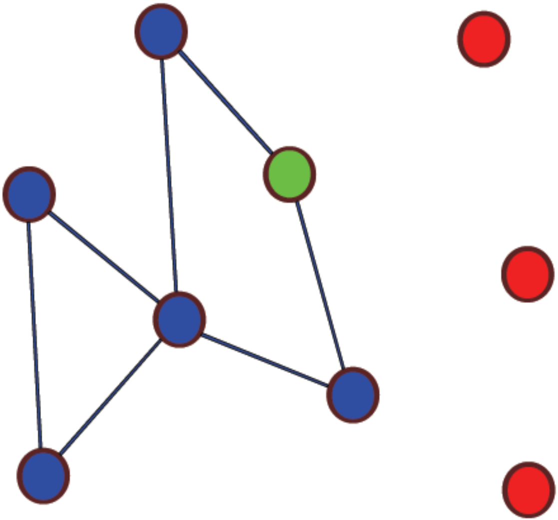
- Motivation: define specific models for each target node to discriminate between its neighbors and the others
- Treat each node independently from the other. Then combine predictions for ranking candidate edges.



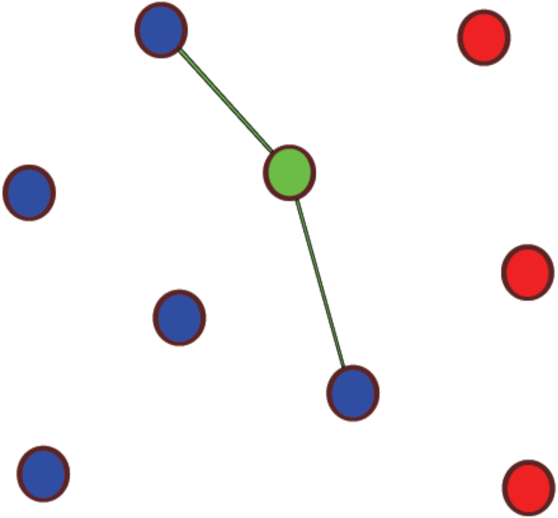
The LOCAL model



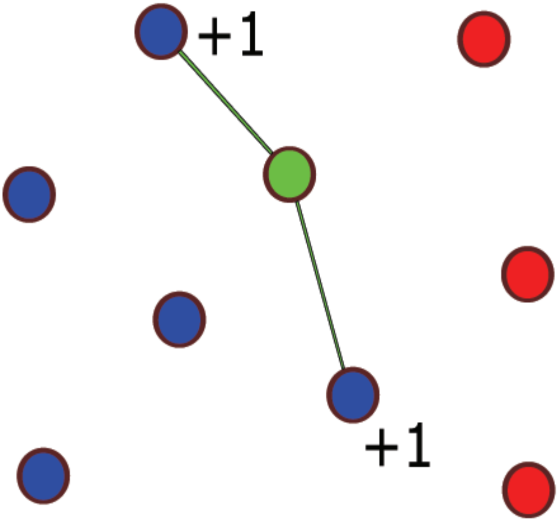
The LOCAL model



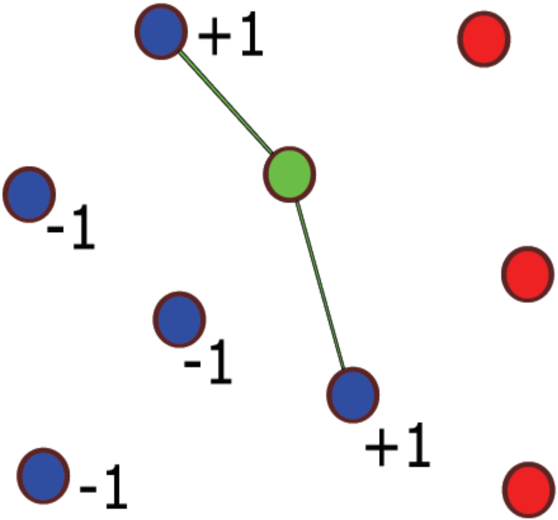
The LOCAL model



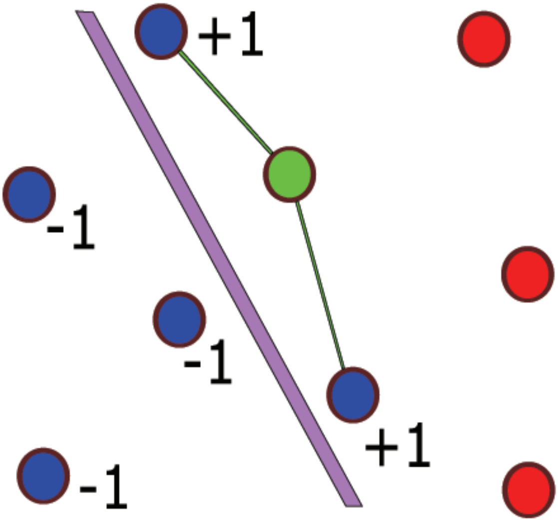
The LOCAL model



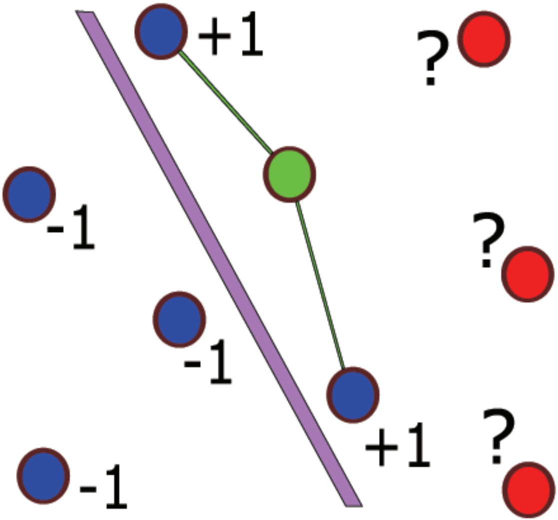
The LOCAL model



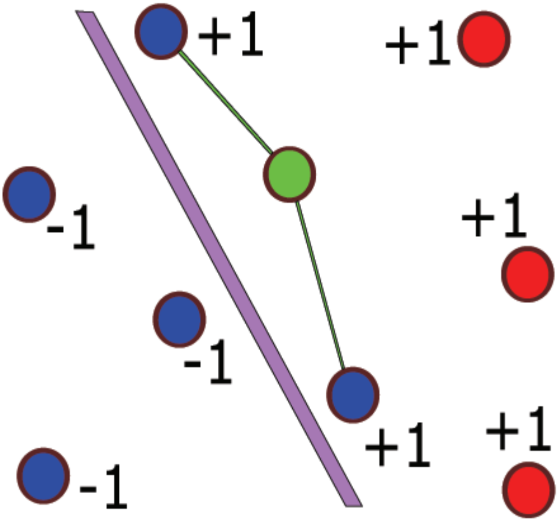
The LOCAL model



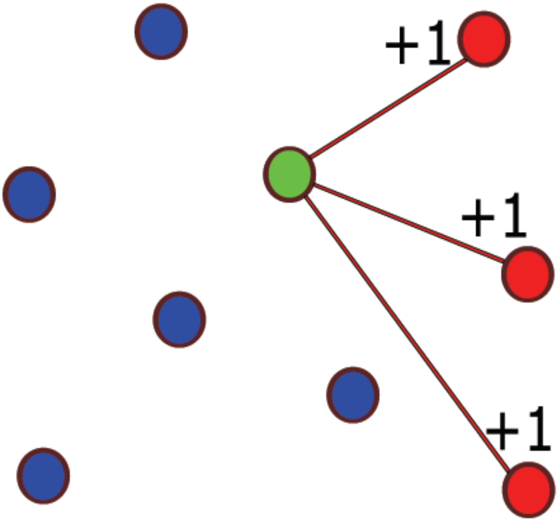
The LOCAL model



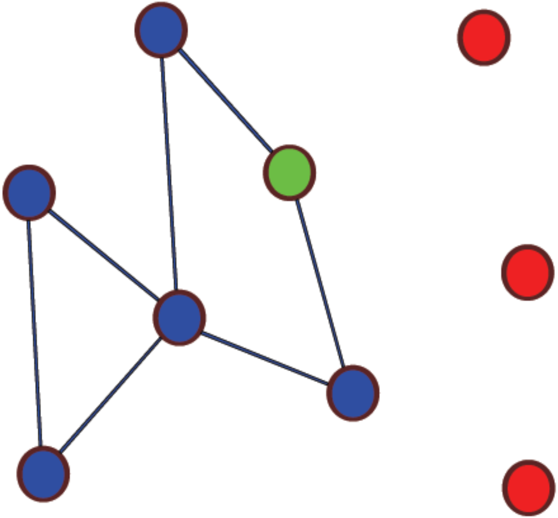
The LOCAL model



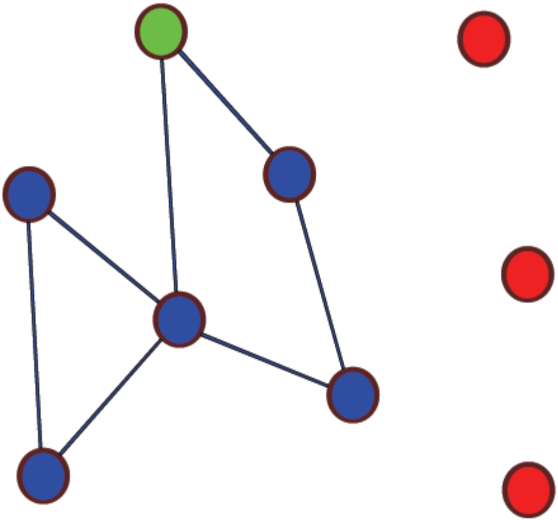
The LOCAL model



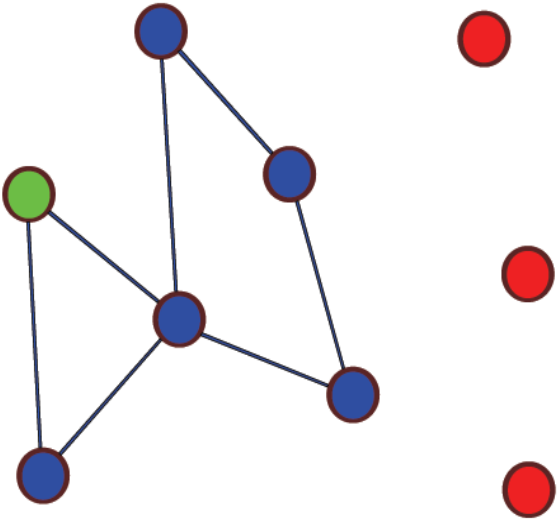
The LOCAL model



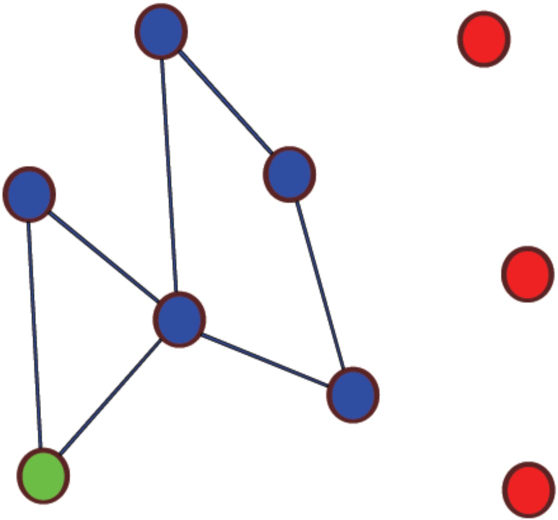
The LOCAL model



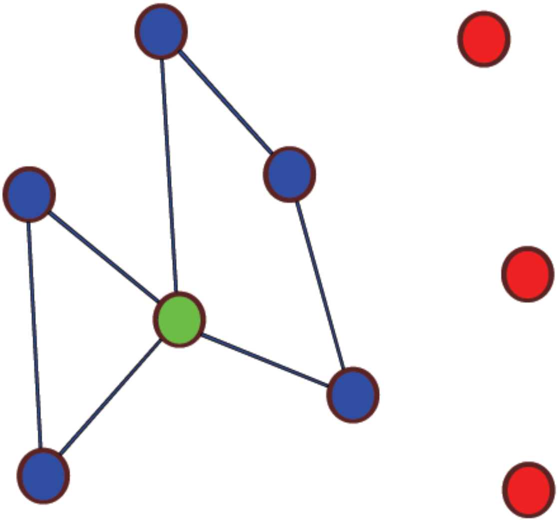
The LOCAL model



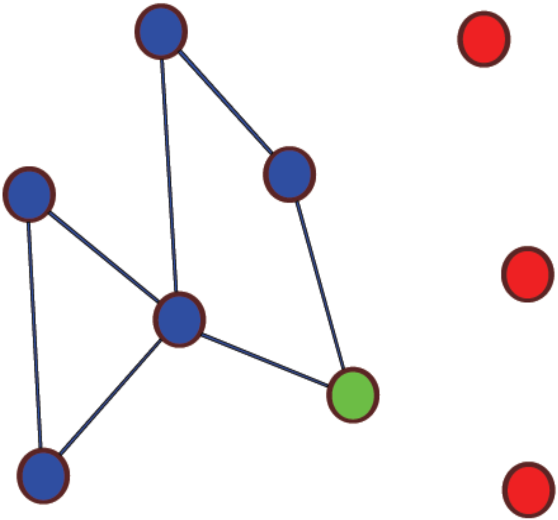
The LOCAL model



The LOCAL model



The LOCAL model



A few remarks

- Weak hypothesis:
  - if A is connected to B,
  - if C is similar to B,
  - then A is likely to be connected to C.
- Computationally: much faster to train N local models with N training points each, than to train 1 model with N^2 training points.
- Caveats:
  - each local model may have very few training points
  - no sharing of information between different local models

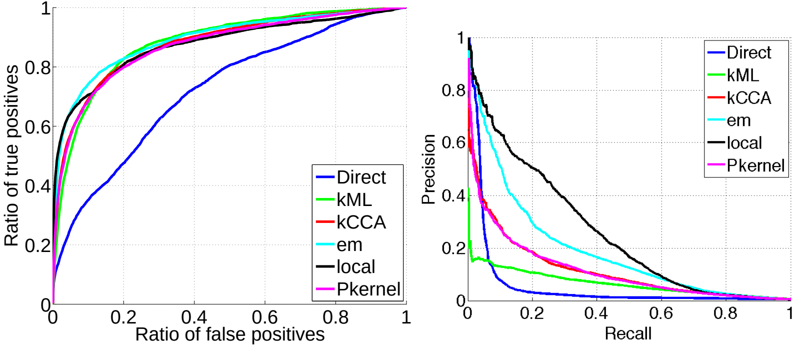
A few remarks

- Weak hypothesis:
  - if A is connected to B,
  - if C is similar to B,
  - then A is likely to be connected to C.
- Computationally: much faster to train N local models with N training points each, than to train 1 model with N^2 training points.
- Caveats:
  - each local model may have very few training points
  - no sharing of information between different local models

A few remarks

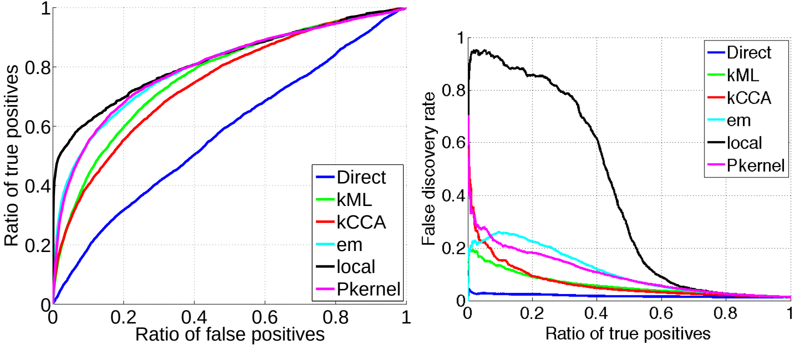
- Weak hypothesis:
  - if A is connected to B,
  - if C is similar to B,
  - then A is likely to be connected to C.
- Computationally: much faster to train N local models with N training points each, than to train 1 model with N^2 training points.
- Caveats:
  - each local model may have very few training points
  - no sharing of information between different local models

Results: protein-protein interaction (yeast)



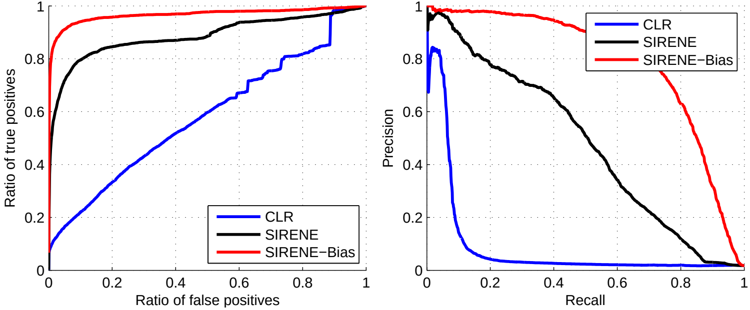
(from Bleakley et al., 2007)

Results: metabolic gene network (yeast)



(from Bleakley et al., 2007)

Results: regulatory network (E. coli)



Method	Recall at 60%	Recall at 80%
SIRENE	**44.5%**	**17.6%**
CLR	7.5%	5.5%
Relevance networks	4.7%	3.3%
ARACNe	1%	0%
Bayesian network	1%	0%

SIRENE = Supervised Inference of REgulatory NEtworks (Mordelet and V., 2008)

Applications: missing enzyme prediction

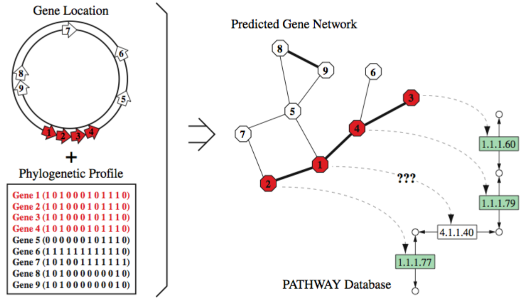
the FEBS Journal

Prediction of missing enzyme genes in a bacterial metabolic network

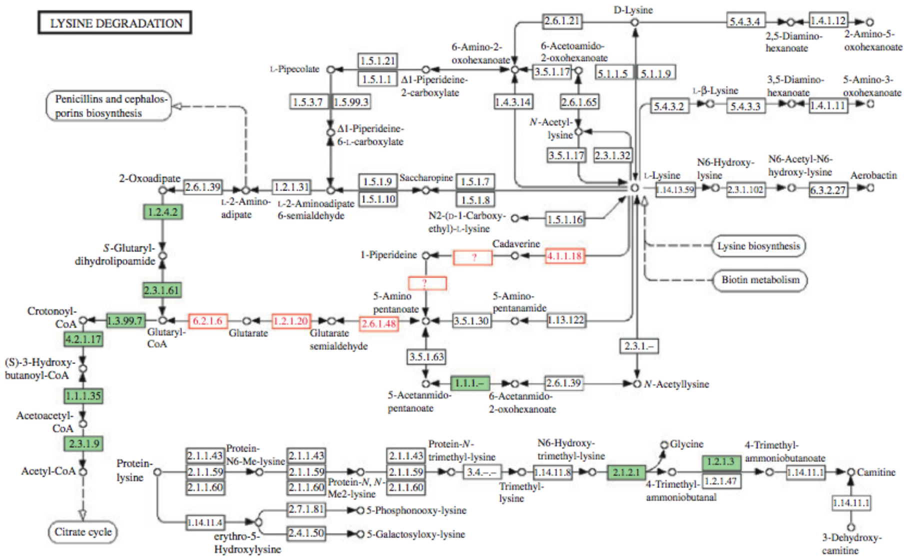
Reconstruction of the lysine-degradation pathway of *Pseudomonas aeruginosa*

Yoshihiro Yamanishi[1], Hisaaki Mihara[2], Motoharu Osaki[2], Hisashi Muramatsu[3], Nobuyoshi Esaki[2], Tetsuya Sato[1], Yoshiyuki Hizukuri[1], Susumu Goto[1] and Minoru Kanehisa[1]

1 Bioinformatics Center, Institute for Chemical Research, Kyoto University, Japan
2 Division of Environmental Chemistry, Institute for Chemical Research, Kyoto University, Japan
3 Department of Biology, Graduate School of Science, Osaka University, Japan



Applications: missing enzyme prediction



Applications: missing enzyme prediction

900 DOI 10.1002/pmic.200600862 *Proteomics* 2007, *7*, 900–909

RESEARCH ARTICLE

Prediction of nitrogen metabolism-related genes in *Anabaena* by kernel-based network analysis

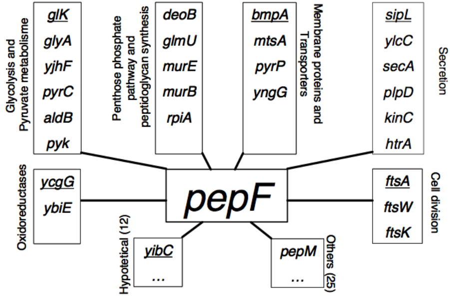
Shinobu Okamoto[1], Yoshihiro Yamanishi[1], Shigeki Ehira[2], Shuichi Kawashima[3], Koichiro Tonomura[1]** and Minoru Kanehisa[1]*

[1] Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji, Japan
[2] Department of Biochemistry and Molecular Biology, Faculty of Science, Saitama University, Saitama, Japan
[3] Human Genome Center, Institute of Medical Science, University of Tokyo, Meguro, Japan

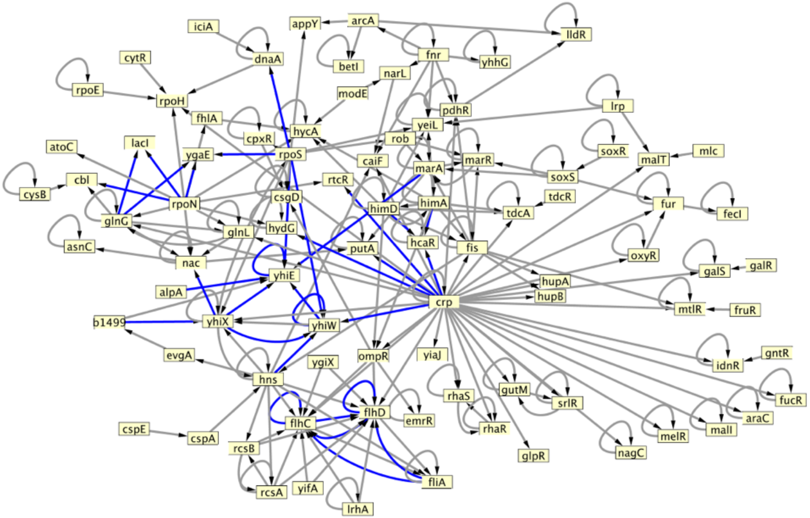
Determination of the role of the bacterial peptidase PepF by statistical inference and further experimental validation

Liliana LOPEZ KLEINE[1,2] , Alain TRUBUIL[1], Véronique MONNET[2]

[1]Unité de Mathématiques et Informatiques Appliquées. INRA Jouy en Josas 78352, France.
[2]Unité de Biochimie Bactérienne. INRA Jouy en Josas 78352, France.



Application: predicted regulatory network (E. coli)



Prediction at 60% precision, restricted to transcription factors (from Mordelet and V., 2008).

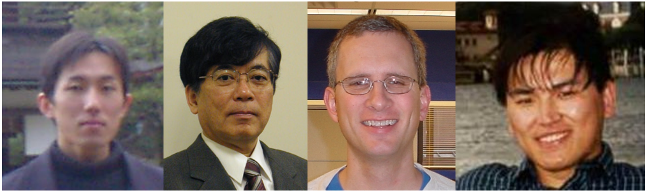
Outline

1. De novo methods
2. Supervised methods
3. Conclusion

Take-home messages

- When the network is known in part, supervised methods can be more adapted than unsupervised ones.
- A variety of methods have been investigated recently (metric learning, matrix completion, pattern recognition).
    - work for any network
    - work with any data
    - can integrate heterogeneous data, which strongly improves performance
- Current research: infer edges simultaneously with global constraints on the graph?

People I need to thank



- Yoshihiro Yamanishi, Minoru Kanehisa (Univ. Kyoto): kCCA, kML
- Jian Qian, Bill Noble (Univ. Washington): pairwise SVM
- Kevin Bleakley, Gerard Biau (Univ. Montpellier), Fantine Mordelet (ParisTech/Curie): local SVM