Learning in Social Networks

E. Viennet

Laboratoire de Traitement et Transport de l'Information
L2TI

Université Paris 13

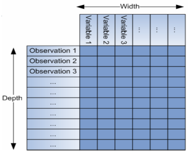
6/5/2009

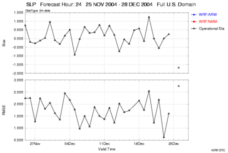
Agenda

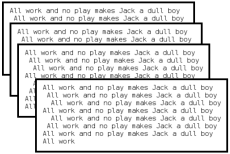
1. Introduction to Social Networks
2. Detection of communities in networks
3. Node classification
4. Kernel methods for graphs

Learning from data

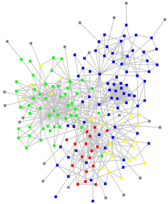
From tables to structured data...








```
<texte id="exemple">
 <titre>Un document</titre>
 <partie>
   <par>
     Ceci est la première partie
   </par>
 </partie>
 <partie>
  ...
```



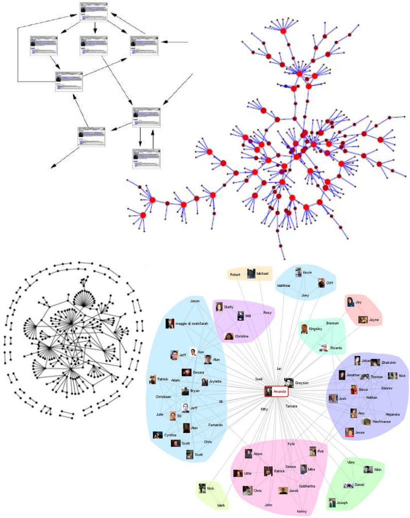
Models: classification, regression, clustering...

Data mining and social networks

Relations, interactions → structure

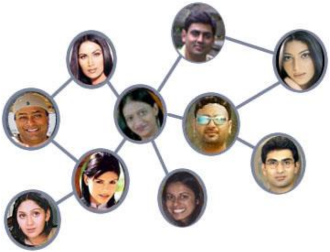
Examples:

- Web
- Semantic networks
- Electronic mail
- Instant messaging (IM)
- Forums
- Telecommunications (cellphones, ...)
- Biology



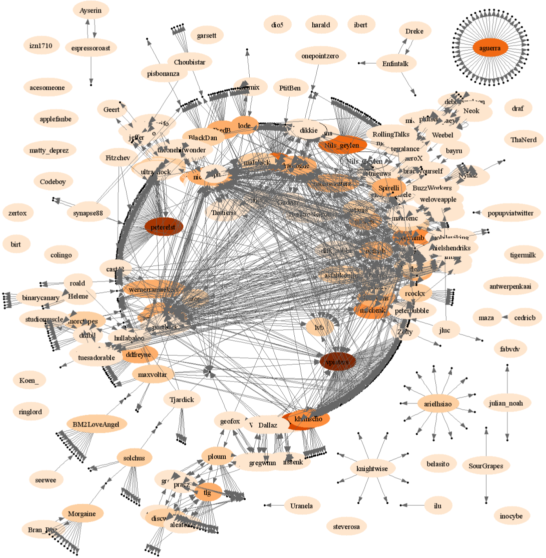
Social networks data is everywhere

- Call networks
- Email networks
- Movie networks
- Coauthor networks
- Affiliation networks
- Friendship networks
- Organizational networks



Firms increasingly are collecting **data** on explicit social networks of consumers

Another example: Twitter Social Network



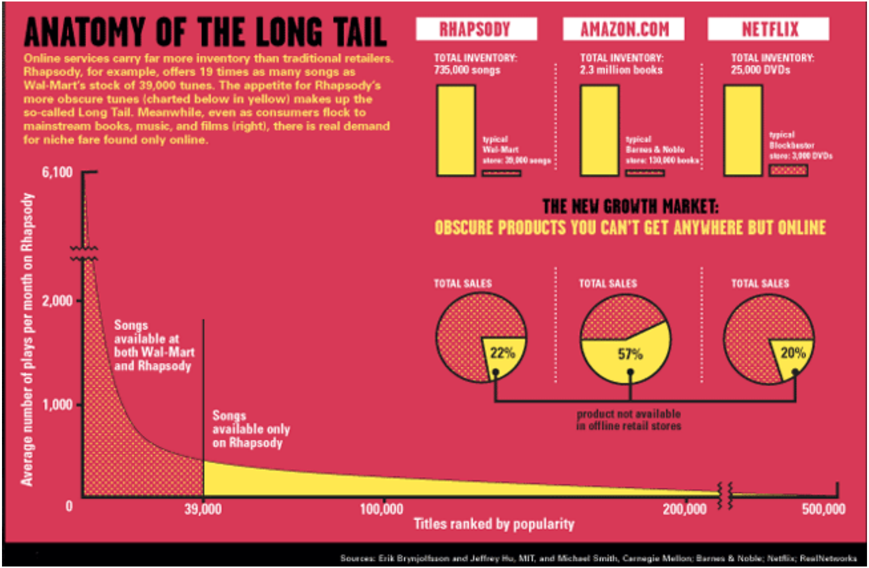
Twitter Friends van Belgische Twitteraars

(2007, Bruno Peeters, Belgium)

Applications & problems

- Social networks: community and structure (animation, targeted marketing)
- WWW: search, information retreival (group web sites or documents)
- Targeted marketing: identify groups of customers or products to make recommandations (targeted advertising, viral marketing)
- Personalization (interfaces, services)
- Epidemiology
- Fraud detection
- Security (counterterrorism)
- ...

Marketing & recommandation: the long tail



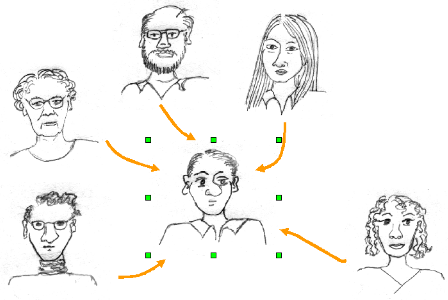
Chris Anderson, The Long Tail, Wired, Issue 12.10 - October 2004

Marketing, recommandation and SN

Need for personalized recommandations !

- $>$ 50% of people do research online before purchasing electronics
- personalized recommendations based on prior purchase patterns and ratings Amazon, “*people who bought x also bought y*”
  - MovieLens, “*based on ratings of users like you...*”
  - Epinions, “*based on the opinions of the raters you trust...*”

We are more influenced by our friends than by strangers !



68% of consumers consult friends and family before purchasing home electronics (Burke 2003)

Some interesting problems for data miners...

- Caracterize networks
- Model diffusion of information (for, e.g., viral marketing)
- Model evolution (link creation)
- Extract information for learning (node classification)

Our objectives today...

1. Give some insight about Social Network Analysis
2. Present some recent advances in community detection
3. Define the node classification problem
4. Show how to define *kernels* for graph data

Typical size of datasets used in the field

	Number of nodes
e-mails of a lab (2 months)	≈ 1000
e-mails (2 years)	≈ 50000
Friendship among bloggers	4.4 millions
Cellular phone calls (CDR)	$\approx$ 20 millions
IM communications	240 millions

Sparse networks: number of links *proportional* to the number of nodes.

What's different about networked data ?

A social netwok is a graph, but:

- nodes can have attributes
- edges (links) may be weighed and/or directed, or not
- so, the similarity between two nodes is $= f(\text{attributes}, \text{links})$
- the network's graph is not a simple random graph (special structural properties)

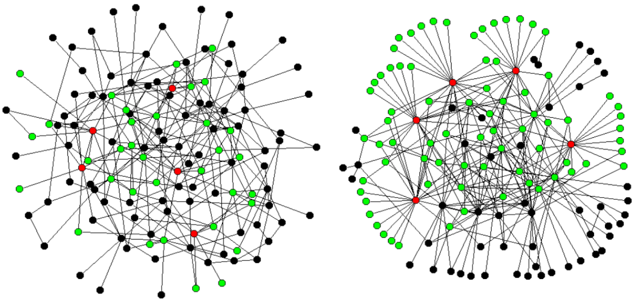
Nodes are not i.i.d. !

Small world effect

The shortest path between two random nodes is on average **small.**

This property is related to the distribution of the degrees of the nodes: *scale-free network* (Barabasi, 2000)

$$P(\text{degree} = k) \propto k^{-\gamma}$$



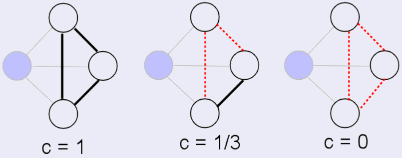
random graph scale-free graph

(Albert et al, 2000)

Common properties characterizing nodes or links

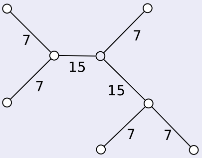
Clustering coefficient

Related to the number of neighbors of a node which are linked together (triangles) (Watts et Strogatz, 1998)



Betweenness

Number of *shortest paths* passing through a given edge (or node)

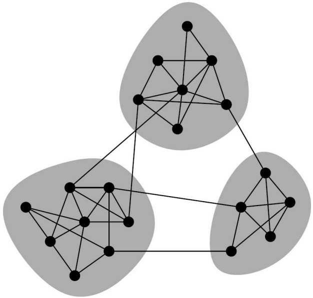


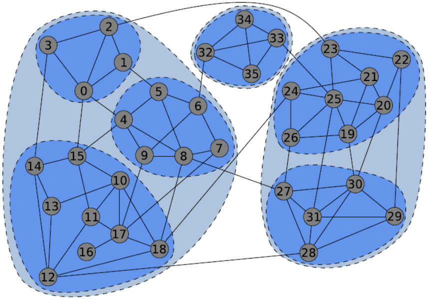
(Newman 2004)

Part 2

Detection of communities in networks

Communities in networks





(P. Pons, 2007)

- Finding communities = partition the graphe in *N* clusters
- Identify = finding the (small) communauty around a given node

Model-based clustering for social networks

Modelize simultanously the distribution of nodes attributes and positions in "*social space*": latent variable model

Representation of the social network

The matrix Y_{ij} describes the links between nodes.
$Z = z_i \in \mathbb{R}^d$ gives the positions of the nodes in *social space* $\mathbb{R}^d$ "social space".

Model-based clustering (continued): the model

Handcock & Raftery, 2006
n nodes, $Y = y_{ij}$ adjacency matrix ("sociomatrix").
Links are considered as independents:

$$P(Y|Z, X, \beta) = \prod_{i \neq j} P(y_{ij}|z_i, z_j, x_{ij}, \beta)$$

where

- X : attributes of nodes (or of pair (i, j))
- β : parameters of the model

Modelization by logistic regression:

$$logit(y_{ij} = 1|z_i, z_j, x_{ij}, \beta) = \beta_0^{\mathrm{T}} x_{ij} - \beta_1 |z_i - z_j|$$

with $\frac{1}{n} \sum_i |z_i|^2 = 1$

Model-based clustering (continued)

Clustering via modelization of the coordinates z_i by gaussian mixture:

$$z_i \propto \sum_{g=1}^{G} \lambda_g \exp(-\frac{|z_i - \mu_g|^2}{2\sigma_g^2}) \quad \text{with } \lambda_g > 0 \text{ and } \sum \lambda_g = 1$$

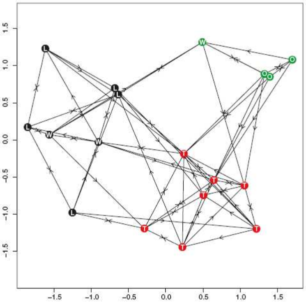
G number of clusters, fixed a priori

Estimation of parameters : maximum likelyhood or bayesian (markov chain or Monte Carlo)
$\leadsto$ *estimation is computationally costly*

Model-based clustering (continued): application

The choice of the number of clusters G can be posed as a model selection problem (e.g. BIC criteria) $\rightsquigarrow$ *slow !*

Links between monks



Sociological study: “friendship” between monks

18 nodes (monks)

$\rightsquigarrow$ 3 groups of monks (match those identified by sociologists)

Model-based clustering (continued): application 2

Links between teenagers in a school

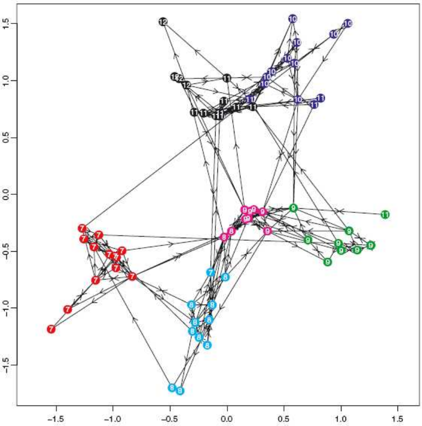


Fig. 7. Bayesian estimates of posterior clusters and latent positions for the friendship network in the adolescent health school: latent clusters are shown by colour with actual grades shown as numbers

Relations between 71 adolescents (here 6 clusters)

Model-based clustering: conclusions

- Complex methods (heavy computations) giving precise results
- Take in account both links and attributes at the same time
- Restricted to problems of small size !

$\Longrightarrow$ we will now focus on “structural” methods (using only links)

Criteria: Modularity

Mesure the quality of a clustering of the graph in c communities

$$Q = \sum_i (d_{ii} - (\sum_j d_{ij})^2)$$

D matrix $c \times c$, with elements d_{ij} giving the proportion of edges linking nodes from community i to nodes of community j

$Q \in [-1, 1]$ measures the density of links inside communities compared to links between communities

Finding structural communities

Lot of recent work and progress...

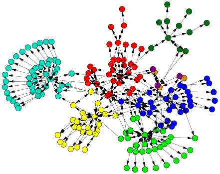
Méthods based on (*betweenness*)

First attempt: Newman & Girvan (2004)

- Repeat:
  1. compute betweeness of edges
  2. cut most important edge
- until no more edges

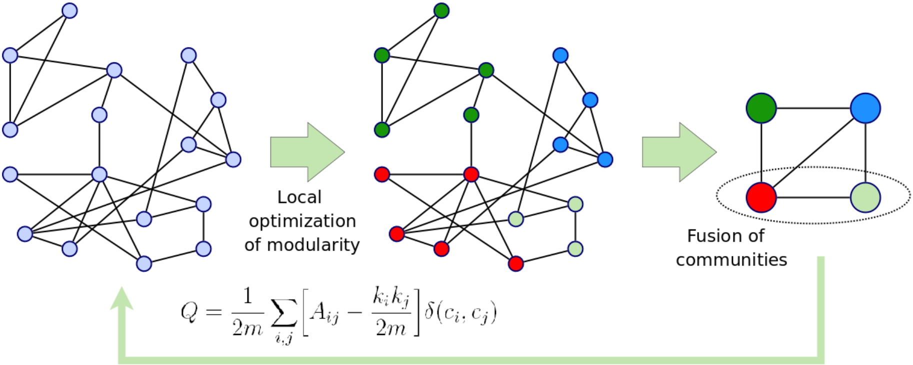
For a sparse graph of size n nodes:

Newman & Girvan	2004	$O(n^3)$
Newman	2004	$O(n^2)$
Wakita & Tsurumi	2007	$O(n \log^2 n)$
Blondel et al. (Louvain)	2008	linear ?



$\rightsquigarrow$ less than 5 minutes for 1 million nodes, or 40 minutes for 23 millions

Finding communities: Louvain method

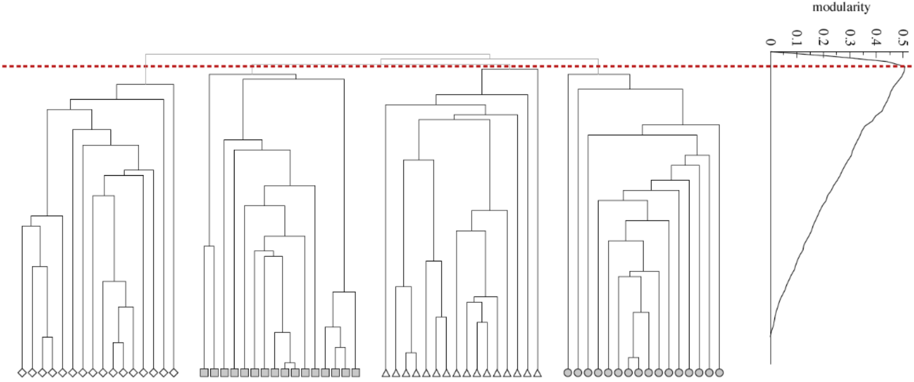


$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

Local optimization by switching labels considering only neighborhood of each node.

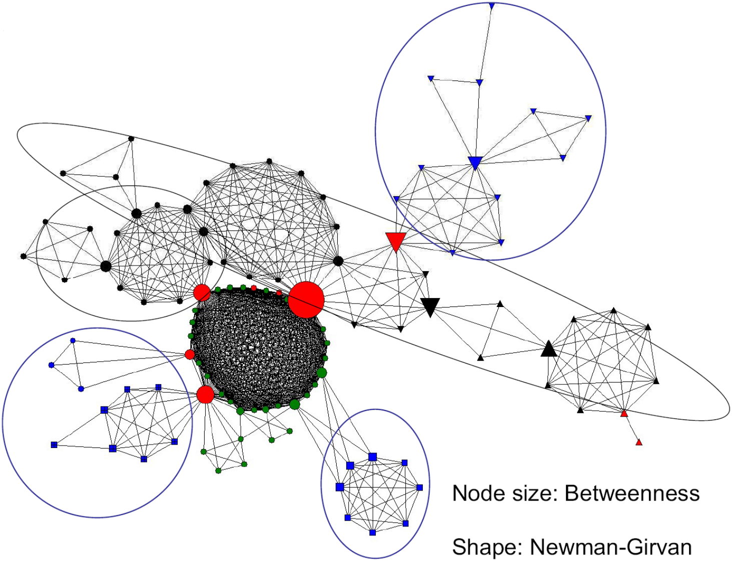
Blondel et al., Fast unfolding of communites in large networks, 2008

Hierarchical communities and modularity



From Newman & Girvan, 2004

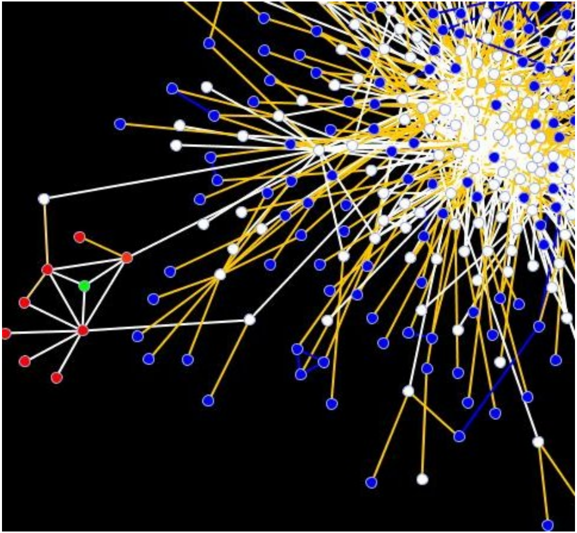
Example (scientists collaboration network)



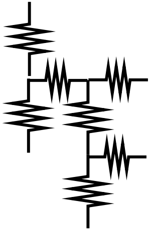
From K. Martin et M. Avnet, 2006.

Identification of communities

Look for a neighborhood (micro-community) around a given node



Identifying communities: a physical approach (Wu & Huberman)



Consider the graph as an electrical circuit
Kirchhoff's law on node C:

$$\sum_{i=1}^{n} I_i = \sum_{i=1}^{n} \frac{V_{D_i} - V_C}{R} = 0$$

If w_{ij} weight of edge, define $R_{ij} = w_{ij}^{-1}$
Fix the tension at two nodes: $V_1 = 1, V_2 = 0$ Then

$$V_i = \frac{1}{k_i} \sum_{j=3}^{n} V_j a_{ij} + \frac{1}{k_i} a_{i1} \quad \text{for } i = 3, \ldots, n$$

k_i : degre of node i, a_{ij} adjacency matrix
This linear equations system can be solved in $O(n^3)$ (slow).

Fast approximate solution

Iterative method:

1. fix $V_1 = 1, V_2 = \cdots = V_n = 0$ (in $O(V)$)
2. update tension of each node (in $O(E)$)
3. repeat step 22

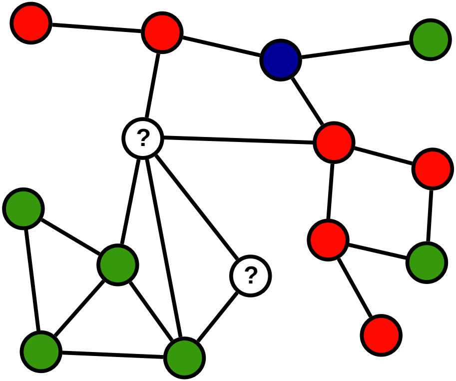
- Precision after step 2 depends only on the number of iteration, not on graph size
- In practice, convergence after about 10 iterations

Part 3

Node classification: learn from your neighboors...

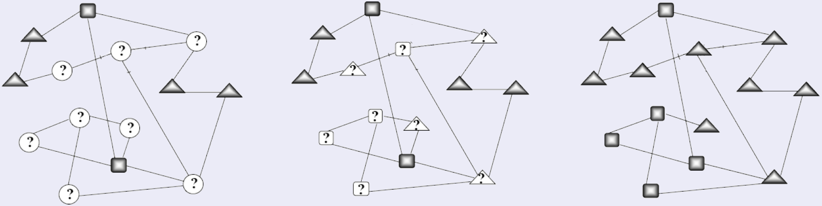
Node classification

Applications: marketing (churn, influence), text categorization, ...



Node classification

Relaxation labeling (Angelova et al 2006)



F1 score grows by 33% vs using only nodes attributes

=> importants gains on various applications

Node classification: a simple & fast approach

RL is slow on large graphs

Idea: to classify nodes based on attributes and "position" in graph, just add new attributes:

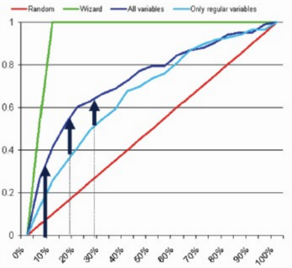
- local graph characteristics (see above: degree, triangles, ...)
- attributes describing the community to which the node belongs

Exemple: KXEN on Telco customers churn

Two models:

1. regular vars only
2. + social network vars

Most significant variable: number of "friends" who churned !



Node classification: a simple & fast approach

RL is slow on large graphs

Idea: to classify nodes based on attributes and "position" in graph, just add new attributes:

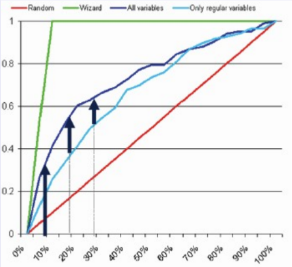
- local graph characteristics (see above: degree, triangles, ...)
- attributes describing the community to which the node belongs

Exemple: KXEN on Telco customers churn

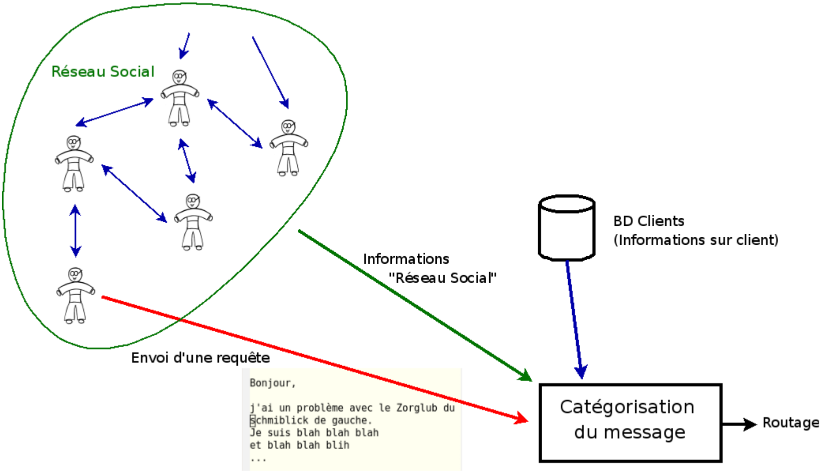
Two models:

1. regular vars only
2. + social network vars

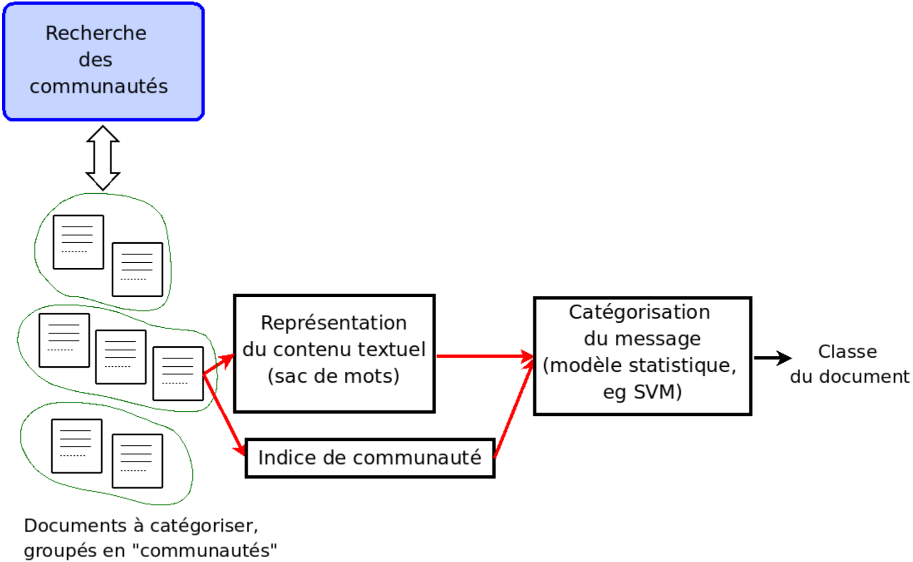
Most significant variable:
number of "friends"
who churned !



Example: text categorization



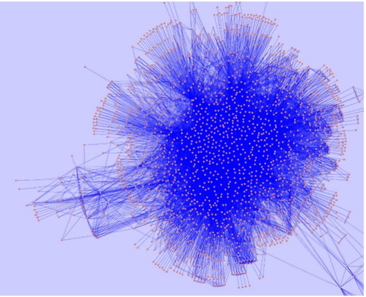
Text categorization (continued)



Application: bug triage (Bugzilla)

Bug tracker for Eclipse project

- Network of developpers
- 10 000 bug reports, 2100 users
- 50 000 links: users working on same bug
- Goal: associate the bug to a software developper



Level	Communities	Modularity
0	2081	0.01
1	229	0.26
2	16	0.36
3	14	0.37

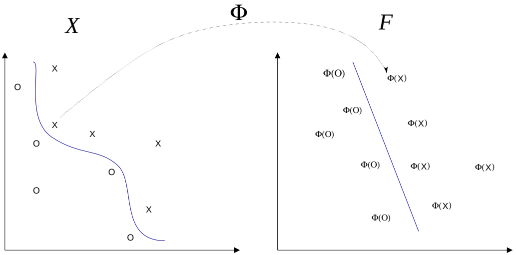
Method	**Performance**
TF-IDF $\rightarrow$ SVM	32%
TF-IDF + Author Community $\rightarrow$ SVM	38%

Part 4

Kernel methods for graphs

Feature space and kernels

Projection in feature space: transformation Φ



Kernel $K(x,y) = < \phi(x), \phi(y) >$

$$\text{Non linear SVM :}\ \hat{y} = \sum_{i \in SV} \alpha_i K(x_i, x) + b$$

$\Rightarrow$ "kernel trick" also used with a lot of models, like PCA, Discriminant Analysis, PLS, ...

$\Rightarrow$ can be applied to problems where no explicit vectorial representation of data points (strings of symbols, trees, ...)

Defining new kernels

Admissibility condition

- symetry: $k(x, y) = k(y, x)$
- semi-definite positive: $\sum \sum c_i c_j k(x_i, x_j) \geq 0$

On can define kernels based on existing kernels:

$$\text{combination:} \quad k(x, y) = \sum w_\alpha \, k_\alpha(x, y) \,, \forall w_\alpha \geq 0$$

$$\text{composition:} \quad k(x, y) = \sum \prod_{d=1}^{D} k_d(x_d, y_d) \quad \text{(Haussler 1999)}$$

Exemples: kernels for sequences, trees, graphs

A simple exemple: a kernel for trees

$$k(t, t') = \sum_{i=0}^{2} \sum_{j=0}^{1} k_c(c_i, c_j)$$

Defining new kernels

Admissibility condition

- symetry: $k(x, y) = k(y, x)$
- semi-definite positive: $\sum\sum c_i c_j k(x_i, x_j) \geq 0$

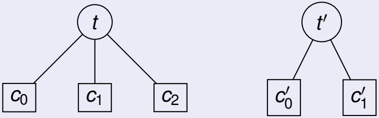
On can define kernels based on existing kernels:

$$\text{combination:} \quad k(x, y) = \sum w_\alpha \, k_\alpha(x, y) \; , \forall w_\alpha \geq 0$$

$$\text{composition:} \quad k(x, y) = \sum \prod_{d=1}^{D} k_d(x_d, y_d) \quad \text{(Haussler 1999)}$$

Exemples: kernels for sequences, trees, graphs

A simple exemple: a kernel for trees



$$k(t, t') = \sum_{i=0}^{2} \sum_{j=0}^{1} k_c(c_i, c_j)$$

Kernel for graph node categorization

K positive semi-definite:

$$\forall f_x, \sum_x \sum_{x'} f_x f_{x'} K(x, x') \geq 0$$

Following Haussler (1999), one can write:

$$e^{\beta H} = \lim_{n \to \infty} (1 + \frac{\beta H}{n})^n \quad (1)$$

$$= I + \beta H + \frac{\beta^2}{2!} H^2 + \cdots \quad (2)$$

H self-adjoint $\Rightarrow$ $K = e^{\beta H}$ positive semi-definite.

Parameter β controls the "locality" of the obtained kernel (diffusion on the graph).

Diffusion kernel

Graph Laplacian: $L = D - A$, $\quad L = \begin{cases} -1 & \text{si } i \sim j \\ d_i & \text{si } i = j \\ 0 & \text{sinon} \end{cases}$

Graph laplacians are often encountered in graph theory

$$\forall w, w^{\mathrm{T}} H w = \sum_{(i,j) \in E} (w_i - w_j)^2$$

Note:

$\frac{\partial}{\partial t}\Psi = \mu \Delta \Psi$: heat diffusion equation
If $K = e^{\beta H}$, on a $\frac{d}{d\beta} K_\beta = -L K_\beta$: heat diffusion on the graph (Kondor & Lafferty 2002).

$K_\beta(i,j)$ can be seen as the energy injected in i received in j, with diffusion parameter β

Diffusion kernel: implementation

$$
\begin{aligned}
K(0) &= I \\
K(\beta) &= \lim_{s \to \infty} \left(I + \frac{\beta L}{s} \right)^{s}
\end{aligned}
$$

Difficulty: K is a dense matrix, even if L is sparse

$\Rightarrow$ hard to use on large graphs

But interesting results have been obtained: exemple on "WebKB" dataset:

- 8275 web pages, 7 classes ($\neq$ universities)
- error rates varies from 8 to 15%, *ignoring page content (texts) !*

Also: applications to transductive learning (suggested by Gärtner et Smola 2007).

Summary

- SNA pose new challenges to the data mining community (non iid data, structure)
- New industrial applications leads to huge volumes of networked data, with a lot of value
- Designing new methods and algorithms is urgent !

Thank you !