A comparison between dissimilarity SOM & kernel SOM for clustering the vertices of a graph

Nathalie Villa-Vialaneix(1) Fabrice Rossi(2)

(1)Institut de Mathématiques de Toulouse, France - nathalie.villa@math.ups-tlse.fr

(2)Projet AxIS, INRIA Rocquencourt, France

September 5th, 2007

Modelization through large graphs

Various fields:

- Computer science: World Wide Web, P2P network...
- Social networks
- Biology: Protein interactions, Neuronal network,...
- Business, management: Transportation networks, Industry partnerships...

Modelization through large graphs

Various fields:

- Computer science: World Wide Web, P2P network...
- Social networks
- Biology: Protein interactions, Neuronal network,...
- Business, management: Transportation networks, Industry partnerships...

Question: Understanding the structure of these large graphs

- Clustering: building relevant homogeneous groups;
- Graph drawing: giving a global representation of the graph.

Modelization through large graphs

Various fields:

- Computer science: World Wide Web, P2P network...
- Social networks
- Biology: Protein interactions, Neuronal network,...
- Business, management: Transportation networks, Industry partnerships...

Question: Understanding the structure of these large graphs

- Clustering: building relevant homogeneous groups;
- Graph drawing: giving a global representation of the graph.

Here: **Self-Organizing Map for nonvectorial data.**

Table of contents

Table of contents

The data

We are given:

- **Data**: $x_1, x_2, \ldots, x_n$ from a (possibly) non vectorial space $\mathcal{G}$;

The data

We are given:

- **Data**: $x_1, x_2, \ldots, x_n$ from a (possibly) non vectorial space $\mathcal{G}$;
- **Kernel**: $k : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ such that k is:
  - symmetric: $k(x, x') = k(x', x)$;
  - positive definite: $\sum_{i,j=1}^m \alpha_i \alpha_j k(x_i, x_j) \geq 0$ for all $m \in \mathbb{N}$, all $(x_i)_i \in \mathcal{G}$ and all $(\alpha_i)_i \in \mathbb{R}$.

The data

We are given:

- **Data**: $x_1, x_2, \ldots, x_n$ from a (possibly) non vectorial space $\mathcal{G}$;
- **Kernel**: $k : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ such that k is:
  - symmetric: $k(x, x') = k(x', x)$;
  - positive definite: $\sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \geq 0$ for all $m \in \mathbb{N}$, all $(x_i)_i \in \mathcal{G}$ and all $(\alpha_i)_i \in \mathbb{R}$.

Main consequence:
$\exists$ Hilbert space $(\mathcal{H}, \langle ., . \rangle)$ and $\phi : \mathcal{G} \rightarrow \mathcal{H}$ such that:

$$k(x, x') = \langle \phi(x), \phi(x') \rangle.$$

The data

We are given:

- **Data**: $x_1, x_2, \ldots, x_n$ from a (possibly) non vectorial space $\mathcal{G}$;
- **Kernel**: $k : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$ such that k is:
  - symmetric: $k(x, x') = k(x', x)$;
  - positive definite: $\sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \geq 0$ for all $m \in \mathbb{N}$, all $(x_i)_i \in \mathcal{G}$ and all $(\alpha_i)_i \in \mathbb{R}$.

Main consequence:
$\exists$ Hilbert space $(\mathcal{H}, \langle ., . \rangle)$ and $\phi : \mathcal{G} \to \mathcal{H}$ such that:

$$k(x, x') = \langle \phi(x), \phi(x') \rangle.$$

Nonvectorial data are mapped onto a vectorial space where a SOM can be performed by the use of k as a scalar product.

kernel SOM [Lau *et al.*, 2006]: prototypes

Practical computation:

- **Prototype** of neuron i is in $\mathcal{H}$ and is of the form

$$p_i = \sum_{j=1}^{n} \gamma_{ji} \phi(x_j);$$

kernel SOM [Lau *et al.*, 2006]: prototypes

Practical computation:

- **Prototype** of neuron i is in $\mathcal{H}$ and is of the form

$$p_i = \sum_{j=1}^{n} \gamma_{ji} \phi(x_j);$$

- ϕ is **implicit** as $\forall\, i, j = 1, \ldots, n$,

$$\|\phi(x_i) - \phi(x_j)\|^2 = k(x_i, x_i) + k(x_j, x_j) - 2k(x_i, x_j);$$

kernel SOM: practical computation (online)

- **assignment step**: for x_l,

$$\arg\min_{j=1,\ldots,M}\left(\sum_{i=1}^{n}\gamma_{ij}k(x_l,x_i)-\sum_{i,i'=1}^{n}\gamma_{ij}\gamma_{i'j}k(x_i,x_{i'})\right)$$

kernel SOM: practical computation (online)

- **assignment step**: for x_l,

$$\arg\min_{j=1,\ldots,M}\left(\sum_{i=1}^{n}\gamma_{ij}k(x_l,x_i)-\sum_{i,i'=1}^{n}\gamma_{ij}\gamma_{i'j}k(x_i,x_{i'})\right)$$

- **representation step**: $p_i^l=\sum_{j=1}^{n}\gamma_{ji}^l\phi(x_j)$:

$$\gamma_{ji}^l=\gamma_{ji}^{l-1}+\alpha(l)h(f^l(x_l),j)\left(\delta_{il}-\gamma_{ji}^{l-1}\right)$$

Another assumptions for the data set

We are given:

- **Data**: $x_1, x_2, \ldots, x_n$ from a (possibly) non vectorial space $\mathcal{G}$;
- **A dissimilarity measure**: $\delta : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ such that:
  - δ is symmetric: $\delta(x, x') = \delta(x', x)$;
  - δ is positive: $\delta(x, x') \geq 0$;
  - $\delta(x, x) = 0$.

Another assumptions for the data set

We are given:

- **Data**: $x_1, x_2, \ldots, x_n$ from a (possibly) non vectorial space $\mathcal{G}$;
- **A dissimilarity measure**: $\delta : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$ such that:
  - δ is symmetric: $\delta(x, x') = \delta(x', x)$;
  - δ is positive: $\delta(x, x') \geq 0$;
  - $\delta(x, x) = 0$.

Adaptation for SOM:

- The **prototypes** are one of the elements of the data set $(x_i)_{i=1,\ldots,n}$;

Another assumptions for the data set

We are given:

- **Data**: $x_1, x_2, \ldots, x_n$ from a (possibly) non vectorial space $\mathcal{G}$;
- **A dissimilarity measure**: $\delta : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ such that:
  - δ is symmetric: $\delta(x, x') = \delta(x', x)$;
  - δ is positive: $\delta(x, x') \geq 0$;
  - $\delta(x, x) = 0$.

Adaptation for SOM:

- The **prototypes** are one of the elements of the data set $(x_i)_{i=1,\ldots,n}$;
- δ is used **instead of the usual distance**.

Dissimilarity SOM: practical computation

- **assignment step**: for x_i,

$$\arg\min_{j=1,\ldots,M} \delta(x_i, p_j^{l-1})$$

- **representation step**:

$$p_j^l = \arg\min_{x\in(x_{i'})_{i'=1,\ldots,n}} \sum_{i=1}^{n} h(f^l(x_i), j)\delta(x_i, x)$$

Link with kernel SOM

A **dissimilarity built from a kernel** k:

$$\begin{aligned}\delta_{\text{mean}}(x,x') &= \|\phi(x)-\phi(x')\|^2\\ &= k(x,x)+k(x',x')-2k(x,x').\end{aligned}$$

Link with kernel SOM

A **dissimilarity built from a kernel** k:

$$\begin{aligned}\delta_{\text{mean}}(x, x') &= \|\phi(x) - \phi(x')\|^2 \\ &= k(x, x) + k(x', x') - 2k(x, x').\end{aligned}$$

Consequence: By generalizing dissimilarity SOM to the case where the prototype of neuron i is of the form

$$p_j = \sum_{i=1}^{n} \gamma_{ji} \phi(x_i);$$

we can derive a **batch version** of kernel SOM from dissimilarity SOM.

From dissimilarity SOM to batch kernel SOM

Assignment step

for x_i,

$$\arg\min_{j=1,\ldots,M} \delta(x_i, p_j^{l-1})$$

Representation step

$$p_j^l = \arg\min_{x \in (x_{i'})_{i'=1,\ldots,n}} \sum_{i=1}^{n} h(f^l(x_i), j)\delta(x_i, x)$$

From dissimilarity SOM to batch kernel SOM

Assignment step

for x_i,

$$\arg\min_{j=1,\ldots,M} \left\| x_i - \sum_{i=1}^{n} \gamma_{ji}\phi(x_i) \right\|$$

Representation step

$$\gamma_j^l = \arg\min_{\gamma\in\mathbb{R}^n} \sum_{i=1}^{n} h(f^l(x_i), j) \left\| x_i - \sum_{l'=1}^{n} \gamma_{l'}\phi(x_{l'}) \right\|^2$$

From dissimilarity SOM to batch kernel SOM

Assignment step

for x_i,

$$\arg\min_{j=1,\dots,M} \sum_{u,u'=1}^{n} \gamma_{ju}\gamma_{ju'} k(x_u, x_{u'}) - 2\sum_{u=1}^{n} \gamma_{ju} k(x_u, x_i)$$

Representation step

$$\gamma_{ji}^{l} = \frac{h(f^l(x_i), j))}{\sum_{i'=1}^{n} h(f^l(x_{i'}, j))}$$

Table of contents

Laplacian and diffusion kernel

Definitions

For a graph with vertices $V = \{x_1, \ldots, x_n\}$ having positive weights $(w_{i,j})_{i,j=1,\ldots,n}$ such that, for all $i, j = 1, \ldots, n$, $w_{i,j} = w_{j,i}$ and $d_i = \sum_{j=1}^{n} w_{i,j}$,

- **Laplacian**: $L = (L_{i,j})_{i,j=1,\ldots,n}$ where

$$L_{i,j} = \begin{cases} -w_{i,j} & \text{if } i \neq j \\ d_i & \text{if } i = j \end{cases};$$

Laplacian and diffusion kernel

Definitions

For a graph with vertices $V = \{x_1, \ldots, x_n\}$ having positive weights $(w_{i,j})_{i,j=1,\ldots,n}$ such that, for all $i, j = 1, \ldots, n$, $w_{i,j} = w_{j,i}$ and $d_i = \sum_{j=1}^n w_{i,j}$,

- **Laplacian**: $L = (L_{i,j})_{i,j=1,\ldots,n}$ where

$$L_{i,j} = \begin{cases} -w_{i,j} & \text{if } i \neq j \\ d_i & \text{if } i = j \end{cases} ;$$

- **Regularization: the diffusion matrix**: for $\beta > 0$, $K^\beta = e^{-\beta L}$.

Laplacian and diffusion kernel

Definitions

For a graph with vertices $V = \{x_1, \ldots, x_n\}$ having positive weights $(w_{i,j})_{i,j=1,\ldots,n}$ such that, for all $i, j = 1, \ldots, n$, $w_{i,j} = w_{j,i}$ and $d_i = \sum_{j=1}^n w_{i,j}$,

- **Laplacian**: $L = (L_{i,j})_{i,j=1,\ldots,n}$ where

$$L_{i,j} = \begin{cases} -w_{i,j} & \text{if } i \neq j \\ d_i & \text{if } i = j \end{cases};$$

- **Regularization: the diffusion matrix**: for $\beta > 0$, $K^\beta = e^{-\beta L}$.

$\Rightarrow$

$$\begin{array}{rccc} k^\beta : & V \times V & \rightarrow & \mathbb{R} \\ & (x_i, x_j) & \rightarrow & K^\beta_{i,j} \end{array}$$

is the **diffusion kernel** (or heat kernel).

Properties

1. **Diffusion on the graph**: $k^\beta(x_i, x_j) \simeq$ quantity of energy accumulated in x_j after a given time if energy is injected in x_i at time 0 and if diffusion is done along the edges.
   $\beta \simeq$ intensity of diffusion;

Properties

1. **Diffusion on the graph**: $k^\beta(x_i, x_j) \simeq$ quantity of energy accumulated in x_j after a given time if energy is injected in x_i at time 0 and if diffusion is done along the edges.
   $\beta \simeq$ intensity of diffusion;
2. **Regularization operator**: for $u \in \mathbb{R}^n \sim V$, $u^T K^\beta u$ is higher for vectors u that vary a lot over “close” vertices of the graph.
   $\beta \simeq$ intensity of regularization (for small β, direct neighbors are more important);

Table of contents

Simulated data

Random generation

50 non-weighted graphs having:

- 5 cliques $(\mathcal{C}_i)_{i=1,\ldots,5}$ with $(n_i)_{i=1,\ldots,5}$ vertices where $n_i \sim \mathcal{P}(50)$;
- for $i = 1, \ldots, 5$, l_i links between $\mathcal{C}_i$ and the other vertices $l_i \sim \mathcal{U}(\{1, \ldots, 100n_i\})$.

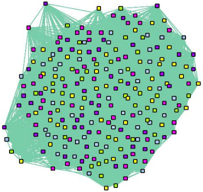
Simulated data

Random generation

50 non-weighted graphs having:

- 5 cliques $(\mathcal{C}_i)_{i=1,\dots,5}$ with $(n_i)_{i=1,\dots,5}$ vertices where $n_i \sim \mathcal{P}(50)$;
- for $i = 1, \dots, 5$, l_i links between $\mathcal{C}_i$ and the other vertices $l_i \sim \mathcal{U}(\{1, \dots, 100n_i\})$.

Example:



Number of vertices: 237
Total of degrees: 12 924
Diameter: 2
Density$\simeq$ 0.46

Compared results ($\beta = 0.1$)

	k-SOM (online)	d-SOM	k-SOM (batch)
Mean energy	**0.10**	0.13	**0.10**
Mean nb of classes	8.04	9	**6.56**
Mean % of good classif.	79.84	77.89	**94.34**
Time of computation (min)	260	80	20

Compared results ($\beta = 0.1$)

	k-SOM (online)	d-SOM	k-SOM (batch)
Mean energy	**0.10**	0.13	**0.10**
Mean nb of classes	8.04	9	**6.56**
Mean % of good classif.	79.84	77.89	**94.34**
Time of computation (min)	260	80	20

	94 %	
	6 %	2 %
		98 %

A medieval social network [Villa *et al.*, 2007]

Graph built from a large corpus of medieval contracts

From a corpus of 1000 agrarian contracts made in South West of France (1250-1350), we built a weighted graph:

- **vertices**: peasants found in the contracts;
- **edges**: number of contracts where two peasants are mentionned together.

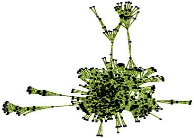
A medieval social network [Villa *et al.*, 2007]

Graph built from a large corpus of medieval contracts

From a corpus
of 1000 agrarian contracts made in South West
of France (1250-1350), we built a weighted graph:

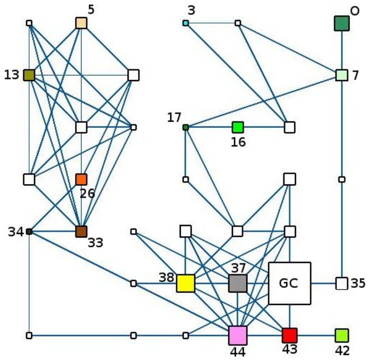
- **vertices**: peasants found in the contracts;
- **edges**: number of contracts where two peasants are mentionned together.

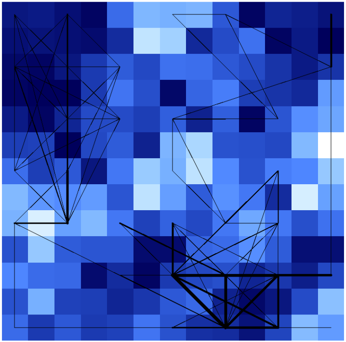
Example:



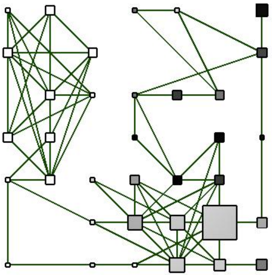
Number of vertices: 515
Number of edges: 4193
Total of weights: 40 329
Diameter: 10
Density: 2,2%

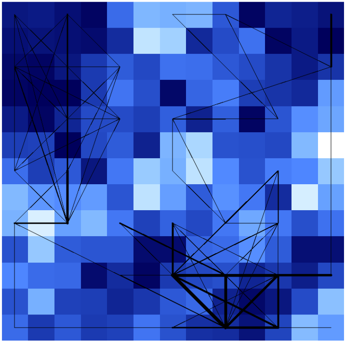
Results on a 7×7 rectangular map





Results on a 7×7 rectangular map





References

Lau, K., Yin, H. & Hubbard, S. (2006).
Kernel self-organising maps for classification.
Neurocomputing, **69**, 2033–2040.

Villa, N., Boulet, R., Rossi, F. & Jouve, B. (2007).
Mining a medieval social network by spectral clustering and kernel self-organizing map.
Submitted.