svcR: An R Package for Support Vector Clustering improved with Geometric Hashing applied to Lexical Pattern Discovery

Nicolas Turenne
INRA, Paris

Abstract

We present a new R package which takes a numerical matrix format as data input, and computes clusters using a support vector clustering method (SVC). We have implemented an original 2D-grid labeling approach to speed up cluster extraction. In this sense, SVC can be seen as an efficient cluster extraction if clusters are separable in a 2-D map. Secondly we showed that this SVC approach using a Jaccard-Radial base kernel can help to classify well enough a set of terms into ontological classes and help to define regular expression rules for information extraction in documents; our case study concerns a set of terms and documents about developmental and molecular biology.

Keywords: unsupervised learning, support vector clustering, lexical clustering, pattern discovery, grid-based labeling, ontology, terminology, jaccard-radial kernel.

1. Introduction

Mining text archives is a great challenge since lots of documents are available and their amount grows in the same way as the capacity of computer storage. Making rules of a domain for knowledge extraction involves efficient features with low semantic ambiguity. It is not an easy task and we try to answer this question by representing vectors of linguistic expressions (i.e. terms) by features and using a scalable density-based distance to cluster the terms.
The first idea for our problem concerns the choice of a density-based method and the improvement of its scalability. Clustering can be a useful knowledge-poor technique to induce organization into scattered data (Jain and Dubes 1988). Non-parametric methods such as support vector machines can be interesting to analyze noisy data by density processing. (Ben-Hur, Horn, Siegelmann, and Vapnik 2001; Schölkopf, Platt, Shawe-Taylor, Smola, and Williamson 2001; Horn 2001) proposed an unsupervised support vector algorithm to enclose data clusters by contours and based it on a radial kernel. Diverse applications have been tested for novelty detection (Eskin, Arnold, Prerau, Portnoy, and Stolfo 2002; Lazarevic, Ozgur, Ertoz, Srivastava, and Kumar 2003), rule extraction (Zhang, Su, Jia, and Chu 2005), désoxyribo-nucleic acid (DNA) and chemical compounds (Bilen 2005; Eveillard and Guermeur 2002) or image processing (Campedel and Moulines 2005). The method of point assignation to contours and related clusters is based on adjacent points between each pair and is time-consuming. Some studies (Park, Ji, Zha, and Kasturi 2004; Puma-Villanueva, Bezerra, Lima, and Zuben 2005) have been proposed to speed-up the method. In particular Yang, Estivill-Castro, and Chalup (2002) and Lee and Lee (2005) proposed an improved method to label clusters, i.e. to assign point to clusters by graph analysis.
We present a new robust method based on the computation of a hash function through surrounding points working with a grid which we map to data using a k-nearest neighbor method. We developed this clustering method under the R platform R Development Core Team (2010), as a package called **svcR**, and we compared our approach to other ones, especially graph-based, on the Iris dataset (**svcR** is available from the Comprehensive R Archive Network at http:

//CRAN.R-project.org/package=svcR"). Kees, Marchiori, and van der Vaart (2003) have also developed a support vector method for clustering but using a divisive way iteratively searching a classical hyperplan separator based on classical support vector machine. The first step tries to separate the data set and a set artificially build in the same space of attribute values and the same size than the data set which is theoretically not justified; it seems that if not many classes are present (2 or 3) and not many attributes describe data, the algorithm seems to find groups, in other cases it tends to find a number of final clusters equal to the number of iterations.
The second idea presented in this paper concerns our original usage of support vector clustering (SVC) clustering methodology cited above for solving a certain form of ambiguity in natural language. Information retrieval (Salton and McGill 1983; Daille 2003) and information extraction (Kushmerick 2000; Soderland 1999) are key methodologies to retrieve information from text archives. But simple keywords may have several senses and assignment of term to conceptual classes should be important (Gale, Church, and Yarowsky 1992; Hearst 1992; Riloff 1993; Grefenstette 1994; Fellbaum 1998). Clustering may be used to reduce the number of variables to take into account in rules for information mining in documents. We base our assumption on two works. Firstly that collocation analysis is useful to understand morphological structure and its link to a conceptual space (Harris 1968; Smadja and McKeown 1990). Secondly that clustering can bring a good approach to build semantic classes with the help of a distance of similarity (Pereira, Tishby, and Lee 1993; Nazarenko, Zweigenbaum, Bouaud, and Habert 1997). This methodology about clustering linguistic terms can help to get common features to build rules for information mining in document archives. Classification of a set of terms requires to represent data as morphological information vectors (terms themselves or parts of terms and how?) and to determine which kernel has to be used to achieve SVC. We try to use whole terms, morphological primitives and bigrams as morphological information. And we try to use the Levenshtein distance and the Jaccard similarity index, Radial basis function, and combination Levenshtein-Radial or Jaccard-Radial kernels to study the clustering effect.
In Section 2, we introduce the methodology of support vector clustering. Section 3 presents the labeling approach and Section 4 gives studies of vector representation and different kernels for term clustering. Finally Section 5 shows evaluation of the technique.

2. Support vector clustering

In this section, we recall the clustering approach.

2.1. Kernel trick

We know a priori classes of items (red circles and yellow squares) and we search a linear frontier in a higher dimensional space. For that, data are transformed using a kernel function (dot product). Preprocess the data with:

$$\begin{aligned} \Phi : X &\to X \\ x &\to x \end{aligned} \tag{1}$$

K is a dot product of the space (Hilbert space, H), and learn the mapping of $\Phi(x)$ to y (class).

$$\begin{aligned} \{x_i\}_{i=0}^{N} \text{ learning data x in X is a multivariate data on } X^d, \\ \text{where d is the number of feature} \end{aligned} \tag{2}$$

$$\langle \Phi(x), \Phi(x') \rangle = K(x, x') \text{ can be computed in } X^d \tag{3}$$

($\|.\|$ is the norm associated to the K dot product)
Usually $dim(X) << dim(H)$

2.2. Optimization

As an extreme view the distribution of data under the scope of unsupervised learning can be interpreted as density estimation. But in our case the approach estimates *quantiles* of a distribution, not its density. In the case of SVC, we determine support vectors to delimit the distribution of points. The goal now points out to find the minimal sphere which surrounds data. One can show: if $\Phi(x)$,...,$\Phi(x_N)$ is separable from the origin in H, then the solution of margin minimization between two classes corresponds to the normal vector of the hyperplan separating the data from the origin with maximum margin.
In our case we try to encapsulate data into a ball. The points inside the ball represent data to classify (first) and the origin represents the second class. Primal problem is written as follows. Let a the (non-fixed) center of the ball, R the radius of the ball and C is a fixed penalty constant controlling the number of data near the ball. Let us minimize:

$$F\left(R, \{r_i\}_{i=1}^{N}\right) = R^2 + C\sum_{i} r_i \tag{4}$$

Under the constraints:

$$\|\Phi(x_i) - a\|^2 \leq R^2 + r_i \ , \text{ where } r_i \geq 0 \text{ for all } i = 1, ..., N \tag{5}$$

a is the center of the ball. The dual problem (for a convex problem) is the Lagrangian written as follows.

$$L(R, a, \{r_i\}_{i=1}^{N}, \{\beta_i\}_{i=1}^{N}, \{\mu_i\}_{i=1}^{N}) = R^2 - \sum_{i=1}^{N} \beta_i (R^2 + r_i - \|\Phi(x_i) - a\|) - \sum_{i=1}^{N} r_i \mu_i + C\sum_{i=1}^{N} r_i \tag{6}$$

Where $\beta_i \geq 0$ and $\mu_i \geq 0$ are Lagrange multipliers. Now we minimize L to find a possible couple $(\widehat{R}, \widehat{a})$ such that:

$$\|\Phi(x_i) - \widehat{a}\|^2 \leq \widehat{R}^2 + r_i \ , \text{ where } r_i \geq 0 \ for \ all \ i = 1, ..., N \tag{7}$$

Dual variables verify:

$$\frac{\partial L(\widehat{R}, \widehat{a})}{\partial R} = 0, \frac{\partial L(\widehat{R}, \widehat{a})}{\partial a} = 0 \tag{8}$$

We make L optimal with the constraints:

$$\{\beta_i\}_{i=1}^{N}, \{\mu_i\}_{i=1}^{N} \tag{9}$$

Hence the center can be written:

$$\widehat{a} = \sum_{i=1}^{N} \beta_i \Phi(x_i) \tag{10}$$

The dual problem is rewritten (after replacing $\widehat{R}, \widehat{a}$ and $\widehat{\mu_i}$):

$$W(\{\beta_i\}_{i=1}^{N}) = \sum_{i=1}^{N} \beta_i K(x_i, x_i) - \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j K(x_i, x_j) \tag{11}$$

which we maximize under the constraints:

$$0 \leq \beta_i \leq C, \ \sum_{j=1}^{N} \beta_i = 1 \text{ for all i} = 1, ..., \text{N} \tag{12}$$

A Gaussian kernel ensures the equivalence between primal and dual forms for convex problem. The function defining a point in the feature space is:

$$\phi(x_i) = z e^{-q \cdot \|x - x_i\|^2}, \text{ where z is constant} \tag{13}$$

And the kernel:

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle = e^{-q \cdot \|x_i - x_j\|^2} \tag{14}$$

The kernel fits with Mercer's Theorem for kernel definition (k needs to be a positive definite operator on $L_2(X)$, X being a compact space). Only the parameter q influences clustering. Three kinds of points results from the dual formulation:

- Inside the ball $\beta_i = 0$ ($\beta_i = C$ and $r_i = 0$)
- Outside the ball $\beta_i = C$ ($\beta_i = 0$ so as to $r_i > 0$) also called bound support vectors (BSV)
- On the ball, $0 < \beta_i < C$, these are support vectors (SV)

The KKT (Karush-Kuhn-Tucker) complementary conditions of Fletcher are:
$r_i > 0, \Rightarrow \beta_i = \frac{1}{\nu N}$, then BSV is an SV.
The set of SV not being BSV is $\left\{ i \mid \text{decision function } R^2(x_i) = \|\Phi(x_i) - \widehat{a}\|^2, \beta_i = 0 \right\}$
For any point x_i:

$$r_j \mu_j = 0 \tag{15}$$

$$\left(\widehat{R}^2 + r_j - \|\Phi(x_i) - \widehat{a}\|^2 \right) \beta_j = 0 \tag{16}$$

If x is a support vector, the radius is:

$$\widehat{R}^2 = \|\Phi(x) - \widehat{a}\|^2 = K(x, x) - 2 \sum_{j=1}^{N} \beta_j K(x, x_j) + \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j K(x_i, x_j) \tag{17}$$

For any point y the distance Ry from the center is:

$$R_y^2 = \|\Phi(y) - \widehat{a}\|^2 = K(x,y) - 2\sum_{j=1}^{N}\beta_j K(y,x_j) + \sum_{i=1}^{N}\sum_{j=1}^{N}\beta_i\beta_j K(x_i,x_j) \tag{18}$$

Hence it is possible to test if y is inside the sphere or not by comparing $\widehat{R}$ and R_y.

2.3. Contour deformation

The value of parameter $\nu = \frac{1}{N.C}$ asymptotically represents a max bound of the BSV rate. Parameter C takes values in $[\frac{1}{N}, 1]$, as C is reduced, more and more points are labeled as outliers. If q increases the Gaussian radius decreases and the number of SV increases. Subsequently if one or more points of a cluster become support vector a specific contour will be generated for the cluster. From a certain value of q, support vectors appear around each cluster.

3. Geometric hashing based labeling

In this section, we describe our mapping methodology to assign data points to clusters.

3.1. 2-d grid assumption

In the previous method only support vectors guide processing to make contours but escape to know if a given point lies inside or outside the contour. Some methods such as describe in the foundation work by Ben-Hur *et al.* (2001), and Yang *et al.* (2002) work with an adjacency matrix defined as follows. Given two points of the data x_i and x_j and $\widehat{R}$ (the radius of the ball), the adjacency matrix A such that:

$$A_j^i = \begin{cases} 1 \text{ if any point between } x_i \text{ and } x_j \text{ is such that } R(x_k) \leq \widehat{R}; \\ 0 \text{ else.} \end{cases} \tag{19}$$

Hence Ben-Hur *et al.* (2001) define a set of points between each pair and calculate if all the points belongs to the sphere or not, and so assign the pair of point to a cluster; In the second method Yang *et al.* (2002) use a graph method to analyze the density areas of the graph defined by the adjacency matrix. We have compared our approach to these ones we call respectively in the following nearest-neighbors (NN) and minimum spanning tree (MST). These methods are time-consuming and we imagined a method based on a geometric hashing function achieved with a grid surrounding data points in the attribute space. Basically according to the SVC method we only compute the radius for the points of the grid (that are hash keys) to build clusters, and as (Datar, Immorlica, Indyk, and Mirrokni 2004) we assume that almost closest points can be associated to a same hash. We use a nearest-neighbor method (Cover and Hart 1967) to associate data points to their hash.

3.2. Algorithm

The basic idea behind random projections is a class of hash functions that are locally sensitive; that is, if two points (a, b) are close, they will have small $|p - q|$ values and they will hash to the same value with a high probability. If they are distant they collide with small probability. We have the following definitions. Let M be the size of the grid, and fixed by the user. A 2-dimension $M \times M$ grid is characterised with a step s. The step s is defined according the minimal/maximal value of two first coordinates obtained by correspondence analysis (COA), $c1$ is the first coordinate, and $c2$ the second coordinate. We use the **ade4** package of R-project to compute COA (Dray and Dufour 2007). Let g_i be a grid point.

Definition 3.1 (s_{ck})
Let s_{ck} be the scale of the grid from correspondance coordinates ck

$$s_{ck} = \frac{(max\left\{g_i^{ck}\right\}_{i=1}^{N} - min\left\{g_i^{ck}\right\}_{i=1}^{N})}{M}, ck = \{c1, c2\} \tag{20}$$

We can define the set of grid points g_M with each point g_i by:

Definition 3.2 (g_G)
Let g_M be the set defined by:

$$g_G(s_{ck}) = \left\{g_i : dim(g_i) = 2 \text{ and } (g_i^{ck} - g_{i+1}^{ck}) = s_{ck},\ i \in [1:G]\right\}, ck = \{c1, c2\} \tag{21}$$

For each point g_i we can assess membership to clusters without specifying which one.

Definition 3.3 Let be $C = \{c_j\}$ the set of clusters, knowing radius R according Equation 18

$$g_i \in C \text{ if } \left(\widehat{R} - R(g_i)\right) \geq 0 \tag{22}$$

We now try to define clusters set with grid points:

Definition 3.4 (C)
We call C the set of clusters. A cluster consists of a grid point and all neighbouring grid points:

$$\begin{aligned} C = \{c_i : \exists j\ g_j \in c_i\ \wedge\ g_k \in c_i \\ \text{if } g_k^{c1} \in \left[g_i^{c1} - 1, g_i^{c1} + 1\right], g_k^{c2} \in \left[g_i^{c2} - 1, g_i^{c2} + 1\right]\ \} \end{aligned} \tag{23}$$

Now we define the ball as the neighborhood of the hash key (X, Y) from which it is assigned a specific cluster reference c_j using a k-nearest neighbor threshold:

Definition 3.5 $B_k(X, Y)$
Let g_G the grid, C the set of clusters and $P(P_{c1}, P_{c2})$ a point with coordinates (X, Y) in the grid space GxG. Then the ball of P $B_k X, Y$ is defined by at least k neighbours belonging to a same cluster:

$$\begin{aligned} B_k(X, Y) = \{c_j : g_i \in c_j \\ \wedge g_i^{c1} \in [X - 1, X + 1], g_i^{c2} \in [Y - 1, Y + 1] \wedge \#i \geq k\} \end{aligned} \tag{24}$$

A family $H = h : F \to G$ is called locality-sensitive if, for any point a, the function $p(u)$ is defined as follows:

Definition 3.6 $p(u)$

$$\begin{aligned} p(u) = Pr_H[h(a) = h(b) = (X, Y) : |a - b| \leq u, \\ E(a_{c1}) = E(b_{c1}) = X, E(a_{c2}) = E(b_{c2}) = Y] \end{aligned} \tag{25}$$

$p(u)$ decreases in u. That is the probability of the collision of points a and b decreases with the distance between them.
After defining a grid on data space, `ClusterLabeling` function achieves the first stage assigning a cluster number to each point of the grid. The calculation of Lagrange coefficient gives the kernel matrix (`MK`). User settles the size of grid `G`, and `MinMax` value in data space can be computed. The main function (`findSvcModel`, described in next chapter) outputs a matrix called `NumPoints`

linking each grid point to a cluster id. The radius `Rc` can be computed according algorithm shown in Table 1.

Algorithm 1

```
Require: kernelmatrix MK, grid size G, MinMaxX min max value of x value in data, MinMaxY
    min max value of y value in data.
Ensure: NumPoints, a GxG vector for each grid point and membership to a cluster id.

    while each GxG Grid point P do {we identify if a point belongs to a possible cluster}
        Associate x, y values to P from MinMaxX and MinMaxY
        Calculate Radius Rp of P , if Rp <= Rc , give ball membership to P
    end while

    while each GxG Grid point P(i) do {we identify cluster id(s)}
        while each P(k) around P(i) of one step do
            if all points P(k) have no cluster membership then
                Create a new cluster vector CV with a new cluster id Cm
                Put CV in a list of cluster vector membership LCV
                Put P(i) to CV and associate Cm in NumPoints
            else
                associate cluster id of P(k) to P(i) in NumPoints
            end if
        end while
    end while

    while each CV(i) in LCV do {we merge closed clusters}
        while each other CV(j) in LCV != CV(i) do
            if CV(i) has distance of one step from CV(j) then
                Merge CV(i) and CV(j)
                Update NumPoints
            end if
        end while
    end while
```

Table 1: Calculate `Rc` radius of the ball using `MK`.

Finally we can assign a cluster label for any point x of the data set according the hash function and the corresponding ball value, defined in Equation 24.

$$f(x) = c_j \text{ if } B_k(h(x)) = c_j \tag{26}$$

`MatchGridPoint` function, presented below, achieves the second stage; computation of $f(x)$ in Equation 26. It returns a vector we call `ClassPoints` associating a cluster id to each data point in the initial dataset (see Table 2).

```
Algorithm 2
Require: data matrix MD, grid size G, MinMaxX, MinMaxY, NumPoints, neighbourhood of a data
    point k.
Ensure: ClassPoints, a vector for data point and membership to a cluster id.
 1: for each point D(i) in MD do
 2:     Calculate Grid coordinate of any D(i) , with MinMaxX, MinMaxY
 3: end for
 4: for each point D(i) in MD do
 5:     Init a score vector SV(i) with dimension of cluster id(s)
 6:     for each Grid Point P(j) in NumPoints do
 7:         if P(i) cluster id = k is found and distance between P(j) and D(i) = k then
 8:             Increment SV(i)(k)
 9:             Associate Max(SV) to Classpoint(i)
10:         end if
11:     end for
12: end for
```

Table 2: `MatchGridPoint` routine.

3.3. Usage of the svcR package

Main function is the `findSvcModel` function. It computes a clustering model and returns it as an R object which is usable to other function for display and export. Let call ret the return object, it covers some information about model parameters as the language coefficients (`getlagrangeCoeff(ret)$A` attribute), the kernel matrix (`getMatrixK(ret)` attribute) and the cluster memberships (`getClassPoints(ret)` attribute). `findSvcModel` takes 10 arguments :

- `data.frame` means `data.frame` parameter in standard use
  or means `data.frame` in `loadMat` use
  or means `DatMat` in `Eval` use, a matrix given as unic argument
- `MetOpt`, optimization control parameter : `optimStoch` (stochastic way of optimization) or `optimQuad` (quadratic way of optimization)
- `MetLab`, labelling method: `gridLabeling` (grid labelling) or `mstLabeling` (mst labelling) or `knnLabeling` (knn labelling)
- `KernChoice`, kernel choice: `KernLinear` (Euclidian) or `KernGaussian` (RBF) or `KernGaussianDist` (Exponential) or `KernDist` (Matrix data as Kernel value)
- `Nu`, *nu* parameter
- `q`, *q* parameter
- `k`, *k* nearest neigbours for grid
- `G`, grid size
- `Cx`, *x* component to display (1 for 1st attribute)
- `Cy`, *y* component to display (2 for 2nd attribute)

If `Cx` and `Cy` are 0 the correspondent analysis is used. The data is given as first argument. The format is data.frame() (i.e. list) as the iris well known dataset. Some R libraries are required as **quadprog** (Berwin, Turlach, and Weingessel 2007) for optimization, **ade4** (Dray and Dufour 2007) and **spdep** (Bivand 2010) for principal component analysis. This an exemple of usage in R :

```
> library("svcR")
> data("iris")

> retA <- findSvcModel(iris, MetOpt = "optimStoch", MetLab = "gridLabeling",
+     KernChoice = "KernGaussian", Nu = 0.5, q = 40, K = 1,
+     G = 5, Cx = 0, Cy = 0)

> plot(retA)
> ExportClusters(retA, "iris")
> findSvcModel.summary(retA)
```

It means as data is the iris data frame. The Kernel choice is radial-based, parameters of SVC technique are $nu = 0.5$ and $q = 40$. Parameters for cluster labeling are $k = 1$ neighbor and grid size of 5×5 points. $Cx = Cy = 0$ means that first two principal components are used. `MetLab` value means that geometric-hashing method is used. `Plot` function permits to visualize clusters. `ExportClusters` outputs clusters in a file with variables names. `findSvcModel.summary` displays size and number of clusters, and averaged attributes for each cluster. Some functions can help the user to navigate in clusters. `ShowClusters(retA)` returns all clusters ordered by their id (cluster 0 is a bag of variables not clusterable), `GetClustersTerm(retA, term = "121")` returns clusters in which "121" is a substring names of a member include in them, and `GetClusterID(retA, Id = 1)` returns the cluster with $Id = 1$.

3.4. Toy example

We used the famous Fisher's Iris data set. It contains 3 classes, 150 variables and 4 attributes. Our clustering extraction is largely based on the topology of points localized on a 2-D map. The dimensions of the maps are found by using a correspondence analysis and we kept the first two coordinates. The Iris data on these projection classes 2 and 3 are not well separated as it shown on Figure 1. So the method can catch well class 1 and from time to time it occurs a "bridge" between class 2 and 3 that links them to form one cluster (Figure 1). The system is not very robust to force a so weak topological boundary. And so several iterations can force cluster 2 and cluster 3 to appear. For a grid size of $G = 13$, we obtain 50% of success after a certain number of run executions.

```
> library("svcR")
> data("iris")

> retB <- findSvcModel(iris, MetOpt = "optimStoch", MetLab = "gridLabeling",
+     KernChoice = "KernGaussian", Nu = 0.5, q = 40, K = 1,
+     G = 13, Cx = 0, Cy = 0)

> plot(retB)
```

The nearest neighbour parameter k is used to find the closest cluster for a given data point. Low values such as $k = 1$ or $k = 2$ give same level of precision evaluation parameter to obtain 3 clusters. But this approach is not sufficient for good level of precision when the size of the Grid is high ($G > 25$) because the distance of peripheral data point is too far from their cluster.

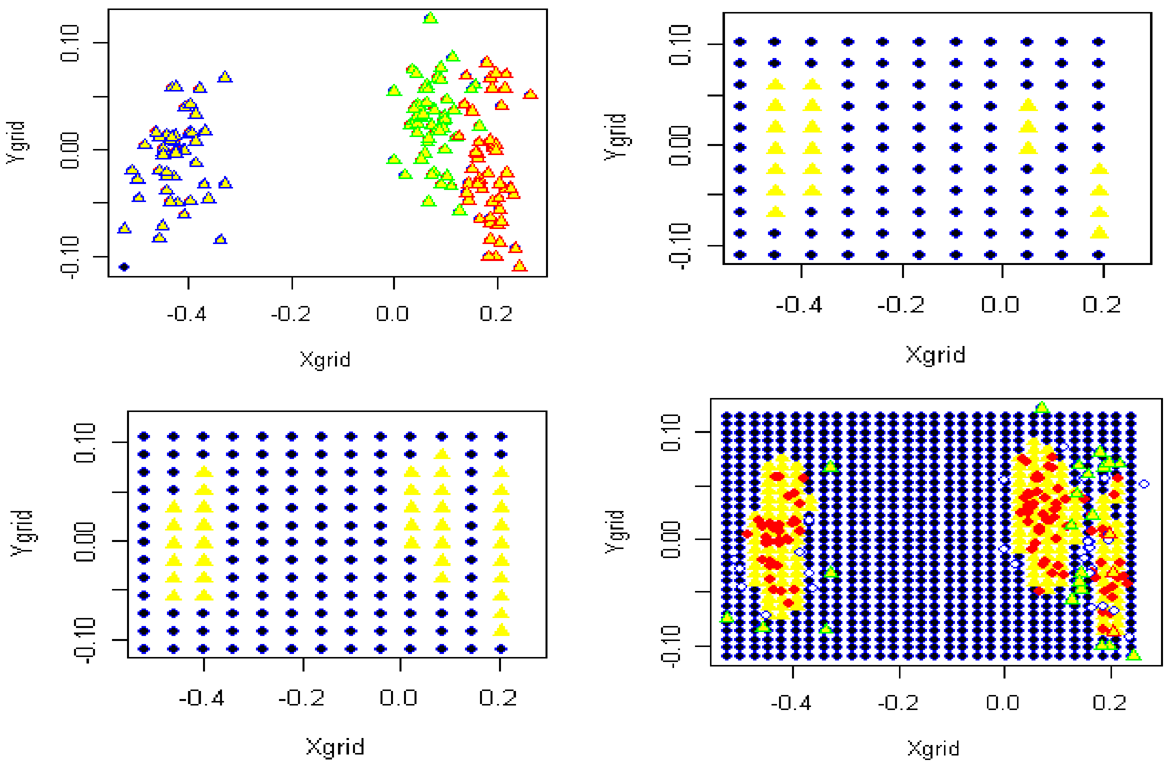


Figure 1: 2-D displays showing: data, clustered grid and data superimposed with clustered grid. Top left: Data plotted with COA $c1 = 1$, $c2 = 2$; Top middle $G = 11$, #unclassified points is 17, #missclassified points is 9; Top right: $G = 13$, #unclassified points is 2, #missclassified points is 7; Bottom: $G = 30$.

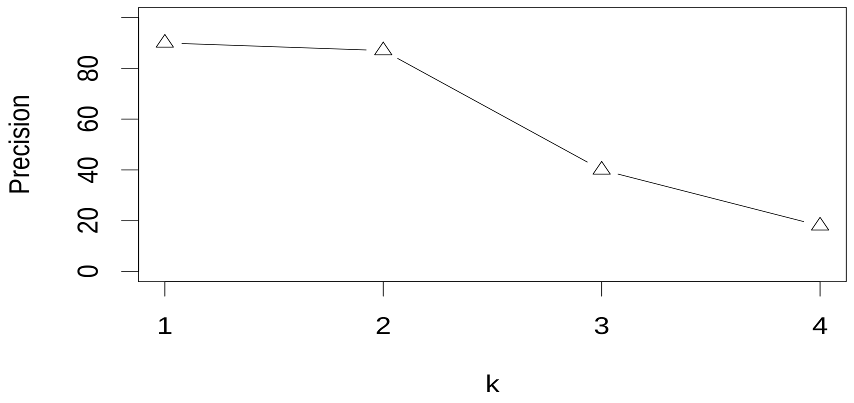


Figure 2: Clustering precision on iris dataset with parameters $Nu = 0.7$, $q = 1200$, $G = 13$.

A second stage of high-distance nearest neighbour should perform well at this size grid. But as we can see on Figure 3 the running time is less interesting when $G > 25$ is less competitive to other approaches. The time speed is shown on Figure 3. We can see that for $G = 13$ and a data size $N < 50$ for any method the run time is almost the same but increases very fast for the NN method. Precison when N is small is stable and very high (Figure 2). Our approach becomes interesting for a much higher amount of data. For the whole Iris Data set our approach is two times faster but the run time depends on the grid size. We can see on Figure 3 that if $G > 20$ it becomes less competitive. We used the **quadprog** package in R-Project for optimization.

4. Representation of term sets and kernels

In this section, we describe our good representation to classify term from a specific domain with an adapted kernel.

4.1. Data, language models and domain knowledge

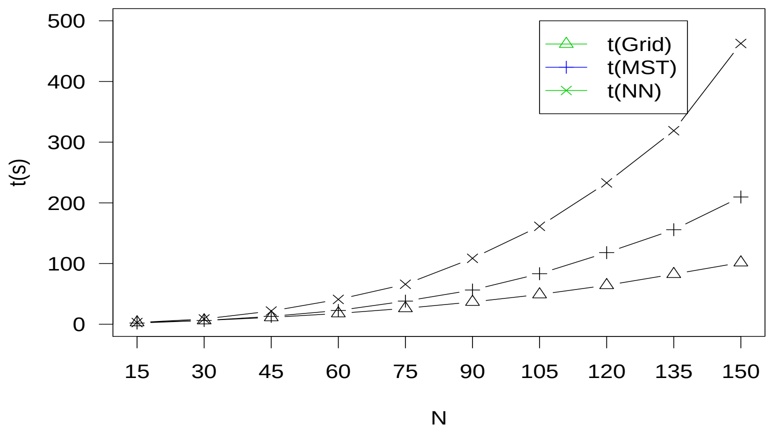
In the previous chapter we have shown that a radial base function can make a suitable clustering. But the data were made of a few attributes and not coming from natural language surrounded by sense ambiguities. We tried to make an attempt to classify terms coming from a specific domain: molecular and developmental biology.

Our linguistic data set consists of 1,893 terms (linguistic phrases) manually extracted from an annotation of 1,471 documents (5,730 sentences) where annotated linguistic phrases describe temporal stages of biological development. The corpus itself has been build manually grabbed from Medline document database about spore coat formation and gene transcription specifically for Bacillus Subtilis species. We define some ideas about the language model studied in next chapter. Let suppose the following phrase ”septal localisation of the division”; it will be supposed to be a term. From this term we can consider different sub-structures. ”septal” and ”localisation” are considered to be distinct words, and for instance ”sept” is supposed to be a radical i.e a sequence of character which can be found in other words. ”septal localisation” is considered as a bigram, i.e. a sequence of two words. ”localisation of the” is considered as a trigram, i.e. a sequence of three words.

Textual corpus we used describes biological knowledge and especially a well known biological model called sporulation. This biological process is activated by a microorganism to be resistant in an environment with starvation. The bacterial is transformed into a resistant sphere with mininum needs and activity. In information extraction from texts gene network reconstruction is a quite interesting field to understand how a gene network is activated. Temporal and spatial information are complementary information useful to understand when gene interactions occured. A well studied biological process as sporulution can be a reference model with both interest:

- Gathering enough molecular information about gene-gene interactions in texts since ten years;
- Being a well described biological model across different stages.

Six main stages describe the sporulation process. At the beginning of the process a frontier called the septum is created and at the last stage an engulfment is created to leave out the bacterial spore. The 1,893 terms have been also classified manually into the 6 biological stages. An average amount of 600 terms can cover a given stage. The problem is related to morphological and fuzzy description of language. Where a strict formal description should used for instance ”stage II” concerning the second stage of biological development, an expert could use ”during the first stages of sporulation” or ”at the onset of sporulation” or ”at stage I-II” or ”after septum formation” ...etc. Moreover complexity of description, we can imagine insofar because 600 terms per class on to only 1,893, is that lots of terms are not exclusive to one stage (i.e. one class). Lots of expressions can designate a stage and often several stages at the same time.



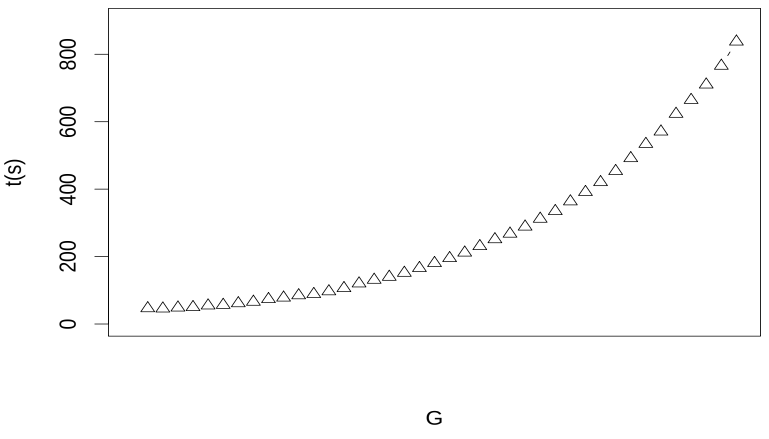


Figure 3: Running time of **svcR** approach according (top) data size (Nu = 0.7, q = 1200, G = 13) and (down) grid gize (N = 150, Nu = 0.7, q = 1200).

Terms (TM)				**Radicals (RD)**	**Bigrams (BG)**	**Trigrams (TG)**
class 1	**class 2**	**class 3**	**class 4**			
insertion into the septum	prespore devel-opment	cortex layer , synthesized between , the forespore inner and , outer membranes	coat, encases the spore	init	cell specific	the mother cell
integrity of the septum	prespore gene expres-sion	cortex peptidoglycan in spores	coats	sept	spore coat	mother cell specific
septal com-part-ment	prespore pro-gramme of gene expres-sion	cortex structure	coats of wild-type spores	prespore	during sporulation	in the mother
septal local-ization of the division	prespore-like cells	cortexless spores	compartment	endospore	and sporulation	mother cell com-partment
septal pepti-dogly-can during cell division	prespore-specific	cortical or vegetative peptidoglycan synthesis	compartment-specific	engulfment	of sporulation	growth and sporula-tion

Figure 4: Samples of terminological data sets.

Why do a clustering method such as SVC could be of interest ? We observed that:

1. Most of terms describing occurrence of a gene activation/inhibation/regulation are not expressed in a simple regular way such as ”at stage 2” or ”at stage 3”. But terminology of temporal knowledge has a variable expressivity;

2. Lots of terms are not exclusive to a stage.

In such usage context, the **svcR** technique could help an expert to build rules about expressions to get equivalence between a set of expressions and a mapping of rule with a specific class. We decided to compare which language model can bring benefit for term description and for each language model which kernel can be also more relevant. We had manually selected a list simple morphological radicals (11 tokens), word bigrams (a restricted sample of 500 on to 1,477) and word trigrams (a restricted sample of 500 on 2,179) from the whole set of terms. Figure 4 gives a sample of some linguistic expressions. In our clustering experiments we first made a sample of 98 terms and 4 classes, similar in size with iris data (Section 3.4).

After viewing which language model (term-radical, term-term, term-bigram, term-trigram) and which kernel are enough efficient, we apply the language model and the kernel to the whole set of 1893 terms.

4.2. Kernels

As terms (that are strings), intrinsically and without textual context, can be statistically compared pairwise (in a Levenshtein way) or using a bag-of-words (in the Jaccard way) we compared these approaches, in addition to robustness due to randomized non null value in the Jaccard case. The Levenshtein distance is an editing distance based on the cost to transform a string into another (Levenshtein 1966). Assume a and b being two strings. Let a_i be the sub-string consisting of the first i symbols of string a where $0 \leq i \leq \|a\|$ and b_j be the sub-string consisting of the first j symbols, iteratively we obtain the Levenshtein distance at position i and j:

$$D_{i,j} = min(D_{i-1,j} + w^I, D_{i-1,j-1} + w^S, D_{i,j-1} + w^D) \tag{27}$$

where w^I, w^D and w^S are weights of insertion, deletion and substitution on operations respectivaly and $D_{0,0} = 0$ Finally $D_{a,b}$ represents the weighted Levenshtein distance. From its expression we define the Levenshtein radial base kernel:

$$LRB(x_1, x_2) = e^{-q \cdot \left\| \sum_{i=1}^{N} (D_{1i} - D_{2i}) \right\|^2} \tag{28}$$

We also define a kernel using only the component of Levenshtein distance between a pair of terms:

$$RBL(x_1, x_2) = e^{-q \cdot \| D_{12} \|} \tag{29}$$

Equation 28 and Equation 29 are a composition of a semi-positive definite kernel (the radial base function) so the final kernels are also semi-definite positive. The Jaccard index is a similarity index (Jaccard 1901) that is useful to assess the similarity between two objects computed only knowing the set of their attributes, and not the whole set of attributes being often huge and not describing the given objects. Its expression is the following knowing that a string s_1 is composed with tokens s_{10},..., s_{1m} and string s_2 is composed with tokens s_{20},..., s_{2k}:

$$J_{12} = J(S_1, S_2) = \frac{|\{S_{10}, ..., S_{1m}\} \cap \{S_{20}, ..., S_{2k}\}|}{|\{S_{10}, ..., S_{1m}\} \cup \{S_{20}, ..., S_{2k}\}|} \tag{30}$$

Hence we define a Jaccard-radial base kernel (JRB) according vector defined with Jaccard index with other terms (the data matrix is symmetric):

$$JRB(x_1, x_2) = e^{-q \cdot \left\| \sum_{i=1}^{N} (J_{1i} - J_{2i}) \right\|^2} \tag{31}$$

We also define a kernel using only the component of Jaccard index between a pair of terms:

$$RBJ(x_1, x_2) = e^{-q \cdot \| J_{12} \|} \tag{32}$$

Equation 31 and Equation 32 are a composition of a semi-positive definite kernel (the radial base function) so the final kernels are also semi-definite positive. Xu and Zhang (2004) and Bilenko and Mooney (2002) have been respectively adapted a kernel approach with a Levenshtein and a Jaccard similarity coefficient and proved their robustness though their classical simplicity. In our data representation we have used four Kernels: the Levenshtein-radial base (LRB), the radial base-Levenshtein (RBL), the Jaccard-radial base (JRB) and the radial base-Jaccard (RBJ). We also have introduced noise in the data matrix such that if the Jaccard coefficient gives 0 we assign a random non null value to the data matrix component. We call this fuzziness, Jaccard+. The vector approach using such distance and index heuristics in natural language processing sets the representation of description by sets of words but property of such sets can be modulated. For instance co-frequency in textual context (with left and right collocations) (lexical-based similarity), or string inclusion between two terms (dictionary-based similarity), or ontological nodes shared between two terms (conceptual-based similarity). We focused on the second way and we compared several cases of dictionary to build the matrix of similarity. As variables to classify of course we used the sets of terms, and as attributes the set of radicals (RD), the sets of terms itself (TM), the sets of bigrams (BG) and the set of trigrams (TG).

4.3. Results

As we see in the Section 2 for the presentation of the SVC method coupled to our geometric approach for cluster extraction, if no clear geometric separation in data occurred on the 2-D map of correspondence analysis coordinates, the method is unsuccessful. Figure 5 shows different plots of the different cross between data attributes and distances. We see on these maps that TM-TM Levenshtein, TM-RD-Jaccard and TM-RD Jaccard+ can produced interesting maps for SVC application. Thus on each set of the data matrix we applied the cluster extraction to compare the efficiency of class retrieval. Figure 6 shows performance of the method. The Jaccard kernel gives best results with a good separation and extraction of classes. And the variant set introducing random noise in the matrix still becomes successful with 2 misclassified items on 98.

```
> library("svcR")
> data("term")

> retC <- findSvcModel(term, MetOpt = "optimStoch", MetLab = "gridLabeling",
+     KernChoice = "KernGaussian", Nu = 0.9, q = 2000,
+     K = 1, G = 13, Cx = 0, Cy = 0)

> plot(retC)
```

Now we adopt the best obtained clustering setting that is a term-radical matrix (language model) and Jaccard Radial base kernel. Now to study scalability efficiency we expand amount of terms and radicals taken into account for Jaccard distance computation. Independently of clusters purity (class homogeneity), impact of features (radicals) is a warranty to make a good separation between similar terms. We do not forget that support vector machine is a non linear method which is efficient only if data are separable. Hence recall that role of features is to make similarity clue between terms, role of Jaccard index is to capture similarity, role of 2D component analysis is to capture main features that make separation between data, and finally role of support vector clustering is to capture bounds of cluster thanks to their geometric separation. Figure 7 shows that too many features do not make separation of data (attribute DName is changing for each four data sets):

```
> library("svcR")
> data("term")

> retD <- findSvcModel(term, MetOpt = "optimStoch", MetLab = "gridLabeling",
+     KernChoice = "KernGaussian", Nu = 0.9, q = 2000,
+     K = 1, G = 13, Cx = 0, Cy = 0)

> plot(retD)
```

But too few features make too few set of clusters. A medium set of features can lead to a good number of clusters. In our case 38 features describing structure of terms induce 15 clusters easily distinguishable visually. Recall the set of terms is made of 1893 terms describing 6 stages of sporulation process as we mentioned in Section 4.1.

Lots of terms belong to several stages (in the sense of classes). Even typical string token relevant of a class can belong to different stages. It is mainly due to biological stage results from microscopy studies, so visual patterns and often a co-occurrence of patterns can be simultaneously typical of a stage but individually we can observe a pattern occurs during several stages as *mother cell* and *compartmentalization* (beginning at stage 2 and staying at stages 3, 4, 5, 6), *engulfment* (beginning at stage 3 and staying at stages 4, 5, 6), *septum* (beginning at stage 1 and staying at stages 2, 3,

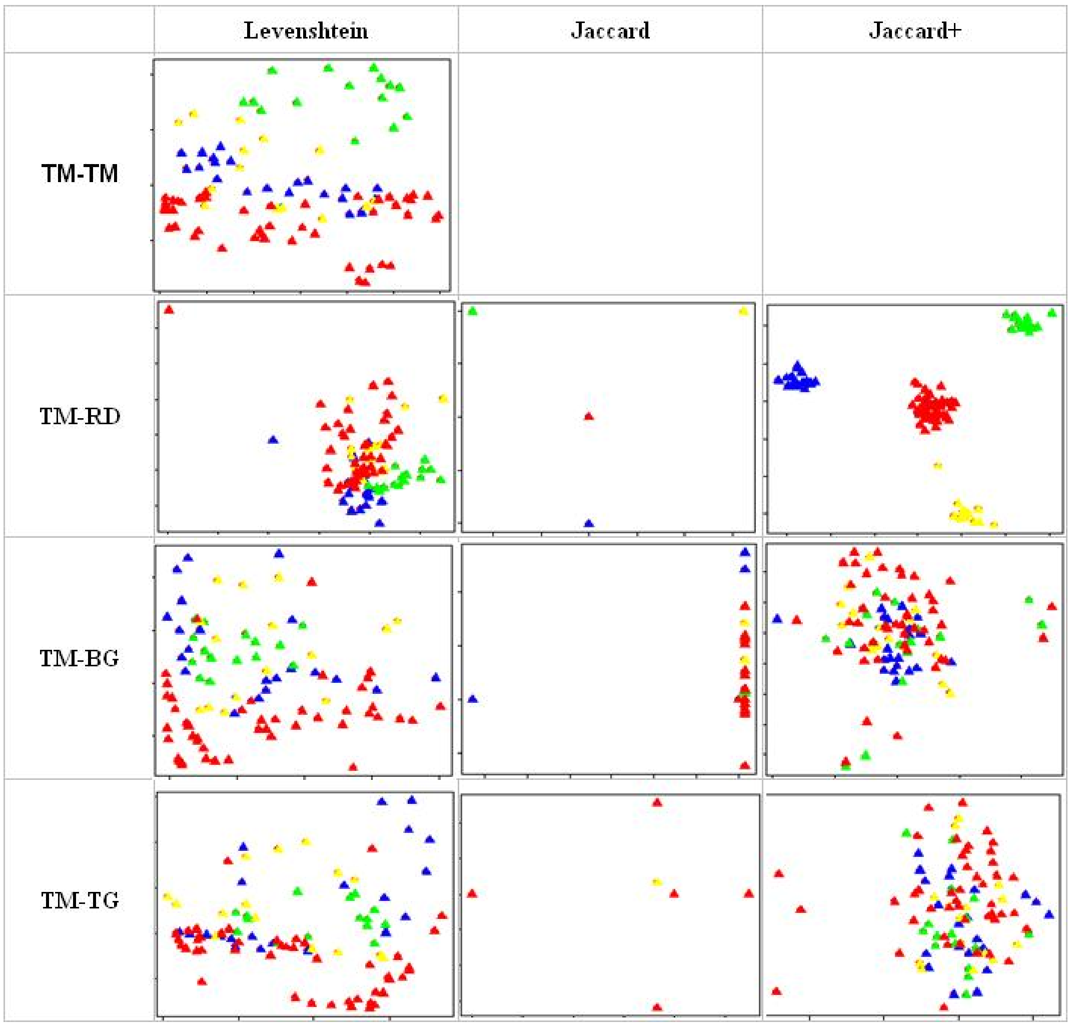


Figure 5: In this table each display is dependant upon features describing linguistic phrases (TM or terms) i.e. with terms themselves (TM-TM), with radicals (TM-RD), with bigrams (TM-BG) or with trigrams (TM-TG), secondly results depends upon kernel used for clustering Levenshtein, Jaccard or Jaccard with artificial noise. Displays represent data classes in green, red, blue, yellow colors and in 2-d maps of the COA components.

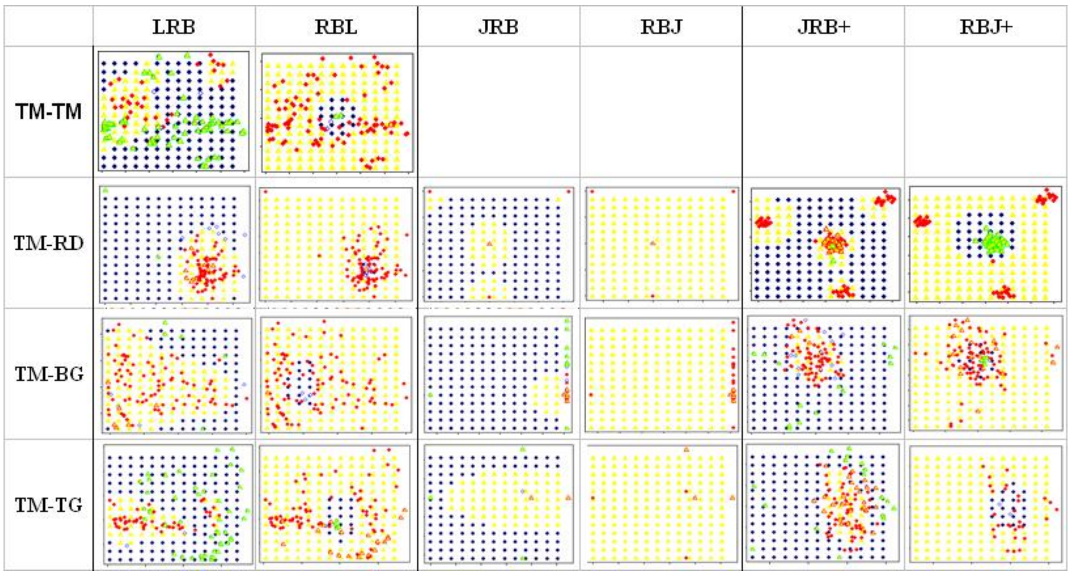


Figure 6: Clustering with SVC-geometric hashing ($Nu = 0.9$, $G = 13$, q chosen at best between 1 and 10000).

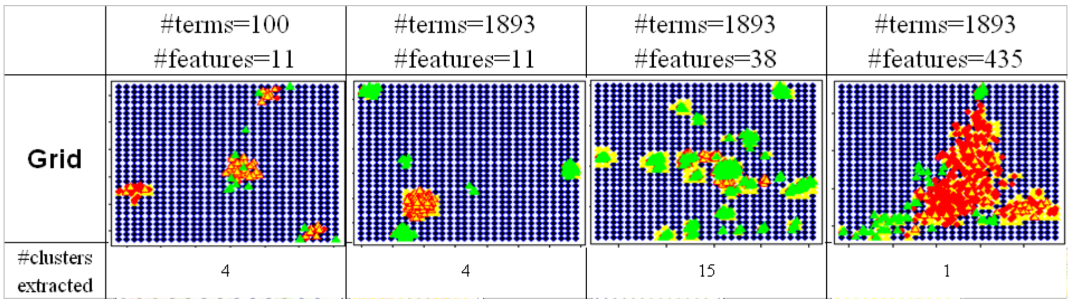


Figure 7: Clustering with SVC-geometric hashing ($Nu = 1$, $G = 30$, $q = 2000$, JRB+); Each column means different number of terms and number of features, datasets sizes increase from left to right. Below are the number of clusters extracted with **svcR**. Yellow color represent clusters, red is data color for major class of a cluster and green is data color but not belonging to major class of a cluster.

4). This property of cross-membership is hard enough to compute as a mapping between a specific term to a unique class. In our results (Figure 7) getting more clusters (15) than classes (6) induces that terms can be misclassified (green points) but make a variety of specific clusters from which we expect they capture patterns association that should be used to define rule of an automaton. For instance among the 15 clusters, a specific one gives the following members :

compartmentalized activation , compartment-specific activation

We can imagine a rule associated to stage 2, 3, 4 and 5 : "compartment activation". Another one gives the following members :

slow postseptational, prespore-specific SpoIIIE synthesis, endospore coat, endospore coat assembly, endospore coat component, forespore coat, from the endospore coat, cortex and/or coat assembly, spore coat and cortex.

We can imagine a rule associated to stages 3, 4, 5 and 6 : "endospore coat", "coat cortex", "cortex coat". From these clusters of terms coat, cort, prespore, endospore, postseptational, forespore are in the sets of features. In this process of lexical rule definition the user plays an important role in such way a cluster do not give information directly exportable as an automatically defined rule. Visualization of clusters by an expert leads to identification of patterns association to include in lexical rules. Especially by the fact that elements taken into account are features and knowledge about features is required to say that these rules will be applicable to a set of classes (biological stages). The methodology makes us to understand, but it is not a discovery, that clustering mixes several components of different categories. Nevertheless it can be efficient to identify relevant features to identify as a lexical pattern to build rules for information extraction, in our case information extraction of a biomolecular process described linguistically and formally by several stages (i.e. a scenario in the domain of biology).

5. Comparison with other techniques

In this section, we discuss behavior of concurrent clustering methods, existing kernels and interpretation of SVC clustering capacity. Below a simple general R utility function, gets outputs of used R clustering functions (k-means, **svcR**, hierarchical) and exploits a data property that is insertion of the class number in each term (as "4 coat protein" meaning "coat protein" belongs to class 4). Hence using grep function it is very easy to find the repartition of classes over clusters :

```
TabEval <- function(Dat) {
M <- matrix(nrow = (max(Dat[]) + 1), ncol = 8, 0)
for( k in 1:max(Dat[]) ){
        Stat <- c()
        Size <- length(Dat[ Dat == k])
        for(i in 1:6) {
                GR   <- grep(i, names(Dat[ Dat == k]) )
                Stat <- c(Stat, 100 * length(GR) / Size )
        }
        Stat= c(Stat, 0, Size )
        M[k,] <- Stat
}
Stat <- c()
for(i in 1:6) {
        Size <- length(Dat[])
        GR <- grep(i, names(Dat[]) )
        Stat <- c(Stat, 100 * length(GR) / Size )
}
Stat <- c(Stat, 0, Size )
M[nrow(M), ] <- Stat
print(M, digits = 1)
```

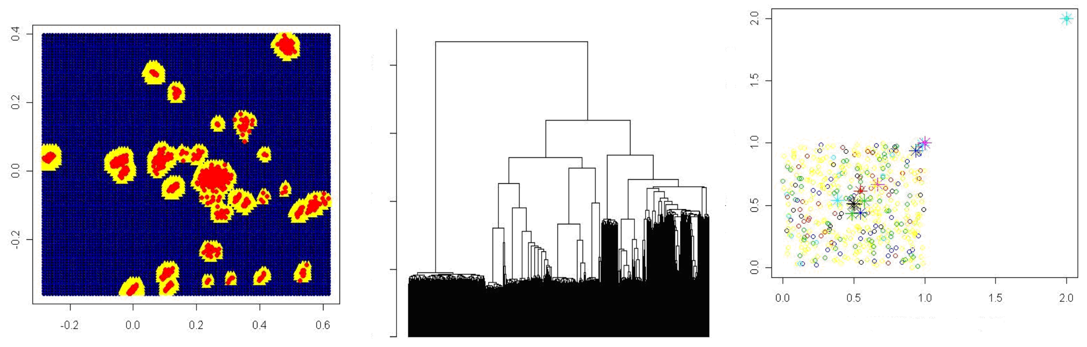


Figure 8: Clustering with SVC-geometric hashing (left), hierarchical agglomerative clustering (centre), k-means (right); Data are 1893 terms with 6 classes and 37 features using a Jaccard-radial base kernel.

5.1. Classical clustering

Algorithms such as k-means (KM) and hierarchical clustering (HC) are widespread poor knowledge techniques using metrics to find automatically clusters in any kind of data. Figure 8 shows graphically how such clusters could be represented.

About **svcR** and KM, 2-dimensional coordinates come from component analysis. On the KM map only centroids represent clusters (as star plotting characters). HC (Figure 8, center) displays a dendrogram where branches mean clustered points and require a cut-off at a level of the tree to catch clusters. In R, we used kmeans function from **stats** package (R Development Core Team 2010):

```
> library("stats")
> data("Cterm")

> res_km <- kmeans(na.omit(Cterm[, 1:1893]), 30)

> plot(Cterm[, 1:2], col = res_km$size)
> points(res_km$centers, col = 1:30, pch = 8, cex = 2)
> TabEval(res_km$cluster)
```

and `hclusterpar` function from **amap** package (Lucas 2007)

```
> library("amap")

> data("Cterm")

> res_HC <- hclusterpar(na.omit(Cterm[, 1:1893]), method = "euclidean",
+     link = "complete", nbproc = 1)

> plot(res_HC, labels = FALSE, hang = -1, main = "Original Tree")
> memb <- cutree(res_HC, k = 30)
> TabEval(memb)
```

As Data contains 6 classes and **svcR** approaches with JRB kernel induces extraction of 17 clusters we settle 30 clusters extraction as settings for both KM and HC function. Figure 9 shows the content of clusters and class distribution for each approach (hierarchical, k-means and SVC). The right column of each result set means the size of each cluster. The last line means distribution over classes from the whole dataset as baseline (it means that 12% of terms belong to class 1, 19% to class 2, 20% to class 3, 20% to class 4, 16% to class 4, 12% to class 6 and size of set is 1893 terms). First we can observe that distribution profile in cluster size is similar between HC and **svcR** :

```
> library("svcR")
> data("Cterm")

> retE <- findSvcModel(Cterm, MetOpt = "optimStoch", MetLab = "gridLabeling",
+     KernChoice = "KernGaussian", Nu = 0.5, q = 2000,
+     K = 1, G = 40, Cx = 0, Cy = 0)

> memb <- retE@ClassPoints
> names(memb) <- retE@Matrice$Var[[1]]
> TabEval(memb)
```

Secondly, looking at over-representation of classes over clusters HC and KM do not achieve better discrimination of terms across the 6 classes some clusters are better over. Language ambiguities seem to be a real bottleneck for all methods when usage is based on a Jaccard-Radial Kernel. But what happens when string kernels are used ?

5.2. String kernels

Lodhi, Saunders, Taylor, Cristianini, and Watkins (2002) and Moschitti (2009) promoted kernel strings to get semantic knowledge from texts. The string kernels calculate similarities between two strings by matching the common substring in the strings. A standard string kernel is the constant one (`SK-constant`) and assess similarities even is characters are matching in any order, and higher is the return value when order is respected and size of matching is bigger. Exact matching of n characters is called spectrum kernel (`SK-spectrum`) (Teo and Vishwanathan 2006). For instance let suppose a string of 29 characters and estimate value of a string with itself, `SK-constant` return 3165, `SK-spectrum` return 430.

If we pick two termes from our biology term data set : `SK-constant` (”inner coat”, ”in the mother cell”) = 22, and `SK-spectrum` (”inner coat”, ”in the mother cell”) = 15 ; another pair give `SK-constant` (”inner coat”, ”initiation of sporulation”) = 27, and `SK-spectrum` (”inner coat”, ”initiation of sporulation”) = 24. Variation between both pairs are not far according string kernels, though terms of the first pair are from one class (class 2) and the other pair compares terms from different classes (class 1 and class 2). We built a kernel matrix using both string kernels and achieved cluster labelling with this similarity information. Result is shown in Figure 10 :

```
> library("svcR")
> data("sk")

> retF <- findSvcModel(sk, MetOpt = "optimStoch", MetLab = "gridLabeling",
+     KernChoice = "KernGaussian", Nu = 0.5, q = 2000,
+     K = 1, G = 40, Cx = 0, Cy = 0)

> findSvcModel.summary(retF)
```

Even if `SK-constant` shows some capability to isolate clusters, a big cluster contains 1600 items, hence 85% of information. Such kernel is though challenging, perhaps including more lexical knowledge.

HC							KM							svcR						
C1	**C2**	**C3**	**C4**	**C5**	**C6**	#	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	#	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	#
17	17	16	17	17	16	481	23	27	16	12	12	9	153	17	17	16	17	17	16	481
22	24	16	14	12	10	152	4	18	24	18	29	6	49	11	17	23	21	13	14	283
14	16	25	22	11	12	199	22	16	20	11	20	11	64	3	25	22	22	22	6	156
27	27	0	7	13	27	15	3	22	22	17	17	18	103	4	20	21	25	25	5	113
17	17	17	17	17	17	78	9	27	18	0	27	18	11	15	31	26	16	6	6	81
30	49	7	7	3	5	61	0	10	31	52	3	3	29	6	21	19	19	19	16	63
13	13	20	20	13	20	15	17	11	19	22	11	20	54	17	17	17	17	17	17	78
12	8	27	15	19	19	26	4	28	20	22	16	10	50	0	28	39	17	6	11	18
4	24	21	21	21	9	219	0	0	48	52	0	0	44	0	0	25	75	0	0	4
25	0	25	25	0	25	4	17	24	17	12	22	7	41	8	75	17	0	0	0	12
4	21	23	23	22	7	137	0	20	20	43	13	3	30	11	14	22	25	14	14	276
4	19	19	19	19	20	84	23	8	8	27	10	23	48	10	24	19	19	19	10	21
60	40	0	0	0	0	10	17	19	18	15	16	16	245	0	25	25	25	25	0	4
12	19	19	19	15	15	26	9	20	7	28	17	20	46	17	17	17	17	17	17	6
5	18	20	27	26	4	104	8	21	19	19	19	15	53	7	20	20	20	20	13	15
10	24	19	19	19	10	21	21	5	19	21	19	16	43	0	20	20	20	20	20	5
14	14	14	14	21	24	29	17	6	11	22	17	28	18	14	14	19	17	19	17	42
3	24	38	18	9	9	34	5	22	20	20	20	13	55	0	0	26	32	21	21	19
7	20	20	20	20	13	15	14	16	16	20	18	14	49	0	25	25	25	25	0	8
11	33	11	22	11	11	9	3	18	26	21	29	3	34	0	25	50	25	0	0	4
0	24	24	24	24	6	17	3	18	24	33	18	3	33	17	17	17	17	17	17	6
0	12	39	41	4	4	51	4	21	14	25	29	7	28	5	18	20	27	26	4	104
0	20	20	20	20	20	25	12	19	23	23	11	12	328	25	75	0	0	0	0	4
0	100	0	0	0	0	1	15	24	20	13	15	13	46	35	34	11	7	7	7	74
0	11	22	22	44	0	9	10	24	26	5	19	17	42	0	23	23	23	23	8	13
0	0	40	20	20	20	5	2	28	21	25	21	4	57							
0	0	25	25	25	25	16	10	18	15	20	32	5	40	*12*	*19*	*20*	*20*	*16*	*12*	*1893*
0	0	25	35	20	20	20	7	15	26	22	26	4	27							
0	0	44	56	0	0	25	17	23	12	12	15	21	48							
0	0	0	80	0	20	5	0	28	20	16	28	8	25							
12	*19*	*20*	*20*	*16*	*12*	*1893*	*12*	*19*	*20*	*20*	*16*	*12*	*1893*							

Figure 9: Class distribution over clusters HC (left), KM (centre) and **svcR** (right). C1 to C6 represent given classes. A line represents the content (i.e. distribution as % of classes) of a cluster. Columns # represent the size of the cluster. Last line represents the content of the whole set of terms.

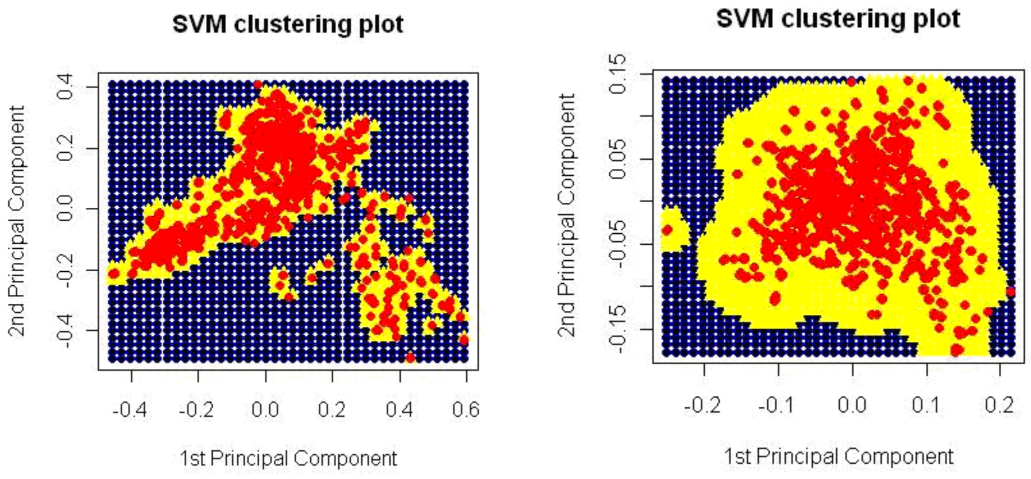


Figure 10: SVC using string kernel-constant (left), or using string kernel-spectrum (right).

5.3. Clusterability of a SVC model

Section 2.1 presents a general framework of a kernel method. It does not mean any assumption about clustering but moreover about classification. Nevertheless SVC is not a new technique in itself. SVC has been seen as one-cluster discovery since a ball in the dual space is targeted. Hence it was described in detail for a long time as a one class approach applied to novelty detection when information is deviating from a block of well known information. In R-project, **kernlab** package (Karatzoglou, Smola, Hornik, and Zeileis 2004; Karatzoglou, Meyer, and Hornik 2006) implements novelty detection task. When running one-class kernel to our dataset it returns a model of 394 support vectors

```
> library("kernlab")
> data("Cterm")

> tsv <- ksvm(Cterm, kernel = "stringdot", kpar = list(length = 5),
+     cross = 3, C = 10)

> tsv
```

, with $nu = 0.2$ and cross-validation 0.205. Our observation is that SVC performs well cluster extraction (or labeling) from a 2-dimensional map, depending on existence condition of clusters. It means that data ought to be separable in the 2-d map. Separability can be managed by composition of a metric with a radial-based function over the whole input matrix dimensions. A possible explanation for capability of SVC to identify clusters is related to the same problem as trying to flatten the skin of an orange onto a tabletop. In this case, projection is a procedure to transform locations and features from the three-dimensional surface onto the two-dimensional paper in a defined and consistent way. The result is some slight bulges and a lot of gaps. The transformation of map information from a sphere to a flat sheet can be accomplished in many ways but no single projection can accurately portray area, shape, scale, and direction. SVC clustering takes origin from capacity within projections to distort.

6. Conclusion

We developed, improved and applied a density and kernel based method called support vector clustering (SVC) we implemented as an R-project package (**svcR**). The package is available from the CRAN R project server (http://cran.r-project.org/ see Software, Packages; **svcR** version 1.4), and downloadable from the R graphical user interface (required R libraries : **quadprog**, **ade4** and **spdep**). First we proved that mapping points in the data space to a grid and using the sphere radius from the attribute space and a k-nearest-neighbor approach improves time consumption for cluster labeling. In this sense, SVC can be seen as an efficient cluster extraction if clusters are separable in a 2-D map. Secondly we found a representation for term clustering using a mixed Jaccard-Radial base kernel and we proved its efficiency with SVC for term clustering in a natural language processing task as lexical classification (i.e. oriented ontology knowledge acquisition). Some investigations remain under R implementation to integrate C functions for matrix acquisition so as to make the toolkit more scalable in data size. Semantic and lexical-based kernels are promising for application in text mining frameworks. Yet it must understand how to select and integrate attributes for the description of terms.

Acknowledgments

Special thanks to Roy Varshavsky, Marine Campedel, Dunyon Lee and Olivier Chapelle for their discussion. The methodology discussed in this paper has been supported by the INRA-1077-SE

grant from the French Institute for Agricultural Research (agriculture, food & nutrition, environment and basic biology).

References

Ben-Hur A, Horn D, Siegelmann H, Vapnik V (2001). “Support Vector Clustering.” *Journal of Machine Learning Research*, **2**, 125–137.

Berwin A, Turlach R, Weingessel A (2007). **quadprog***: Functions to solve Quadratic Programming Problems.* R package version 1.5-3, URL http://CRAN.R-project.org/package=quadprog.

Bilen B (2005). “Support Vector Clustering of Microarray Gene Expression Data.” Technical Report, Bilkent University, Turkey. URL http://www.cs.bilkent.edu.tr/~guvenir/courses/cs550/Workshop/Biter_Bilen.pdf.

Bilenko M, Mooney RJ (2002). “Learning to Combine Trained Distance Metrics for Duplicate Detection in Databases.” Technical Report AI 02-296, Artificial Intelligence Lab, University of Texas at Austin. URL http://www.cs.utexas.edu/~ml/papers/marlin-tr-02.pdf.

Bivand R (2010). **spdep***: Spatial Dependence, Weighting Schemes, Statistics and Models.* R package version 0.5-14, URL http://CRAN.R-project.org/package=spdep.

Campedel M, Moulines E (2005). “Méthodologie de Sélection de Caractéristiques pour la Classification d’Images Satellitaires.” In *Proceedings of the Conférence Nationale sur l’Apprentissage (CAP’05), Nice, France*, pp. 107–108. URL http://www.cmi.univ-mrs.fr/~fdenis/cap05/programme.html.

Cover T, Hart P (1967). “Nearest Neighbor Pattern Classification.” *IEEE Transactions on Information Theory*, **13**(1), 21–27.

Daille B (2003). “Conceptual Structuring through Term Variations.” In DM F Bond A Korhonen, A Villacicencio (eds.), *Proceedings of the 38th International Conference of Association for Computational Linguistics (ACL’03), Workshop on Multiword Expressions: Analysis, Acquisition and Treatment, Sapporo, Japan.* URL http://acl.ldc.upenn.edu/acl2003/.

Datar M, Immorlica N, Indyk P, Mirrokni V (2004). “Locality-Sensitive Hashing Scheme based on P-stable Distributions.” In *Proceedings of the 20th Annual Symposium on Computational Geometry (SCG’04), New York, USA*, pp. 253–262. ACM New York, NY, USA. URL http://www.informatik.uni-trier.de/~ley/db/conf/compgeom/compgeom2004.html.

Dray S, Dufour A (2007). “The **ade4** Package: Implementing the Duality Diagram for Ecologists.” *Journal of Statistical Software*, **22**(4), 1–20. URL http://www.jstatsoft.org/v22/i04/.

Eskin E, Arnold A, Prerau M, Portnoy L, Stolfo S (2002). “A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data.” In D Barbara, S Jajodia (eds.), *Applications of Data Mining in Computer Security*, pp. 77–102. Kluwer.

Eveillard D, Guermeur Y (2002). “Statistical Processing of SELEX Results.” Proceedings of the 10th International Conference on Intelligent Systems for Molecular Biology (ISMB’02), Poster Session, Edmonton, Canada. URL http://www.iscb.org/cms_addon/Conferences/ismb2002/.

Fellbaum C (1998). *WordNet : An Electronic Lexical Database.* MIT Press.

Gale WA, Church KW, Yarowsky D (1992). “A Method for Disambiguating Word Senses in a Large Corpus.” *Computers and the Humanities*, **26**, 415–439.

Grefenstette G (1994). "SEXTANT: Extracting Semantics from Raw Text: Implementation Details, Heuristics." *Integrated Computer-Aided Engineering*, **1**, 527–536.

Harris Z (1968). *Mathematical Structure of Language*. John Wiley & Sons.

Hearst M (1992). "Automatic Acquisition of Hyponyms from Large Text Corpora." In *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92), Nantes, France*, pp. 539–545. URL http://acl.ldc.upenn.edu/C/C92/.

Horn D (2001). "Clustering via Hilbert Space." *Physica A*, **302**(1), 70–79.

Jaccard P (1901). "Etude Comparative de la Distribution Florale dans une Portion des Alpes et des Jura." *Bulletin de la Société Vaudoise des Sciences Naturelles*, **37**, 241–272.

Jain AK, Dubes RC (1988). *Algorithms for Clustering Data*. Englewood Cliffs, New Jersey, Prentice Hall.

Karatzoglou A, Meyer D, Hornik K (2006). "Support Vector Machines in R." *Journal of Statistical Software*, **15**(9), 1–28. URL http://www.jstatsoft.org/v15/i09/.

Karatzoglou A, Smola A, Hornik K, Zeileis A (2004). "**kernlab** – An S4 Package for Kernel Methods in R." *Journal of Statistical Software*, **11**(9), 1–20. URL http://www.jstatsoft.org/v11/i09/.

Kees J, Marchiori E, van der Vaart A (2003). "Finding Clusters using Support Vector Classifiers." In *Proceedings of the 18th ESANN-European Symposium on Artificial Neural Networks (ESANN'03), Bruges, Belgium*, pp. 23–25. URL http://www.dice.ucl.ac.be/esann/index.php?pg=esann03_programme.

Kushmerick N (2000). "Wrapper Induction: Efficiency and Expressiveness." *Artificial Intelligence*, **118**(1), 15–68.

Lazarevic A, Ozgur A, Ertoz L, Srivastava J, Kumar V (2003). "A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection." In *Proceedings of the 3rd SIAM International Conference on Data Mining (ICDM'03), San Francisco, USA*, pp. 25–36. URL http://www.siam.org/meetings/sdm03/.

Lee J, Lee D (2005). "An Improved Cluster Labeling Method for Support Vector Clustering." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**(3), 461–464.

Levenshtein VI (1966). "Binary Codes Capable of Correcting Insertions and Reversals." *Soviet Physics Doklady*, **10**(8), 707–710.

Lodhi H, Saunders C, Taylor JS, Cristianini N, Watkins C (2002). "Support Vector Clustering." *Journal of Machine Learning Research*, **2**, 419–444.

Lucas A (2007). **amap**: *Another Multidimensional Analysis Package*. R package version 0.8-2, URL http://mulcyber.toulouse.inra.fr/projects/amap/.

Moschitti A (2009). "Syntactic and Semantic Kernels for Short Text Pair Categorization." In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL'09) , Athens, Greece*, pp. 576–584. Association for Computational Linguistics Morristown, NJ, USA. URL http://www.eacl2009.gr/Conference/.

Nazarenko A, Zweigenbaum P, Bouaud J, Habert B (1997). "Corpus-Based Identification and Refinement of Semantic Classes." *Journal of the American Medical Informatics Association*, **4**(suppl), 585–589.

Park J, Ji X, Zha H, Kasturi R (2004). "Support Vector Clustering combined with Spectral Graph Partitioning." In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04) , Cambridge, UK*, pp. 581–584. URL http://www.informatik.uni-trier.de/~ley/db/conf/icpr/index.html.

Pereira F, Tishby N, Lee L (1993). "Distributional Clustering of English Words." In *Proceedings of the 31th Conference of the Association for Computational Linguistics (ACL'93), Jerusalem, Israel*, pp. 183–190.

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Puma-Villanueva WJ, Bezerra GB, Lima CA, Zuben FJV (2005). "Improving Support Vector Clustering with Ensembles." In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'05), Montreal, Canada*, pp. 13–15. URL http://computer-vision.org/ijcnn05/.

Riloff E (1993). "Automatically Constructing a Dictionary for Information Extraction Tasks." In *Proceedings of the 11th National Conference on Artificial Intelligence (NCAI'93) , Washington, USA*, pp. 811–816. AAAI Press/The MIT Press. URL http://www.aaai.org/Conferences/AAAI/aaai93.php.

Salton G, McGill MJ (1983). *Introduction to Modern Information Retrieval*. McGraw Hill, New York.

Schölkopf B, Platt J, Shawe-Taylor J, Smola A, Williamson R (2001). "Estimating the Support of a High-Dimensional Distribution." *Neural Computation*, **33**, 1443–1471.

Smadja FA, McKeown KR (1990). "Automatically Extracting and Representing Collocations for Language Generation." In *Proceedings of the 28th International Conference of Association for Computational Linguistics (ACL'90), Pittsburgh, USA*, pp. 574–579. Association for Computational Linguistics Morristown, NJ, USA. URL http://www.informatik.uni-trier.de/~ley/db/conf/acl/acl90.html.

Soderland S (1999). "Learning Information Extraction Rules for Semi-Structured and Free Text." *Machine Learning*, **34**(1-3), 233–272.

Teo CH, Vishwanathan SVN (2006). "Fast and Space Efficient String Kernels using Suffix Arrays." In *Proceedings of the International Conference on Machine Learning (ICML'06) Pittsburgh, USA*, pp. 929–936. ACM New York, NY, USA. URL http://www.machinelearning.org/icml2006_proc.html.

Xu J, Zhang X (2004). "Kernels Based on Weighted Levenshtein Distance." In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'04), Budapest, Hungary*, pp. 3015–3018. URL http://www.Conferences.hu/budapest2004/.

Yang J, Estivill-Castro V, Chalup S (2002). "Support Vector Clustering through Proximity Graph Modelling." In *Proceedings of the 9th International Conference on Neural Information Processing,(ICONIP'02) , Singapore*, pp. 898–903. URL http://www3.ntu.edu.sg/Iconip02/.

Zhang Y, Su H, Jia T, Chu J (2005). "Rule Extraction from Trained Support Vector Machines." In *Proceedings of the 9th International Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD'05), Hanoi, Vietnam*, pp. 61–70. URL http://www.jaist.ac.jp/PAKDD-05/.