Semi-supervised learning using multiple clusterings with limited labeled data

Germain Forestier[a], Cédric Wemmert[b,*]

[a] *MIPS, University of Haute-Alsace, France*
[b] *ICube, University of Strasbourg, France*

Abstract

Supervised classification consists in learning a predictive model using a set of labeled samples. It is accepted that predictive models accuracy usually increases as more labeled samples are available. Labelled samples are generally difficult to obtain as the labelling step if often performed manually. On the contrary, unlabeled samples are easily available. As the labeling task is tedious and time consuming, users generally provide a very limited number of labeled objects. However, designing approaches able to work efficiently with a very limited number of labeled samples is highly challenging. In this context, semi-supervised approaches have been proposed to leverage from both labeled and unlabeled data.

In this paper, we focus on cases where the number of labeled samples is very limited. We review and formalize eight semi-supervised learning algorithms and introduce a new method that combine supervised and unsupervised learning in order to use both labeled and unlabeled data. The main idea of this method is to produce new features derived from a first step of data clustering. These features are then used to enrich the description of the input data leading to a better use of the data distribution. The efficiency of all the methods is compared on various artificial, UCI datasets, and on the classification of a very high resolution remote sensing image. The experiments reveal that our method shows good results, especially when the number of labeled sample is very limited. It also confirms that combining labeled and unlabeled data is very useful in pattern recognition.

Keywords: semi-supervised learning, classification, pattern recognition, remote sensing

1. Introduction

The number of available labeled samples is very important in supervised classification. If too few examples are given to a supervised learning algorithm, the induced predictive model will generally have poor performance. Unfortunately, in many real-world applications, labeled samples are difficult to obtain due to the cost of manual labeling. Moreover, in many applications, the user only gives few examples and the system has to find similar objects in a database (*e.g.* content-base image retrieval, on-line web-page recommendation, etc.). In these cases, only few labeled samples are available although many unlabeled data are present (*i.e.* all the other instances in the database). In web-page recommendation for example, the user labels only interesting pages. It is not possible to ask him to produce other samples as he might not know other pages interesting him. The same problem appears with on-line shopping services, when a customer buys a product and the system targets to automatically advise him for other related products. The system only knows which products the user bought and their ratings (*i.e.* collaborative filtering). Furthermore, in the problems involving the user for visual content analysis, the amount of available knowledge is generally very limited.

*ICube Laboratory, 300 bd Brant, CS 10413 - F-67412 Illkirch Cedex, France
wemmert@unistra.fr, +33 3 68 85 45 81

Another challenging task is to obtain a high classification accuracy when the ratio between available labeled data and the number of features is unbalanced. If the number of features is high, standard classifiers will need many training samples to perform an accurate classification. This observation is known as the Hughes phenomenon [1]. In the remote sensing field, the hyperspectral sensors produce data with multiple spectral bands, up to 200 (which means 200 floating values for each pixel). With such data, more details can be observed in the land cover, *i.e.* the number of classes of interest is increased. More features and more classes call for more samples, which are generally expensive and time consuming to acquire. The same observation can be made with the object-based analysis of Very High Spatial Resolution (VHSR) images. With these images, a first step of segmentation is generally performed to build regions (*i.e.* homogeneous set of pixels). These regions are then characterized by multiple features (*e.g.* spectral, spatial or contextual). Thus, the dataset is often composed of objects described by a high number of characteristics, but with very few samples.

Despite the fact that labeled samples are rare and insufficient compared to the data space dimension, unlabeled samples are generally available in important quantities. Some research efforts revealed that these samples can be used to improve supervised classification [2, 3, 4, 5, 6, 7]. Moreover, as stated by Zhou [8], semi-supervised learning and ensemble learning are complementary, and stronger learning machines can be generated by leveraging unlabeled data and classifiers combination.

The method proposed in this paper is slightly different from the existing ones, as we use many unsupervised classifications to create new features to describe the labeled samples. Then, a supervised classification algorithm is applied in this new data space. As an unsupervised classification creates clusters that tend to maximize intra-cluster similarity and inter-cluster dissimilarity, no labeled sample is needed, but no label is assigned to the different clusters. The clustering could be seen as a way to summarize the distribution of the samples. It is sometimes used to reduce the data before the classification step. Moreover, as unsupervised learning can only use data distribution, the classes must respect, to some extends, this distribution. If two classes form a highly homogeneous cluster in the feature space, one can not expect a simple clustering to separate them. Even if our approach can weaken this condition, it must be kept in mind.

In this article, we first present some related work and motivate our method in Section 2. The contribution of this paper is also to draw an overview of the classical semi-supervised approaches in a common formalization, and to present our new way of dealing with datasets having very few samples (Section 3). They are also compared through many experiments on a large set of artificial datasets and classical datasets from the UCI repository (Section 4). The results are compared with other semi-supervised algorithms and classical supervised methods to quantify the improvement gained from the unsupervised clustering pre-processing. We also present some results obtained on a real dataset extracted from a VHSR remote sensing image of an urban area of Strasbourg (France). Finally, we conclude and draw some perspectives about this work in Section 5.

2. Related work

Multiple previous efforts have shown that unlabeled data can help to improve the classification accuracy when very few labeled samples are available. As proposed by Gabrys and Petrakieva [9] or Bouchachia [10], these methods can be grouped into three main families: (1) *pre-labeling* approaches, where unlabeled data are labeled with an initial classifier trained on the set of labeled objects, (2) *post-labeling* approaches, where clusters of the whole dataset are labeled estimating their composition in terms of labeled objects, and (3) *semi-supervised* approaches, which consist in dealing with both labeled and unlabeled data at the same time during the clustering process.

2.1. Pre-labeling approaches

A first way to exploit unlabeled objects is the co-training method defined in [11]. The main idea is to use two complementary classification methods to iteratively label the unlabeled data. This assumes that two independent and complementary feature sets exist on the data.

To extend this method and to avoid the independence and redundancy of the feature sets (which is not realistic in real-life problems), Goldman and Zhou presented in [12] a co-training strategy which uses

unlabeled data to improve the performance of a supervised classifier. Their method uses two different supervised learners which can select some unlabeled data to label the other learner in an iterative way. Experiments have shown that the method increases the accuracy of the ID3 algorithm. In [13], another version based on the Expectation Maximization (EM) algorithm is proposed and applied to text classification.

More recently, Raskutti et al. [14] exposed a co-training method that does not necessary need two complementary supervised learning algorithms. The idea is to produce an alternate view of the data by performing an unsupervised classification algorithm on all the dataset (*i.e.* labeled and unlabeled). Then, the original view and the view built from the clustering are used to create two independent predictors for co-training.

In [15], the authors presented a co-training approach assuming to have two views of the data. The method uses the correlation between the two views to produce extra positive and negative samples in an iterative process. Experiments were performed when only one labeled sample is available. They show that the method overcome other co-training approaches.

Unlike co-training, ASSEMBLE [16] can build semi-supervised ensemble of any size, and does not require the domain to have multiple views. ASSEMBLE incorporates self-labeled examples in a boosting framework. At each iteration of ASSEMBLE, examples from the unlabeled set are labeled by the current ensemble and added to the training set.

2.2. Post-labeling methods

The idea of these approaches is to first produce a clustering of the dataset and then to evaluate the purity of each cluster, by calculating their composition in terms of label. Any clustering algorithm can be used in the first step. In [17], a general approach based on the Expectation Maximization (EM) algorithm is proposed.

Other methods [18, 19] work on the optimization of the purity in the clusters extracted from the dataset. They propose to use a Genetic Algorithm (GA) to iteratively refine the class membership of the unlabeled objects so that the maximum a posteriori based predicted labels of the data in the labeled dataset are in agreement with the known labels.

2.3. Semi-supervised clustering

In [20], an empirical study of various semi-supervised learning techniques on a variety of datasets is presented. Different experiments were made to evaluate the influence of the size of the labeled and unlabeled sets, or the effect of noise in the samples. The paper concludes that the performance of the methods is heavily dependent of the field of application and the nature of the dataset. However, using labeled *and* unlabeled samples improves the accuracy in most of the cases.

Basu et al. [21] have defined two variants of the Kmeans algorithm, dealing with labeled data. The main idea is to used the labeled objects to guide the seeding step of the algorithm. In the first method, the seeds are calculated according to the labeled data and the algorithm is ran classically, while in the second case, the labeled objects used for the seeding stay in their initial cluster.

Another approach was exposed in [22]. The idea is to simultaneously compute the clustering and the classification, instead of proceeding in two sequential steps. To achieve this goal, the authors define an objective function evaluating not only the classification but also the clustering ability by mixing two terms: the misclassification rate (for the supervised part) and the clustering impurity (for the unsupervised aspect). A quite similar method is presented in [23] and applied to real marketing datasets.

Ao et al. [24] proposed to combine supervised and unsupervised models via unconstrained probabilistic embedding. In their approach, they considered an ensemble problem in which outputs coming from models developed in the supervised and unsupervised modes are combined to improve the classification accuracy of supervised model ensemble.

2.4. Ensemble clustering methods

Ensemble clustering methods consist in building a consensus clustering from multiple different clusterings. The task of merging multiple clustering results is considered more complicated than merging multiple

classification models as there is no common set of classes that can be used to merge the decisions. Consequently, multiple methods have been proposed to (1) generate the set of clusterings to merge and (2) obtain a single clustering result [25, 26, 27, 28, 29].

A strong interest has been given in evaluating diversity present in the ensemble [30, 31]. The underlying idea is that merging highly similar clusterings is not likely to improve the final result as all the input clusterings would bring the same information. Thus, different strategies can be used to increase the diversity within the ensemble [32]. For example, using different clustering methods, or the same method with different parameters can lead to the generation of different results.

More recently, Yu *et al.* [33, 34, 35] proposed a new approach of semi-supervised constraint ensemble clustering. The idea is to transfer expert knowledge into pairwise constraints used to make a feature selection to avoid noise degradation of the results. Then, a selection of a subset of representative clustering results is extracted and they are merged to obtain the final solution.

Sampling the instances, or the attributes, can also be used to generate multiple views [36] of the same dataset and to cluster them individually to obtain multiple clustering results. Random projection [37] is also an interesting approach to generate diverse results. In this paper, we choose to combine different strategies to control the diversity in the generated clusterings on the unlabeled data. We used two different clustering methods (*i.e.*, Kmeans and EM), with different parameters (*i.e.*, number of clusters) and with different combination of features as input (*i.e.*, with or without original features).

2.5. Remote sensing applications

As mentioned previously, semi-supervised methods have been applied in many different domains (*e.g.* text, web or image processing). In this paper, we are interested in assessing these methods for remote sensing image classification. In this field, some previous work to take advantage of unlabeled data in the classification process were undertaken by [38, 39, 40]. They proposed three methods to incorporate simultaneously labeled and unlabeled samples in parametric, nonparametric and semi-parametric classifiers. They also gave some results on a small extract of an AVIRIS remote sensing image, to show the enhancement of the classification performance.

Later, in [41], the authors presented a new covariance matrix estimator. It produces a higher accuracy classification than standard covariance matrix estimation methods, when using a limited training data set. Some experiments on the classification of an agricultural zone of the Nevada, based on an AVIRIS image, showed the efficiency of the estimator on real problems.

Jia et al. [42] presented a new method to deal with hyperspectral data. The idea is to cluster the training data and then, to associate spectral clusters to information classes to build a cluster-space classification. Then, each pixel is classified according to its cluster membership and the membership of the cluster to information classes.

Morgan et al. [43] proposed another approach to solve the problem of the small amount of labeled data. It consists in using a feature reduction scheme that adaptively adjusts itself to the size of the samples dataset. Some experiments are presented on hyperspectral data obtained with the HyMap sensor (Hyperspectral Mapper). The results show that even if the feature space is reduced and the number of samples is very low, the produced classification has a high accuracy.

More recently, a novel approach using ensemble of semi-supervised classifiers was proposed by Roy et al. [44] for change detection in remotely sensed images. The novelty of this method is to use multiple classifiers instead of using a single weak classifier. The classifiers are trained using selected unlabeled and few labeled objects. A consensus is computed between the classifiers' results to help the choice of the unlabeled patterns for the next training step.

The method presented in this paper is slightly different from the ones presented above. All the co-training methods use the labeled and unlabeled samples together in the training step. If more labeled samples are available, the entire training step needs to be computed again, which is often costly. In our approach, the unsupervised classification can be seen as a pre-processing step, which is performed only once. Then, depending on the availability of labeled samples, the supervised classification can be computed. This training part is very efficient as the number of samples is very low.

3. Semi-supervised methods

In this section, we formally present the different approaches used in our experiments. Let X denote a set of n data objects $x_j \in X$. We consider a q-class classification problem with m labeled and l unlabeled objects where m is very low and $l >> m$.
Let L be the set of labeled objects of X:

$$L = \{(x_1, y_1), \ldots, (x_m, y_m)\} \tag{1}$$

where $y_i \in \{1, \ldots, q\}$ are the class values of the samples.
Let U be the set of unlabeled objects of X:

$$U = \{(x_{m+1}, 0), \ldots, (x_{m+l}, 0)\} \tag{2}$$

where 0 means that there is no label assigned to this object.

The objective of the semi-supervised classification is to build a classifier based on the training dataset X. This classifier is a function associating one of the q classes to any object x. It can be formally defined as:

$$y = C_X(x) \ : \ y \in \{1, \ldots, q\} \tag{3}$$

3.1. Pre-labeling methods

The first methods are called *pre-labeling methods*. The idea is to perform an initial classification C_L on the labeled data only (L). Then, the unlabeled data are labeled according to this classification.

In the *static labeling* (**SL**) [9] method, the unlabeled data are all labeled in one step by simply applying C_L to U. A new dataset W is built as:

$$W = \{(x_j, y_j) \ : \ y_j = C_L(x_j), x_j \in U\} \tag{4}$$

Then, in a second step, the final classification is computed as:

$$y = C_{L \cup W}(x) \tag{5}$$

The algorithm of this method is described in Algorithm 1.

Algorithm 1 Static labeling (**SL**)

1: build a classification C_L given the labeled dataset L
2: let $W = \{(x_j, y_j) \ : \ y_j = C_L(x_j), x_j \in U\}$
3: build the final classifier $C_{L \cup W}$

Another pre-labeling approach is the *dynamic labeling* (**DL**) [9] method. As for the static labeling method, a classifier C_L is build according to the labeled dataset. Then, instead of labeling all the objects of U, they are iteratively labeled, one sample at a time. One object x_j of U is chosen and labeled according to C_L. Then it is added to L and a new classifier is trained with $L \cup \{x_j\}$. The process iterates until all unlabeled samples are labeled. The algorithm is presented in Algorithm 2.

The order in which the objects are chosen can be defined according to the confidence in the classification of each one. The definition of the most confidently classified object depends on the classification method used.

3.2. Post-labeling methods

At the opposite of pre-labeling methods that use the unlabeled objects to enhance an initial classifier, post-labeling methods first build a clustering of all the objects of the dataset, but without considering their label. Then, the unlabeled data of each cluster are labeled to the majority class of their cluster. Let K_l,

Algorithm 2 Dynamic labeling (**DL**)

let $U' = U$ and $W' = \varnothing$
build a classifier $C_{L \cup W'}$
for all $x_j \in U'$ chosen according to their class confidence **do**
  $W' := W' \cup \{(x_j, t_j) \ : \ t_j = C_{L \cup W'}(x_j)\}$
  $U' := U' \setminus \{(x_j, 0)\}$
end for
build the final classifier $C_{L \cup W'}$

$l = 1, \ldots, k$, denote the clusters produced by the initial clustering on the dataset, and c_{lj}, $j = 1, \ldots, q$ the number of labeled objects from class j in the cluster l:

$$c_{lj} = \|\{(x_i, y_i) \in L \ : \ (x_i, y_i) \in K_l, y_i = j\}\| \tag{6}$$

Cluster labeling by majority. The post-labeling method presented hereafter is called *cluster labeling by majority* (**CLM**) [9] and is described in Algorithm 3. It is composed of three steps.

- The first step consists in labeling all clusters containing at least one labeled sample. The label assigned to each of these clusters is the majority class of all labeled objects of the cluster.
- The second step labels the clusters containing no labeled sample with the label of the most similar already labeled cluster. The similarity measure $\Delta(K_j, K_k)$ depends on the clustering method and estimates the similarity between two clusters K_l and K_k.
- Finally, in the third step, the final classifier is build according to all the new labeled objects.

Algorithm 3 Cluster labeling by majority (**CLM**)

build a clustering $K = \{K_l, l = 1 \ldots k\}$ on X
let $LK := \varnothing$
for all K_l, $l = 1 \ldots k$ **do**
  if $\sum_{j=1}^{q} c_{lj} \neq 0$ **then**
    $y_{K_l} = \arg\max_{j \in \{1 \ldots q\}} (c_{lj})$
    $W_l = \{(x_i, y_{K_l}), x_i \in K_l\}$
    $LK := LK \cup K_l$
  end if
end for
for all $K_u \ : \ \sum_{j=1}^{q} c_{uj} = 0$ **do**
  $K_m = \arg\max_{K_l \in LK} (\Delta(K_l, K_u))$
  label all objects in cluster K_u with label y_{K_m}
  $W_u = \{(x_i, y_{K_m}), x_i \in K_u\}$
end for
build the final classifier $C_{W_1 \cup \ldots \cup W_k}$

Purity optimization. Another family of methods [19] is based on the optimization of the *purity* of each cluster found by an initial clustering K on the dataset. The evaluation of the purity Π of a cluster is based on two criteria:

- the class impurity which measures the percentage of *minority examples* in the different clusters of K;
- the number of clusters k, which should be kept low in a majority of cases.

The *minority examples* are labeled objects belonging to a class that is not the majority class of their cluster. As previously defined, the majority class of a cluster K_l is $y_{K_l} = \arg\max_{j \in \{1 \dots q\}} (c_{lj})$. Thus, the minority examples $m(K_l)$ of a cluster K_l can be defined as:

$$m(K_l) = \{(x_i, y_i) \in K_l \ : \ y_i \neq y_{K_l}\} \tag{7}$$

and the minority examples $M(K)$ of the clustering K:

$$M(K) = \{m(K_l) \ : \ \forall l \in [1 \dots k]\} \tag{8}$$

The purity can be defined as:

$$\Pi(K) = \mathsf{impurity}(K) + \eta \times \mathsf{penalty}(k) \tag{9}$$

where

$$\mathsf{impurity}(K) = \frac{\|M(K)\|}{n}$$

and

$$\mathsf{penalty}(k) = \begin{cases} \sqrt{\frac{k-q}{n}} & k \geqslant q \\ 0 & k < q \end{cases}$$

The parameter η ($0 < \eta < 2$) determines the penalty associated to the number of clusters k.

The first algorithm defined by Eick et al. [19] is a greedy algorithm called *Single Representative Insertion/Deletion Steepest Decent Hill Climbing with Randomized Restart* (**SRIDHCR**) [19]. Some objects are randomly selected (between q and $2q$ objects) to be the initial representatives of the clusters, which are created by affecting each object to its closer representative. Then, one object is added to the set of representative or one representative object is removed from this set. The quality $\Pi(K)$ is evaluated and the algorithm iterates until no significant improvement is observed on the quality of the clustering. The algorithm is run r times and the best solution is kept. It is completely described in Algorithm 4.

Algorithm 4 Single Representative Insertion/Deletion Steepest Decent Hill Climbing with Randomized Restart) (**SRIDHCR**)

for $i = 1$ to r **do**
  $\mathsf{rep} := \{(x_i, y_i) \in L \ : \ \text{randomly chosen}\}$ and $q \leqslant \|\mathsf{rep}\| \leqslant 2q$
  while not terminated **do**
    let $s^1 = \mathsf{rep} \cup (x_i, y_i) \ : \ x_i \notin \mathsf{rep}$
    let $s^2 = \mathsf{rep} \setminus (x_i, y_i) \ : \ x_i \in \mathsf{rep}$
    let $S = \arg\min_{s \in \{s1, s2\}} \Pi(s)$
    if $\Pi(S) < \Pi(\mathsf{rep})$ **then**
      $\mathsf{rep} := S$
    else if $\Pi(S) = \Pi(\mathsf{rep})$ and $\|S\| > \|\mathsf{rep}\|$ **then**
      $\mathsf{rep} := S$
    else
      terminate
    end if
  end while
end for
the best solution is kept

The second algorithm, called Supervised Clustering using Evolutionary Computing (**SCEC**) [19], tries to find the best representatives set by using an evolutionary computing approach. A first set of ps clusterings is randomly generated and genetic operators are then applied to create the next generations. Three operators

are used in the SCEC algorithm:

- mutation: a representative object is replaced by another object that is not in the representative set of one solution;
- crossover: a new solution is created from two initial solutions; the intersection of the representatives of the two initial solutions are included in the new solution, and each representative only present in one of the initial solutions is included with a probability of 50% in the new solution;
- copy: a solution of the current generation is copied in the new generation.

The new representatives are randomly chosen according to a k-tournament technique among the new representatives built with the operators. The process iterates over a fixed number N of generations. The algorithm is described in Algorithm 5.

Algorithm 5 Supervised Clustering using Evolutionary Computing (**SCEC**)

$pop_0 = \{S_r = \{s_i^r = (x_i, y_i) \in L \ : \ \text{randomly chosen}\}$
$1 \leqslant r \leqslant ps\}$
for $i = 1$ to N **do**
  $mu_i = \texttt{mutation}(pop_{i-1})$
  $cr_i = \texttt{crossover}(pop_{i-1})$
  $co_i = \texttt{copy}(pop_{i-1})$
  $pop_i = \text{select\_by\_k\_tournament}(mu_i, cr_i, co_i)$
  for all $S_r \in pop_i$ **do**
    let K^r be the clustering corresponding to the representatives S_r
    for all $l \in [1 \dots k]$ **do**
      $K_l^r = \{(x_j, y_j) \ : \ \texttt{dist}(x_j, s_l^r)$ is minimal $\}$
    end for
    compute $\Pi(S_r)$
  end for
end for
the best solution S_r is kept

3.3. Semi-supervised clustering

The last type of approaches, called *semi-supervised clustering* approaches, incorporates labeled and unlabeled data at the same time. The idea is that the clustering of the data is guided by the labeled samples.

Refined clustering. The *refined clustering* (**RC**) [9] method consists in two steps:

1. creation of a new fully labeled dataset from both labeled and unlabeled objects ($L \cup U$);
2. application of a nearest neighbor classifier to learn the new dataset.

The first step is very important and is composed of two stages:

1. creation of a new dataset, including both labeled and unlabeled data subsets and representing as much as possible information;
2. labeling of this dataset by the CLM technique described previously.

To create the new dataset, a divisive approach is proposed. The idea is to split the existing clusters containing more than one type of class label. A threshold is used to define the minimal representative ratio of a class in a cluster. Moreover, to prevent minority classes from disappearing, minority classes only present in one cluster are conserved by splitting the cluster into two. The complete algorithm of the method is presented in Algorithm 6.

Algorithm 6 Refined clustering (**RC**)

build a clustering $K = \{K_l, l = 1 \dots k\}$ on X with k relatively small
for all K_l, $l = 1 \dots k$ **do**
  if $\sum_{j=1}^{q} c_{lj} \neq 0$ **then**
    while $nbc > 1$ **do**
      $nbc := 1$
      for all $m \in [1 \dots q]$ **do**
        let $r_m = \frac{c_{lm}}{\sum_{j=1}^{q} c_{lj}}$
        if $r_m < \Theta$ and $\forall i \in [1 \dots k], i \neq l, g_{im} = 0$ **then**
          $K' := split(K, l)$
        else if $r_m \geqslant \Theta$ **then**
          $nbc := nbc + 1$
          $K' := split(K, l)$
        end if
      end for
    end while
  end if
end for
apply **CLM** to K'

Seeding methods. Finally, we used in this study the *seeding methods* proposed by Basu et al. [21]. These methods are variants of the well-known Kmeans partitioning method. The objective of the Kmeans method is to generate a k-partitioning $K = \bigcup_{i=1}^{k} K_i$ of the dataset X. Each cluster K_i is represented by its gravity center μ_i. As previously mentioned, we consider a q-class classification problem of a dataset $X = L \cup U$ where L are labeled objects. The idea is to guide the Kmeans algorithm by using the labeled objects L as initial seeds. A first q-partitioning $\{S_i\}_{i=1\dots q}$ of L is calculated by grouping the objects of L having a same label into one cluster. An assumption is made that corresponding to each cluster S_i, there is at least one seed point.

The first method, *Seeded-Kmeans* (**SK**) [21], simply uses this seed partition instead of a random initialization. Then, the algorithm is run without any modification. The complete algorithm of the SK method is presented in Algorithm 7.

Algorithm 7 Seeded-Kmeans (**SK**)

$\mu_i^{(0)} = \frac{1}{|S_i|} \sum_{x \in S_i} x$ for $i = 1 \dots q$
$t = 0$
repeat
  $K^{(t+1)} = \left\{ K_i^{(t+1)}, i = 1 \dots q \right\}$
  where $K_i^{(t+1)} = \left\{ x \in X \ : \ i = \arg\min_h \|x - \mu_h^{(t)}\|^2 \right\}$
  $\mu_i^{(t+1)} = \frac{1}{|K_i^{(t+1)}|} \sum_{x \in K_i^{(t+1)}} x$ for $i = 1 \dots q$
  $t = t + 1$
until convergence

The second method, called *Constrained-Kmeans* (**CK**) [21], the seed clustering is used as in the *Seeded-Kmeans* method, to initialize the clustering. However, the labeled objects are not reassigned during the execution of the algorithm. They are constrained to stay in their initial cluster. The algorithm is given in Algorithm 8.

Algorithm 8 Constrained-Kmeans (**CK**)

1: $\mu_i^{(0)} = \frac{1}{|S_i|} \sum_{x \in S_i} x$ for $i = 1 \dots q$
2: $t = 0$
3: **repeat**
4: $K^{(t+1)} = \left\{ K_i^{(t+1)}, i = 1 \dots q \right\}$
where
$K_i^{(t+1)} = S_i \cup \left\{ x \in X \ : \ i = \arg\min_h \|x - \mu_h^{(t)}\|^2 \right\}$
5: $\mu_i^{(t+1)} = \frac{1}{|K_i^{(t+1)}|} \sum_{x \in K_i^{(t+1)}} x$ for $i = 1 \dots q$
6: $t = t + 1$
7: **until** convergence

3.4. Semi-supervised learning enhanced by multiple clusterings

The method that we propose, called *Semi-supervised learning enhanced by multiple clusterings* (**SLEMC**), could be categorized as a post-labeling method. Indeed, it tries to improve the classification by first producing a clustering of the dataset. The clustering, computed on all the labeled and unlabeled objects, regroups the similar instances together, maximizing the intracluster similarity and the intercluster dissimilarity. If the classes are well separated in the feature space, we should be able to associate to each cluster one of the classes, using the class of the labeled samples belonging to the cluster.

Unfortunately, in real world problems, classes are generally not well separated. It is then possible to have samples from different classes in one cluster, or no sample in others. To address this issue, the proposed method uses a combination of multiple clusterings.

We consider here b clustering methods $\{C_k\}_{1 \leq k \leq b}$. Each method is an instance of one clustering algorithm (*e.g.* Kmeans, EM, SOM, etc.), having its own parameters and producing a partition $K_k = \{K_k^1, \dots, K_k^n\}$ of the n objects from the dataset X composed of both labeled and unlabeled samples.

Each object $x_i \in X$ is described by a vector of p features $(a_1^i, \dots, a_p^i)$. The idea is to affect to each labeled sample $(x_i, y_i) \in L, \ \forall i : 1 < i < m$, a new features vector:

$$v(x_i) = \left(a_1^i, \dots, a_p^i, K_1^i, \dots, K_b^i, y_i\right) \tag{10}$$

where K_j^i is the cluster affected in the j^{th} clustering partition K_j to x_i, and y_i is the label of the sample x_i (see equation 1).

Then, a predictive model $C_V : X \to \{1, \dots, q\}$ where $\{1, \dots, q\}$ are the class values of the samples (see equation 3), can be induced from this new dataset $V = \{v(x_i)\}_{i=1}^m$, using a classical supervised learning method (*e.g.* Naive Bayes).

The description of each unlabeled object x_j of U is then extended in the same way:

$$v'(x_j) = \left(a_1^j, \dots, a_p^j, K_1^j, \dots, K_b^j, 0\right) \tag{11}$$

The predictive model C_V, learned previously on the labeled samples, can be applied for each unlabeled object $x_j \in U$. The algorithm presenting the complete classification process is presented in algorithm 9.

The computational cost of the proposed method is directly linked to the clustering algorithms and the supervised learning algorithm that are used in Algorithm 9. It is interesting to note that the clustering step can be performed offline, as the model only needs to be trained once. Then, on runtime, the affectation of unlabeled samples to the clusters can be performed without computing the entire clustering again. As we assume that the number of labeled sample is very low, it is not likely that their use in the clustering process would have significantly affected the clustering result. In the following experiments, we used Naive Bayes to train the model, which has a standard complexity of $O(nd)$ for n points in d dimensions. Consequently, the complexity of SLEMC increases linearly with the addition of new dimensions obtained using the clustering algorithms (*i.e.* KMeans, EM, etc.).

Algorithm 9 Semi-supervised learning enhanced by multiple clusterings (**SLEMC**)

apply b clustering algorithms $\{C_k\}_{1 \leq k \leq b}$ to the dataset X
each method C_k produces a partition $K_k = \{K_k^1, \ldots, K_k^n\}$ of the n objects
for all $(x_i, y_i) \in L$ **do**
  $v(x_i) = \left(a_1^i, \ldots, a_p^i, K_1^i, \ldots, K_b^i, y_i\right)$
end for
apply a supervised learning method to produce a predictive model C_V from $V = \{v(x_i)\}_{i=1}^m$
affect the features vector $v'(x_j) = \left(a_1^j, \ldots, a_p^j, K_1^j, \ldots, K_b^j, 0\right)$ to each $x_j \in U$
use C_V to label all objects of U

4. Experiments

To evaluate the efficiency of the SLEMC approach, three sets of experiments were carried out. First on artificial datasets (Section 4.1) where the data and the samples are well correlated to the class information (*i.e.* the ideal case), then on standard UCI datasets (Section 4.2), and finally on a real application in the domain of remote sensing image analysis (Section 4.3). For all experiments, we compare the SLEMC method to semi-supervised methods presented in Section 3. Comparison is also performed with supervised classifiers, using only the labeled objects for the learning (*i.e.* the unlabeled data were ignored). Several supervised algorithms were selected, from different families of learning methods: the standard tree inducer C4.5, Naive Bayes (NB) and a 1-nearest-neighbor (1-NN) algorithm.

To apply SLEMC, the number of clusterings to compute on the dataset has to be fixed (*i.e.* how many attributes will have each object in the new data space). Then, the different clustering methods have to be chosen. Four different configurations were tested, to study the importance of the number of clusterings. The four configurations are the followings:

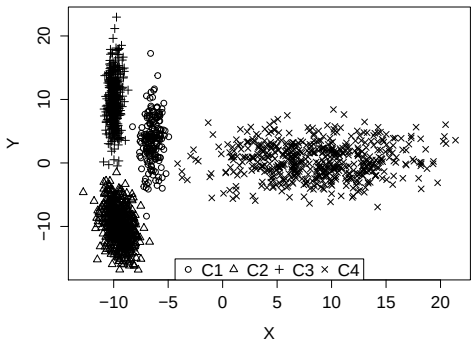
1. *Simple*: one EM (Expectation-Maximization);
2. *Low*: one EM and one KMeans;
3. *Medium*: two EM and two KMeans;
4. *High*: c EM and c KMeans (with c the number of classes in the dataset).

Note that only EM and KMeans algorithms were chosen for convenience (*i.e.* low complexity) but other clustering algorithms could have been used. Furthermore, as the method aim at leveraging from the class distribution in the data space, these two algorithms are particularly well suited. The supervised method used to produce the learning model is Naive Bayes in all experiments.

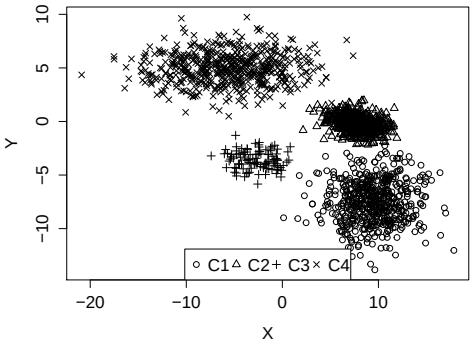
Moreover, we also evaluate an alternative setup for SLEMC, that keeps the description along with adding new features for each object. These configurations are referred as *Simple+*, *Low+*, *Medium+* and *High+*. Each method was run with a number of clusters equals to the number of classes actually present in the dataset except for the *High* configuration where the clustering method had k clusters ($k \in \{2, \ldots, c\}$, randomly chosen).

For each experiment, each dataset was split into two sets, each composed of 50% of the objects. The first set was used as data with unknown label for the unsupervised learning task of the semi-supervised methods. The second set was used to select few samples considered as the available knowledge. The rest of the second set was used to evaluate the methods (*i.e.* compute accuracy measure). We chose to evaluate the method when 2, 4, 8 and 16 samples per class were available. We carried out the experiments for each of these configurations (2, 4, 8 and 16 samples) for each dataset.

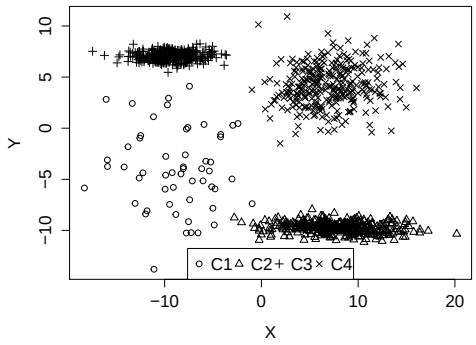
As the number of available labeled samples is very low the performance can greatly differ depending on their selection. Consequently, the experiments were carried out 30 times and the results were averaged. At each execution, we split randomly the datasets in two sets (50% for the unsupervised learning, 50% in which we selected the available labeled samples and used the remaining objects for the evaluation).



(a) 2d-4c-no1 dataset.



(b) 2d-4c-no5 dataset.



(c) 2d-4c-no8 dataset.

Figure 1: Example of artificial datasets used in the experiments.

For the sake of readability, only the top 3 accuracies are presented for each experiment in result tables. However, the full results are available on the companion web page of the paper [1].

4.1. Evaluation on artificial benchmarks

The first set of experiments were carried out on artificial datasets, generated by the software provided by Julia Handl [45]. Three different datasets configurations were used:

- `2d-4c-noX`: 4 classes described by 2 attributes;
- `10d-4c-noX`: 4 classes described by 10 attributes;
- `2d-10c-noX`: 10 classes described by 2 attributes.

For all of them, 10 different datasets were randomly generated (no0, no1, ... no9). Figure 1 shows three datasets generated for the 2d4c series.
For each experiment, the accuracy of the model was calculated. Tables 1, 2 and 3 present the top 3 accuracies respectively for the series 2d-4c, 10d-4c and 2d-10c.

[1] `http://germain-forestier.info/src/is2015/` (Accessed: 4 January 2016)

From these tables, one can notice that the *High* configuration of our method outperforms all the other approaches as it appears most of the time in rank 1 for multiple datasets. This is even more visible for the configurations where very few (2 or 4) samples are available. For example in Table 1 and Table 2, the configuration *High* is - always - ranked first when 2 samples are available and is first 9 times out of 10 in Table 3. One can also notice that when the number of labeled sample increase (8, 16) the method that keep the description along with the new features (*i.e.* configurations with the "+") tend to perform better.

We also computed for the three sets of artificial datasets (`2d-4c-noX`, `2d-10c-noX` and `10d-4c-noX`) the average ranking among the datasets. This is computed by summing the rank of each method for each experiment and dividing the result by the number of experiments. The Table 4 shows these results. In this table, one can observe that the configuration "High" performs particularly well as it is always in first position when the number of labelled samples available is low (2 or 4). When the number of available samples increases (8 and 16) the DL method provides better results.

4.2. Evaluation on UCI datasets

In this section, we compare our approach to the semi-supervised methods presented previously on various datasets of the UCI repository [46]. The Table 5 summarizes information on each dataset. Note that it is not guarantee that the class distribution is correlated with the data clusters distribution. With such datasets, the information provided by the clustering will not give any relevant insight to enhance the model learning.

The results for each dataset and for each experiment are presented in Tables 7 and 8, where the values are the accuracy of the top 3 methods for each configuration. The number of samples used is presented in brackets at the beginning of each line.

In these tables, one can observe that the proposed methods outperformed, most of the time, the supervised learning and the other semi-supervised approaches, especially when the number of samples is very low (2 or 4 samples per class). The best results are obtained on the *wine* dataset, where our methods are 8 times in the top 3 for the 12 configurations. On *ionosphere*, *anneal*, *heart*, *cardio* and *thyroid*, they are 6 times cited for the 12 configurations.

It is also important to notice that for some configurations and some datasets the difference of accuracy between our method and the other semi-supervised methods appears to be important. For example, for the configuration (2) on the dataset *optidigits*, our method with the *High* configuration gives 54.77% of accuracy while the other semi-supervised present in the top 3 (DL and 1-NN) respectively gives 50.85% and 48.88%. This can also be noted on the *wine* dataset for the configuration (2) as our method gives 88.40% and 83.25% respectively for the *High* and the *Medium* configurations, while the SCEC method only obtained 79.96%.

Our method using the *High* configuration (*i.e.* c EM and c KMeans) seems to give the best results as it appears 23 times considering all the top 3. This result enforces the intuitive feeling that adding more clusterings improves the result, as the objects are described with more details (*i.e.* have more attributes).

Moreover, one can also observe that most of the time, the configurations without keeping the description of the objects (*Low*, *Simple*, *Medium* and *High*) give good results when the number of available labeled objects is low (2 and 4), whereas the configurations which keep the description along with adding new features (*Low+*, *Simple+*, *Medium+* and *High+*) give better results when the number of available labeled objects is higher (8 and 16).

This can be explained by observing that, when the number of available labeled samples is low, their description is too weak to build an efficient classifier. However, as the features added from the clustering algorithms are computed on 50% of the dataset, the descriptions are more relevant. When the number of available labeled samples increases, the description of the objects (*i.e.* the original features) are more meaningful, and are more precise than the rough information offered by the clusterings.

As stated in the introduction, if the data space of the dataset is not correlated with the class information, using clustering is useless. This affirmation can be studied on the *segment* and *robot* datasets, where the proposed semi-supervised approach obtains worse results than 1-NN or C4.5, regardless of the number of available samples.

Among the other semi-supervised methods, the DL gave particularly good results as it is present 22 times in the top 3. It is followed by the SL which is present 5 times. These results tend to show that the pre-labeling give better results than the other ones on these datasets.

Table 1: Top 3 accuracies for the experiments on the 2d-4c series.

dataset	Rank 1	Rank 2	Rank 3
2d-4c-no0 (2)	**High** 94.878	DL 94.313	**Medium** 93.992
2d-4c-no0 (4)	**High** 97.973	DL 97.164	SCEC 96.705
2d-4c-no0 (8)	**High** 99.356	DL 99.027	1-NN 98.146
2d-4c-no0 (16)	DL 99.635	**High** 99.537	1-NN 98.849
2d-4c-no1 (2)	**High** 93.446	DL 89.339	1-NN 88.142
2d-4c-no1 (4)	**High** 99.292	DL 96.545	**Medium** 96.268
2d-4c-no1 (8)	DL 99.768	**High+** 99.286	**Medium+** 99.261
2d-4c-no1 (16)	DL 99.775	**Simple+** 99.767	**Medium+** 99.767
2d-4c-no2 (2)	**High** 92.543	DL 85.093	**Medium** 84.984
2d-4c-no2 (4)	DL 96.967	**High** 96.888	1-NN 95.19
2d-4c-no2 (8)	**High+** 99.606	**Medium+** 99.483	**High** 99.392
2d-4c-no2 (16)	**High+** 99.893	**Medium+** 99.866	**Low+** 99.831
2d-4c-no3 (2)	**High** 92.174	**Medium** 87.427	DL 86.695
2d-4c-no3 (4)	**High** 98.128	DL 97.298	**Medium** 97.165
2d-4c-no3 (8)	**High** 99.191	DL 99.0	1-NN 98.386
2d-4c-no3 (16)	DL 99.576	**High** 99.188	1-NN 99.148
2d-4c-no4 (2)	**High** 96.123	**Medium** 91.631	DL 90.946
2d-4c-no4 (4)	**High** 94.867	DL 93.703	**Medium** 93.357
2d-4c-no4 (8)	**High** 99.641	DL 99.273	1-NN 98.722
2d-4c-no4 (16)	**High** 99.791	DL 99.637	**Medium** 99.591
2d-4c-no5 (2)	**High** 92.477	**Medium** 88.577	DL 87.697
2d-4c-no5 (4)	**High** 94.752	DL 93.431	1-NN 92.44
2d-4c-no5 (8)	**High** 95.803	DL 95.268	1-NN 94.77
2d-4c-no5 (16)	DL 99.437	**High** 99.423	1-NN 99.095
2d-4c-no6 (2)	**High** 92.358	DL 90.066	SCEC 89.328
2d-4c-no6 (4)	**High** 97.162	DL 94.926	1-NN 94.27
2d-4c-no6 (8)	**High** 98.728	DL 98.271	1-NN 97.441
2d-4c-no6 (16)	DL 99.985	1-NN 99.681	SL 99.548
2d-4c-no7 (2)	**High** 96.088	DL 91.105	1-NN 90.296
2d-4c-no7 (4)	**High** 99.078	**Medium** 95.745	DL 95.671
2d-4c-no7 (8)	DL 99.465	**High** 99.226	**Low+** 99.032
2d-4c-no7 (16)	DL 99.737	SL 99.626	**Simple+** 99.626
2d-4c-no8 (2)	**High** 90.843	DL 89.037	**Medium** 87.273
2d-4c-no8 (4)	DL 94.08	**High** 93.706	**Medium** 92.151
2d-4c-no8 (8)	DL 99.328	**High+** 99.264	**Medium+** 99.183
2d-4c-no8 (16)	DL 99.772	**High+** 99.772	**Medium+** 99.603
2d-4c-no9 (2)	**High** 89.415	**Medium** 85.878	DL 85.305
2d-4c-no9 (4)	**High** 97.242	DL 95.588	1-NN 93.579
2d-4c-no9 (8)	DL 99.473	**High+** 99.087	**High** 98.887
2d-4c-no9 (16)	DL 99.608	SL 99.224	RC 99.16

The Table 6 shows the average ranking for two sets of the UCI datasets: Set 1 {iris, wine, ionosphere, waveform, cardio, optidigts, mushroom} and Set 2 {wdbc, pima, anneal, heart, trans, robot, vehicle, thyroid, liver}. The first set contains the datasets where the classes are following the data distribution, while in the second they are not. As one can see in this table, for the first set, the proposed method provides the best results and comes first in average ranking. For the second set, the 1-NN method and DL methods provide the best results. However, the average rankings in the second set (*e.g.* 6.0) are much higher than the average

Table 2: Top 3 accuracies for the experiments on the 10d-4c datasets.

dataset	Rank 1	Rank 2	Rank 3
10d-4c-no0 (2)	**High** 93.486	DL 91.041	**Medium** 90.539
10d-4c-no0 (4)	**High** 90.57	DL 89.97	**Medium** 89.193
10d-4c-no0 (8)	**High** 99.501	DL 98.917	**Medium** 98.84
10d-4c-no0 (16)	DL 99.983	**High** 99.876	**Medium** 99.685
10d-4c-no1 (2)	**High** 93.118	**Medium** 90.382	DL 90.076
10d-4c-no1 (4)	**High** 98.814	DL 97.86	**Medium** 96.37
10d-4c-no1 (8)	**High** 99.525	DL 98.75	1-NN 97.939
10d-4c-no1 (16)	DL 100.0	**High** 99.906	RC 99.536
10d-4c-no2 (2)	**High** 91.99	**Medium** 90.114	DL 87.576
10d-4c-no2 (4)	**High** 98.58	DL 97.635	1-NN 94.522
10d-4c-no2 (8)	DL 99.988	**High** 99.911	**Medium** 98.664
10d-4c-no2 (16)	DL 96.66	**High** 96.591	SL 96.546
10d-4c-no3 (2)	**High** 88.762	DL 86.992	**Medium** 85.782
10d-4c-no3 (4)	**High** 98.618	**Medium** 96.919	DL 96.256
10d-4c-no3 (8)	**High** 99.284	DL 99.21	**Medium** 98.561
10d-4c-no3 (16)	DL 99.976	CLM 99.743	SL 99.639
10d-4c-no4 (2)	**High** 93.987	**Medium** 91.515	DL 89.727
10d-4c-no4 (4)	**High** 98.547	DL 97.197	**Medium** 95.033
10d-4c-no4 (8)	DL 99.063	**High** 99.025	**Medium** 98.647
10d-4c-no4 (16)	DL 99.901	**High** 99.69	**Medium** 98.97
10d-4c-no5 (2)	**High** 96.662	**Medium** 93.447	DL 89.859
10d-4c-no5 (4)	**High** 98.837	DL 98.533	**Medium** 98.391
10d-4c-no5 (8)	DL 100.0	**High** 99.873	**Medium** 99.826
10d-4c-no5 (16)	DL 100.0	**High+** 99.966	**Medium+** 99.961
10d-4c-no6 (2)	**High** 93.234	**Medium** 90.744	DL 87.484
10d-4c-no6 (4)	DL 96.624	1-NN 96.008	**High** 95.786
10d-4c-no6 (8)	DL 98.562	**High** 98.306	1-NN 97.958
10d-4c-no6 (16)	DL 99.678	1-NN 99.448	**High** 99.397
10d-4c-no7 (2)	**High** 90.284	**Medium** 87.291	DL 86.367
10d-4c-no7 (4)	DL 98.122	**High** 97.535	**Medium** 97.097
10d-4c-no7 (8)	DL 99.587	**High** 98.984	**Medium** 98.286
10d-4c-no7 (16)	DL 96.601	RC 95.793	SL 95.624
10d-4c-no8 (2)	**High** 96.022	DL 94.245	**Medium** 90.742
10d-4c-no8 (4)	**High** 95.697	DL 94.142	**Medium** 94.08
10d-4c-no8 (8)	DL 99.729	**High** 99.686	**Medium** 98.48
10d-4c-no8 (16)	**High** 99.838	DL 99.776	**Medium** 99.524
10d-4c-no9 (2)	**High** 94.589	DL 89.319	**Medium** 89.249
10d-4c-no9 (4)	**High** 94.732	DL 94.404	**Medium** 93.576
10d-4c-no9 (8)	DL 96.548	**High** 96.305	**Medium** 96.231
10d-4c-no9 (16)	DL 99.873	**Medium+** 99.865	**High+** 99.865

rankings in the first set (*e.g.* 1.71). It means that for this set, it is more difficult to find a single method that outperforms the others. This is mainly due to the fact that most the semi-supervised methods compared in this paper make also the assumption that the data distribution is following the classes distribution.

We also carried out an experiment to study the importance of the amount of unlabeled data in our methods. We made some experiments in which we used 50% of the dataset as unlabeled data (as in the previous experiment) but also made experiments with 25% and 10% of available unlabeled data. We used

Table 3: Top 3 accuracies for the experiments on the 2d-10c series.

dataset	Rank 1	Rank 2	Rank 3
2d-10c-no0 (2)	**High** 82.603	DL 81.135	**Medium** 80.494
2d-10c-no0 (4)	**High** 92.049	DL 91.146	1-NN 89.918
2d-10c-no0 (8)	DL 98.503	**High** 97.696	1-NN 97.296
2d-10c-no0 (16)	DL 99.401	**High** 98.99	1-NN 98.675
2d-10c-no1 (2)	**High** 83.994	DL 82.526	1-NN 80.401
2d-10c-no1 (4)	**High** 94.664	DL 94.366	1-NN 91.698
2d-10c-no1 (8)	DL 94.774	**High+** 94.148	**Medium+** 93.903
2d-10c-no1 (16)	DL 99.064	**Medium+** 98.916	**High+** 98.883
2d-10c-no2 (2)	**High** 86.16	DL 84.689	1-NN 82.726
2d-10c-no2 (4)	**High** 91.594	DL 91.577	1-NN 89.808
2d-10c-no2 (8)	DL 98.388	**High+** 97.702	**High** 97.695
2d-10c-no2 (16)	DL 95.992	**High+** 95.992	**Medium+** 95.958
2d-10c-no3 (2)	**High** 87.203	DL 83.754	1-NN 82.976
2d-10c-no3 (4)	**High** 95.38	DL 95.012	1-NN 93.843
2d-10c-no3 (8)	DL 97.767	**High** 97.24	1-NN 97.015
2d-10c-no3 (16)	**High+** 99.18	DL 99.05	**Medium+** 98.917
2d-10c-no4 (2)	**High** 86.224	DL 81.573	1-NN 80.373
2d-10c-no4 (4)	**High** 96.087	DL 95.25	1-NN 92.966
2d-10c-no4 (8)	DL 99.161	**High** 98.86	1-NN 97.738
2d-10c-no4 (16)	DL 96.388	**High+** 96.128	**Medium+** 95.881
2d-10c-no5 (2)	**High** 86.937	DL 86.667	1-NN 84.91
2d-10c-no5 (4)	DL 95.057	**High** 94.187	1-NN 93.188
2d-10c-no5 (8)	DL 90.649	**High** 89.76	1-NN 89.343
2d-10c-no5 (16)	DL 94.983	**High+** 94.611	**High** 94.284
2d-10c-no6 (2)	**High** 87.379	DL 86.416	1-NN 85.233
2d-10c-no6 (4)	DL 96.23	**High** 96.13	1-NN 95.212
2d-10c-no6 (8)	DL 98.725	**High** 98.301	1-NN 98.118
2d-10c-no6 (16)	DL 99.639	**Low+** 99.402	**Simple+** 99.377
2d-10c-no7 (2)	DL 85.94	**High** 85.371	1-NN 84.026
2d-10c-no7 (4)	DL 91.84	**High** 90.61	1-NN 90.152
2d-10c-no7 (8)	DL 98.659	1-NN 97.191	**High+** 97.158
2d-10c-no7 (16)	DL 99.54	**High+** 98.91	**Medium+** 98.8
2d-10c-no8 (2)	**High** 78.789	DL 76.083	**Medium** 73.937
2d-10c-no8 (4)	DL 94.513	**High** 94.503	1-NN 91.705
2d-10c-no8 (8)	DL 95.08	**High** 93.927	**High+** 93.335
2d-10c-no8 (16)	DL 95.943	**High+** 95.548	**Medium+** 95.475
2d-10c-no9 (2)	**High** 88.696	DL 86.647	1-NN 85.887
2d-10c-no9 (4)	**High** 95.156	DL 95.086	1-NN 94.145
2d-10c-no9 (8)	DL 98.443	**High** 97.686	1-NN 97.565
2d-10c-no9 (16)	DL 99.111	**High** 99.01	1-NN 98.488

the *iris* dataset and we run the experiment for 2, 4 and 8 labeled samples per class. The results are presented in Figure 2 where each curve represents the evolution of the accuracy according to the amount of unlabeled samples available. As expected, the method leverages from unlabeled samples, and the results improve as more unlabeled samples are available.

Table 4: Average ranking of top 3 methods for the artificial datasets.

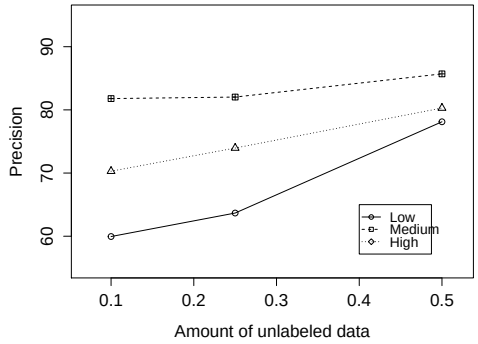
dataset	Rank 1	Rank 2	Rank 3
2d-4c-noX (2)	**High=1.0**	DL=2.4	**Medium=3.5**
2d-4c-noX (4)	**High=1.2**	DL=1.9	**Medium=3.6**
2d-4c-noX (8)	DL=1.8	**High=2.4**	**High+=4.6**
2d-4c-noX (16)	DL=1.5	**High+=4.7**	**Medium+=5.3**
2d-10c-noX (2)	**High=1.1**	DL=1.9	1-NN=3.2
2d-10c-noX (4)	**High=1.4**	DL=1.6	1-NN=3.0
2d-10c-noX (8)	DL=1.0	**High=3.4**	**High+=3.7**
2d-10c-noX (16)	DL=1.1	**High+=2.9**	**Medium+=3.8**
10d-4c-noX (2)	**High=1.0**	**Medium=2.4**	DL=2.6
10d-4c-noX (4)	**High=1.3**	DL=1.9	**Medium=3.1**
10d-4c-noX (8)	DL=1.3	**High=1.7**	**Medium=3.2**
10d-4c-noX (16)	DL=1.1	**High=4.3**	SL=5.0

Table 5: Information about the different datasets

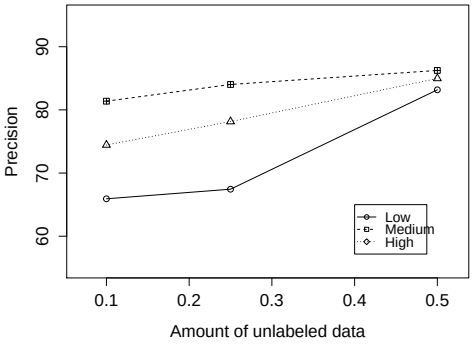
dataset	#classes	#attributes	#objects
iris	3	5	150
wine	3	14	178
wdbc	2	31	569
pima	2	9	768
ionosphere	2	35	351
anneal	5	39	898
waveform	3	22	5000
heart	2	35	349
cardio	10	36	2126
thyroid	3	22	3772
optidigts	10	63	5620
trans	2	5	748
robot	4	5	5456
vehicle	4	19	846
segment	7	20	2310
liver	2	7	345
mushroom	2	23	8124

Table 6: Average ranking of top 3 methods for two sets of UCI datasets: Set 1 {iris, wine, ionosphere, waveform, cardio, optidigts, mushroom} and Set 2 {wdbc, pima, anneal, heart, trans, robot, vehicle, thyroid, liver}.

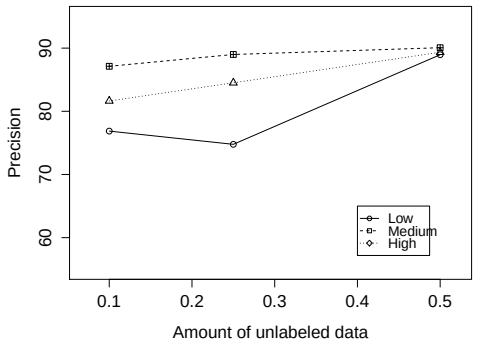
dataset	Rank 1	Rank 2	Rank 3
Set 1 (2)	**High=1.71**	**Medium**=2.71	1-NN=4.0
Set 1 (4)	**High=2.28**	DL=3.28	1-NN=3.42
Set 1 (2,4)	**High=2.0**	1-NN=3.71	DL=3.78
Set 2 (2)	1-NN=6.0	SL=7.0	DL=7.17
Set 2 (4)	1-NN=5.64	DL=6.9	SCEC=7.0
Set 2 (2,4)	1-NN=5.82	DL=7.05	SCEC=7.14



(a) Experiments on *iris* with 2 samples per class.



(b) Experiments on *iris* with 4 samples per class.



(c) Experiments on *iris* with 8 samples per class.

Figure 2: Experiments with variable amount of available unlabeled data.

4.3. Remote Sensing data evaluation

As presented in the introduction, in the field of remote sensing classification, the problem of the low ratio between the number of features and the number of samples is really important when dealing with hyperspectral data or with very high resolution images. Indeed, in this last case, the classification is basically computed in two steps: a segmentation of the image is performed, producing a set of regions (*i.e.* groups of homogeneous pixels); then, these regions are characterized by many features (spectral attributes, geometrical features, spatial attributes, etc.). Thus, we have a new dataset to classify, composed of few regions (compared to the number of pixels in the image), but characterized by more features.

We present here an experiment on the classification of an extract of a very high remote sensing image of an urban area of the city of Strasbourg (France). The input data is a pan-sharpened Quickbird© image with 4 spectral bands and a spatial resolution of 0.7 meter, *i.e.* a pixel on the image represents a square of 0.7×0.7m on the ground. The image and the expert annotations used for the learning task are shown on Figure 3.

After having computed a segmentation on this image using the watershed segmentation algorithm [47, 48], we characterized each region by the 23 following features:

- 4 features representing the mean and the standard deviation of each spectral channel of the pixels composing the region;

Table 7: Top 3 accuracies for the experiments on the UCI datasets (1).

dataset	Rank 1	Rank 2	Rank 3
iris (2)	**High** 80.821	**Medium** 77.246	1-NN 75.362
iris (4)	1-NN 89.577	DL 89.471	RC 87.725
iris (8)	**Low+** 93.007	DL 92.876	**Simple+** 92.876
iris (16)	DL 94.444	SL 94.321	**Low+** 94.321
wine (2)	**High** 88.394	**Medium** 83.253	SCEC 79.96
wine (4)	**High** 88.009	**Medium** 87.706	DL 87.489
wine (8)	**High** 90.051	DL 89.385	**Medium** 89.231
wine (16)	NB 84.553	**Simple+** 84.309	**Medium+** 84.065
wdbc (2)	**Medium** 89.369	**High** 88.0	CLM 87.429
wdbc (4)	DL 81.92	CLM 81.655	RC 81.498
wdbc (8)	DL 89.502	CLM 89.167	SL 89.055
wdbc (16)	SL 77.817	**Medium+** 77.712	**High+** 77.632
pima (2)	**Medium** 59.491	DL 58.193	1-NN 57.798
pima (4)	DL 63.67	1-NN 63.378	SCEC 63.369
pima (8)	DL 64.13	SL 62.255	NB 62.192
pima (16)	NB 67.33	SL 67.301	DL 66.847
ionosphere (2)	**High** 65.517	1-NN 65.107	DL 64.308
ionosphere (4)	**High** 63.293	SCEC 62.974	**Medium** 62.854
ionosphere (8)	SL 72.935	DL 72.327	**High+** 71.95
ionosphere (16)	**Medium+** 69.65	**Low+** 69.487	NB 69.441
anneal (2)	**Simple** 74.088	**Low+** 73.707	**Medium+** 73.638
anneal (4)	**Medium+** 64.627	1-NN 64.58	**Low+** 64.471
anneal (8)	1-NN 55.187	C45 53.267	**Low+** 52.702
anneal (16)	1-NN 54.202	C45 54.127	NB 52.984
waveform (2)	**High** 61.55	DL 60.306	1-NN 57.828
waveform (4)	DL 67.712	**High** 64.204	1-NN 63.165
waveform (8)	DL 73.669	SL 71.832	**High+** 70.685
waveform (16)	**Simple+** 79.246	NB 79.126	**Low+** 79.016
heart (2)	1-NN 63.157	**Low+** 62.961	**Medium+** 62.961
heart (4)	**Low+** 68.213	**High+** 68.213	NB 68.193
heart (8)	NB 66.076	**Low+** 63.608	**Medium+** 63.586
heart (16)	CLM 74.038	NB 73.615	RC 72.535
cardio (2)	**High** 89.143	1-NN 87.76	**Medium** 85.008
cardio (4)	1-NN 95.002	**High** 94.865	**Medium** 91.786
cardio (8)	1-NN 95.534	**High** 94.652	SCEC 92.981
cardio (16)	C45 96.667	1-NN 96.641	**High** 96.183

- 4 features representing the mean and the standard deviation of the value of each spectral channel over the sum of all channels values;
- 2 features representing the mean and the standard deviation of the mean of all spectral channels;
- 2 features calculated as the mean and the standard deviation of the NDVI (*Normalized Difference Vegetation Index*) of the pixels composing the region;
- 1 feature representing the area covered by the region;
- 1 feature calculated as the elongation of the shape represented by the region;

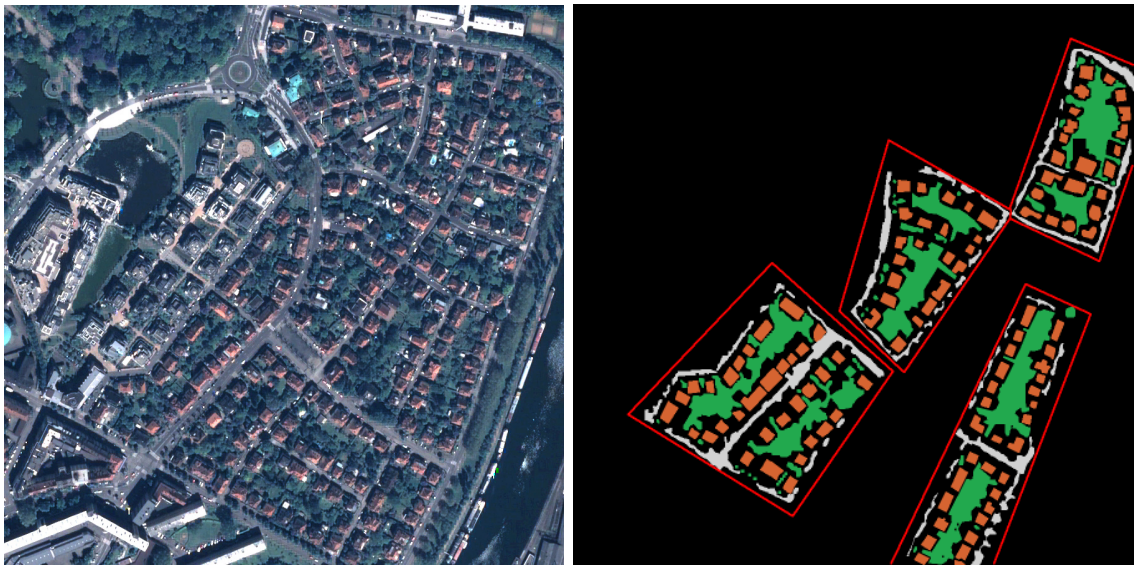
Table 8: Top 3 accuracies for the experiments on the UCI datasets (2).

dataset	Rank 1	Rank 2	Rank 3
thyroid (2)	NB 86.188	**Simple+** 86.188	**Low+** 86.188
thyroid (4)	NB 77.287	SL 76.382	C45 75.948
thyroid (8)	**High+** 67.879	SL 67.854	**Medium+** 67.811
thyroid (16)	**Simple+** 46.627	NB 46.612	**Low+** 46.607
optdigits (2)	**High** 54.765	DL 50.845	1-NN 48.879
optdigits (4)	**High** 77.561	DL 76.945	1-NN 75.001
optdigits (8)	1-NN 53.09	DL 52.469	**High** 51.618
optdigits (16)	1-NN 46.097	**High** 44.806	DL 44.387
trans (2)	C45 74.685	SK 73.775	CK 73.775
trans (4)	**Simple** 73.206	SK 72.641	CK 72.641
trans (8)	CLM 68.399	RC 68.389	NB 68.222
trans (16)	C45 71.452	CLM 71.326	RC 71.277
robot (2)	C45 63.574	SL 56.363	**Low+** 55.118
robot (4)	C45 64.175	NB 58.029	**Low+** 57.999
robot (8)	C45 79.028	**Simple+** 68.5	NB 68.484
robot (16)	C45 77.603	NB 68.27	SL 68.011
vehicle (2)	SCEC 38.024	1-NN 37.606	DL 35.992
vehicle (4)	1-NN 40.786	SCEC 39.615	C45 37.952
vehicle (8)	1-NN 34.851	C45 33.572	**Simple+** 32.268
vehicle (16)	C45 21.114	1-NN 20.929	SRIDHCR 18.561
segment (2)	1-NN 56.375	**High** 54.309	DL 52.571
segment (4)	1-NN 57.249	DL 54.655	SRIDHCR 54.543
segment (8)	1-NN 48.065	C45 47.27	**High+** 46.372
segment (16)	1-NN 58.038	C45 57.507	SRIDHCR 54.155
liver (2)	SCEC 51.865	**Medium+** 51.429	**High+** 51.429
liver (4)	**Simple+** 54.736	SCEC 54.634	NB 54.492
liver (8)	1-NN 51.923	C45 51.496	NB 51.261
liver (16)	C45 54.857	1-NN 54.19	**High** 52.5
mushroom (2)	**Medium** 78.472	SK 75.88	CK 75.88
mushroom (4)	**High+** 81.285	**High** 81.221	DL 80.887
mushroom (8)	DL 91.211	1-NN 90.997	SL 89.342
mushroom (16)	1-NN 80.835	C45 80.022	DL 79.509

- 1 feature measuring the fitting of the shape represented by the region and its oriented bounding box.

Finally, the dataset generated (dataset *remote sensing*) was composed of 186 objects described by 15 features, divided into three classes. As for the previous experiments, we chose three different configurations to study the importance of the number of clusterings. The three configurations are referred as *Low* (two EM clusterers), *Medium* (two EM and two KMeans) and *High* (6 EM and 6 KMeans). We also compare the result with three classical supervised classification methods (C4.5, 1-Nearest Neighbor and Naive Bayes) and the other semi-supervised approaches presented in section 3. Again, the experiment is performed 30 times and the results are averaged. At each run, the different sets are filled with randomly chosen samples. The results obtained with different number of samples are given in Table 9.

Again, these results confirm that *High* configuration improves the result, as it gives the best results when only few samples are available. It confirms that a minimal number of samples are needed by supervised methods to produce efficient results. But this lack of information can be balanced by the use of many clusterings on the unlabeled data.



(a) Extract of the Quickbird© image (Strasbourg, France - 900×900 pixels - 0.7m per pixel - 2001)

(b) Manual annotations used as samples

Figure 3: Data used for the `remote sensing` dataset

Table 9: Top 3 accuracies for the experiments on the `remote sensing` datasets.

dataset	Rank 1	Rank 2	Rank 3
remote (2)	**High** 84.783	**Medium** 82.802	1-NN 82.657
remote (4)	**High** 90.106	1-NN 90.0	SCEC 88.889
remote (8)	1-NN 89.412	SCEC 87.059	DL 86.993
remote (8)	1-NN 92.876	SRIDHCR 88.954	**High** 88.627

5. Conclusion

In this paper, we presented how multiple clustering results can be combined along with a supervised classifier, in order to achieve better results than some classical semi-supervised and supervised algorithms. The presented method has shown good results when the number of labeled samples is very low, and when unlabeled samples are available. We also give a presentation of the main methods of semi-supervised clustering in a same formalization and compared them objectively.

The experiments presented in this paper give some insights on how clustering results can help the supervised learning task and is a way to incorporate unlabeled data into the training process. An experiment on object-based remote sensing image processing highlights the suitability of such an approach for visual content analysis and understanding.

In future work, we are planing to study how the characteristics of the clustering results used in the method influence the classification accuracy. We plan to study ensemble clustering generation methods in order to generate multiple clusterings with high diversity. The diversity in the input clusterings seems to be the key to reach better accuracy as it provide complementary information. We also plan to study other clustering methods and see how they compete with the ones used in this paper. Finally, we also plan to further investigate how clustering can support the supervised classification process [49] for example for data editing.

Supplementary materials

- Java package containing the source code for the proposed method. (Java ARchive file) – `http://germain-forestier.info/src/is2016/` (Accessed: 20 May 2016)

References

[1] G. Hughes, On the mean accuracy of statistical pattern recognizers, IEEE Transactions on Information Theory 14 (1) (1968) 5 – 63.

[2] R. Raina, A. Battle, H. Lee, B. Packer, A. Y. Ng, Self-taught learning: Transfer learning from unlabeled data, in: International Conference on Machine Learning, ACM, New York, NY, USA, 2007, pp. 759–766.

[3] M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: A geometric framework for learning from labeled and unlabeled examples, J. Mach. Learn. Res. 7 (2006) 2399–2434.

[4] A. Shrivastava, V. M. Patel, R. Chellappa, Non-linear dictionary learning with partially labeled data, Pattern Recognition 48 (11) (2015) 3283 – 3292.

[5] S. Anand, S. Mittal, O. Tuzel, P. Meer, Semi-supervised kernel mean shift clustering, Pattern Analysis and Machine Intelligence, IEEE Transactions on 36 (6) (2014) 1201–1215.

[6] G. Huang, S. Song, J. Gupta, C. Wu, Semi-supervised and unsupervised extreme learning machines, Cybernetics, IEEE Transactions on 44 (12) (2014) 2405–2417.

[7] A. Shrivastava, S. Singh, A. Gupta, Constrained semi-supervised learning using attributes and comparative attributes, in: Computer Vision ECCV 2012, Vol. 7574 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 369–383.

[8] Z.-H. Zhou, When semi-supervised learning meets ensemble learning, Frontiers of Electrical and Electronic Engineering in China 6 (1) (2011) 6–16.

[9] B. Gabrys, L. Petrakieva, Combining labelled and unlabelled data in the design of pattern classification systems, International journal of approximate reasoning 35 (3) (2004) 251–273.

[10] A. Bouchachia, Learning with partly labeled data., Neural Computing and Applications 16 (3) (2007) 267–293.

[11] A. Blum, T. Mitchell, Combining labeled and unlabeled data with co-training, in: Proceedings of the Workshop on Computational Learning Theory, 1998, pp. 92–100.

[12] S. Goldman, Y. Zhou, Enhancing supervised learning with unlabeled data, in: International Conference on Machine Learning, 2000, pp. 327–334.

[13] K. Nigam, A. K. McCallum, S. Thrun, T. M. Mitchell, Text classification from labeled and unlabeled documents using EM, Machine Learning 39 (2/3) (2000) 103–134.

[14] B. Raskutti, H. L. Ferr, A. Kowalczyk, Combining clustering and co-training to enhance text classification using unlabelled data., in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 620–625.

[15] Z.-H. Zhou, D.-C. Zhan, Q. Yang, Semi-supervised learning with very few labeled training examples., in: AAAI International Conference on Artificial Intelligence, 2007, pp. 675–680.

[16] K. P. Bennett, A. Demiriz, R. Maclin, Exploiting unlabeled data in ensemble methods., in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 289–296.

[17] Z. Ghahramani, M. I. Jordan, Supervised learning from incomplete data via an em approach, in: Advances in Neural Information Processing Systems 6, Morgan Kaufmann, 1994, pp. 120–127.

[18] R. Kothari, V. Jain, Learning from labeled and unlabeled data using a minimal number of queries, IEEE Transactions on Neural Networks 14 (6) (2003) 1496–1505.

[19] C. Eick, N. Zeidat, Z. Zhao, Supervised clustering–algorithms and benefits, in: IEEE International Conference on Tools with Artificial Intelligence, 2004, pp. 774–776.

[20] N. V. Chawla, G. J. Karakoulas, Learning from labeled and unlabeled data an empirical study across techniques and domains., Journal of Artificial Intelligence Research 23 (2005) 331–366.

[21] S. Basu, A. Banerjee, R. J. Mooney, Semi-supervised clustering by seeding, in: International Conference on Machine Learning, 2002, pp. 27–34.

[22] W. Cai, S. Chen, D. Zhang, A simultaneous learning framework for clustering and classification, Pattern Recognition 42 (7) (2009) 1248–1286.

[23] M. Deodhar, J. Ghosh, A framework for simultaneous co-clustering and learning from complex data, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 250–259.

[24] X. Ao, P. Luo, X. Ma, F. Zhuang, Q. He, Z. Shi, Z. Shen, Combining supervised and unsupervised models via unconstrained probabilistic embedding, Information Sciences 257 (0) (2014) 101 – 114.

[25] L. I. Kuncheva, S. T. Hadjitodorov, L. P. Todorova, Experimental comparison of cluster ensemble methods, in: International Conference on Information Fusion, IEEE, 2006, pp. 1–7.

[26] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, The Journal of Machine Learning Research 3 (2003) 583–617.

[27] A. L. Fred, A. K. Jain, Combining multiple clusterings using evidence accumulation, Pattern Analysis and Machine Intelligence, IEEE Transactions on 27 (6) (2005) 835–850.

[28] G. Forestier, P. Gancarski, C. Wemmert, Collaborative clustering with background knowledge, Data & Knowledge Engineering 69 (2) (2010) 211–228.

[29] G. Forestier, C. Wemmert, P. Gancarski, Towards conflict resolution in collaborative clustering, in: IEEE International Conference Intelligent Systems, IEEE, 2010, pp. 361–366.
[30] L. Kuncheva, S. T. Hadjitodorov, et al., Using diversity in cluster ensembles, in: IEEE international conference on Systems, Man and Cybernetics, Vol. 2, IEEE, 2004, pp. 1214–1219.
[31] S. T. Hadjitodorov, L. I. Kuncheva, L. P. Todorova, Moderate diversity for better cluster ensembles, Information Fusion 7 (3) (2006) 264–275.
[32] H. Liu, T. Liu, J. Wu, D. Tao, Y. Fu, Spectral ensemble clustering, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 715–724.
[33] Z. Yu, H. Chen, J. You, H.-S. Wong, J. Liu, L. Li, G. Han, Double selection based semi-supervised clustering ensemble for tumor clustering from gene expression profiles, IEEE/ACM Transactions on Computational Biology and Bioinformatics 11 (4) (2014) 727–740.
[34] Z. Yu, P. Luo, J. You, H. Wong, H. Leung, S. Wu, J. Zhang, G. Han, Incremental semi-supervised clustering ensemble for high dimensional data clustering, IEEE Transactions on Knowledge and Data Engineering PP (99) (2015) 1–1.
[35] Z. Yu, L. Li, J. Liu, G. Han, Hybrid adaptive classifier ensemble, IEEE Transactions on Cybernetics 45 (2) (2015) 177–190.
[36] X. Zhao, N. Evans, J.-L. Dugelay, A subspace co-training framework for multi-view clustering, Pattern Recognition Letters 41 (2014) 73–82.
[37] F. Yang, X. Li, Q. Li, T. Li, Exploring the diversity in cluster ensemble generation: Random sampling and random projection, Expert Systems with Applications 41 (10) (2014) 4844–4866.
[38] B. M. Shahshahani, D. Landgrebe, The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon, IEEE Transactions on Geoscience and Remote Sensing 32 (5) (1994) 1087–1095.
[39] N. Durand, S. Derivaux, G. Forestier, C. Wemmert, P. Gançarski, O. Boussaid, A. Puissant, Ontology-based object recognition for remote sensing image interpretation, in: IEEE International Conference on Tools with Artificial Intelligence, Vol. 1, IEEE, 2007, pp. 472–479.
[40] G. Forestier, A. Puissant, C. Wemmert, P. Gançarski, Knowledge-based region labeling for remote sensing image interpretation, Computers, Environment and Urban Systems 36 (5) (2012) 470–480.
[41] J. P. Hoffbeck, D. A. Landgrebe, Covariance matrix estimation and classification with limited training data., IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (7) (1996) 763–767.
[42] X. Jia, R. J.A., Cluster-space representation for hyperspectral data classification, IEEE Transactions on Geoscience and Remote Sensing 40 (3) (2002) 593–598.
[43] J. T. Morgan, J. Ham, M. M. Crawford, A. Henneguelle, J. Ghosh, Adaptive feature spaces for land cover classification with limited ground truth data., International Journal of Pattern Recognition and Artificial Intelligence 18 (5) (2004) 777–799.
[44] M. Roy, S. Ghosh, A. Ghosh, A novel approach for change detection of remotely sensed images using semi-supervised multiple classifier system, Information Sciences 269 (0) (2014) 35 – 47.
[45] J. Handl, J. Knowles, Cluster generators for large high-dimensional data sets with large numbers of clusters (2005).
[46] D. Newman, S. Hettich, C. Blake, C. Merz, UCI repository of machine learning databases (1998).
[47] S. Derivaux, G. Forestier, C. Wemmert, S. Lefevre, Supervised image segmentation using watershed transform, fuzzy classification and evolutionary computation, Pattern Recognition Letters 31 (15) (2010) 2364–2374.
[48] G. Forestier, S. Derivaux, C. Wemmert, P. Gancarski, An evolutionary approach for ontology driven image interpretation, in: European Workshop on Evolutionary Computation in Image Analysis and Signal Processing, Vol. 4974, Springer, 2008, pp. 295–304.
[49] F. Petitjean, G. Forestier, G. Webb, A. E. Nicholson, Y. Chen, E. Keogh, et al., Dynamic time warping averaging of time series allows faster and more accurate classification, in: IEEE International Conference on Data Mining, IEEE, 2014, pp. 470–479.