Enhancing K-Means Using Class Labels

Billy Peralta, Pablo Espinace, and Alvaro Soto

Pontificia Universidad Católica de Chile

bmperalt@uc.cl, pespinac@ing.puc.cl, asoto@ing.puc.cl

May 9, 2013

Abstract

Clustering is a relevant problem in machine learning where the main goal is to locate meaningful partitions of unlabeled data. In the case of labeled data, a related problem is supervised clustering, where the objective is to locate class-uniform clusters. Most current approaches to supervised clustering optimize a score related to cluster purity with respect to class labels. In particular, we present Labeled K-Means (LK-Means), an algorithm for supervised clustering based on a variant of K-Means that incorporates information about class labels. LK-Means replaces the classical cost function of K-Means by a convex combination of the joint cost associated to: (i) A discriminative score based on class labels, and (ii) A generative score based on a traditional metric for unsupervised clustering. We test the performance of LK-Means using standard real datasets and an application for object recognition. Moreover, we also compare its performance against classical K-Means and a popular K-Medoids-based supervised clustering method. Our experiments show that, in most cases, LK-Means outperforms the alternative techniques by a considerable margin. Furthermore, LK-Means presents execution times considerably lower than the alternative supervised clustering method under evaluation.

Keywords: Supervised Clustering, K-Means, K-Medoids.

1 Introduction

Techniques to divide a set of N data instances into K groups are known as clustering algorithms. Clustering algorithms are commonly used in an unsupervised learning framework where the goal is to minimize an error function with respect to a given distance metric, for example, intra-cluster distance. A robust model for clustering is to use a mixture of Gaussians [4] that has the ability to capture complex relationships among the data using a sound statistical approach. Despite its advantages, this technique tends to be slow, mainly due to the calculation of a covariance matrix. A faster and simpler clustering technique is the K-Means algorithm [21] that uses a hard assignment of data points to clusters and assumes a spherical covariance. While the simplicity of the K-Means algorithm is one of the main reasons for its popularity [38], its lower computational complexity with respect to alternative clustering techniques is also a desirable feature for intensive clustering tasks, e.g., the acquisition of codewords for visual recognition [17].

In contrast to traditional clustering, supervised clustering is applied to labeled data. Here, the goal is to find clusters with a high purity, where the purity of a cluster is defined as the percentage of data in a cluster that belongs to its most frequent class. Figure 1 shows an illustrative toy example corresponding to a 2D dataset with three spatial clusters and two classes. After applying both unsupervised and supervised clustering, it can be observed that unsupervised clustering ignores class labels (Figure 1.a), while supervised clustering generates clusters that focus on a particular class (Figure 1.b).

Eick et al. [11] enumerate several applications of supervised clustering, such as dataset compression, distance metric learning, or classification refinement, among others. As an example, supervised clustering can be used to identify customers profiles according to ordinal measures (e.g. age, salary, marital status) by identifying clusters that are homogeneous with respect to their buying behavior in terms of particular product categories (labels). Further uses of supervised clustering can be found in the areas of genetics and finance [32].

Current algorithms for supervised clustering usually have the form of a K-Medoids algorithm. Due to the use of medoids, this type of methods is more resistant to outliers than schemes based on a K-Means strategy, however, they have the drawback of being considerably slower. In particular, assuming a fixed number of iterations, the K-Medoids algorithm has a quadratic complexity in terms of the number of data instances, while in the case of K-Means this complexity is only linear [4].

In this work we present a new method for supervised clustering that is based on two main hypotheses: i) For a wide variety of applications a combination of supervised and unsupervised information can lead us to more informative clusters, and ii) A supervised clustering method based on a K-Means type of algorithm can allow us to overcome the speed limitations of current methods based on a K-Medoids clustering strategy. Following these hypotheses, the main contributions of this paper are: i) Presenting LK-Means, a new supervised clustering algorithm that extends the K-Means algorithm to incorporate instance labels, ii) Empirical evidence showing that LK-Means outperforms the K-Means algorithm and a K-Medoid supervised clustering method, as measured by several popular metrics commonly used to access clustering quality, and iii) Empirical evidence showing that, in terms of execution time, our method is more efficient than a supervised clustering technique based on a K-Medoid clustering strategy.

The rest of this paper is organized as follows. Section 2 describes two baseline methods, K-Means and K-Medoids algorithms, and relevant previous works. Section 3 presents LK-Means, the proposed supervised clustering approach. Section 4 presents and discusses experimental results using several benchmark datasets and an application to the case of object recognition. Finally, Section 5 presents our main conclusions and future avenues of research.

2 Background

2.1 K-Means and K-Medoids

K-Means algorithm is one of the most popular clustering techniques. This algorithm partitions N data instances into K clusters, where the number of clusters K has to be known a priori. Specifically, given a dataset X with N data instances $x_i \in \boldsymbol{R}^d$, $i \in [1 \ldots N]$, K-Means algorithm partitions$X$ into K cluster C_k, $k \in [1 \ldots K]$, by minimizing the following cost function:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} \delta_{nk} \left\| x_n - u_k \right\|^2 , \tag{1}$$

where the indicator function δ_{nk} is given by:

$$\delta_{nk} = \begin{cases} 1 & x_n \in C_k \\ 0 & \text{otherwise,} \end{cases}$$

$\| \cdot \|$ refers to L2-norm and u_k corresponds to the mean of cluster k.

Optimal parameters u_k are found by minimizing Equation (1) using a gradient descent approach. This results in an iterative procedure that alternates between assigning data instances to cluster centers, and re-estimating cluster centers given the new assignations. Convergence to a local minimum of Equation (1) is granted by the gradient descent type of exploration and the finite set of possible assignations of data instances to clusters. In particular, assuming a fixed number of iterations and dimensions, the computational complexity of K-Means is $O(NK)$. Algorithm 1 summarizes the main steps of the K-Means algorithm.

While K-Means for fixed numbers of iterations and dimensions has a linear computational complexity with respect to the number of data instances, the computation of the centroids is sensitive to outliers [4]. To alleviate this problem, the K-Medoids

Algorithm 1 : K-Means algorithm.

```
1. Randomly select $K$ data instances as initial means.
2. Associate each data instance with the cluster of its nearest mean and calculate the
   cost function using Equation (1).
3. Calculate the new means as the centroids of the $K$ new partitions.
4. Repeat steps 2 and 3 until there is no change in the cost evaluation (or the cost
   change is below a suitable threshold).
```

algorithm uses a more robust procedure to find the cluster centers, but this procedure has a quadratic complexity with respect to the number of data instances. In particular, K-Medoids minimizes a score that is similar to the one used by K-means, but it considers a more general distance metric $\nu(x, x')$ between data instances x and x', as shown in Equation (2). An example of metric $\nu(x, x')$ is the Euclidean distance, used in K-Means, or the Jaccard distance, commonly used in applications related to transactional databases [23].

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} \delta_{nk} \nu(x_n, u_k) \tag{2}$$

In contrast to K-means, K-Medoids minimizes Equation (2) with respect to parameters u_k by calculating a matrix that stores the distances between all pairs of data instances. Specifically, initially K-Medoids randomly chooses a set of K data instances as the initial set of K medoids and calculates the distance matrix between all the data instances. It then replaces each medoid with all non-medoid points and calculates all possible configurations costs according to Equation (2). Next, it chooses as the new medoids the ones corresponding to the configuration with the lowest cost. Finally, the method repeats the search over the non-medoids elements until the medoids do not change. The procedure is summarized in Algorithm 2.

Assuming a fixed number of iterations and dimensions, the computational complexity of K-Medoids is $O(K(N-K)^2)$. This implies that in general the K-Medoids algorithm is slower than K-Means.

Algorithm 2 : K-Medoid algorithm.

```
1. Randomly select $k$ data instances as initial medoids.
2. Associate each data instance to its most similar medoid and calculate the cost using
   Equation (2).
3. for each medoid $m$ do
4.    for each non-medoid $o$ do
5.       Swap $m$ and $o$ and compute the cost of the configuration.
6.    end for
7. end for
8. Select the set of elements corresponding to the configuration with the lowest cost.
9. Repeat 3 through 8 until there is no change in the set of medoids.
```

2.2 Related work

Semi-supervised clustering uses labeled and unlabeled data to find clusters that maximize a score related to cluster purity with respect to known class labels. Semi-supervised clustering methods can be divided into two groups: Similarity based methods and search-based methods [3]. Similarity-based methods use a modified distance function that considers the labels of classified examples and then uses a traditional clustering algorithm. On the other hand, search-based methods modify clustering algorithms themselves to accommodate for labeled instances, but do not change the distance function [5].

In terms of supervised clustering, all available records have labels. Tishby et al. propose an agglomerative clustering algorithm [35] using the notion of "information bottleneck" [34]. This technique minimizes the information loss of the clustering related to a class conditional distribution. Embrechts et al. [10] propose a genetic algorithm for a version of K-Means where the goal of the search process is to obtain clusters that minimize cluster dispersion and cluster impurity. Cohn et al. [7] modify the popular EM algorithm for incorporating similarity and dissimilarity constraints. They assume the presence of a human oracle that guides the clustering process. Basu et. al. [3] modify the K-means algorithm to cope with class knowledge. They use a careful initialization based on the neighborhood of the data instances.

Sinkkonen et al. [32] propose a method called discriminative clustering that minimizes distortion within clusters. In their work, distortion is related to the loss of mutual information among classes and clusters, which is caused by representing each cluster by a prototype. This technique seeks to produce clusters that are internally as homogeneous as possible with respect to a class conditional distribution. The resulting minimization is complex and they have to resort to approximations or simulated annealing methods to find suitable solutions.

Jordan et al. [39] (and similarly Shental et al. [2]) transform training examples into constraints based on the observation that instances of different classes should have a distance larger than a given threshold. Then, they derive a modified distance metric that minimizes the distance between data instances considering the constraints. Finally, they use a K-Means algorithm in conjunction with the modified distance metric to compute clusters.

Eick et. al. [11] formally introduce the term supervised clustering. Their work proposes supervised versions of some clustering algorithms, such as K-Medoids and divisive clustering. In particular, the SRIDHCR algorithm (Single Representative Insertion/Deletion Steepest Decent Hill Climbing with Randomized Restart) shows good performance in their experiments when compared to alternative techniques, thus, we choose this method as the baseline for comparison in our work.

Ye et. al. [40] present a discriminative version of K-Means. They simultaneously solve linear discriminant analysis (LDA) and K-Means optimization using matrix algebra. An advantage of this method is that it makes a feature transformation using LDA properties. For each iteration, their method needs to solve an optimization problem using linear search. Unfortunately, they do not show any measure of the speed of their method.

In relation to extensions of K-Means, Deelers and Auwatanamongkol [9] propose a scheme to initialize the K-Means algorithm using a recursive strategy that, considering the data axis with highest variance, progressively divides the data until they obtain a suitable number of clusters. Shanmugasundaram and Sukumaran [31] introduce a related scheme to initialize the K-Means algorithm, where they divide the data into two smaller cells considering the data axis with highest variance and keeping the two cells as far apart as possible. This procedure is repeated until one can obtain a prefixed number of clusters. Kumar et al. [18] enhance the K-Means algorithm by considering particular data structures (red-black tree and min-heap) that allow them to reduce computational time. These previous works are valuable in terms of improving the initialization and time processing capabilities of the traditional k-means algorithm, however, these works do not consider labeled data as in our technique. In this sense, these techniques can be considered as complementary to our work.

In a related research task, Lasserre et al. [19] propose the idea of a convex combination of unsupervised and supervised information in machine learning. They introduce a Bayesian framework to combine unlabeled and labeled data, where they find that under limited training data, the best performance is given by a combination of both views. Here, we also follow a similar idea but in the context of a supervised version of the K-Means algorithm leading to a different optimization problem and solution. As shown by our experiments, our proposed strategy provides several advantages with respect to alternative techniques for supervised clustering.

3 Labeled K-Means

Following Eick et. al. [11], several supervised clustering methods follow a K-Medoids approach that is very time consuming. Inspired by [19], we propose LK-Means, a K-Means like algorithm with a modified cost function that considers a convex combination of both, a class-dependent and non-class-dependent cost functions.

We assume a labeled dataset X with N training instances (x_i, y_i), where $x_i \in R^d$, $y_i \in [1, \dots, L]$, and $i \in [1 \dots N]$. We assume that the clustering problem requires

K clusters. LK-Means replaces the tradicional K-Means cost function in Equation (1) with the following function:

$$J(u_k^l, \delta_{nk}^l) = \sum_{n=1}^{N} \left[\alpha \sum_{k=1}^{K} \sum_{l=1}^{L} \delta_{nk}^l \left\| x_n - u_k^l \right\|^2 \rho_k^l + (1-\alpha) \sum_{k=1}^{K} \delta_{nk} \left\| x_n - u_k \right\|^2 \right] \quad (3)$$

where δ_{nk}^l refers to the supervised indicator function that assigns instance x_n to mean u_k^l, which in turn corresponds to the mean of data instances in cluster k with label l. ρ_k^l represents a prior factor for data instances with label l inside cluster k, δ_{nk} refers to the unsupervised indicator functions, and u_k corresponds to the mean of all data instances in cluster k. Equation (3) represents a convex combination, where parameter α in the range $[0, 1]$ manages the balance between the supervised and unsupervised clustering scores.

In particular, prior factor ρ_k^l for data instances with label l inside cluster k is defined as:

$$\rho_k^l = \frac{\sum_{n=1}^{N} \delta_{nk}^l}{\sum_{n=1}^{N} \delta_{nk}} \quad (4)$$

ρ_k^l represents the confidence of label l in cluster k, with values in the range $[0, 1]$. When this weight is near one, cluster k tends to contain only elements with label l. In the opposite case, when this weight is near zero, cluster k tends to contain no elements with label l.

The unsupervised indicator function δ_{nk} for data instance x_n and cluster C_K is defined as:

$$\delta_{nk} = \begin{cases} 1 & \text{if} \quad x_n \in C_k \\ 0 & \text{otherwise} \end{cases}$$

In terms of each unsupervised mean u_k, it is defined as the weighted mean over all supervised means u_k^l for the corresponding cluster k:

$$u_k = \sum_{l=1}^{L} \rho_k^l u_k^l \tag{5}$$

To find the optimal parameters: δ_{nk}^l and u_k^l, we minimize Equation (3) using a block coordinate descent approach that resembles the operation of the K-Means algorithm. Specifically, we alternate optimizations of Equation (3), first with respect to δ_{nk}^l and then with respect to u_k^l. Following the K-Means terminology, we call these steps **assignment** and **update**-steps, respectively. We refer now to each of these steps.

In terms of the **assignment-step**, cost function J in Equation (3) considers each data instance n in separate terms of the main sum, therefore, we can independently optimize J with respect to each indicator δ_{nk}^l. Furthermore, in the **assignment-step** we fix the value of the supervised means u_k^l and, as a consequence, we also fix the values of the unsupervised means u_k. As a result, the assignment that minimizes the cost function J is given by:

$$\delta_{nk}^l = \begin{cases} 1 & \text{if} \quad k = \text{argmin}_j \left[\alpha\delta_{nj}^l \left\|x_n - u_j^l\right\|^2 \rho_j^l + (1-\alpha)\delta_{nj} \left\|x_n - u_j\right\|^2\right] \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

In terms of initialization, initial values for the supervised indicator functions δ_{nk}^l are calculated using a Laplace smoothing. We use this procedure to avoid empty values for the supervised means vectors which can appear in the case of clusters without elements of the corresponding class. In this case, the supervised mean of a missing class is given by the unsupervised cluster because all elements have a value near to zero.

Specifically, we apply a Laplace smoothing according to:

$$\delta_{nk}^l = \frac{\lambda_{nk}^l + \gamma}{1 + LK\gamma}, \tag{7}$$

where λ^l_{nk} is defined as:

$$\lambda^l_{nk} = \begin{cases} 1 & \text{if} \quad x_n \in C_k \ \wedge \ y_n = l \\ 0 & \text{otherwise.} \end{cases}$$

We apply a Laplace smoothing [26] with a constant $\gamma = 0.001$. This small constant can be interpreted as the global uncertainty about the label of an element.

In terms of the **update-step**, we only need to find the optimal supervised means since each unsupervised mean u_k is a function of the corresponding supervised means u^l_k. Applying the corresponding partial derivatives to Equation (3), we have:

$$\frac{\partial}{\partial u^l_k} J = \sum_{n=1}^{N} -2\alpha\delta^l_{nk}\left(x_n - u^l_k\right)\rho^l_k + \sum_{n=1}^{N} -2(1-\alpha)\delta_{nk}\left(x_n - u_k\right)\rho^l_k \quad (8)$$

By rearranging components in Equation (8) and setting the derivative to zero, we obtain:

$$\sum_{n=1}^{N} \alpha\delta^l_{nk}x_n - \sum_{n=1}^{N} \alpha\delta^l_{nk}u^l_k + \sum_{n=1}^{N}(1-\alpha)\delta_{nk}x_n - \sum_{n=1}^{N}(1-\alpha)\delta_{nk}u_k = 0 \quad (9)$$

Assuming iteration t and that we are computing the optimization for the supervised mean of a given class label l', we use the previous supervised means $u^{l(t-1)}_k$ to approximate $u^{(t)}_k$ by updating only the maximized component $u^{l'}_k$ and fixing the rest. Then, $u^{(t)}_k = \sum_{l=1, l\neq l'}^{L} \rho^{l(t-1)}_k u^{l(t-1)}_k + \rho^{l'(t-1)}_k u^{l'(t)}_k = u^{(t-1)}_k - \rho^{l'(t-1)}_k u^{l'(t-1)}_k + \rho^{l'(t-1)}_k u^{l'(t)}_k$. Renaming the variables associated to previous iterations $u^{(t-1)}_k$, $u^{l(t-1)}_k$ and $\rho^{l(t-1)}_k$ as $\tilde{u}_k$, $\tilde{u}^l_k$ and $\tilde{\rho}^l_k$, respectively, and considering that the optimization is computed for l=l', we have:

$$\sum_{n=1}^{N} \alpha\delta^l_{nk}x_n - \sum_{n=1}^{N} \alpha\delta^l_{nk}u^l_k + \sum_{n=1}^{N}(1-\alpha)\delta_{nk}x_n - \sum_{n=1}^{N}(1-\alpha)\delta_{nk}(\tilde{u}_k - \tilde{\rho}^l_k\tilde{u}^l_k + \tilde{\rho}^l_k u^l_k) = 0 \quad (10)$$

Then, by rearranging the components, we have:

$$u_k^l \alpha \sum_{n=1}^{N} \delta_{nk}^l + u_k^l \tilde{\rho}_k^l (1-\alpha) \sum_{n=1}^{N} \delta_{nk} = \alpha \sum_{n=1}^{N} \delta_{nk}^l x_n + (1-\alpha) \sum_{n=1}^{N} \delta_{nk} \left(x_n - \tilde{u}_k + \tilde{\rho}_k^l \tilde{u}_k^l \right) \quad (11)$$

Finally we obtain:

$$u_k^l \quad = \quad \frac{\alpha \sum_{n=1}^{N} \delta_{nk}^l x_n + (1-\alpha) \sum_{n=1}^{N} \delta_{nk} \left(x_n - \tilde{u}_k + \tilde{\rho}_k^l \tilde{u}_k^l \right)}{\alpha \sum_{n=1}^{N} \delta_{nk}^l + (1-\alpha) \tilde{\rho}_k^l \sum_{n=1}^{N} \delta_{nk}} \quad (12)$$

Equation (12) has a straightforward interpretation. If we have $\alpha = 1$, then only supervised information is considered. On the other hand, if $\alpha = 0$ then the resulting clusters correspond to the unsupervised solution provided by the traditional K-Means algorithm. We summarize our procedure in Algorithm 3.

Algorithm 3 Labeled K-Means Algorithm

```
1. Initialize K initial means randomly.
2. Associate each data instance with its nearest mean and consider its class.
3. Compute supervised means $u_k^l$ using Equation (12).
4. Compute unsupervised means $u_k$ using Equation (5).
5. Compute indicatrices $\delta_{nk}^l$ considering Equation (6).
6. Compute the cost $J$ using Equation (3).
7. Repeat 3 to 6 until there is no change in the cost evaluation (or cost change is below
   a threshold).
```

In terms of convergence, the assignment-step given by Equation (6) can only decrease the value of the relevant cost function in Equation (3). Similarly, the update step provides new parameter values that also decrease this cost function. Furthermore, given that set of possible assignments of training instances to clusters is finite, the procedure in Algorithm 3 can not decrease forever. As a consequence, it is possible to guarantee that the proposed algorithm will converge to a local or global optimum of the relevant cost function.

In our model, we do not consider specific strategies to deal with noisy or missing data. However, standard preprocessing strategies do exist to deal with these problems, and they can be used to complement our technique [16].

4 Experiments and Results

4.1 Experiments in general datasets

In this Section, we test the performance of LK-Means using diverse datasets. In particular, we use 8 real data sets from the UCI Machine Learning Repository [1]: Iris, Heart, Glass, Diabetes, Silhouttes, Segment, Ionosphere, and Sonar. Table 1 shows the main details for these datasets. We normalize all these datasets to the range $[0, 1]$. Following the regular implementation of K-means, we use Euclidean distance as the similarity metric. All experiments are performed on a PC with 2.0 Ghz Pentium IV processor with 2GB of RAM memory.

We compare our algorithm against classical K-Means and SRIDHCR. SRIDHCR is a K-Medoids algorithm based on a discriminative metric with random re-initialization if it detects a local minimum. We choose SRIDHCR because it shows good performance in relation to other supervised clustering methods [11]. We compare these algorithms in terms of clustering quality and computational time. In particular, Meilă [24] shows that there is not a single best metric to compare the outputs of clustering algorithms. There are alternative metrics for evaluating clustering quality, such as F-measure [6], Jaccard index [28], or Fowlkes Mallows index [14], however, we follow the metrics suggested in [24]. Consequently, we assess clustering quality using 4 different metrics commonly used to validate clustering results [24]: Adjusted Mutual Information (AMI) [36], Adjusted Variation of Information (AVI) [36], Mirkin distance (MD) [25], and Adjusted Rand Index (ARI) [15]. AMI and AVI are variations of mutual information (MI), while MD and ARI are variations of Rand Index (RI). All these metrics do not make any assumption about the form of the clusters. Also, they are in the range $[0, 1]$, where higher values indicate a better clustering, except in the case of

MD where small values indicate better results.

In our experiments, we use cross-validation with ten folds (10-CV) to validate the results of each algorithm. In terms of selecting a suitable number of clusters K, it is possible to use previous strategies proposed in the context of the K-Means algorithm [22]. Also, it is possible to relate the selection of K to the number of known classes. Here, we do not focus in proposing new strategies to choose this value, and we run our experiments testing different numbers of clusters. For each dataset, we select values for K equally spaced according to 4 intervals beginning from the number of classes L to the upper bound $\left\lceil \sqrt{number\ of\ records/2} \right\rceil$. This upper bound is obtained from the rule of thumb" of clustering [22]. For example, in the case of Heart dataset, as it has 2 classes and 270 instances, we test $K \in \{2, 5, 8, 11, 14\}$.

For each of the 4 clustering metrics considered here, we test the performance of LK-Means using parameter α with values $\{0.8, 0.9, 1.0\}$, and for each of the tests considered here, we report the average performance for these 3 values. In the case of SRIDHCR, we choose the best parameter β (see [11] for details) according to 3-CV in a grid with 11 values: 0 to 2.0 with a step of 0.2. K-Means does not require more parameters than the number of clusters. It is important to note that to be fair with K-Means, we do not optimize the value of parameter α in LK-Means. This is because when α approaches zero LK-Means behaves exactly like K-Means, therefore, by optimizing α, LK-Means can always at least match the performance of K-Means. Consequently, in all tests, we just consider high values of α to stress the relevance of the supervised information. To check if our results are statistically significant, in each case we use a paired Student's t-test (Behrens-Fisher problem [29]) to compare the results of LK-Means against the performance of each of the alternative techniques.

Table 2 shows our results using AMI metric. Considering the average AMI results for all values of K under test, our method outperforms K-Means and SRIDHCR in most of the cases, with the exceptions of the Silhouttes and Ionosphere datasets where

K-Means shows better performance. Considering only cases with confidence $\geq$ 75%, a paired t-Student test shows that for Iris, Heart, Glass, Segment, and Sonar datasets, LK-Means has better AMI than the nearest competitor with 84%, 100%, 88%, 91%, and 75% of confidence, respectively. On the other hand, K-Means shows the best performance in Ionosphere and Silhouttes datasets with 95% and 84% confidence, respectively.

Table 3 shows our results using AVI metric. By considering all datasets, we can observe that, on average, again LK-Means outperforms the other algorithms in most of the cases. Similarly to the results with AMI metric, the worst relative results for LK-Means is given for the case of the Ionosphere dataset. Considering only cases with confidence $\geq$ 75%, a paired t-Student test finds that LK-Means in Iris, Heat, Glass, Segment, and Sonar datasets has greater AVI than the nearest competitor with 87%, 100%, 94%, 75% and 80% of confidence, respectively. On the other hand, K-Means shows the best performance in Ionosphere and Silhouttes datasets with 96% and 95% confidence, respectively.

Table 4 shows results using MD metric. In terms of average results, in half of the 8 datasets LK-Means is the winner, while K-Means shows best performance in the rest of the datasets. In general, we notice that under MD metric there is not a clear winner between LK-Means and K-Means, and results depend on the type of dataset. Considering only cases with confidence $\geq$ 75%, a paired t-Student test shows that LK-Means in Heart, Diabetes, and Sonar datasets has lower MD than the nearest competitor with 100%, 92% and 87% of confidence, respectively. On the other hand, K-Means shows the best performance in Glass, Silhouttes, and Ionosphere datasets with 97%, 98%, and 95% of confidence, respectively.

Table 5 summarizes results using ARI metric. In average LK-Means is the winner in 5 of the 8 datasets. while K-Means shows best performance in 2 datasets, and SRIDCHR in one. Considering only cases with confidence $\geq$ 75%, a paired t-Student test finds that LK-Means in Iris, Heart, Glass, Segment, and Sonar datasets has better ARI than the nearest competitor with 75%, 100%, 95%, 98% and 93% of confidence, respectively. On the other hand, K-Means shows the best performance in Ionosphere dataset with 99% confidence.

Considering the different metrics and datasets used to evaluate clustering quality, the previous results indicate that in general LK-Means outperforms the alternative techniques under consideration. However, the superior performance of LK-Means depends on the type of dataset and the validation metric under consideration. In terms of the different datasets, in general LK-Means shows superior performance in most of them with the exception of Silhouttes and Ionosphere, where the unsupervised clustering strategy of K-Means leads to better clusters. We believe that, in general, the performance of LK-Means is closely related to the pertinence of our hypothesis that homogeneity in class information leads to more informative clusters. Clearly, the validity of this hypothesis depends of the application under consideration, particularly, the semantic of the data labels under consideration. In terms of the 4 metrics used to evaluate clustering quality, LK-Means outperforms clearly the alternative algorithms in the case of AMI and AVI metrics, and to a lesser degree in the case of ARI metric. In the case of MD metric, for the values of α under consideration, LK-Means is unable to improve the results of K-Means. Following the observations in [37], MD metric and, to a lesser extend, ARI metric are affected by cluster size, therefore, they have a bias that affect their performance. As recommended in [37], AMI and AVI produce more stable and suitable results. Coincidentally, in our case AMI and AVI produce similar results and they provide stronger support to the superiority of LK-Means with respect to the alternative techniques.

Additionally, we test the sensibility of performance respect to α by measuring adjusted mutual information (AMI). We consider values of α in the interval: 0.1 to 1.0 with a step of 0.1. In order to facilitate the analysis of results, we consider two representative datasets. Specifically, we choose Diabetes and Glass datasets because in our experiments they represent cases where, under the AMI metric, LK-Means and K-Means alternate the best performance for different values of K. In both cases, we use a fixed number of clusters. We choose the number of clusters using the classical silhouette method [30], where the cardinality of the set of clusters is selected to maximize the average silhouette of the clusters.

Figure 3(a) shows the relationship between α and AMI for Diabetes dataset. The best result for LK-Means is obtained when α=0.9 with a corresponding value of $AMI = 0.092$. The worst result is for alpha= 0.1 with a corresponding value of $AMI = 0.051$. For this dataset, K-Means obtains a value of $AMI = 0.050$, therefore, there is a big advantage in favor of LK-Means. On the other hand, Figure 3(b) shows the relationship between α and AMI for dataset Glass. In this case, the best results are obtained with low values of α (0.1 and 0.2). In particular, the best result for LK-Means is obtained when α=0.1 with a corresponding values of $AMI = 0.171$. For this dataset, K-Means obtains a value of $AMI = 0.168$. Consequently, both algorithms show a similar behavior. This is expected because, according to Equation (3), for values of α near zero LK-Means behaves like K-Means.

The processing time for the different algorithms is summarized in Table 6. As expected, K-Means is faster than the other methods, however, it is relevant to see that LK-Means is visibly faster than SRIDHCR. For example, for 12 clusters in the Silhouttes dataset, K-Means, LK-Means and SRIDHCR use approximately 0.1, 5, and 715 seconds, respectively. The reason for the slowness of SRIDHCR is that K-Medoids requires a distance matrix to be calculated between all the records, while K-Means and LK-Means only require the distance measures from the means to all records. Considering the good results of the AMI score, we can see that LK-Means is capable of combining the speed of K-Means and the semantic gain to incorporate data labels during the clustering process.

4.2 Experiments in object recognition

In this Section we apply LK-Means to the task of codebook generation for a visual recognition task. Currently, the Bag-of-Visual-Words (BoVW) scheme is one of the most popular approaches for visual object recognition [33]. Under this approach, the generation of a suitable codebook plays a key role. In general, most BoVW approaches build the codebook using a clustering algorithm, mainly K-Means. Interestingly, although class labels are usually available, these are not considered during the codebook generation. This suggests a suitable scenario to test the advantages that a supervised clustering technique, such as LK-Means, can offer to provide more discriminative codebooks.

Following the previous intuition, we compare the performance of LK-Means against K-Means for the task of codebook generation in object recognition applications. As a testbed, we select 4 object recognition datasets that are commonly used to benchmark object recognition techniques. These datasets are: UIUC, DARMSTADT, VEH-CALT, and OUTDOOR. UIUC contains 2 object classes: cars and backgorund. DARMSTADT contains 3 object classes: motorbike, cow, and cars [20]. VEH-CALTECH is a subset of CALTECH-101 (VEH-CALT) dataset [12], including 4 object classes: airplane, car, helicopter, and motorbike. Finally, OUTDOOR contains images of 8 types of outdoor scenes [27]. Table 7 shows relevant details for all these datasets. Following a standard implementation of K-means, we use Euclidean distance as the main similarity metric for all our test.

Following standard procedures for BoVW schemes [33], we use Histogram of Gradients (HoG) as a basic visual feature [8]. In particular, we obtain the HoG descriptors using patches of 32x32 pixels. These patches are selected using a sliding window process over a fix grid on each input image. In particular, we use the variant UOCTTI of HoG proposed by Felzenswalb et al. [13]. UOCTTI considers a compressed representation of HoG given by 31 dimensions. For each dataset, we use the HoG descriptors of a set of training patches to build codebooks using K-Means and LK-Means. In the case of LK-Means, we assign to each patch the label of the object class that generates the patch. We evaluate the discriminative properties of the resulting codebooks using them to train a category-object classifier. As a classifier, we use the popular linear Support Vector Machine (SVM), as in [8]. In relation to the training process, we use 15 random images for training and 15 images for testing. In order to evaluate the sensibility of our results in terms of the number of clusters, we consider the following number of codewords: $K = \{50, 100, 150, 200, 250\}$.

Table 8 shows the average accuracy achieved by the resulting classifier. These results are obtained using a 20-hold-out scheme and a fixed value of $\alpha = 0.8$. We select this value of α extrapolating the results of the previous Section, and as a good compromise between the supervised and unsupervised terms in Equation (3). In Table 8, we can observe that LK-Means outperforms K-Means in almost all cases; and in the few cases where K-Means shows superior performance the difference in accuracies is less than 1.0%. Furthermore, we observe that the positive difference in favor of LK-Means increases with the number of clusters, indicating that LK-Means benefits more that K-Means from a greater flexibility in the search for relevant patterns.

Finally, Figure 2 shows some visual codewords resulting from the VEH-CALT dataset. We present the top-six most discriminative words according to the Fisher discriminant score [4]; and considering $K = 200$ for both algorithm. In Figure 2, each visual codeword is represented by its four nearest patches. In Figure 2, it is possible to observe in each row that, in general, LK-Means provides more discriminative codewords than K-Means.

5 Conclusions

In this paper we introduce LK-Means, an extension of the classical K-Means algorithm to the case of supervised clustering. As a main search strategy, LK-Means optimizes a convex combination of class dependent and non-class dependent cost functions. Experiments using a set of standard benchmark datasets and 4 different metrics to assess clustering quality, show that, on average, LK-Means outperforms classical K-Means and SRIDCHR algorithms. In the cases of AMI and AVI metrics, in most of our tests LK-Means outperforms the alternative algorithms by a large margin. In the case of ARI metric, on average, LK-Means also outperforms K-Means and SRIDCHR but by a narrower margin. In the case of MD metric, LK-Means and K-Means present mixed results. As it has been noticed in previous works, MD is negatively affected by clustering size, and it is in general less robust than metrics such as AMI and AVI. Additionally, we show an application of LK-Means as a codebook generator for object recognition applications. We consider several common benchmark datasets, and in all cases LK-Means outperforms K-Means, demonstrating the relevance of considering class information to find meaningful clusters.

Interestingly, our results indicate that the advantages of LK-Means over K-Means depends on the type of dataset. This is closely related to our hypothesis that homogeneity in class information leads to more informative clusters, which depend on the semantic of the data labels. For example, in the case of the object recognition application, where one expects a high correlation between clusters of visual features and object categories, the advantages of using LK-Means instead of regular K-Means are more clear. This observation offers a "rule of thumb" to set the value of the parameter α. For a dataset where it is expected a high correlation between class information and cluster composition α should be close to 1, increasing the relevance of class information.

In relation to time, our experiments show that LK-Means presents an attractive computational performance, being considerably faster than the alternative supervised clustering method considered in this work. In relation to future work, we plan to increase the reliability of the model by modifying the cost function of LK-Means to accommodate cluster shape. We also plan to extend this work to manage fuzzy labels inside of our model. Finally, we also plan to extend the idea behind LK-Means to the case of subspace clustering, which can provide a suitable extended search space to find relevant class-dependent clusters.

References

[1] A. Asuncion and D. Newman. UCI machine learning repository, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 2007.

[2] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proceedings of International Conference of Machine Learning*, pages 11–18, 2003.

[3] S. Basu, M. Bilenko, and R. Mooney. Comparing and unifying search-based and similarity-based approaches to semi-supervised clustering. In *Proceedings of Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 29–42, 2003.

[4] C. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007.

[5] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of Computational Learning Theory*, pages 92–100, 1998.

[6] P. Raghavan C. Manning and H Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[7] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. Technical Report TR2003-1892, Cornell University, 2003.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[9] S. Deelers and A. Auwatanamongkol. Enhancing k-means algorithm with initial cluster centers derived from data partitioning along the data axis with the highest variance. In *International Journal of Electrical and Computer Engineering*, pages 247–252, 2007.

[10] A. Demiriz, K. Benett, and M. Embrechts. Semi-supervised clustering using genetic algorithms. In *Procedings of Artificial Neural Networks in Engineering*, pages 809–814, 1999.

[11] C. Eick, N. Zeidat, and Z. Zhao. In *International Conference on Tools with Artificial Intelligence*, pages 774–776.

[12] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Workshop of Generative Model Based Vision*, 2004.

[13] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis Machine Intelligence*, pages 1627–1645, 2010.

[14] E. Fowlkes and C. Mallows. A Method for Comparing Two Hierarchical Clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.

[15] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[16] H. Jiawei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., 2005.

[17] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *Proceedings of International Conference on Computer Vision*, pages 604–610, 2005.

[18] R. Kumar, R. Puran, and J. Dhar. Enhanced k-means clustering algorithm using red black tree and min-heap. In *International Journal of Innovation, Management and Technology*, pages 49–54, 2011.

[19] J. Lasserre, C. Bishop, and T. Minka. Principled hybrids of generative and discriminative models. pages 87–94, 2006.

[20] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision*, 2004.

[21] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[22] K. Mardia, J. Kent, and J. Bibby. *Multivariate Analysis*. Academic Press, London, first edition, 1979.

[23] Z. Markov and D. Larose. *Data mining the Web : uncovering patterns in Web content, structure, and usage*. John Wiley and Sons, 2007.

[24] M. Meilă. Comparing clusterings: an axiomatic view. In *Proceedings of International Conference on Machine Learning*, pages 577–584, 2005.

[25] B. Mirkin. *Mathematical Classification and Clustering*. Kluwer Academic Press, 1996.

[26] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. In *Journal of Machine Learning*, pages 103–134, 1999.

[27] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal on Computer Vision*, pages 145–175, 2001.

[28] A. Rajaraman and J. Ullman. *Mining of massive datasets*. Cambridge University Press, 2012.

[29] J. Rice. *Mathematical Statistics and Data Analysis, 2nd ed.* Duxbury Press, 1994.

[30] P. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(1):53–65, 1987.

[31] R. Shanmugasundaram and S. Sukumaran. Enhancing k-means algorithm with semi-unsupervised centroid selection method. In *International Journal of Computer Science and Information Security*, pages 337–343, 2010.

[32] J. Sinnkkonen, S. Kaski, and J. Nikkila. Discriminative clustering: Optimal contingency tables by learning metrics. In *Proceedings of European Conference on Machine Learning*, pages 418–430. Springer, 2002.

[33] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision (ICCV)*, 2003.

[34] N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of Allerton Conference on Communications and Computation*, pages 368–377, 1999.

[35] N. Tishby. and N. Slonim. Document clustering using word clusters via information bottleneck method. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 208–215, 2000.

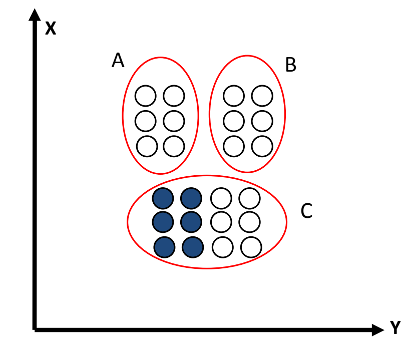
[36] N. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of International Conference on Machine Learning*, pages 1073–1080, New York, USA, 2009.

[37] N.X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(3):2837–2854, 2010.

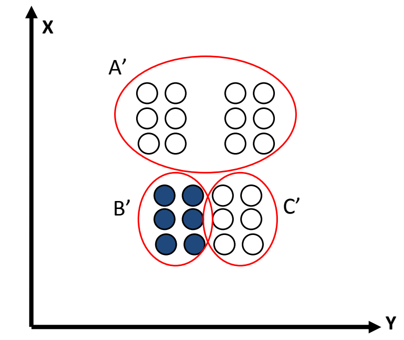
[38] X. Wu, V. Kumar, J. Ross, J. Ghosh, Q. Yang, H. Motoda, J. McLachlan, A. Ng, B. Liuand P. Yu, Z. Zhou, M. Steinbach, D. Hand, and D. Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14:1–37, 2007.

[39] E. Xing, A. Ng., M. Jordan, and S. Russell. Distance metric learning with applications to clustering with side information. *Advances in Neural Information Processing Systems*, pages 505–512, 2003.

[40] J. Ye, Z.Zhao, and M. Wu. Discriminative k-means for clustering. In *Advances in Neural Information Processing Systems*, pages 1649–1656, 2008.

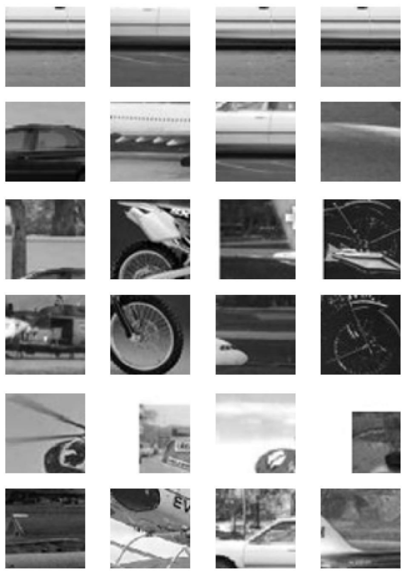


(a) Traditional clustering

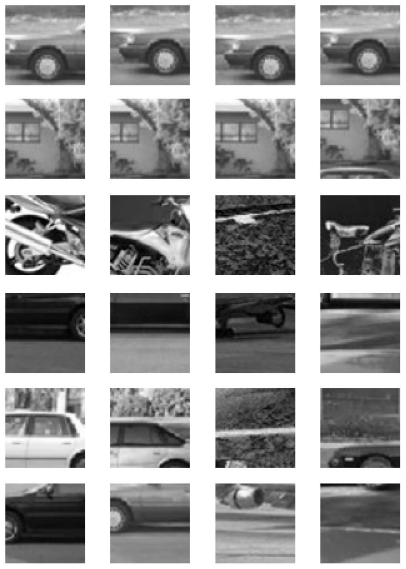


(b) Supervised clustering

Figure 1: Toy example comparing traditional and supervised clustering. There are two class labels denoted by clear and dark circles. We see that while classical clustering finds spatial clusters, supervised clustering finds clusters that are uniform with respect to class labels.



(a) Top-six codebooks obtained with K-Means.

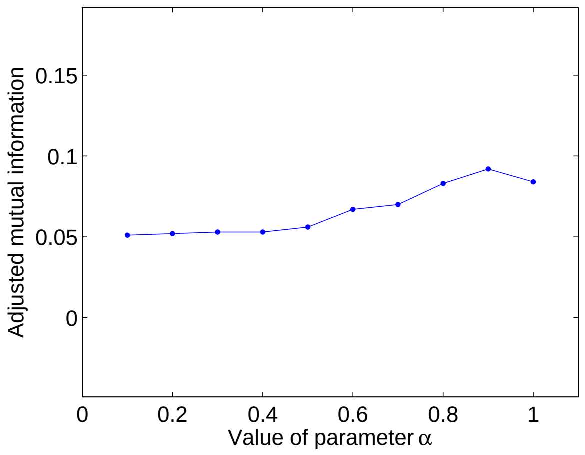


(b) Top-six codebooks obtained with LK-Means.

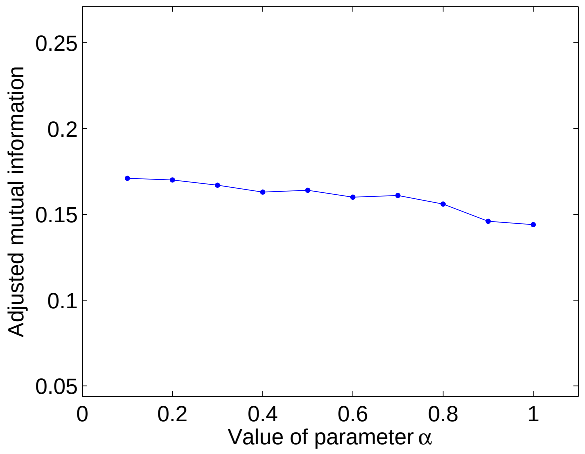
Figure 2: Top-six most discriminative codewords according to Fisher score for codebooks obtained with K-Means and LK-Means, respectively. Each row shows the 4 nearest pathces to each codeword. It is possible to observe that, in general, LK-Means provides more discriminative codewords than K-Means.

Table 1: Datasets details.

Dataset name	**# objects**	**# dimensions**	**# classes**
Iris	150	4	3
Heart	270	13	2
Glass	214	9	6
Diabetes	768	8	2
Silhouttes	846	18	4
Segment	2310	19	7
Ionosphere	351	33	2
Sonar	208	60	2



(a) **Diabetes dataset**(with 2 clusters)



(b) **Glass dataset** (with 11 clusters)

Figure 3: Comparison of sensibility of adjusted mutual information (AMI) respect to parameter α. Figure 3(a) shows that the best result for Diabetes dataset is obtained with α equal to 0.9, AMI=0.092, which is almost the double than the result with K-Means, AMI=0.050. On contrast, Figure 3(b) shows that in case of Glass dataset, the best result is obtained with α equal to 0.1, AMI=0.171, which is slightly greater than performance with K-Means, AMI=0.168. These results show that the discriminativity of clustering is dependant of data and parameter α.

Table 2: Adjusted Mutual Information results for real datasets using 10-CV. In average, LK-Means usually outperforms K-Means and SRIDHCR with variable confidence.

Method	**Number of clusters**					
Iris	***k=3***	***k=5***	***k=7***	***k=9***	***k=11***	***Mean***
K-Means	**0.765**	0.515	0.359	0.385	0.242	0.453
SRIDHCR	0.196	0.260	0.204	0.236	0.241	0.227
LK-Means	0.655	**0.538**	**0.497**	**0.451**	**0.387**	**0.505**
Heart	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	0.273	0.145	0.087	0.079	0.045	0.126
SRIDHCR	0.011	0.082	0.078	0.104	0.097	0.074
LK-Means	**0.293**	**0.212**	**0.137**	**0.134**	**0.104**	**0.176**
Glass	***k=6***	***k=7***	***k=8***	***k=9***	***k=10***	***Mean***
K-Means	**0.168**	**0.166**	0.145	**0.136**	0.126	0.148
SRIDHCR	0.093	0.132	0.138	0.092	0.106	0.112
LK-Means	0.148	0.159	**0.156**	0.132	**0.149**	**0.156**
Diabetes	***k=2***	***k=7***	***k=12***	***k=17***	***k=22***	***Mean***
K-Means	0.050	0.059	**0.050**	**0.046**	**0.045**	0.049
SRIDHCR	**0.113**	0.049	0.044	0.043	0.041	0.058
LK-Means	0.086	**0.068**	0.047	0.040	0.043	**0.060**
Silhouttes	***k=4***	***k=8***	***k=12***	***k=16***	***k=20***	***Mean***
K-Means	**0.120**	0.125	**0.137**	0.124	0.114	**0.124**
SRIDHCR	0.076	0.107	0.132	**0.141**	0.116	0.114
LK-Means	0.112	**0.129**	0.128	0.118	**0.117**	0.120
Segment	***k=7***	***k=14***	***k=21***	***k=28***	***k=35***	***Mean***
K-Means	**0.578**	0.505	0.465	0.411	0.374	0.467
SRIDHCR	0.446	0.522	0.469	0.428	0.392	0.451
LK-Means	0.548	**0.551**	**0.492**	**0.439**	**0.411**	**0.488**
Ionosphere	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	0.169	**0.181**	**0.140**	**0.179**	**0.124**	**0.159**
SRIDHCR	0.053	0.112	0.069	0.082	0.075	0.078
LK-Means	**0.174**	0.177	0.125	0.156	0.108	0.148
Sonar	***k=2***	***k=4***	***k=6***	***k=8***	***k=10***	***Mean***
K-Means	0.001	0.022	**0.051**	0.032	0.019	0.025
SRIDHCR	0.012	0.001	0.050	0.019	0.020	0.020
LK-Means	**0.094**	**0.036**	0.017	**0.039**	**0.058**	**0.049**

Table 3: Adjusted Variation of Information results for real datasets using 10-CV. In average, LK-Means usually outperforms competitors with variable confidence.

Method	**Number of clusters**					
Iris	***k=3***	***k=5***	***k=7***	***k=9***	***k=11***	***Mean***
K-Means	**0.781**	0.612	0.454	0.511	0.352	0.542
SRIDHCR	0.224	0.286	0.221	0.261	0.280	0.254
LK-Means	0.715	**0.613**	**0.591**	**0.560**	**0.485**	**0.592**
Heart	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	0.275	0.207	0.131	0.126	0.075	0.163
SRIDHCR	0.011	0.112	0.115	0.163	0.159	0.112
LK-Means	**0.300**	**0.271**	**0.188**	**0.201**	**0.164**	**0.225**
Glass	***k=6***	***k=7***	***k=8***	***k=9***	***k=10***	***Mean***
K-Means	0.193	**0.188**	0.168	**0.155**	0.150	0.170
SRIDHCR	0.100	0.149	0.150	0.109	0.128	0.127
LK-Means	**0.216**	0.183	**0.187**	0.154	**0.179**	**0.184**
Diabetes	***k=2***	***k=7***	***k=12***	***k=17***	***k=22***	***Mean***
K-Means	0.045	0.087	**0.080**	**0.077**	**0.078**	0.073
SRIDHCR	**0.117**	0.071	0.068	0.068	0.068	0.078
LK-Means	0.105	**0.092**	0.069	0.065	0.069	**0.080**
Silhouttes	***k=4***	***k=8***	***k=12***	***k=16***	***k=20***	***Mean***
K-Means	**0.122**	**0.151**	**0.182**	**0.174**	**0.166**	**0.159**
SRIDHCR	0.076	0.107	0.132	0.141	0.116	0.114
LK-Means	0.121	0.149	0.164	0.157	0.163	0.151
Segment	***k=7***	***k=14***	***k=21***	***k=28***	***k=35***	***Mean***
K-Means	**0.601**	0.575	0.578	0.538	0.511	0.561
SRIDHCR	0.561	0.583	0.568	0.544	0.513	0.554
LK-Means	0.570	**0.615**	**0.580**	**0.549**	**0.531**	**0.569**
Ionosphere	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	0.173	**0.238**	**0.201**	**0.258**	**0.195**	**0.213**
SRIDHCR	0.059	0.132	0.084	0.106	0.106	0.097
LK-Means	**0.182**	0.225	0.173	0.212	0.160	0.190
Sonar	***k=2***	***k=4***	***k=6***	***k=8***	***k=10***	***Mean***
K-Means	0.001	0.029	**0.076**	0.050	0.031	0.037
SRIDHCR	0.012	0.001	0.065	0.028	0.033	0.028
LK-Means	**0.099**	**0.048**	0.026	**0.056**	**0.075**	**0.061**

Table 4: Mirkin distance (MD) results for real datasets using 10-CV. In half of cases, LK-Means usually outperforms K-Means and SRIDHCR with variable confidence.

Method	**Number of clusters**					
Iris	***k=3***	***k=5***	***k=7***	***k=9***	***k=11***	***Mean***
K-Means	**0.098**	**0.126**	0.171	0.148	0.179	**0.144**
SRIDHCR	0.393	0.349	0.352	0.333	0.328	0.351
LK-Means	0.158	0.158	**0.148**	**0.147**	**0.158**	0.154
Heart	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	0.354	0.406	0.433	0.428	0.455	0.415
SRIDHCR	0.491	0.450	0.451	0.445	0.450	0.457
LK-Means	**0.347**	**0.361**	**0.402**	**0.403**	**0.419**	**0.386**
Glass	***k=6***	***k=7***	***k=8***	***k=9***	***k=10***	***Mean***
K-Means	**0.287**	0.303	**0.266**	**0.269**	0.268	**0.279**
SRIDHCR	0.325	**0.302**	0.309	0.298	0.286	0.304
LK-Means	0.324	0.314	0.317	0.291	**0.267**	0.302
Diabetes	***k=2***	***k=7***	***k=12***	***k=17***	***k=22***	***Mean***
K-Means	0.448	0.473	0.494	0.505	0.513	0.487
SRIDHCR	**0.406**	0.493	0.493	0.506	0.512	0.482
LK-Means	0.419	**0.471**	**0.492**	**0.504**	**0.503**	**0.478**
Silhouttes	***k=4***	***k=8***	***k=12***	***k=16***	***k=20***	***Mean***
K-Means	**0.341**	**0.274**	**0.250**	**0.242**	**0.237**	**0.269**
SRIDHCR	0.391	0.319	0.281	0.264	0.262	0.303
LK-Means	0.381	0.287	0.261	0.258	0.246	0.287
Segment	***k=7***	***k=14***	***k=21***	***k=28***	***k=35***	***Mean***
K-Means	**0.142**	0.115	**0.107**	0.110	0.114	0.118
SRIDHCR	0.149	0.111	0.112	0.120	0.124	0.123
LK-Means	0.154	**0.104**	**0.107**	**0.109**	**0.111**	**0.117**
Ionosphere	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	0.401	**0.393**	**0.432**	**0.408**	**0.438**	**0.414**
SRIDHCR	0.429	0.420	0.450	0.445	0.474	0.444
LK-Means	**0.398**	0.403	0.440	0.411	0.444	0.419
Sonar	***k=2***	***k=4***	***k=6***	***k=8***	***k=10***	***Mean***
K-Means	0.523	0.469	**0.447**	0.449	0.457	0.469
SRIDHCR	0.479	0.506	0.474	0.503	0.480	0.488
LK-Means	**0.455**	**0.460**	0.458	**0.444**	**0.440**	**0.451**

Table 5: Adjusted Rand Index (ARI) results for real datasets using 10-CV. In average, LK-Means usually outperforms K-Means and SRIDHCR with variable confidence.

Method	**Number of clusters**					
Iris	***k=3***	***k=5***	***k=7***	***k=9***	***k=11***	***Mean***
K-Means	**0.754**	**0.592**	0.424	0.477	0.322	0.513
SRIDHCR	0.190	0.259	0.196	0.221	0.233	0.220
LK-Means	0.644	0.568	**0.552**	**0.527**	**0.457**	**0.550**
Heart	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	0.293	0.155	0.093	0.097	0.038	0.135
SRIDHCR	0.019	0.110	0.099	0.111	0.099	0.088
LK-Means	**0.315**	**0.257**	**0.164**	**0.155**	**0.119**	**0.202**
Glass	***k=6***	***k=7***	***k=8***	***k=9***	***k=10***	***Mean***
K-Means	0.151	0.124	0.131	**0.124**	0.106	0.127
SRIDHCR	0.074	0.092	0.104	0.079	0.091	0.088
LK-Means	**0.168**	**0.134**	**0.143**	0.119	**0.137**	**0.140**
Diabetes	***k=2***	***k=7***	***k=12***	***k=17***	***k=22***	***Mean***
K-Means	0.094	**0.093**	0.062	0.045	0.033	0.654
SRIDHCR	**0.182**	0.059	**0.068**	**0.048**	0.041	**0.796**
LK-Means	0.150	0.089	0.060	0.043	**0.045**	0.774
Silhouttes	***k=4***	***k=8***	***k=12***	***k=16***	***k=20***	***Mean***
K-Means	**0.084**	0.098	**0.109**	**0.103**	**0.101**	**0.099**
SRIDHCR	0.051	0.082	**0.109**	0.110	0.088	0.088
LK-Means	0.082	**0.103**	0.108	0.098	0.100	0.098
Segment	***k=7***	***k=14***	***k=21***	***k=28***	***k=35***	***Mean***
K-Means	**0.460**	0.426	0.396	0.346	0.294	0.384
SRIDHCR	0.446	0.483	0.410	0.326	0.278	0.389
LK-Means	0.447	**0.502**	**0.444**	**0.388**	**0.357**	**0.428**
Ionosphere	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***	***Mean***
K-Means	**0.198**	**0.219**	**0.147**	**0.197**	**0.139**	**0.180**
SRIDHCR	0.115	0.163	0.112	0.120	0.080	0.118
LK-Means	0.196	0.199	0.130	0.189	0.124	0.168
Sonar	***k=2***	***k=4***	***k=6***	***k=8***	***k=10***	***Mean***
K-Means	0.001	0.016	**0.050**	0.032	0.016	0.023
SRIDHCR	0.042	0.001	0.048	0.018	0.025	0.027
LK-Means	**0.103**	**0.044**	0.034	**0.052**	**0.059**	**0.058**

Table 6: Speed in seconds for real datasets using 10-CV. LK-Means is considerably faster than SRIDHCR. Even though K-Means is faster than LK-Means, the difference is not very high and is compensated by an increase in clusters quality.

Method	**Number of clusters**				
Iris	***k=3***	***k=5***	***k=7***	***k=9***	***k=11***
K-Means	0.004	0.009	0.009	0.010	0.011
SRIDHCR	35.502	54.315	72.967	90.713	108.689
LK-Means	0.047	0.095	0.074	0.068	0.087
Heart	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***
K-Means	0.008	0.012	0.016	0.017	0.020
SRIDHCR	45.232	95.696	146.066	195.522	247.243
LK-Means	0.044	0.096	0.141	0.190	0.169
Glass	***k=6***	***k=7***	***k=8***	***k=9***	***k=10***
K-Means	0.010	0.018	0.014	0.015	0.030
SRIDHCR	88.823	102.429	115.432	128.478	154.635
LK-Means	0.400	0.356	0.516	0.565	0.548
Diabetes	***k=2***	***k=7***	***k=12***	***k=17***	***k=22***
K-Means	0.023	0.101	0.125	0.110	0.113
SRIDHCR	124.984	356.097	585.961	815.086	1063.223
LK-Means	0.230	1.556	2.686	3.997	4.339
Silhouttes	***k=4***	***k=8***	***k=12***	***k=16***	***k=20***
K-Means	0.123	0.098	0.093	0.106	0.137
SRIDHCR	251.755	464.094	715.121	887.593	1311.306
LK-Means	1.492	3.398	4.775	5.937	9.384
Segment	***k=7***	***k=14***	***k=21***	***k=28***	***k=35***
K-Means	0.245	0.307	0.394	0.494	0.622
SRIDHCR	1105.172	2108.405	3173.733	4086.154	4972.139
LK-Means	13.272	11.170	37.325	40.543	7.154
Ionosphere	***k=2***	***k=5***	***k=8***	***k=11***	***k=14***
K-Means	0.009	0.030	0.027	0.029	0.031
SRIDHCR	61.963	139.254	212.244	263.724	329.487
LK-Means	0.118	0.337	0.441	0.605	0.923
Sonar	***k=2***	***k=4***	***k=6***	***k=8***	***k=10***
K-Means	0.008	0.013	0.014	0.016	0.017
SRIDHCR	39.390	68.486	97.831	127.679	165.534
LK-Means	0.058	0.106	0.118	0.279	0.332

Table 7: Details of real object datasets.

Dataset name	**# objects**	**# classes**
UIUC	1050	2
DARMSTADT	327	3
VEH-CALT	1801	4
OUTDOOR	2600	8

Table 8: Recognition accuracy using a 20-hold-out evaluation scheme. In average, LK-Means overcomes K-Means in almost all datasets and all configurations. The advantage of LK-Means is more evident for cases with a large number of codewords (K >150).

Dataset	**Method**	**Number of clusters**				
		50	***100***	***150***	***200***	***250***
UIUC	K-Means	**79.33**	78.17	81.17	77.33	77.50
	LK-Means	78.50	**80.83**	**82.17**	**81.33**	**83.00**
DARMSTADT	K-Means	86.33	86.78	86.44	86.89	89.33
	LK-Means	86.33	**89.00**	**87.89**	**87.67**	**91.67**
VEH-CALT	K-Means	71.83	79.08	78.16	79.33	81.03
	LK-Means	**74.75**	**80.83**	**79.75**	**81.00**	**82.75**
OUTDOOR	K-Means	50.79	**57.00**	55.50	57.88	57.46
	LK-Means	**52.13**	55.71	**57.58**	**58.50**	**60.38**