Pattern Recognition 47 (2014) 2505–2516

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

The MinMax *k*-Means clustering algorithm

Grigorios Tzortzis *, Aristidis Likas

Department of Computer Science & Engineering, University of Ioannina, Ioannina 45110, Greece

ARTICLE INFO

Article history:
Received 18 February 2013
Received in revised form
20 January 2014
Accepted 28 January 2014
Available online 6 February 2014

Keywords:
Clustering
k-Means
k-Means initialization
Balanced clusters

ABSTRACT

Applying *k*-Means to minimize the sum of the intra-cluster variances is the most popular clustering approach. However, after a bad initialization, poor local optima can be easily obtained. To tackle the initialization problem of *k*-Means, we propose the MinMax *k*-Means algorithm, a method that assigns weights to the clusters relative to their variance and optimizes a weighted version of the *k*-Means objective. Weights are learned together with the cluster assignments, through an iterative procedure. The proposed weighting scheme limits the emergence of large variance clusters and allows high quality solutions to be systematically uncovered, irrespective of the initialization. Experiments verify the effectiveness of our approach and its robustness over bad initializations, as it compares favorably to both *k*-Means and other methods from the literature that consider the *k*-Means initialization problem.

1. Introduction

Clustering is a fundamental problem in data analysis that arises in a variety of fields, such as pattern recognition, machine learning, bioinformatics and image processing [1,2]. It aims at partitioning a set of instances into homogeneous groups, i.e. the intra-cluster similarities are high while the inter-cluster similarities are low, according to some clustering objective. However, exhaustively searching for the optimal assignment of instances to clusters is computationally infeasible and usually a good local optimum of the clustering objective is sought.

One of the most well-studied clustering algorithms is *k*-Means [3], which minimizes the sum of the intra-cluster variances. Its simplicity and efficiency have established it as a popular means for performing clustering across different disciplines. Even an extension to kernel space has been developed [4,5] to enable the identification of non-linearly separable groups. Despite its wide acceptance, *k*-Means suffers from a serious limitation. Its solution heavily depends on the initial positions of the cluster centers, thus after a bad initialization it easily gets trapped in poor local minima [6,7]. To alleviate this shortcoming, *k*-Means with multiple random restarts is often employed in practice.

Several methods attempt to overcome the sensitivity to the initialization in a more principled way. A first group of methods applies special techniques aiming at systematically avoiding partitionings of poor quality during the restarts. In [8], the initial centers are selected through a stochastic procedure such that the entire data space is covered. Theoretical guarantees are provided about the capability of the method to approximate the optimal clustering. Two approaches that start from random centers and penalize clusters relative to the winning frequency of their representatives are presented in [9,10]. Discouraging clusters to which several points have already been assigned from attracting new points in the subsequent steps has a regularizing effect. Centers that were initially ill-placed and are currently underutilized can actively participate in the solution on the following steps, which obstructs outlier clusters from forming and in effect balances the sizes of the clusters. Some other, analogous, strategies can be found in [11,12].

A second group of methods attempts to eliminate the dependence on random initial conditions, hence restarts are not anymore necessary. Global *k*-Means [13] and its modifications [14,15] are incremental approaches that start from a single cluster and at each step a new cluster is deterministically added to the solution according to an appropriate criterion. A kernel-based version of global *k*-Means is also available [16,17]. In [18] and its extension [19], spectral clustering is applied to locate the global optimum of a relaxed version of the *k*-Means objective, by formulating the problem as a trace maximization. Although these algorithms are not susceptible to bad initializations, they are computationally more expensive.

In this paper we propose MinMax *k*-Means, a novel approach that tackles the *k*-Means initialization problem by altering its objective. Our method starts from a randomly picked set of centers and tries to minimize the maximum intra-cluster variance instead of the sum of the intra-cluster variances. Specifically, *a weight is associated with each cluster*, such that clusters with larger variance[1] are allocated higher weights, and a weighted version of the sum of the intra-cluster variances criterion is derived. Different notions of weights have been

* Corresponding author. Tel.: +30 26510 08838; fax: +30 26510 08882.
E-mail addresses: gtzortzi@cs.uoi.gr (G. Tzortzis), arly@cs.uoi.gr (A. Likas).

[1] To avoid cluttering the text, we shall also refer to the intra-cluster variances, simply, as the variances of the clusters.

http://dx.doi.org/10.1016/j.patcog.2014.01.015
0031-3203 © 2014 Elsevier Ltd. All rights reserved.

exploited in the literature across several k-Means variants. In fuzzy c-means and Gaussian mixture models [20] weights are used to compute the degree of cluster membership of the instances, while in other variants weights are assigned to features, or groups of features, such that the tasks of clustering and feature selection are simultaneously performed [21,22]. Also, in [23], a weighting factor is added to each instance in order to detect outliers.

The per cluster weights predispose our algorithm towards primarily minimizing those clusters that currently exhibit a large variance, in essence confining the occurrence of large variance clusters in the outcome, and are *learned automatically*, together with the cluster assignments. The proposed method alternates between a minimization step, resembling the k-Means procedure, and an additional maximization step, in which the weights are calculated using closed-form expressions. By applying this weighting mechanism, results become less affected by the initialization and *solutions of high quality can be more consistently discovered*, even after starting from a bad initial set of centers. In addition, the obtained clusters are balanced with respect to their variance.

The presented algorithm also *incorporates a parameter* p, whose value must be specified prior to execution, that adjusts the degree of its bias towards penalizing large variance clusters. When $p=0$, k-Means, which has a zero bias, can be deduced as a special case of our method. A practical framework extending MinMax k-Means to automatically adapt this parameter to the dataset has been also developed in this work, so that the hidden group structures in the data can be successfully uncovered.

Experiments are conducted on several diverse datasets, including images, handwritten digits, proteins and patient records. MinMax k-Means is compared to k-Means, as well as to k-Means++ [8] and pifs k-Means [10] that evade degenerate optima, the first by methodically picking the initial centers and the second by balancing the cluster sizes. Our empirical evaluation reveals the effectiveness of the proposed clustering scheme in restricting the emergence of large variance clusters and producing superior solutions compared to the other three approaches, while restarted from random initializations. Furthermore, we observe that our algorithm constitutes a very promising technique for initializing k-Means.

The rest of this paper is organized as follows. We next briefly describe k-Means, while in Section 3 the proposed MinMax k-Means algorithm is presented and its properties are analyzed. Section 4 introduces our practical framework for setting the p parameter. The experiments follow in Section 5, before the concluding remarks of Section 6.

2. *k*-Means

To partition a dataset $\mathcal{X}=\{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i \in \Re^d$ into M disjoint clusters, $\{\mathcal{C}_k\}_{k=1}^M$, k-Means [3] minimizes the sum of the intra-cluster variances (1), where $\mathcal{V}_k=\sum_{i=1}^N \delta_{ik}\|\mathbf{x}_i-\mathbf{m}_k\|^2$ and $\mathbf{m}_k=\sum_{i=1}^N \delta_{ik}\mathbf{x}_i/\sum_{i=1}^N \delta_{ik}$ are the variance[2] and the center of the k-th cluster, respectively, and δ_{ik} is a cluster indicator variable with $\delta_{ik}=1$ if $\mathbf{x}_i \in \mathcal{C}_k$ and 0 otherwise.

$$\mathcal{E}_{sum}=\sum_{k=1}^M \mathcal{V}_k=\sum_{k=1}^M\sum_{i=1}^N \delta_{ik}\|\mathbf{x}_i-\mathbf{m}_k\|^2 \tag{1}$$

Clustering proceeds by alternating between assigning instances to their closest center and recomputing the centers, until a local minimum is (monotonically) reached.

Despite its simplicity and speed, k-Means has some drawbacks, with the most prominent being the dependence of the solution on the choice of initial centers [6,7]. Bad initializations can lead to poor local minima, thus multiple random restarts are usually executed to circumvent the initialization problem. Often, the solutions returned by the restarts significantly vary in terms of the achieved objective value, ranging from good to very bad ones, particularly for problems with a large search space (e.g. many clusters and dimensions). Therefore, numerous runs of the algorithm are required to increase the possibility of locating a good local minimum.

3. MinMax *k*-Means

As discussed in Section 2, the sensitivity of k-Means to initialization and the diverse solutions uncovered during the restarts make it difficult to find a good partitioning of the data. Motivated by this, we propose the optimization of a different objective and a new methodology that allows k-Means to *produce high quality partitionings more systematically*, while restarted from random initial centers.

3.1. The maximum variance objective

Consider a dataset $\mathcal{X}=\{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i \in \Re^d$ to be split into M disjoint clusters, $\{\mathcal{C}_k\}_{k=1}^M$. Instead of minimizing the *sum* of the intra-cluster variances (1), we propose to minimize the *maximum* intra-cluster variance

$$\mathcal{E}_{max}=\max_{1\le k\le M}\mathcal{V}_k=\max_{1\le k\le M}\left\{\sum_{i=1}^N \delta_{ik}\|\mathbf{x}_i-\mathbf{m}_k\|^2\right\}, \tag{2}$$

where $\mathcal{V}_k$, $\mathbf{m}_k$ and δ_{ik} are defined as in (1).

The rationale for this approach is the following: the summation over all clusters in the k-Means objective (1) allows for similar $\mathcal{E}_{sum}$ values to be achieved either by having a few clusters with large variance that are counterbalanced by others with small variance, or by having a moderate variance for all clusters. This means that the relative differences among the cluster variances are not taken into account. Note that the variance of a cluster is a measure of its quality. The above remark does not hold when minimizing $\mathcal{E}_{max}$ though, as the first case above would lead to a higher objective value. Hence, when minimizing $\mathcal{E}_{max}$, large variance clusters are avoided and *the solution space is now restricted towards clusters that exhibit more similar variances*.

The previous observation has two important implications. Since k-Means minimizes $\mathcal{E}_{sum}$, it cannot distinguish between the two cases, thus a bad initialization yields a poor solution that is characterized by substantially different variances among the returned clusters; a result of natural groups getting merged (large variance clusters) and others getting split (small variance clusters), or of outlier clusters being formed.[3] As explained, the maximum intra-cluster variance objective $\mathcal{E}_{max}$ is less likely to converge to such solutions, hence it is *easier to overcome a bad initialization*. Thus, we expect a k-Means type algorithm coupled with this objective to be able to uncover better group structures more consistently during the restarts. An example is illustrated in Fig. 1.

Additionally, a balancing effect on the clusters occurs. Balanced outcomes have been pursued in different ways in the literature. For example, in [10] k-Means and spherical k-Means are modified to penalize clusters in proportion to the number of instances assigned to them, while in [24,25] a graph cut criterion is optimized which favors the creation of subgraphs where the sums of the edge weights within the subgraphs (subgraph associations) are similar. In our case, balancing is done with regard to the

[2] In this work, we define cluster variance as the sum, and not the average, of the squared distances from the instances belonging to the cluster to its center.

[3] Let us clarify that a solution with quite different variances on the clusters is not necessarily a bad one. There are datasets where the natural groups exhibit such structure. We simply claim that such behavior also arises after a bad initialization, where some groups are merged and others are split.

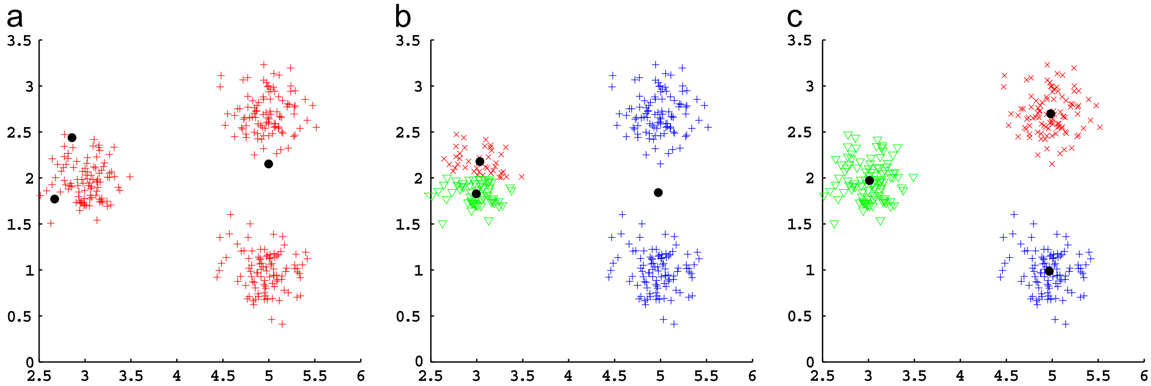

Fig. 1. Example (a) of a bad initialization that (b) leads to a poor k-Means solution, consisting of clusters with significantly different variances. On the contrary, our method, which is based on the notion of the maximum intra-cluster variance, manages to correctly locate the clusters (c) starting from the same initial centers. Different symbols and colors represent the cluster assignments and centers. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

variance of the clusters and not the number of cluster instances. As many real life applications demand partitionings of comparable size for subsequent data analysis [10], this is a nice and desired property of the presented objective (2). Note that a known shortcoming of k-Means is its tendency to produce skewed solutions, i.e. clusters with widely varying number of instances and/or near-empty clusters, especially for data with many dimensions and clusters, since the solution space vastly expands in this case [10,20].

3.2. A relaxed maximum variance objective

Despite the aforementioned advantages, directly minimizing the maximum intra-cluster variance $\mathcal{E}_{max}$ poses a non-trivial optimization problem. To tackle this problem, the objective is relaxed so it can be readily optimized in a k-Means iterative fashion. Specifically, we construct *a weighted formulation* $\mathcal{E}_w$ of the sum of the intra-cluster variances (3), where greater emphasis, i.e. a higher weight w_k, is placed on clusters with large variance, to mimic the behavior of the maximum variance criterion (2).

$$\mathcal{E}_w = \sum_{k=1}^{M} w_k^p \mathcal{V}_k = \sum_{k=1}^{M} w_k^p \sum_{i=1}^{N} \delta_{ik}\|\mathbf{x}_i - \mathbf{m}_k\|^2,$$

$$w_k \geq 0, \quad \sum_{k=1}^{M} w_k = 1, \quad 0 \leq p < 1 \tag{3}$$

In contrast to $\mathcal{E}_{max}$, now, all clusters contribute to the objective, albeit to different degrees regulated by the w_k values (the w_k^p values, to be precise). Obviously, the more a cluster contributes (higher weight), the more intensely its variance will be minimized, as in this way a bigger reduction of the objective is possible. Note that the weights are not constants, but parameters that must be optimized together with the cluster labels. We treat weights as parameters to allow their values to accurately reflect the variance of their respective clusters at each iteration during training and constrain them to sum to unity to avoid overfitting and get a meaningful optimization problem. The p exponent is a user specified constant that takes values in the range $[0, 1)$ and controls the sensitivity of the weight updates to the relative differences of the cluster variances, i.e. how strongly these differences are echoed by the weights. We shall shortly provide more insight into the $\mathcal{E}_w$ objective and thoroughly explain the role of p.

The general goal of clustering is to produce a partitioning with low intra-cluster variances (compact clusters) and at the same time we wish to rigorously guard against solutions in which large variance clusters occur, analogously to $\mathcal{E}_{max}$. In order for the relaxed objective to penalize large clusters, a higher variance should lead to a higher weight, which can be realized by maximizing $\mathcal{E}_w$ with respect to the weights. The resulting optimization problem is a min-max problem of the form

$$\min_{\{\mathcal{C}_k\}_{k=1}^M} \max_{\{w_k\}_{k=1}^M} \mathcal{E}_w,$$

$$\text{s.t. } w_k \geq 0, \quad \sum_{k=1}^{M} w_k = 1, \quad 0 \leq p < 1. \tag{4}$$

We propose an iterative algorithm, called MinMax k-Means, that alternates between the $\mathcal{C}_k$ and w_k optimization steps to get a local optimum of $\mathcal{E}_w$ and the corresponding derivations are presented next. Note that p is not part of the optimization and must be fixed a priori.

3.2.1. Minimization step

By keeping the weights fixed, new cluster assignments and representatives $\mathbf{m}_k$ are calculated. For the assignments, because the terms of $\mathcal{E}_w$ involving the cluster indicator variables δ_{ik} for the i-th instance are independent of the other instances, the optimization is straightforward, giving

$$\delta_{ik} = \begin{cases} 1, & k = \text{argmin}_{1 \leq k' \leq M} w_{k'}^p \|\mathbf{x}_i - \mathbf{m}_{k'}\|^2 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Hence, each instance is assigned to the cluster whose weighted distance from the representative to the instance is the smallest. Moreover, it is evident that as the weight w_k increases, only instances that are very close to the representative $\mathbf{m}_k$ are assigned to the k-th cluster.

To estimate the representatives, the derivatives of the objective function with respect to $\mathbf{m}_k$ are set to zero, which yields

$$\mathbf{m}_k = \frac{\sum_{i=1}^N \delta_{ik}\mathbf{x}_i}{\sum_{i=1}^N \delta_{ik}}. \tag{6}$$

As for k-Means, the representatives coincide with the centroids of the clusters and are independent of the weights.

3.2.2. Maximization step

To update the weights for given cluster assignments and centers, the weight constraints (4) are incorporated into the objective via a Lagrange multiplier and the derivatives with respect to w_k are set to zero. It is easy to verify that the constrained objective is concave with respect to the weights when $0 \leq p < 1$, hence their optimal values that maximize $\mathcal{E}_w$ given the current partitioning can be determined. After some manipulation the following closed-form solution emerges

$$w_k = \mathcal{V}_k^{1/(1-p)} \Bigg/ \sum_{k'=1}^{M} \mathcal{V}_{k'}^{1/(1-p)}, \quad \text{where } \mathcal{V}_k = \sum_{i=1}^{N} \delta_{ik}\|\mathbf{x}_i - \mathbf{m}_k\|^2. \tag{7}$$

As $1/(1-p) > 0$, since $0 \le p < 1$, it can be observed that the larger the cluster variance $\mathcal{V}_k$ the higher the weight w_k.

3.3. Discussion

In this section some aspects of the MinMax k-Means algorithm and its relaxed objective (3) are analyzed in more detail. According to (7), for a given partitioning of the data the weights are set proportionally to the cluster variances. In the subsequent minimization step, the assignment of instances to clusters is made using the weighted distance from the cluster centers (5). Apparently, for highly weighted clusters, the weighted distance of their representatives from the instances increases. Consequently, a cluster with large variance may lose some of its current instances that are away from its center (instances on the periphery of the cluster) and its variance is expected to decrease. At the same time, low variance clusters, due to the small weights, may also acquire instances that are not close to their centers and their variance will increase. Therefore, the iterative procedure of MinMax k-Means *punishes large variance clusters and operates towards clusters with similar variances*, resembling the maximum variance objective (2) whose advantages are carried over.

MinMax k-Means requires initial values for the cluster representatives and the weights. At the start no information about the variance of the clusters is available and the weights should be uniformly initialized, i.e. $w_k = 1/M$. Similar to k-Means, the solution depends on the initialization of the centers and multiple restarts are necessary. However, as $\mathcal{E}_w$ shares the same properties with $\mathcal{E}_{max}$, high quality solutions are anticipated on a regular basis compared to k-Means.

Regarding the p values, the most natural choice would be to propose a method where $p=1$. For $p=1$ the estimation of the weights simplifies to

$$w_k = \begin{cases} 1, & k = \mathrm{argmax}_{1 \le k' \le M} \mathcal{V}_{k'} \\ 0 & \text{otherwise}. \end{cases} \tag{8}$$

Obviously, in each iteration only the highest variance cluster receives a non-zero weight and thus in the following minimization step all its instances will be randomly assigned (5) to one of the other, zero-weight, clusters, which clearly signifies a degenerate case. If $p > 1$ is selected, the relaxed objective becomes convex with respect to the weights, thus the weight updates, which take the same form as in (7), will minimize $\mathcal{E}_w$ instead of maximizing it as required by (4). Therefore, only $0 \le p < 1$ can be permitted.

As for the effect of the p exponent $(0 \le p < 1)$, based on (7) it can be shown that the greater (smaller) the p value the less (more) similar the weight values become, as the relative differences of the variances among the clusters are enhanced (suppressed). This remark also holds for the w_k^p values, which are the actual coefficients used in the relaxed objective (3). To demonstrate the above in detail, the ratio between any two weights, $w_k/w_{k'}$, can be considered as an indicator of their similarity. The more this ratio tends to 1 the more similar the weights. Assume a fixed clustering, i.e. fixed cluster variances $\mathcal{V}_k$ and $\mathcal{V}_{k'}$. From (7),

$$\frac{w_k}{w_{k'}} = \left(\frac{\mathcal{V}_k}{\mathcal{V}_{k'}}\right)^{1/(1-p)}$$

and

$$\frac{w_k^p}{w_{k'}^p} = \left(\frac{\mathcal{V}_k}{\mathcal{V}_{k'}}\right)^{p/(1-p)},$$

$0 \le p < 1$. As p increases, the value of the $1/(1-p)$ and $p/(1-p)$ exponents grows, thus the relative differences of the cluster variances are enhanced and both ratios deviate more from 1, i.e. the weights and coefficients w_k^p attain less similar values (the exact opposite holds when p is decreased). In other words, p adjusts how intensely the differences of the cluster variances are reflected on the weights.

Therefore, for a high p value, large variance clusters accumulate considerably higher w_k and w_k^p values compared to low variance clusters, resulting in an objective that severely penalizes clusters with high variance. Note that an extremely high p may force clusters with large variance to lose most, or even all their instances, as their enormous weights will excessively distance the instances from their centers (5), something not desired of course. On the other hand, for $p=0$, all w_k^p coefficients equal 1, hence the differences of the cluster variances are ignored and actually the k-Means criterion is recovered, which permits high variance clusters. As shown in Section 3.1, preventing the appearance of large variance clusters is helpful in evading poor solutions after a bad initialization and also balances the clusters. However, this tactic may prove problematic when natural groups with different amounts of variance exist in the dataset, a common scenario in practice, as it will hinder the clustering process from unveiling the true structure of the data. We believe that intermediate p values provide a good compromise, since high variance clusters will be admitted up to a certain extent. In a nutshell, *the p exponent controls how strongly the relaxed objective of MinMax k-means restricts the occurrence of large variance clusters*, allowing its adaptation to the dataset. This is an important advantage over the maximum variance objective $\mathcal{E}_{max}$, whose strictness over large variance clusters cannot be adjusted.

4. Improving MinMax *k*-Means

A crucial limitation of the MinMax k-Means algorithm is the treatment of the p exponent as a predefined constant. While from the above discussion it is clear that a moderate p is preferable, this is a rough assessment that hardly provides an indication as to which exact p values suit a specific dataset. Therefore, manually selecting an appropriate p is not trivial and requires repeating the clustering for several p values. This task becomes even harder given the dependence of the solution on the initial centers for a particular p.

To circumvent this limitation, we devise a practical framework that extends MinMax k-Means to automatically adapt the exponent to the dataset, while alternating between the minimization and maximization steps as before. Specifically, we begin with a small p (p_{init}) that after each iteration is increased by step p_{step}, until a maximum value is attained (p_{max}). After p_{max} is reached, clustering continues without changing p. The idea behind this strategy is that the clusters formed during the first steps are heavily influenced by the initialization and should be allowed to freely evolve without considering their differences in variance, thus a small p is desirable (we set $p_{init} = 0$). As clustering progresses, p is gradually increased to restrain large variance clusters that persist in the solution and result in poor outcomes, especially after a bad initialization. Note that such a progressive punishment of large variance clusters is not possible when p is fixed a priori. Moreover, since clusters with high variance must not be completely eliminated in order to correctly uncover the intrinsic structures in the data (Section 3.3), extremely high values for p_{max} should be avoided.

As p grows, large variance clusters are susceptible to relinquishing most of their current instances (see Section 3.3). If an empty or singleton cluster appears, it will receive zero weight in the maximization step as $\mathcal{V}_k = 0$. This will cause all the dataset instances to be assigned to it in the subsequent minimization step (5) and the clustering process will collapse. This situation indicates that p has attained a very high value for the particular dataset. Whenever an empty or singleton cluster emerges, irrespective of whether p_{max} has been reached or not, we decrease p by p_{step}, revert back to the cluster assignments corresponding to the previous p value and resume clustering from there. Note that

p is never increased again in the following iterations. This manipulation of the p exponent has the same effect as setting p_{max} to be equal to the reduced p value from the beginning (here the adjustment is done automatically though) and actually shows that the presented framework is not very sensitive to the choice of p_{max}, as p will stop increasing when necessary.

To enhance the stability of the MinMax k-Means algorithm, a memory effect could be added to the weights:

$$w_k^{(t)} = \beta w_k^{(t-1)} + (1-\beta)\left(\mathcal{V}_k^{1/(1-p)} \Big/ \sum_{k'=1}^{M} \mathcal{V}_{k'}^{1/(1-p)}\right), \quad 0 \le \beta \le 1, \tag{9}$$

where β controls the influence of the previous iteration weights to the current update, allowing for smoother transitions of the weight values between consecutive iterations. It should be stressed that when memory is applied ($\beta > 0$), the newly derived weights no more correspond to their optimal values for the current partitioning, in contrast to the case where memory is not employed (see Section 3.2.2). However, this does not negatively affect our method, as convergence to a local optimum cannot be guaranteed, irrespective of the use of memory, since at every iteration both a minimization and a maximization of the objective is performed. On the contrary, our empirical evaluation has shown that memory is beneficial in several cases and that fewer empty clusters are created.

The change of the relaxed objective's value (3) between two consecutive iterations serves as the termination criterion. When this change is less than a tiny value ϵ, we stop. However, as mentioned above, convergence cannot be theoretically ensured, therefore we also stop if a predefined number of iterations t_{max} is exceeded. In practice we observed that convergence was achieved in many of our experiments. The pseudocode for the complete MinMax k-Means algorithm is shown in Algorithm 1.

Algorithm 1. MinMax k-Means.

```
Input: Dataset $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$, Initial centers $\{\mathbf{m}_k^{(0)}\}_{k=1}^M$, Number of clusters $M$, Secondary parameters (see text) $p_{max}$, $p_{step}$, $\beta$, $\epsilon$, $t_{max}$
Output: Cluster assignments $\{\delta_{ik}\}_{i=1,\dots,N,k=1,\dots,M}$, Final centers $\{\mathbf{m}_k\}_{k=1}^M$
Set $t=0$
Set $p_{init}=0$
Set $w_k^{(0)} = 1/M,\ \forall k = 1,\dots,M$
Set $empty$ = false //No empty or singleton clusters yet detected.
$p = p_{init}$
repeat
  $t = t+1$
  for $i=1$ to $N$ do //Update the cluster assignments.
    for $k=1$ to $M$ do
      $\delta_{ik}^{(t)} = \begin{cases} 1, & k = \text{argmin}_{1 \le k' \le M} (w_{k'}^{(t-1)})^p \|\mathbf{x}_i - \mathbf{m}_{k'}^{(t-1)}\|^2 \\ 0 & \text{otherwise} \end{cases}$
    end for
  end for
  if empty or singleton clusters have occurred at time $t$ then // Reduce $p$.
    $empty$ = true
    $p = p - p_{step}$
    if $p < p_{init}$ then
      return NULL
    end if
    //Revert to the assignments and weights corresponding to the reduced $p$.
    $\delta_{ik}^{(t)} = [\Delta^{(p)}]_{ik},\ \forall k = 1,\dots,M,\ \forall i = 1,\dots,N$
    $w_k^{(t-1)} = [\mathbf{w}^{(p)}]_k,\ \forall k = 1,\dots,M$
  end if
  for all $\mathbf{m}_k$, $k=1$ to $M$ //Update the centers.
    $\mathbf{m}_k^{(t)} = \sum_{i=1}^N \delta_{ik}^{(t)} \mathbf{x}_i / \sum_{i=1}^N \delta_{ik}^{(t)}$
  end for
  if $p < p_{max}$ and $empty$ = false then //Increase $p$.
    $\Delta^{(p)} = [\delta_{ik}^{(t)}]$ //Store the current assignments in matrix $\Delta^{(p)}$.
    $\mathbf{w}^{(p)} = [w_k^{(t-1)}]$ //Store the previous weights in vector $\mathbf{w}^{(p)}$.
    $p = p + p_{step}$
  end if
  for all $w_k$, $k=1$ to $M$ do //Update the weights.
    $w_k^{(t)} = \beta w_k^{(t-1)} + (1-\beta)((\mathcal{V}_k^{(t)})^{1/(1-p)} / \sum_{k'=1}^M (\mathcal{V}_{k'}^{(t)})^{1/(1-p)})$,    where
    $\mathcal{V}_k^{(t)} = \sum_{i=1}^N \delta_{ik}^{(t)} \|\mathbf{x}_i - \mathbf{m}_k^{(t)}\|^2$
  end for
until $|\mathcal{E}_w^{(t)} - \mathcal{E}_w^{(t-1)}| < \epsilon$ or $t \ge t_{max}$
return $\{\delta_{ik}^{(t)}\}_{i=1,\dots,N,k=1,\dots,M}$, $\{\mathbf{m}_k^{(t)}\}_{k=1}^M$
```

5. Empirical evaluation

The performance of MinMax k-Means[4] is studied on several datasets and we wish to investigate if indeed its relaxed objective (3) limits the occurrence of large variance clusters and how effective the proposed method is in overcoming bad initializations and attaining good solutions more regularly than k-Means.

To demonstrate the above, first, a comparison to the basic k-Means algorithm is made. As already discussed, k-Means does not consider the relative differences of the clusters, allowing high variance clusters to emerge. Also, its solution is greatly affected by the initial centers. Hence, this comparison will provide strong evidence on the effectiveness of MinMax k-Means. Moreover, we also experiment with two k-Means variants, called k-Means++ and partially incremental frequency sensitive (pifs) k-Means.

In k-Means++ [8] a stochastic procedure is employed to pick the initial cluster centers and then k-Means is executed from these centers. Specifically, given that $k-1$ centers have already been selected, instance $\mathbf{x}_i$ may be selected as the k-th initial center with a probability that is proportional to its minimum distance from the $k-1$ centers. The above procedure aims at selecting initial centers that cover the entire data space, thus providing better initializations to k-Means (compared to random starts), and, therefore, constituting a worthy competitor against which to measure our method.

Pifs k-Means [10] explicitly penalizes clusters in proportion to the number of instances already assigned to them, according to the following cluster update rule[5]:

$$\delta_{ik} = \begin{cases} 1, & k = \operatorname{argmin}_{1 \le k' \le M} |\mathcal{C}_{k'}| \, \|\mathbf{x}_i - \mathbf{m}_{k'}\|^2 \\ 0 & \text{otherwise}, \end{cases} \tag{10}$$

where $|\mathcal{C}_k|$ is the current size of the k-th cluster. Based on (10), the larger the cluster the lower the chance of an instance being acquired by that cluster. Thus, clusters are balanced in terms of their size, which has been shown to decrease the sensitivity to bad initializations [9,10]. Remember (Section 3.3), that MinMax k-Means, implicitly, through its weighting strategy, operates towards clusters with similar variances. Therefore, it is interesting to examine how these two different balancing approaches compare against each other.

5.1. Datasets

Six popular datasets are utilized in our empirical study for which the ground-truth is available. Their features are normalized to zero mean and unit variance, unless stated otherwise.

Coil-20 [26] contains 72 images taken from different angles for each of the 20 included objects. As in [27], SIFT descriptors [28] are first extracted from the images which are then represented by the bag of visual words model using 1000 visual words and the data vectors are normalized to unit length. For our purposes, three subsets of Coil-20 were created, Coil1 (objects 3, 9 and 10), Coil2 (objects 15, 18 and 19) and Coil3 (objects 2, 4, 7, 10 and 11).

Multiple features & Pendigits are two collections of handwritten digits (0–9) from the UCI repository [29]. Multiple features digits (200 per class) are described in terms of six different feature sets and we select two of them, namely pixel averages and profile correlations. Pendigits consists of 10 992 instances (roughly 1100 samples per numeral) in 16-dimensional space.

Table 1
Main characteristics of the tested datasets.

Dataset	Instances	Features	Classes	Balanced
Coil1 & Coil2	216	1000	3	Yes
Coil3	360	1000	5	Yes
Multiple features—pixel averages	2000	240	10	Yes
Multiple features—profile correlations	2000	216	10	Yes
Pendigits	10 992	16	10	Almost
Olivetti	900	2500	10	Yes
Ecoli	307	7	4	No
Dermatology	366	34	6	No

Olivetti is a face database of 40 individuals with ten 64×64 grayscale images per individual. Based on [30], we only retain the first 100 images, belonging to ten persons, and apply the same preprocessing. Specifically, each image is smoothed using a Gaussian kernel and then rotated by -10°, 0° and 10° and scaled by a factor of 0.9, 1.0 and 1.1, resulting in 900 images. Finally, a central 50×50 window of the images is kept and its pixels are normalized to zero mean and 0.1 variance.

Ecoli (UCI) [29] includes 336 proteins from the *E. coli* bacterium and seven attributes, calculated from the amino acid sequences, are provided. Proteins belong to eight categories according to their cellular localization sites. Four of the classes are extremely underrepresented and are not considered in our evaluation. Note that classes differ in size, i.e. it is an unbalanced dataset.

Dermatology (UCI) [29] is composed of 366 patient records that suffer from six different types of the Eryhemato-Squamous disease. Each patient is described by both clinical and histopathological features (34 in total). This dataset is also unbalanced.

A summary of the datasets is provided in Table 1.

5.2. Experimental protocol

All tested algorithms, apart from k-Means++, are restarted 500 times from the same randomly chosen initial centers. For k-Means++, the stochastic initialization procedure is executed 500 times. The number of clusters is set equal to the number of classes in each dataset, throughout the experiments. To evaluate the quality of the returned solutions, the maximum cluster variance $\mathcal{E}_{max}$, defined in (2), and the sum of the cluster variances $\mathcal{E}_{sum}$, defined in (1), serve as the main performance measures and their average and standard deviation over the 500 runs is reported. Note that $\mathcal{E}_{sum}$ favors k-Means and k-Means++ in the comparisons, since this is the objective optimized by these two methods. Likewise, $\mathcal{E}_{max}$ favors our framework which optimizes a relaxed version of (2). Since the ground-truth is available, the achieved NMI score (11)[6] is also reported. Higher NMI values indicate a better match between the cluster labels and the class labels.

$$NMI = \frac{2\sum_{k=1}^{M}\sum_{h=1}^{C}\frac{n_k^h}{N}\log\frac{n_k^h N}{\sum_{i=1}^{M} n_i^h \sum_{i=1}^{C} n_k^i}}{\mathcal{H}_M + \mathcal{H}_C} \tag{11}$$

Moreover, to assess the computational complexity of the algorithms, their average execution time (in seconds) is reported.

In a second series of experiments, the cluster centers derived by each execution of MinMax k-Means and pifs k-Means are used to initialize a subsequent k-Means run. This allows us to determine if k-Means performance can be improved when initialized by these

[4] Matlab code is available at: http://www.cs.uoi.gr/~gtzortzi.

[5] Note that the exact cluster update rule proposed in [10] contains an additional $d \ln|\mathcal{C}_{k'}|$ term, where d is the dataset dimensionality. However, better results were obtained without using this term in our experiments.

[6] N is the dataset size, M is the number of clusters, C is the number of classes, n_k^h is the number of points in cluster k belonging to class h, and $\mathcal{H}_M$, $\mathcal{H}_C$ is the entropy of the clusters and the classes, respectively.

two approaches and also facilitates the comparison of the tested methods under a common objective ($\mathcal{E}_{sum}$).

For MinMax *k*-Means, some additional parameters must be fixed prior to execution (p_{max}, p_{step}, β, ϵ and t_{max}). Our method is not particularly sensitive to either p_{max} or p_{step}. Regarding p_{max}, p stops increasing when empty or singleton clusters are detected. For p_{step}, one should simply avoid a large step which will cause abrupt changes to the p value between consecutive iterations. Thus, we do not fine-tune these two parameters for each dataset and for all the experiments we set $p_{max}=0.5$ and $p_{step}=0.01$. Note that empty clusters appear quite often for the selected p_{max} value, indicating that it is already set to a high value. For β, we tried three different levels of memory, $\beta \in \{0, 0.1, 0.3\}$, and present the corresponding results. Finally, $\epsilon = 10^{-6}$ and $t_{max}=500$ for all experiments.

5.3. Performance analysis

The comparison of the algorithms across the various datasets is shown in Tables 2–10, where MinMax *k*-Means and pifs *k*-Means are abbreviated as MinMax and pifs, respectively. Tables 2(b)– 10(b), labeled as "method + *k*-Means", refer to the experiments where *k*-Means is initialized from the solution of the method designated by the corresponding row. For example, we denote as MinMax+*k*-Means (pifs+*k*-Means), the method where MinMax *k*-Means (pifs *k*-Means) is executed first and its solution is used to initialize a subsequent run of *k*-Means. Of course, reinitializing *k*-Means with its own, or the *k*-Means++ solution has no effect and the results are just replicated from Tables 2(a)– 10(a) for readers' convenience. Superscripts asterisk (*), dagger (†) and double dagger (‡) denote that MinMax *k*-Means has a statistically significant difference to *k*-Means, *k*-Means++ and pifs *k*-Means, respectively, according to the *t*-test (the significance level is taken as 0.05). A line above (below) these symbols stands for a higher (lower) average, respectively.

From the tables two main observations can be made. First, all memory levels of MinMax *k*-Means exhibit better (smaller) $\mathcal{E}_{max}$ than *k*-Means, *k*-Means++ and pifs *k*-Means for every dataset (Tables 2(a)– 10(a)), but Pendigits. This clearly displays that the relaxed objective (3) minimizes large variance clusters and mimics the maximum variance criterion (2). Note also that *k*-Means, when

Table 2
(a) Comparative results on the Coil1 dataset. (b) Comparative results on the Coil1 dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	NMI	Time
(a)				
MinMax ($\beta=0$)	$46.61 \pm 0.00^{\underline{*}\underline{\dagger\ddagger}}$	$119.02 \pm 0.00^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	$0.88 \pm 0.00^{\underline{\dagger\ddagger}}$	0.54 (0.54)
MinMax ($\beta=0.1$)	$45.75 \pm 0.00^{\underline{*}\underline{\dagger\ddagger}}$	$119.24 \pm 0.00^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	$0.87 \pm 0.00^{\underline{*}\underline{\dagger\ddagger}}$	0.42 (0.42)
MinMax ($\beta=0.3$)	$\mathbf{45.04 \pm 0.00}^{\underline{*}\underline{\dagger\ddagger}}$	$119.40 \pm 0.00^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	$0.87 \pm 0.00^{\underline{*}\underline{\dagger\ddagger}}$	0.42 (0.42)
***k*–Means**	66.33 ± 19.46	121.24 ± 7.12	0.89 ± 0.16	**0.07**
***k*–Means++**	64.92 ± 18.83	121.01 ± 7.18	0.90 ± 0.16	0.09
Pifs	53.43 ± 0.00	**117.82 ± 0.00**	**1.00 ± 0.00**	**0.07**
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta=0$)	$\mathbf{53.43 \pm 0.00}^{\underline{*}\underline{\dagger}}$	$\mathbf{117.82 \pm 0.00}^{\underline{*}\underline{\dagger}}$	$\mathbf{1.00 \pm 0.00}^{\overline{*}\overline{\dagger}}$	0.58 (0.58)
MinMax ($\beta=0.1$)	$\mathbf{53.43 \pm 0.00}^{\underline{*}\underline{\dagger}}$	$\mathbf{117.82 \pm 0.00}^{\underline{*}\underline{\dagger}}$	$\mathbf{1.00 \pm 0.00}^{\overline{*}\overline{\dagger}}$	0.45 (0.45)
MinMax ($\beta=0.3$)	$\mathbf{53.43 \pm 0.00}^{\underline{*}\underline{\dagger}}$	$\mathbf{117.82 \pm 0.00}^{\underline{*}\underline{\dagger}}$	$\mathbf{1.00 \pm 0.00}^{\overline{*}\overline{\dagger}}$	0.46 (0.46)
***k*–Means**	66.33 ± 19.46	121.24 ± 7.12	0.89 ± 0.16	**0.07**
***k*–Means++**	64.92 ± 18.83	121.01 ± 7.18	0.90 ± 0.16	0.09
Pifs	**53.43 ± 0.00**	**117.82 ± 0.00**	**1.00 ± 0.00**	0.09

Table 3
(a) Comparative results on the Coil2 dataset. (b) Comparative results on the Coil2 dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	NMI	Time
(a)				
MinMax ($\beta=0$)	$58.74 \pm 0.36^{\underline{*}\underline{\dagger\ddagger}}$	$\mathbf{154.49 \pm 1.04}^{\underline{*}\underline{\dagger\ddagger}}$	$\mathbf{0.95 \pm 0.14}^{\overline{*}\overline{\dagger\ddagger}}$	1.94 (0.47)
MinMax ($\beta=0.1$)	$\mathbf{57.14 \pm 0.35}^{\underline{*}\underline{\dagger\ddagger}}$	$155.09 \pm 0.85^{\underline{*}\underline{\ddagger}}$	$0.91 \pm 0.13^{\overline{*}\overline{\dagger\ddagger}}$	0.45 (0.44)
MinMax ($\beta=0.3$)	$58.73 \pm 0.42^{\underline{*}\underline{\dagger\ddagger}}$	$154.56 \pm 1.09^{\underline{*}\underline{\dagger\ddagger}}$	$0.94 \pm 0.14^{\overline{*}\overline{\dagger\ddagger}}$	0.52 (0.52)
***k*–Means**	77.46 ± 18.74	155.49 ± 2.26	0.80 ± 0.16	**0.09**
***k*–Means++**	74.33 ± 16.87	155.18 ± 1.85	0.82 ± 0.16	0.10
Pifs	59.48 ± 1.11	155.96 ± 1.61	0.75 ± 0.19	0.11
Method+*k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta=0$)	$\mathbf{58.95 \pm 0.97}^{\underline{*}\underline{\dagger\ddagger}}$	$\mathbf{154.38 \pm 0.93}^{\underline{*}\underline{\dagger\ddagger}}$	$\mathbf{0.95 \pm 0.13}^{\overline{*}\overline{\dagger\ddagger}}$	1.97 (0.50)
MinMax ($\beta=0.1$)	$59.03 \pm 1.70^{\underline{*}\underline{\dagger\ddagger}}$	$154.39 \pm 0.94^{\underline{*}\underline{\dagger\ddagger}}$	$\mathbf{0.95 \pm 0.13}^{\overline{*}\overline{\dagger\ddagger}}$	0.47 (0.46)
MinMax ($\beta=0.3$)	$59.11 \pm 1.98^{\underline{*}\underline{\dagger\ddagger}}$	$154.42 \pm 0.96^{\underline{*}\underline{\dagger\ddagger}}$	$0.94 \pm 0.14^{\overline{*}\overline{\dagger\ddagger}}$	0.55 (0.55)
***k*–Means**	77.46 ± 18.74	155.49 ± 2.26	0.80 ± 0.16	**0.09**
***k*–Means++**	74.33 ± 16.87	155.18 ± 1.85	0.82 ± 0.16	0.10
Pifs	62.84 ± 7.06	155.58 ± 1.50	0.77 ± 0.19	0.14

Table 4
(a) Comparative results on the Coil3 dataset. (b) Comparative results on the Coil3 dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(a)				
MinMax ($\beta = 0$)	58.00 ± 0.27$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	245.95 ± 0.71$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.99 ± 0.03**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	3.29 (0.74)
MinMax ($\beta = 0.1$)	57.90 ± 0.25$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**245.64 ± 0.75**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.99 ± 0.03**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	5.46 (0.81)
MinMax ($\beta = 0.3$)	**53.24 ± 0.40**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	249.82 ± 0.24$^{\overline{\dagger}\overline{\ddagger}}$	0.94 ± 0.01$^{\overline{*}\overline{\dagger}}$	3.36 (0.82)
***k*–Means**	101.95 ± 29.81	249.64 ± 5.64	0.88 ± 0.08	**0.15**
***k*–Means++**	96.35 ± 28.37	249.13 ± 5.45	0.89 ± 0.07	0.18
Pifs	58.39 ± 1.07	246.47 ± 2.52	0.95 ± 0.08	0.20
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta = 0$)	58.30 ± 2.85$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	245.41 ± 0.41$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.99 ± 0.02**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	3.33 (0.82)
MinMax ($\beta = 0.1$)	58.26 ± 2.72$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	245.41 ± 0.41$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.99 ± 0.02**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	5.51 (0.90)
MinMax ($\beta = 0.3$)	**58.03 ± 1.77**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**245.40 ± 0.23**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.99 ± 0.01**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	3.40 (0.89)
***k*–Means**	101.95 ± 29.81	249.64 ± 5.64	0.88 ± 0.08	**0.15**
***k*–Means++**	96.35 ± 28.37	249.13 ± 5.45	0.89 ± 0.07	0.18
Pifs	64.12 ± 9.50	245.68 ± 2.01	0.96 ± 0.06	0.25

Table 5
(a) Comparative results on the Multiple features (pixel averages) dataset. (b) Comparative results on the Multiple features (pixel averages) dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(a)				
MinMax ($\beta = 0$)	149.60 ± 9.56$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	1239.33 ± 6.19$^{\overline{\dagger}\overline{\ddagger}}$	0.68 ± 0.03$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	2.59 (2.00)
MinMax ($\beta = 0.1$)	146.73 ± 14.70$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	1240.49 ± 8.61$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.68 ± 0.03$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	2.36 (1.98)
MinMax ($\beta = 0.3$)	**145.00 ± 17.17**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	1243.09 ± 13.05$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.68 ± 0.04$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	2.22 (1.50)
***k*–Means**	222.50 ± 33.95	1238.36 ± 12.51	0.71 ± 0.04	**0.66**
***k*–Means++**	219.63 ± 36.34	**1237.24 ± 11.18**	0.71 ± 0.04	0.80
Pifs	150.75 ± 4.47	1237.84 ± 4.31	**0.72 ± 0.05**	1.03
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta = 0$)	202.03 ± 23.73$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	1230.64 ± 5.56$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.72 ± 0.03$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	2.87 (2.28)
MinMax ($\beta = 0.1$)	200.20 ± 23.89$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	1230.52 ± 5.38$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.72 ± 0.03$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	2.66 (2.28)
MinMax ($\beta = 0.3$)	198.91 ± 24.51$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	**1229.77 ± 4.27**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.72 ± 0.03$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	2.55 (1.83)
***k*–Means**	222.50 ± 33.95	1238.36 ± 12.51	0.71 ± 0.04	**0.66**
***k*–Means++**	219.63 ± 36.34	1237.24 ± 11.18	0.71 ± 0.04	0.80
Pifs	**177.06 ± 21.25**	1232.07 ± 3.53	**0.74 ± 0.04**	1.30

Table 6
(a) Comparative results on the Multiple features (profile correlations) dataset. (b) Comparative results on the Multiple features (profile correlations) dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(a)				
MinMax ($\beta = 0$)	**118.60 ± 7.63**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**966.96 ± 8.43**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.69 ± 0.04$^{\underline{\ddagger}}$	2.84 (1.98)
MinMax ($\beta = 0.1$)	150.97 ± 52.71$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	1004.81 ± 52.86$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.67 ± 0.04$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	3.93 (1.82)
MinMax ($\beta = 0.3$)	120.21 ± 15.16$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	972.86 ± 13.50$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	0.69 ± 0.04$^{\underline{\ddagger}}$	2.13 (1.03)
***k*–Means**	179.22 ± 41.17	970.18 ± 15.90	0.69 ± 0.04	**0.49**
***k*–Means++**	175.74 ± 37.88	968.81 ± 15.43	0.69 ± 0.03	0.63
Pifs	133.29 ± 10.57	974.54 ± 5.63	**0.71 ± 0.04**	1.00
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta = 0$)	155.36 ± 16.13$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	958.28 ± 6.97$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.70 ± 0.03$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	3.05 (2.19)
MinMax ($\beta = 0.1$)	154.59 ± 13.08$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	957.78 ± 6.54$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.70 ± 0.03$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	4.17 (2.04)
MinMax ($\beta = 0.3$)	**153.97 ± 12.22**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**957.63 ± 6.28**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.70 ± 0.03$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	2.37 (1.26)
***k*–Means**	179.22 ± 41.17	970.18 ± 15.90	0.69 ± 0.04	**0.49**
***k*–Means++**	175.74 ± 37.88	968.81 ± 15.43	0.69 ± 0.03	0.63
Pifs	160.16 ± 11.63	962.93 ± 3.56	**0.72 ± 0.04**	1.26

Table 7
(a) Comparative results on the Pendigits dataset. (b) Comparative results on the Pendigits dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(a)				
MinMax ($\beta = 0$)	**7769.50 ± 1249.80**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	61 140.86 ± 659.81$^{\overline{\dagger}\underline{\ddagger}}$	0.68 ± 0.01$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	2.72 (2.23)
MinMax ($\beta = 0.1$)	17 497.21 ± 5431.65$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	71 599.61 ± 5066.73$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.64 ± 0.03$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	4.79 (1.47)
MinMax ($\beta = 0.3$)	8849.21 ± 1706.73$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	62 345.44 ± 1266.36$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.69 ± 0.01$^{\underline{\ddagger}}$	2.27 (0.91)
***k*–Means**	11 576.43 ± 3125.47	61 024.17 ± 1333.92	0.69 ± 0.02	**0.55**
***k*–Means++**	11 857.89 ± 3039.04	**60 940.96 ± 1294.01**	0.69 ± 0.02	0.56
Pifs	8623.37 ± 329.35	61 895.12 ± 643.98	**0.70 ± 0.01**	3.06
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta = 0$)	9403.21 ± 2760.33$^{\underline{*}\underline{\dagger}}$	60 681.71 ± 710.50$^{\underline{*}\underline{\dagger}}$	0.69 ± 0.01$^{\underline{\ddagger}}$	2.88 (2.39)
MinMax ($\beta = 0.1$)	9835.21 ± 2444.54$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	60 447.71 ± 751.37$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.70 ± 0.01$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	4.99 (1.60)
MinMax ($\beta = 0.3$)	**9258.11 ± 2590.49**$^{\underline{*}\underline{\dagger}}$	**60 366.92 ± 731.99**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	0.69 ± 0.01$^{\underline{\ddagger}}$	2.50 (1.07)
***k*–Means**	11 576.43 ± 3125.47	61 024.17 ± 1333.92	0.69 ± 0.02	**0.55**
***k*–Means++**	11 857.89 ± 3039.04	60 940.96 ± 1294.01	0.69 ± 0.02	0.56
Pifs	9289.79 ± 672.91	60 722.65 ± 684.59	**0.71 ± 0.00**	3.25

Table 8
(a) Comparative results on the Olivetti dataset. (b) Comparative results on the Olivetti dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(a)				
MinMax ($\beta = 0$)	1217.72 ± 55.18$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**11 016.58 ± 44.35**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.34 ± 0.04**	7.80 (7.43)
MinMax ($\beta = 0.1$)	1207.91 ± 86.61$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	11 019.11 ± 83.40$^{\underline{*}\underline{\dagger}}$	**0.34 ± 0.04**	7.26 (7.10)
MinMax ($\beta = 0.3$)	**1198.19 ± 92.13**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	11 019.25 ± 69.03$^{\underline{*}\underline{\dagger}}$	**0.34 ± 0.04**	6.50 (6.22)
***k*–Means**	1610.49 ± 152.77	11 034.37 ± 61.38	**0.34 ± 0.03**	**2.40**
***k*–Means++**	1624.46 ± 158.38	11 031.70 ± 64.07	**0.34 ± 0.03**	2.82
Pifs	1305.87 ± 36.61	11 024.36 ± 45.72	**0.34 ± 0.03**	2.97
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta = 0$)	1383.35 ± 120.45$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	10 985.52 ± 41.70$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.34 ± 0.04**	8.61 (8.24)
MinMax ($\beta = 0.1$)	1374.73 ± 117.89$^{\underline{*}\underline{\dagger}}$	10 984.49 ± 41.86$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.34 ± 0.04**	8.04 (7.88)
MinMax ($\beta = 0.3$)	1367.46 ± 116.57$^{\underline{*}\underline{\dagger}}$	**10 980.86 ± 42.48**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.34 ± 0.04**	7.33 (7.05)
***k*–Means**	1610.49 ± 152.77	11 034.37 ± 61.38	**0.34 ± 0.03**	**2.40**
***k*–Means++**	1624.46 ± 158.38	11 031.70 ± 64.07	**0.34 ± 0.03**	2.82
Pifs	**1362.69 ± 101.90**	10 993.37 ± 40.90	**0.34 ± 0.03**	3.91

Table 9
(a) Comparative results on the Ecoli dataset. (b) Comparative results on the Ecoli dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(a)				
MinMax ($\beta = 0$)	5.29 ± 0.15$^{\underline{*}\underline{\dagger}}$	15.94 ± 0.24$^{\overline{*}\overline{\dagger}\underline{\ddagger}}$	0.58 ± 0.01$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	0.18 (0.06)
MinMax ($\beta = 0.1$)	5.02 ± 0.25$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	15.72 ± 0.04$^{\underline{\ddagger}}$	0.57 ± 0.01$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	0.11 (0.05)
MinMax ($\beta = 0.3$)	**4.80 ± 0.00**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	15.73 ± 0.00$^{\underline{\ddagger}}$	0.58 ± 0.00$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	0.05 (0.05)
***k*–Means**	6.38 ± 0.88	**15.68 ± 0.54**	**0.61 ± 0.02**	0.02
***k*–Means++**	6.60 ± 1.58	15.79 ± 1.02	**0.61 ± 0.03**	**0.01**
Pifs	5.30 ± 0.28	16.19 ± 0.15	0.55 ± 0.01	0.04
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax ($\beta = 0$)	6.29 ± 0.11$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	15.40 ± 0.03$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.63 ± 0.00**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.19 (0.06)
MinMax ($\beta = 0.1$)	6.29 ± 0.00$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	**15.39 ± 0.00**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.63 ± 0.00**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.12 (0.06)
MinMax ($\beta = 0.3$)	6.29 ± 0.00$^{\underline{*}\underline{\dagger}\overline{\ddagger}}$	**15.39 ± 0.00**$^{\underline{*}\underline{\dagger}\underline{\ddagger}}$	**0.63 ± 0.00**$^{\overline{*}\overline{\dagger}\overline{\ddagger}}$	0.05 (0.05)
***k*–Means**	6.38 ± 0.88	15.68 ± 0.54	0.61 ± 0.02	0.02
***k*–Means++**	6.60 ± 1.58	15.79 ± 1.02	0.61 ± 0.03	**0.01**
Pifs	**6.04 ± 0.35**	15.65 ± 0.19	0.61 ± 0.02	0.05

Table 10
(a) Comparative results on the Dermatology dataset. (b) Comparative results on the Dermatology dataset when *k*-Means is initialized by the solution returned by MinMax *k*-Means and pifs *k*-Means.

Method	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(a)				
MinMax ($\beta=0$)	$1513.85 \pm 316.42^{\underline{*\dagger\ddagger}}$	$\mathbf{5672.82 \pm 272.21}^{\underline{*\dagger\ddagger}}$	$\mathbf{0.82 \pm 0.03}^{\overline{\ddagger}}$	0.37 (0.18)
MinMax ($\beta=0.1$)	$1439.76 \pm 296.76^{\underline{*\dagger\ddagger}}$	$5685.16 \pm 237.35^{\underline{*\dagger\ddagger}}$	$\mathbf{0.82 \pm 0.03}^{\overline{\ddagger}}$	0.37 (0.16)
MinMax ($\beta=0.3$)	$\mathbf{1368.05 \pm 347.04}^{\underline{*\dagger\ddagger}}$	$5703.26 \pm 195.87^{\underline{*\dagger\ddagger}}$	$\mathbf{0.82 \pm 0.01}^{\overline{\ddagger}}$	0.49 (0.16)
***k*–Means**	2247.59 ± 804.75	5885.92 ± 542.49	**0.82 ± 0.07**	0.10
***k*–Means++**	2134.54 ± 681.34	5800.23 ± 448.38	**0.82 ± 0.07**	0.11
Pifs	1650.13 ± 91.99	6057.18 ± 50.62	0.80 ± 0.01	**0.08**
Method + *k*–Means	$\mathcal{E}_{max}$	$\mathcal{E}_{sum}$	**NMI**	**Time**
(b)				
MinMax($\beta=0$)	$1683.33 \pm 402.51^{\underline{*\dagger\ddagger}}$	$5578.95 \pm 295.56^{\underline{*\dagger}\overline{\ddagger}}$	$0.86 \pm 0.04^{\overline{*\dagger}\underline{\ddagger}}$	0.42 (0.23)
MinMax ($\beta=0.1$)	$1609.88 \pm 379.81^{\underline{*\dagger\ddagger}}$	$5548.56 \pm 263.49^{\underline{*\dagger}\overline{\ddagger}}$	$0.86 \pm 0.03^{\overline{*\dagger}\underline{\ddagger}}$	0.42 (0.21)
MinMax ($\beta=0.3$)	$\mathbf{1395.32 \pm 109.48}^{\underline{*\dagger\ddagger}}$	$\mathbf{5441.13 \pm 107.40}^{\underline{*\dagger\ddagger}}$	$\mathbf{0.87 \pm 0.01}^{\overline{*\dagger}}$	0.54 (0.21)
***k*–Means**	2247.59 ± 804.75	5885.92 ± 542.49	0.82 ± 0.07	**0.10**
***k*–Means++**	2134.54 ± 681.34	5800.23 ± 448.38	0.82 ± 0.07	0.11
Pifs	1761.13 ± 358.36	5496.97 ± 207.25	**0.87 ± 0.02**	**0.10**

initialized by our algorithm, leads to clusters with lower $\mathcal{E}_{max}$ for most datasets (Tables 2(b)– 10(b)). However, *k*-Means optimizes the sum of the variances and does not consider the maximum variance. Hence, it is reasonable in this case that $\mathcal{E}_{max}$ increases compared to that before employing *k*-Means and that pifs+ *k*-Means produces equal or better $\mathcal{E}_{max}$ scores than MinMax+ *k*-Means for half of the datasets.

Second, our method outperforms *k*-Means for all the metrics (apart from execution time) reported in Tables 2–10, demonstrating its ability to attain good partitionings on a more frequent basis. To add to the above, MinMax+*k*-Means obtains lower $\mathcal{E}_{sum}$ and higher NMI values than *k*-Means, i.e. *k*-Means converges to better local minima when initialized by MinMax *k*-Means. Actually, the main difference between *k*-Means and MinMax+*k*-Means is that some restarts of the former return solutions with excessively high $\mathcal{E}_{sum}$ (its higher standard deviation is indicative of that), while for the latter such poor outcomes do not emerge, illustrating the robustness of MinMax *k*-Means over bad initializations.

Considering *k*-Means++, its stochastic initialization process improves performance, as lower $\mathcal{E}_{max}$ and $\mathcal{E}_{sum}$ (and equal NMI) values are acquired on most cases compared to the randomly restarted *k*-Means. When put up against MinMax *k*-Means though, similar conclusions to those mentioned above for *k*-Means can be drawn, further establishing the potential of the presented framework. It is of particular interest that MinMax+*k*-Means yields better $\mathcal{E}_{sum}$ and NMI scores on every dataset, despite *k*-Means++ carefully picking the initial centers. This definitely reveals that the centers outputted by MinMax *k*-Means consist good initializations for *k*-Means.

The proposed algorithm is also superior to pifs *k*-Means. Specifically, it always reaches a lower $\mathcal{E}_{max}$ (the exception being Multiple features-profile correlations for $\beta=0.1$ and Pendigits for $\beta=0.1$ and $\beta=0.3$), while for $\mathcal{E}_{sum}$ it gets ahead on four of the nine datasets and it is only beaten two times. For the remaining three (Coil3, Multiple features-profile correlations and Pendigits), there is at least one memory level for which pifs *k*-Means is outperformed. As $\mathcal{E}_{max}$ is biased towards MinMax *k*-Means and $\mathcal{E}_{sum}$ is optimized by neither algorithm, to get a more meaningful and fair comparison we should focus on MinMax+*k*-Means and pifs+*k*-Means. In this case, $\mathcal{E}_{sum}$ is the most informative measure, since it coincides with the *k*-Means objective, and consistently MinMax+*k*-Means edges ahead (apart from Dermatology when $\beta=0$ or $\beta=0.1$), signifying that the MinMax *k*-Means solutions are of higher quality and thus when fed to *k*-Means improved local optima are attained. In terms of NMI, they are closely matched, each achieving a better score than the other on half of the datasets (Tables 2–10). Note that apart from Ecoli and Dermatology, all other datasets consist of classes of equal size, thus we would expect pifs *k*-Means, which explicitly balances the cluster sizes, to have the upper hand for this metric. Therefore, we can conclude that balancing the variance of the clusters is a more effective strategy.

Table 11
Percentage (%) of MinMax *k*-Means restarts over all nine datasets for which empty or singleton clusters never occur, in relation to the memory level.

Memory level	**Percentage**
$\beta=0$	14.96
$\beta=0.1$	54.37
$\beta=0.3$	91.19

By examining how memory affects the results, the following pattern arises. As the amount of memory grows, a greater reduction of $\mathcal{E}_{max}$ is possible, which is usually accompanied by an increase over $\mathcal{E}_{sum}$ (Tables 2(a)– 10(a)). This can be explained from Table 11, which depicts a remarkable rise on the number of restarts that are free of empty or singleton clusters as memory increases. When no empty or singleton clusters are detected, p reaches p_{max} in our framework and, remember, that for higher p values large variance clusters are heavily punished, while less effort is put into minimizing the sum of the cluster variances. Two datasets severely deviate from the previous pattern, Multiple features-profile correlations and Pendigits, for which the use of memory (especially $\beta=0.1$) yields partitionings of very poor quality. NMI-wise, $\beta=0.1$ seems to be slightly worse than $\beta=0$ and $\beta=0.3$. For MinMax+*k*-Means, the setting where $\beta=0.3$ always displays (apart from Coil2) a better or, at least, equal score for $\mathcal{E}_{max}$, $\mathcal{E}_{sum}$ and NMI than the other two β settings. However, the performance differences between the memory levels for MinMax+*k*-Means are small and, in general, not statistically significant on most datasets. Hence, larger memory seems to only slightly boost efficacy when initializing *k*-Means.

The average execution time per run (in seconds) unveils, as anticipated, that *k*-Means is the fastest method, followed by *k*-Means++, pifs *k*-Means and MinMax *k*-Means. MinMax

k-Means is slower than k-Means by a factor ranging between 3 and 6, depending on the dataset. This higher execution time is a direct consequence of our method requiring more iterations to converge, due to the process employed for adapting p to the data, and also the fact that for some restarts convergence is not achieved, hence t_{max} iterations are performed. Note that t_{max} is set to a high value in the experiments ($t_{max} = 500$). For this reason, the execution time for only those restarts that do converge is also shown (in parentheses) and for Coil3, Multiple features-profile correlations, Pendigits, Ecoli and Dermatology a significant reduction is observed. However, MinMax k-Means is still more time consuming.

Overall, the experimental evaluation has revealed that MinMax k-Means is superior to k-Means, k-Means++ and pifs k-Means, although it incurs a higher computational cost. Importantly, our method guards against large variance clusters and evades poor solutions after bad initializations. Furthermore, it constitutes a sound approach for initializing k-Means. This superior performance has been attained for general p_{max} and p_{step} values that were not tailored to each particular dataset, which greatly enhances the applicability of the presented algorithm. Regarding the use of memory, a higher memory further limits large variance clusters as well as the occurrence of empty or singleton clusters, but increases $\mathcal{E}_{sum}$ and its gains when used to initialize k-Means are small. We could argue that memory is helpful, but not considerably, and even without memory ($\beta = 0$) solutions of very good quality can be obtained. As already discussed, the convergence of MinMax k-Means cannot be theoretically guaranteed. However, for the conducted experiments about 60% of the restarts across all datasets do converge, empirically validating that runs which stop at a local optimum of the relaxed objective (3) are frequently encountered. Finally, a note on the Olivetti dataset, where the compared methods attain identical NMI scores (Tables 8 (a)–(b)): despite the NMI being equal on average, many of the individual restarts exhibit significant differences across the different methods.

6. Conclusions

We have proposed the MinMax k-Means algorithm, a principled approach to circumvent the initialization problem associated with k-Means. Weights are assigned to the clusters in proportion to their variance and a weighted version of the k-Means objective is optimized to restrain large variance clusters from appearing in the solution. A user specified p exponent is utilized to control the strictness of our method over large variance clusters. By punishing large variance clusters, bad initializations can be readily overcome to consistently uncover partitionings of high quality, irrespective of the initial choice of the cluster centers. Additionally, clusters are balanced in terms of their variance, which may prove useful as many data analysis scenarios require groups of roughly the same size. Training involves a min-max problem that is iteratively solved, where the weights are updated in the maximization step to accurately reflect the variances of the clusters at each iteration. Moreover, we have presented a methodology for adjusting the p exponent to the underlying dataset properties, so that the intrinsic group structures can be identified, which greatly facilitates the application of our algorithm.

To draw reliable conclusions, MinMax k-Means was extensively tested on various datasets. Results demonstrate its robustness over bad initializations and its efficacy, as for most cases it outperforms (in terms of clustering quality) all three compared methods, namely k-Means, k-Means++ [8] and pifs k-Means [10]. Furthermore, we noticed that k-Means solutions can be significantly improved when initialized by MinMax k-Means, suggesting an important additional usage of our approach. Overall, MinMax k-Means appears to be a very competitive and easy to employ method for dealing with the sensitivity to initialization of k-Means.

As for future work, we plan to extend MinMax k-Means to kernel-based clustering [2], so that nonlinearly separable clusters can be detected in the data. Also, it would be interesting to explore other possible ways of automatically determining the p value and compare them to the methodology proposed in this paper.

Conflict of interest statement

None declared.

Acknowledgments

This work was supported and co-financed by the European Union (European Regional Development Fund—ERDF) and Greek national funds through the Operational Program “THESSALY—MAINLAND GREECE AND EPIRUS 2007–2013” of the National Strategic Reference Framework (NSRF 2007–2013).

References

[1] R. Xu, D.C. Wunsch II, Survey of clustering algorithms, IEEE Trans. Neural Netw. 16 (3) (2005) 645–678.
[2] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognit. 41 (1) (2008) 176–190.
[3] S.P. Lloyd, Least squares quantization in PCM, IEEE Trans. Inf. Theory 28 (2) (1982) 129–136.
[4] B. Schölkopf, A.J. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (5) (1998) 1299–1319.
[5] I.S. Dhillon, Y. Guan, B. Kulis, Kernel k-means, spectral clustering and normalized cuts, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004, pp. 551–556.
[6] J.M. Peña, J.A. Lozano, P. Larrañaga, An empirical comparison of four initialization methods for the k-means algorithm, Pattern Recognit. Lett. 20 (10) (1999) 1027–1040.
[7] M.E. Celebi, H.A. Kingravi, P.A. Vela, A comparative study of efficient initialization methods for the k-means clustering algorithm, Expert Syst. Appl. 40 (1) (2013) 200–210.
[8] D. Arthur, S. Vassilvitskii, k-Means++: the advantages of careful seeding, in: ACM-SIAM Symposium on Discrete Algorithms (SODA), 2007, pp. 1027–1035.
[9] C.-D. Wang, J.-H. Lai, J.-Y. Zhu, A conscience on-line learning approach for kernel-based clustering, in: International Conference on Data Mining (ICDM), 2010, pp. 531–540.
[10] A. Banerjee, J. Ghosh, Frequency-sensitive competitive learning for scalable balanced clustering on high-dimensional hyperspheres, IEEE Trans. Neural Netw. 15 (3) (2004) 702–719.
[11] P.S. Bradley, U.M. Fayyad, Refining initial points for k-means clustering, in: International Conference on Machine Learning (ICML), 1998, pp. 91–99.
[12] T. Su, J.G. Dy, In search of deterministic methods for initializing k-means and gaussian mixture clustering, Intell. Data Anal. 11 (4) (2007) 319–338.
[13] A. Likas, N.A. Vlassis, J.J. Verbeek, The global k-means clustering algorithm, Pattern Recognit. 36 (2) (2003) 451–461.
[14] A.M. Bagirov, Modified global k-means for minimum sum-of-squares clustering problems, Pattern Recognit. 41 (10) (2008) 3192–3199.
[15] A.M. Bagirov, J. Ugon, D. Webb, Fast modified global k-means algorithm for incremental cluster construction, Pattern Recognit. 44 (4) (2011) 866–876.
[16] G. Tzortzis, A. Likas, The global kernel k-means clustering algorithm, in: International Joint Conference on Neural Networks (IJCNN), 2008, pp. 1977–1984.
[17] G. Tzortzis, A. Likas, The global kernel k-means algorithm for clustering in feature space, IEEE Trans. Neural Netw. 20 (7) (2009) 1181–1194.
[18] H. Zha, X. He, C.H.Q. Ding, M. Gu, H.D. Simon, Spectral relaxation for k-means clustering, in: Advances in Neural Information Processing Systems (NIPS), 2001, pp. 1057–1064.
[19] C.H.Q. Ding, X. He, K-means clustering via principal component analysis, in: International Conference on Machine Learning (ICML), 2004, pp. 225–232.
[20] P.-N. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining, Addison-Wesley, Boston, MA, USA, 2005.
[21] J.Z. Huang, M.K. Ng, H. Rong, Z. Li, Automated variable weighting in k-means type clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (5) (2005) 657–668.
[22] D.S. Modha, W.S. Spangler, Feature weighting in k-means clustering, Mach. Learn. 52 (3) (2003) 217–237.

[23] A. Keller, Fuzzy clustering with outliers, in: International Conference of the North American Fuzzy Information Processing Society (NAFIPS), 2000, pp. 143–147.

[24] C.H.Q. Ding, X. He, H. Zha, M. Gu, H.D. Simon, A min-max cut algorithm for graph partitioning and data clustering, in: International Conference on Data Mining (ICDM), 2001, pp. 107–114.

[25] F. Nie, C.H.Q. Ding, D. Luo, H. Huang, Improved minmax cut graph clustering with nonnegative relaxation, in: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), 2010, pp. 451–466.

[26] S.A. Nene, S.K. Nayar, H. Murase, Columbia Object Image Library (COIL-20), Technical Report CUCS-005-96 1996.

[27] A. Kalogeratos, A. Likas, Dip-means: an incremental clustering method for estimating the number of clusters, in: Advances in Neural Information Processing Systems (NIPS), 2012, pp. 2402–2410.

[28] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.

[29] A. Frank, A. Asuncion, UCI machine learning repository, 2010. URL ⟨http://archive.ics.uci.edu/ml⟩.

[30] B.J. Frey, D. Dueck, Clustering by passing messages between data points, Science 315 (2007) 972–976 ⟨www.psi.toronto.edu/affinitypropagation⟩.

Grigorios Tzortzis received the B.Sc. and M.Sc. degrees in Computer Science from the University of Ioannina, Greece, in 2006 and 2008, respectively. Currently he is a Ph.D. candidate at the Department of Computer Science & Engineering, University of Ioannina, Greece. His research interests include machine learning, pattern recognition, multi-view learning and data mining.

Aristidis Likas received the Diploma degree in Electrical Engineering and the Ph.D. degree in Electrical and Computer Engineering from the National Technical University of Athens, Greece, in 1990 and 1994, respectively. Since 1996, he has been with the Department of Computer Science & Engineering, University of Ioannina, Greece, where he is currently a Professor. His research interests include machine learning, pattern recognition, multimedia, and bioinformatics.