Pattern Recognition Letters 128 (2019) 427–432

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

High-dimensional data clustering by using local affine/convex hulls[☆]

Hakan Cevikalp[*]

Eskisehir Osmangazi University, Electrical and Electronics Engineering, Meselik, Eskisehir, 26480, Turkey

ARTICLE INFO

Article history:
Received 6 April 2019
Revised 9 October 2019
Accepted 10 October 2019
Available online 11 October 2019

MSC:
Clustering
Affine hull
Convex hull

ABSTRACT

In this paper we propose a novel clustering algorithm that uses local affine/convex hulls for high-dimensional data clustering. In high-dimensional spaces, the sparse and irregular distributions make the nearest-neighbor distances unreliable (hole artifact), and this deteriorates the clustering performance. Therefore, there is a need to fill in these gaps between the nearest samples. To this end, we use local affine/convex hulls of the nearest neighbors of a given sample to fill in the holes, and this greatly improves the Euclidean distance metric and the clustering accuracy. The proposed method can also be seen as the local extension of the well-known iterative subspace clustering algorithms in which the entire cluster is approximated with a single linear/affine subspace. Experimental results show that the proposed method is efficient and it outperforms other subspace clustering algorithms on a wide range of datasets.

1. Introduction

Data clustering can be defined as splitting data into clusters based on their similarities. Therefore, this task requires discovering the unknown hidden groups of data samples. Clustering has found wide application areas including machine learning, pattern recognition, computer sciences, medical sciences and economics. However, clustering is a difficult task since it is both data dependent and ill-posed problem. For instance, optimizing clustering criteria like the minimum squared error that is used by many clustering algorithms is theoretically NP-hard. Since it is also data dependent, there is not a single clustering algorithm that can work well for all problems. As a consequence, many clustering algorithms have been proposed and there is a vast number of surveys for analyzing and categorization of clustering algorithms [1,2].

Clustering is initially originated as an unsupervised method which only uses unlabeled data, thus many clustering algorithms have been proposed for this setting [1–4]. However, recently there is a growing interest in application of clustering algorithms for both semi-supervised and supervised applications. In semi-supervised applications, in addition to the unlabeled data, there is also a limited amount of labeled data or some side-information given in terms of similarity/dissimilarity constraints [5,6]. Supervised clustering methods on the other hand use completely labeled data to arrange data for further process such as classification [7,8].

In this paper we focus on unsupervised clustering of high-dimensional data. Clustering of high-dimensional data is more problematic compared to the clustering of low-dimensional data samples. Because, majority of the clustering algorithms rely on pair-wise Euclidean distances between the data samples, and the Euclidean distances become unreliable in high-dimensional spaces because of sparse and irregular distributions of data samples [9–12]. In particular, the authors [10] theoretically show that the distance to the nearest data point approaches to the distance to the farthest data point as the dimensionality increased so that Euclidean distance between pairs of samples becomes meaningless for high-dimensional spaces. This effect makes high-dimensional nearest neighbor based clustering methods erratic, which degrades the performance of clustering methods. In supervised case, this problem is circumvented by approximating each class set with a points set that fills in the sparse regions between examples of the same class. Any convex set containing samples of a class has this property. Both global and local convex sets such as affine/convex hulls [13], bounding hyper-spheres [14], and bounding hyper-disks [12] have been used for this purpose, and these studies report significant improvements over the Euclidean distance. In this study we also use local affine/convex hulls to fill in the sparse regions, but we focus on a more complicated scenario where we do not have any class labels. To this end, we propose an iterative algorithm where we utilize an initial guess of cluster memberships and try to improve the cluster assignments in each iteration.

Related Work: As stated above, the feature space is very sparse in high-dimensional spaces and it is hard to distinguish high density regions from low-density regions which complicates the data clustering problem. Subspace clustering is the most common

[☆] Handled by Associate Editor-Haibin Yan.

[*] Corresponding author.
E-mail address: hakan.cevikalp@gmail.com

https://doi.org/10.1016/j.patrec.2019.10.007
0167-8655/© 2019 Elsevier B.V. All rights reserved.

technique to overcome this limitation. Subspace clustering splits the data samples into groups such that each group contains only data samples lying in the same low-dimensional subspace of the given high-dimensional feature space. This problem has recently received great attention especially in computer vision since many common datasets used for motion segmentation, hand-written recognition and face clustering in different illumination conditions can be modeled by mixture of linear/affine subspaces. As a result, many subspace clustering methods have been proposed. Among these, iterative approaches such as k-subspaces [15], k-means Projective Clustering [16], Median k-Flats [17] alternate between assigning points to linear/affine subspaces and updating subspace parameters based on the newly assigned data points to each subspace.

Statistical approaches such as Mixtures of Probabilistic Principal Component Analysis (MPPCA) [18] and Multi-Stage Learning (MSL) [19] approximate each subspace with a Gaussian distribution and update cluster memberships and Gaussian distribution parameters by using Expectation Maximization (EM) algorithm. There are also algebraic methods such as Generalized PCA [20] and its robust variant [21] which formulate the subspace clustering problem as a high-order polynomial fitting problem. Majority of the subspace methods [22–25] use spectral clustering and these methods differ in the way that how they create the affinity matrix. For example, [22,25] use sparse combination coefficients to create affinity matrix, [24] creates an affinity matrix using similarities between local linear subspaces, and [23] uses low-rank representation for constructing affinity matrix [26] creates many local best-fit affine subspaces and apply a greedy selection algorithm to select the ones that best fit to the given data. Interested reader is referred to survey [3] for more information on subspace clustering.

Our proposed method can be seen as similar to [22,24–26] in the sense that we compute the distances based on the nearest affine/convex reconstruction weights rather than Euclidean distances. However, our method differs from these methods in a way that we use different local models (as opposed to the linear subspaces or sparse representation) and we iteratively update each cluster, thus the similarities/distances may change in each iterations (however all other methods compute the similarities only once at the beginning and use it in spectral clustering). Therefore, our method can be seen as a variant of k-means clustering algorithm where the nearest mean distance is interchanged with the nearest local affine/convex hull distances.

In addition, motivated by the great success of deep neural networks in supervised classification, there are many attempts to adopt deep learning for unsupervised clustering [27–29]. For example, [28] proposes Deep Embedded clustering method that simultaneously learns feature representation and cluster assignments using deep neural networks. To this end, the authors integrate KL divergence loss into a deep autoencoder network. In [29], the same network is used together with additional reconstruction loss to improve the clustering accuracy. Caron et al. [27] introduces a very simple iterative network where a convolutional neural network (CNN) is used to extract features and k-means clustering algorithm is used to assign pseudo labels to the samples. Then, the network is re-trained by using the pseudo labels returned by the k-means clustering algorithm and this procedure is repeated until convergence. More information on clustering methods using deep neural networks can be found in the survey paper of [30].

2. Method

Similar to local best-fit flats clustering [26], k-means Projective Clustering and k-subspaces clustering algorithms, we also need pre-defined clusters before running the proposed algorithm. To initialize the algorithm, we can use any clustering method such as k-means clustering or k-means Projective Clustering. In our experiments, we always initialized our algorithm with k-means Projective Clustering, and k-means Projective Clustering method is initialized with k-means clustering method. For each data point in the dataset, we compute the nearest m neighbors from each cluster and compute the distances from that point to the affine/convex hulls of those neighbors. Then, the data point is assigned to the cluster yielding the minimum affine/convex hull distance. This procedure is repeated until the convergence. The algorithm of the proposed clustering method is summarized in Algorithm 1. Using local affine/convex hulls for distance computation rather than using classical Euclidean distances greatly improves the results. Fig. 1 illustrates the visual comparison of Euclidean distance to the local affine/convex hull based distances. We define how to compute affine and convex hull distances below.

Algorithm 1 Local Affine/Convex Hull Clustering Algorithm.

```
Initialize
Initial Clusters $S_c = \{\mathbf{X}_c\}_{c=1}^{K}$, $\mathbf{D}$ the distance matrix including pairwise distances, $n$ is the number of all data samples, $m$ is the number of the nearest samples, $T$ is the maximum iteration number
Description:
  for $t \in 1, \dots, T$ do
    for $i \in$ randperm($n$) do
      – $\mathbf{x}_i = \mathbf{X}(:, i)$;
      for $c \in 1, \dots, K$ do
        – Determine the $m$-nearest samples of $\mathbf{x}_i$ from cluster $c$ by using distance matrix $\mathbf{D}$;
        – Find the affine/convex hull distances using equations (4)/(6);
      end for
      – Assign $\mathbf{x}_i$ to the cluster yielding the smallest distance;
    end for
    if the assignments do not change, break;
  end for
```

2.1. Computing the affine/convex hull distances

Affine hull (affine subspace) of m-nearest point set $\{\mathbf{x}_{c1}, \dots, \mathbf{x}_{cm}\}$ of a data point $\mathbf{x}$ belonging to cluster c is defined as,

$$H_c^{affine}(\mathbf{x}) = \left\{ \mathbf{p} \middle| \mathbf{p} = \sum_{i=1}^{m} \alpha_{ci}\mathbf{x}_{ci}, \ \sum_{i=1}^{m} \alpha_{ci} = 1, \ \alpha_{ci} \in \mathbb{R} \right\}. \qquad (1)$$

By choosing a reference point $\boldsymbol{\mu}_c$ (such as mean, $\boldsymbol{\mu}_c = (1/m)\sum_{i=1}^{m}\mathbf{x}_{ci}$) from the point set, the local affine hull of the nearest neighbors can also be written as

$$H_c^{affine}(\mathbf{x}) = \left\{ \mathbf{p} \middle| \mathbf{p} = \boldsymbol{\mu}_c + \mathbf{U}_c\mathbf{v}_c, \ \mathbf{v}_c \in \mathbb{R}^l \right\}. \qquad (2)$$

Here, $\mathbf{U}_c$ is a local orthonormal basis for the directions spanned by affine subspace and $\mathbf{v}_c$ is a vector of free parameters that determines the coordinates for the points within the subspace, expressed with respect to the basis $\mathbf{U}_c$. Numerically $\mathbf{U}_c$ is obtained by applying the Singular Value Decomposition (SVD) to $[\mathbf{x}_{c1} - \boldsymbol{\mu}_c \dots, \mathbf{x}_{cm} - \boldsymbol{\mu}_c]$ and l is the dimension of the basis. In order to assign a data point $\mathbf{x}$ to a cluster, we need to find the minimum distances between data point and local linear affine hulls of each cluster. Then, the data point will assigned to the cluster whose local affine hull is closest to $\mathbf{x}$. The minimum distance between $\mathbf{x}$ and a local linear affine hull is computed as

$$d(\mathbf{x}, H_c^{affine}(\mathbf{x})) = \arg\min_{\mathbf{p} \in H_c^{affine}(\mathbf{x})} \|\mathbf{x} - \mathbf{p}\| = \arg\min_{\mathbf{v}_c} \|\mathbf{x} - \boldsymbol{\mu}_c - \mathbf{U}_c\mathbf{v}_c\|, \quad c = 1, \dots, K. \qquad (3)$$

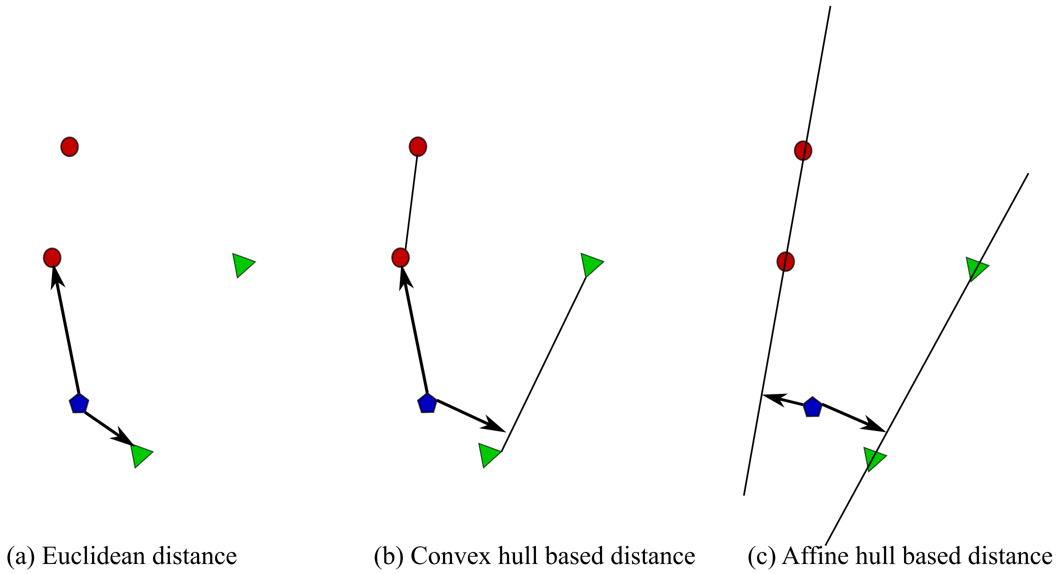
(a) Euclidean distance
(b) Convex hull based distance
(c) Affine hull based distance

Fig. 1. Visual comparison of distances: Nearest distances based on (a) Euclidean distance, (b) convex hull distance, (c) affine hull distance. The closest distance from any sample to an affine/convex hull is the norm of displacement from that sample to the closest point on the hull. Observe how the distances change by each selected method.

The minimization of above optimization problem leads to $\mathbf{v}_c = \mathbf{U}_c(\mathbf{x} - \boldsymbol{\mu}_c)$. It should be noted that $\mathbf{P}_c = \mathbf{U}_c\mathbf{U}_c^\top$ defines orthogonal projection operator onto the difference subspace of local neighbors. Therefore, the final distance becomes

$$d(\mathbf{x}, H_c^{affine}(\mathbf{x})) = \|(\mathbf{I} - \mathbf{P})(\mathbf{x} - \boldsymbol{\mu}_c)\|, \tag{4}$$

where $\mathbf{I}$ is the identity matrix. See [13] for more details.

For convex hulls, the convex hull of a point set is the convex span of the samples in that point set and it can be obtained by adding an additional nonnegativity constraint to (1) as,

$$H_c^{convex}(\mathbf{x}) = \left\{ \mathbf{p} | \mathbf{p} = \sum_{i=1}^{m} \alpha_{ci}\mathbf{x}_{ci}, \ \sum_{i=1}^{m} \alpha_{ci} = 1, \ \alpha_{ci} \geq 0 \right\}. \tag{5}$$

Finding the distance from a point to the convex hull of a point set requires the solution of the following quadratic programming problem

$$\begin{aligned} &\arg\min_{\boldsymbol{\alpha}_c} \ \frac{1}{2}\|\mathbf{x} - \mathbf{X}_c\boldsymbol{\alpha}_c\|^2 \\ &\quad \text{s.t.} \ \sum_{i=1}^{m} \alpha_{ci} = 1, \ \alpha_{ci} \geq 0, \end{aligned} \tag{6}$$

where $\mathbf{X}_c$ is the matrix whose columns are the nearest neighbors of point $\mathbf{x}$ coming from the cluster c. Once the optimal coefficient vector $\boldsymbol{\alpha}_c^*$ is found, $\|\mathbf{x} - \mathbf{X}_c\boldsymbol{\alpha}_c^*\|$ determines the minimum distance from point $\mathbf{x}$ to the local convex hull of the cluster c.

2.2. Algorithm complexity

The proposed method requires finding the nearest neighbors from all clusters for each sample in the data set. Computation of the nearest neighbors in each iteration is computationally expensive, thus we compute all pair-wise distances at the beginning of the algorithm and save them in a distance matrix, **D**. During finding the nearest neighbors, we sort the pre-computed distances and determine the nearest neighbors efficiently. Computational complexity of finding distances from each sample to the local affine/convex hulls is related to $O(m^2)$, where m is the number of the nearest neighbors and it must be set by the user. For small values of m, the training complexity will be very low, but for higher values the complexity can be high. In our experiments, to determine the value of m, we tried $m = 5, 10, 15, 20$ values on a validation set and fixed m to the value yielding the highest accuracy.

3. Experiments

We tested the proposed clustering algorithm, Local Subspace Clustering (LSC) using local affine and convex hulls, on four different datasets: Hopkins 155, Cifar10, ESOGU Videos, and MNIST. We compared the proposed method to k-means clustering and other state-of-the-art subspace clustering algorithms including k-means Projective Clustering (PC) [16], Sparse Subspace Clustering (SSC) [22], Sparse Manifold Clustering (SMC) [25], Robust Subspace Clustering by Low-Rank Representation (LRR) [23], Ordered Subspace Clustering (OSC) [31]. We use k-means clustering algorithm to initialize PC, and the result of PC is used to initialize the other subspace clustering algorithms including the proposed one that needs initialization. To measure the clustering accuracy, we used the classification accuracy which is adopted by other subspace clustering papers. In addition, we also use the most common metric F-measure, which is widely used to evaluate clustering results. F-measure is the harmonic mean of the pair-wise precision and recall measures. We compute precision and recall over pair of samples and consider for the samples whether they are assigned to the same cluster and whether they contain the same class membership based on the ground-truth data labels. Let A denote the set of images assigned to the same cluster and let B denote the set of images that contain the same class category. With $|A|$ denoting the cardinality of A, the measures are defined as:

$$\text{Precision} = \frac{|A \cap B|}{|A|}, \ \text{Recall} = \frac{|A \cap B|}{|B|},$$

$$\text{F-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

3.1. Experiments on MNIST digit database

The MNIST digit dataset consists of 70 K handwritten digit samples, each of size 28 × 28 pixels. For this dataset, 60 K samples are allocated for training and the remaining 10 K samples are reserved for test. We applied clustering algorithms to the training and test

Table 1
Classification rates (CR) (%) and F-scores on MNIST digit database.

Methods	Training data		Test data	
	CR	F-score	CR	F-score
LSC-aff. hull	**89.9**	**83.8**	**95.7**	**91.7**
LSC-conv. hull	89.7	83.5	95.5	91.3
k-means	62.2	49.2	60.7	47.4
PC	87.5	79.7	94.8	90.0
SSC	OOM	OOM	-	-
SMC	-	-	61.4	58.6
LRR	-	-	11.4	18.2
OSC	OOM	OOM	11.2	21.6
DEC ([28])	86.6	-	82.4	-
DCEC ([29])	**89.0**	-	**85.3**	-

Table 2
Classification rates (CR) (%) and F-scores on the Hopkins 155 database.

Methods	CR	F-score
LSC-affine hull	91.92	89.26
LSC-convex hull	91.26	88.56
k-means	79.68	76.11
PC	91.13	88.33
SSC	**96.78**	**96.08**
SMC	82.03	78.43
LRR	94.73	92.86
OSC	80.79	76.70

sets separately since some clustering methods had memory problems for 60K training data.

The accuracies are given in Table 1. In addition to the subspace methods, we also give accuracies of the clustering methods using deep neural networks at the bottom of the table. Majority of the tested subspace clustering algorithms either fail to converge or have memory problems on 60K training data. The proposed clustering method using local affine hulls achieves the best accuracy in all cases. All other state-of-the-art subspace clustering methods SMC, LRR and OSC return very poor results compared to our proposed method, which clearly demonstrates the superiority of the proposed clustering algorithm. The proposed methods also outperform recent state-of-the-art deep clustering methods on both training and test data. Especially, the performance difference is very significant on the test data.

3.2. Experiments on the Hopkins 155 database

The Hopkins 155 dataset includes 156 sequences of two or three motions where each motion corresponds to a subspace. Each sequence is a sole clustering task, therefore there are 156 clustering tasks in total for this dataset. The data are slightly corrupted by noise, but do not have any missing entries or outliers. The classification accuracies and F scores for all 156 sequences are given in Table 2. SSC method achieves the highest accuracy followed by LRR and the proposed clustering methods. *k*-means clustering is the worst performing method.

3.3. Experiments on the Cifar10 database

Cifar10 dataset includes 60K 32×32 small images of 10 objects: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. 50K samples are used as training and they are split into 5 batches whereas the remaining 10K samples are used for testing. We used 4096 dimensional CNN features. To extract CNN features, all images are first resized to 256×256 and then we used Caffe [32] implementation of the CNN described by Krizhevsky et al. [33] by using the identical setting used for ILSVRC 2012 classification with the exception that the base learning rate was set to 0.001.

In our experiments we combined all samples in each batch with the test samples (20K samples in total) and conducted clustering for each batch separately. The results are given in Table 3 are quite mixed. The SSC method that performed best on the Hopkins dataset did not return a solution in 4 weeks, which clearly demonstrates that this method does not scale well with the data set size. Similarly, OSC method produced “out of memory” (OOM) problem although we use a very powerful workstation with 256 GB RAM for conducting experiments. The best accuracies are achieved by *k*-means, SMC and the proposed clustering method using convex hulls. LRR method is the worst performing method.

3.4. Experiments on ESOGU face videos

ESOGU Face Videos [34] includes videos of 285 people captured in two sessions separated by at least three weeks. Four short videos were captured under four different scenarios in each session. We used the videos recorded with free head movements under normal illumination conditions as shown in Fig. 2. Face images are manually cropped from videos and they are resized to 120×90 pixels. There are many proposed descriptors for extracting features from face images [35–38]. In our experiments, in addition to the classical LBP (local binary patterns) features [38], we also used recently proposed context-aware local binary features [35] for extracting features from face images.

For clustering, we use face images of randomly chosen 5 people. This is repeated 5 times and the resulting accuracies for LBP and context-aware local binary features are given in Table 4. As in the previous case, SSC failed to converge. For LBP features, SMC, PC, and the proposed clustering methods achieve the best accuracies whereas SMC, OSC and the proposed methods yield the best accuracies for context-aware local binary features. For LBP features, SMC significantly outperforms all other clustering methods for trials 1 and 2, but perform badly on trials 3 and 4. PC and the proposed methods yield similar accuracies. LRR and OSC methods return very poor accuracies on LBP features. For context-aware local binary features, *k*-means clustering returns very poor results since

Table 3
Classification rates (CR) (%) and F-scores on Cifar10 database.

Methods	Batch 1		Batch 2		Batch 3		Batch 4		Batch 5	
	CR	F-score	CR	F-score	CR	F-score	CR	F-score	CR	F-score
LSC-aff. hull	86.0	72.8	86.5	73.8	85.7	72.4	85.8	72.7	85.1	71.6
LSC-conv. hull	85.1	71.5	85.3	71.3	85.5	72.4	85.6	72.5	**85.5**	**72.4**
k-means	**86.1**	**73.5**	**86.7**	74.4	**86.0**	73.4	**86.2**	**73.8**	85.3	72.3
PC	84.5	70.6	85.1	71.8	84.8	71.4	84.7	71.2	83.9	69.9
SSC	-	-	-	-	-	-	-	-	-	-
SMC	85.8	73.4	86.6	**74.6**	**86.0**	**73.7**	86.0	73.7	85.1	72.3
LRR	75.2	60.1	76.4	61.1	70.5	60.5	75.6	60.1	74.4	58.6
OSC	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM

Table 4
Classification rates (CR) (%) and F-scores on ESOGU face videos database.

Methods	LBP Features									
	Trial 1		Trial 2		Trial 3		Trial 4		Trial 5	
	CR	F-score	CR	F-score	CR	F-score	CR	F-score	CR	F-score
LSC-aff. hull	83.4	72.7	83.3	77.5	**83.3**	**73.1**	89.2	79.7	**90.1**	**83.6**
LSC-conv. hull	83.3	72.7	83.4	77.6	83.2	**73.1**	89.2	79.7	**90.1**	**83.6**
k-means	83.1	70.1	66.3	59.7	63.7	66.3	57.8	49.6	70.2	76.1
PC	83.4	72.7	83.5	77.6	**83.3**	**73.1**	**89.3**	**79.9**	89.9	83.3
SSC	-	-	-	-	-	-	-	-	-	-
SMC	**97.2**	**94.8**	**96.1**	**94.6**	72.4	59.0	69.5	74.6	84.9	83.3
LRR	43.0	37.4	52.9	53.1	42.7	44.0	46.4	46.3	49.4	54.3
OSC	54.0	43.2	73.0	76.3	58.3	58.2	55.1	44.0	64.0	58.5
	Context aware local binary features									
	Trial 1		Trial 2		Trial 3		Trial 4		Trial 5	
	CR	F-score	CR	F-score	CR	F-score	CR	F-score	CR	F-score
LSC-aff. hull	92.4	91.5	87.7	87.4	85.2	84.6	**100**	**100**	86.5	85.5
LSC-conv. hull	92.4	91.5	87.5	87.2	85.7	84.8	**100**	**100**	86.4	85.2
k-means	72.4	65.5	83.8	82.1	83.0	82.3	71.9	67.9	76.9	69.7
PC	83.0	81.7	83.8	82.6	89.8	83.7	95.2	94.5	80.0	74.3
SSC	-	-	-	-	-	-	-	-	-	-
SMC	**99.4**	**99.2**	**99.0**	**98.2**	99.8	99.0	83.4	78.6	85.0	84.7
LRR	98.0	97.2	68.4	65.0	79.9	73.5	92.1	91.4	85.0	84.7
OSC	**99.4**	**99.2**	**99.0**	**98.2**	**99.9**	**99.0**	**100**	**100**	**92.4**	**92.3**

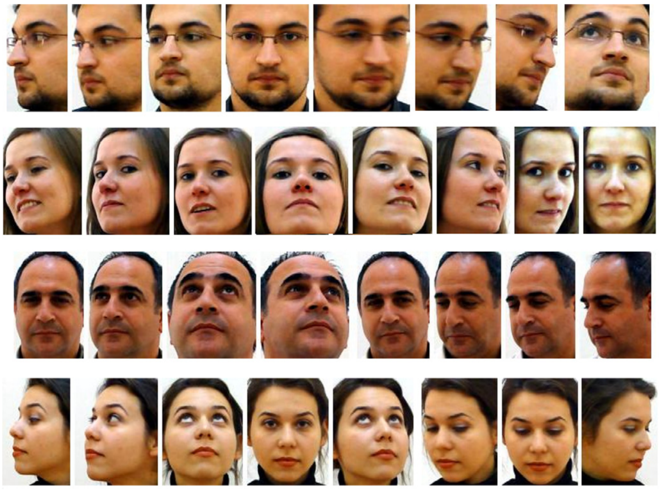

Fig. 2. Some image frames extracted from videos.

the dimension of the feature space is very high (i.e., the dimensionality is 32 K). However, these features are more discriminative compared to LBPs, and the most tested subspace methods as well as the proposed methods achieve better accuracies compared to the ones obtained for LBP features. Especially, poorly performing LRR and OSC methods on LBP features, return very high accuracies on context-aware features.

3.5. Comparison of the speed of the algorithms

Table 5
Test times of the clustering methods.

Methods	Time (s)
LSC-affine hull	5.6
LSC-convex hull	37.2
k-means	0.3

We also conducted experiments to compare the speed of the proposed algorithms with the *k*-means clustering algorithm. To this end, we created 5 classes including 200 examples per class in 100-dimensional space. All classes have the same covariance structure but their centers are shifted for separation (each class is respectively shifted as −5, −2.5, 0, 2.5 and 5). As stated earlier, computational complexity of the proposed methods largely depends on the number of the nearest neighbors, *m*. In all experiments, we set it to a moderate value of 15. For a fair comparison with the *k*-nearest mean clustering, we started our algorithms completely from random initializations (It should be noted that this is generally not a good idea since the proposed methods are very sensitive to the initialization. This is done only for fair comparison in these tests).

We repeated the experiments 10 times and the final test times are the averages over these 10 trials. The test times are given in Table 5. Since the proposed algorithms are started from random initializations, they are slow compared to *k*-means clustering. The proposed affine hull based clustering is approximately 7 times faster than the proposed convex hull based clustering. This is expected since computing distances from samples to the affine hulls requires simple multiplications whereas small size quadratic programming problems must be solved for computing distances to the convex hulls.

4. Conclusion

In this paper, we introduced a novel method for high-dimensional data clustering. The proposed method uses local affine/convex hulls constructed from the nearest neighbors of samples, and cluster assignment of each sample is made based on the shortest distances to these local hulls. This greatly improves the clustering accuracy of the methods using pair-wise Euclidean distances. Experimental results demonstrate several facts: First of all, there is not a single clustering algorithm that works well for all problems, and the results are mostly data dependent. We observed that the best performing methods change for even different partitions of the same type of data. Yet, the proposed clustering methods perform well on most of the tested datasets and achieve the best accuracies or comparable results to the best performing method. We also realized that majority of the state-of-the-art subspace clustering methods do not scale well with data set size and fail to converge for even moderate sized datasets. Furthermore, some methods had insufficient memory issues for most of the datasets. In contrast, the proposed method did not have any of these problems.

Declaration of Competing Interest

The authors declare that they do not have any financial or non-financial conflict of interests.

References

[1] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognit. Lett. 31 (2010) 651–666.
[2] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A.Y. Zomaya, S. Foufou, A. Bouras, A survey of clustering algorithms for big data: taxonomy and empirical analysis, IEEE Trans. Emerg. Top. Comput. 2 (2014) 267–279.
[3] R. Vidal, Subspace clustering, IEEE Signal Process. Mag. 28 (2011) 52–68.
[4] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Trans. Neural Netw. 16 (2005) 645–678.
[5] M. Bilenko, S. Basu, R.J. Mooney, Integrating constraints and metric learning in semi supervised clustering, ICML, 2004.
[6] E.P. Xing, A.Y. Ng, M.I. Jordan, S. Russell, Distance metric learning with application to clustering with side-information, NIPS, 2003.
[7] H. Cevikalp, New clustering algorithms for the support vector machine based hierarchical classification, Pattern Recognit. Lett. 31 (2010) 1285–1291.
[8] V. Vural, J.G. Dy, A hierarchical method for multi-class support vector machines, ICML, 2004.
[9] A. Hinneburg, C.C. Aggarwal, D.A. Keim, What is the nearest neighbor in high dimensional spaces? Proceedings of the 26th International Conference on Very Large DataBases, 2000.
[10] K. Beyer, J. Goldstein, R. Ramakrishnan, U. Shaft, When is nearest neighbor meaningful, Lect. Notes Comput. Sci. 1540 (1999) 217–235.
[11] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data, Data Min. Knowl. Discov. 1 (2005) 5–33.
[12] H. Cevikalp, B. Triggs, R. Polikar, Nearest hyperdisk methods for high-dimensional classification, ICML, 2008.
[13] H. Cevikalp, D. Larlus, M. Neamtu, B. Triggs, F. Jurie, Manifold based local classifiers: linear and nonlinear approaches, J. Signal Process. Syst. 61 (2010) 61–73.
[14] D. Tax, R. Duin, Support vector data description, Mach. Learn. 54 (2004) 45–66.
[15] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, D. Kriegman, Clustering appearances of objects under varying illumination conditions, CVPR, 2003.
[16] P.K. Agarwal, N.H. Mustafa, K-means projective clustering, Proceedings of the Twenty-Third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2004.
[17] T. Zhang, A. Szlam, G. Lerman, Median *k*-flats for hybrid linear modeling with many outliers, ICCV Workhops, 2009.
[18] M. Tipping, C. Bishop, Mixtures of probabilistic principal component analyzers, Neural Comput. 11 (1999) 443–482.
[19] A. Gruber, Y. Weiss, Multibody factorization with uncertainty and missing data using the em algorithm, CVPR, 2004.
[20] R. Vidal, Y. Ma, S. Sastry, Generalized principal component analysis (gpca), IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005) 1–15.
[21] A.Y. Yang, S.R. Rao, Y. Ma, Robust statistical estimation and segmentation of multiple subspaces, CVPR Workshops, 2006.
[22] E. Elhamifar, R. Vidal, Sparse subspace clustering, CVPR, 2009.
[23] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, ICML, 2010.
[24] J. Yan, M. Pollefeys, A general framework for motion segmentation: independent, articulated, rigid, non-rigid, degenerate and non-degenerate, European Conference on Computer Vision, 2006.
[25] E. Elhamifar, R. Vidal, Sparse manifold clustering and embedding, NIPS, 2011.
[26] T. Zhang, A. Szlam, Y. Wang, G. Lerman, Hybrid linear modeling via local best-fit flats, Int. J. Comput. Vis. 1000 (2012) 217–240.
[27] M. Caron, P. Bojanowski, A. Joulin, M. Douze, Deep clustering for unsupervised learning of visual features, European Conference on Computer Vision (ECCV), 2018.
[28] J. Xie, R. Girshick, A. Farhadi, Unsupervised deep embedding for clustering analysis, ICML, 2016.
[29] X. Guo, X. Liu, E. Zhu, J. Yin, Deep clustering with convolutional autoencoders, International Conference on Neural Information Processing, 2017.
[30] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, J. Long, A survey of clustering with deep learning: from the perspective of network architecture, IEEE Access 6 (2018) 39501–39514.
[31] S. Tierney, J. Gao, Y. Guo, Subspace clustering for sequential data, CVPR, 2014.
[32] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: convolutional architecture for fast feature embedding, (2014). arXiv:1408.5093.
[33] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, NIPS, 2012.
[34] M. Yalcin, H. Cevikalp, H.S. Yavuz, Towards large-scale face recognition based on videos, ICCV Workhsops, 2015.
[35] Y. Duan, J. Lu, J. Feng, J. Zhou, Context-aware local binary feature learning for face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 40 (2018) 1139–1153.
[36] Y. Duan, J. Lu, Z. Wang, J. Feng, J. Zhou, Learning deep binary descriptor with multi-quantization, IEEE Trans. Pattern Anal. Mach. Intell. 41 (2019) 1924–1938.
[37] J. Lu, V.E. Liong, J. Zhou, Simultaneous local binary feaure learning and encoding for homogeneous and heterogeneous face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 40 (2018) 1979–1993.
[38] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary patterns: application to face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 28 (2006) 2037–2041.