Combining Feature Extraction and Clustering for Better Face Recognition

Salim Afra and Reda Alhajj

Abstract In this paper, we study the performance of face clustering approaches using different feature extraction techniques. This study will highlight best practices for handling faces of terrorists and criminals in an approach which we are working on to trace and red flag potential cases. Given as input images containing faces of people, face clustering divides them into K groups/clusters with each group containing images expected to represent almost the same person. Face clustering is very important, especially in forensic investigations where millions of images are available in crime scenes to be investigated. We study the performance of face clustering by first choosing different feature extraction techniques to capture information from faces. Feature extraction techniques are employed to check which face representation works better in describing faces as input to clustering algorithms. We also used Rank Order clustering algorithm which is known for its good accuracy when clustering face images along with other traditional clustering techniques. We evaluated the performance of feature extraction techniques and clustering algorithms using four datasets (JAFFE, AT&T, LFW, and YaleB); each imposing different challenges for face clustering with varying image environment and for datasets of different sizes. These datasets challenge clustering algorithms and feature extraction techniques in run time and clustering accuracy. Experimental results show the effectiveness of Rank Order clustering in terms of accuracy for small datasets while its run time performance degrades for larger datasets. K-means performed poorly on the LFW dataset. OpenFace performed the best in describing face images, especially on large datasets compared to other feature extraction techniques. The latter method reported high accuracy margin that is big and acceptable feature extraction time.

Keywords Face clustering · Face recognition · Feature extraction · Image processing

S. Afra · R. Alhajj (✉)
Department of Computer Science, University of Calgary, Calgary, AB, Canada
e-mail: salim.afra@ucalgary.ca; alhajj@ucalgary.ca

M. Kaya et al. (eds.), *Social Network Based Big Data Analysis and Applications*, Lecture Notes in Social Networks, https://doi.org/10.1007/978-3-319-78196-9_11

1 Introduction

Face recognition involves detecting and verifying persons' identity by processing digital images and frames extracted from videos. Face recognition systems are becoming more popular due to rapidly advancing technology which made it affordable to capture and store a large number of images at low cost. They have various applications and benefits, including homeland security where video surveillance systems detect and recognize criminals or intruders. Video surveillance systems that are able to recognize people from a captured video stream are becoming more important, especially with incidents related to crimes, for example, the Boston marathon attack [12]. In such incidents, thousands of images are collected by video surveillance cameras and then inspectors analyze faces residing inside frames.

Clustering of people plays an important role during the investigation of crimes. In crowded areas, a large number of persons may pass in a specific location where a video surveillance system will keep on capturing image frames which can be in the order of millions. The same person may appear hundreds of times in frames which are not necessarily all consecutive. Thus, clustering of people will be perfect to apply for the following two main reasons:

- Filtering Data: By excluding images where no person is detected in surveillance camera frames.These images should be discarded during the investigation. The remaining images will be trimmed to concentrate only on persons appearing in frames, mainly their faces.
- Organizing Data: Here images of the same person in different location scenes will be identified to belong to the same cluster. This way, an investigator interested in tracking a specific person will only concentrate on a specific cluster and may proceed to identify and investigate other related suspects.

In order to cluster people in video frames, we use face as the identifier because a face is the most distinctive key to person's identity [5]. Clustering faces is a challenging process and dependable not only on the clustering algorithm invoked, but also on the feature extraction technique used. Both have several challenges to cope with. Feature representation challenges are inherited from limitations of visual features due to several factors, including low resolution of face images, changes in illumination between images, capturing a person from different viewpoints, cluttered background, etc., while clustering challenges may be attributed to the fact that the expected number of people in an input frame is not known in advance. This may cause a problem for some clustering techniques, mainly those which require a number of clusters as input. Another issue is that the number of images of different people is unbalanced. For instance, some people may appear in a few frames while others may exist in many frames; this aspect is challenging for some clustering techniques as discussed in Sect. 3.3.

Face recognition has recently received considerable attention as evident by the number of face recognition algorithms described in the literature. However, clustering of face images has not received enough attention yet. As a result, existing literature lacks on efforts which investigate an appropriate match between feature extraction techniques and clustering algorithms. Motivated by this, the work described in this paper evaluates the performance of various feature extraction and clustering techniques using a number of datasets of face images. By doing so, we seek to have a better idea on which feature extraction technique works better with a given clustering algorithm with respect to time and clustering accuracy.

The rest of the paper is organized as follows. Section 2 discusses existing literature related to face clustering. Section 3 describes the methodology used in face clustering along with feature extraction and clustering algorithms to be used in performance evaluation. Section 4 presents the datasets used in the evaluation. Section 5 includes the experiments and results. Section 6 is conclusions.

2 Related Work

Clustering has been applied in pattern recognition and has been successfully used in many different fields. Face clustering analysis has not received much attention as the clustering of faces depends not only on the clustering algorithm used but also on the feature extraction technique invoked. Different feature extraction techniques described in the literature can be applied as a preprocessing step for face clustering, but there is no widely accepted feature representation technique.

Several studies (e.g., [9, 24]) have evaluated the performance of a feature representation technique in association with a single clustering algorithm. For instance, Ho et al. [9] used Spectral clustering to evaluate the performance of local gradients with pixel intensity as a feature vector for face representation. Zhao et al. [24] used Hierarchical clustering to cluster photos in a personal gallery. They used a combination of features to represent a face image based on information extracted from face, body, and context information. Zhu et al. presented a new clustering algorithm specifically for face clustering [26]; it is called Rank Order distance clustering. Clustering is achieved by measuring the dissimilarity between two faces based on their neighborhood information. This is one of the clustering algorithms evaluated in this study. It is described in detail in Sect. 3.3.

Other studied investigated the effect of using feature representation techniques in combination with clustering algorithms. For instance, Heisele et al. [8] classified faces using Support Vector Machine (SVM) to evaluate three different feature representation techniques, namely, component-based method and two global methods for face recognition. In their evaluation, they used ROC curves of these feature representation techniques for formal and rotated faces. They determined that the component-based method outperformed the two global methods. While in [19] they present analysis similar to our study by checking the performance of different feature extraction techniques and clustering algorithms. They used component-based features and compared with a commercial face matcher. After extracting features from a face, they then apply three different clustering algorithms, namely, K-means, Spectral clustering, and Rank Order distance, which we have used in this study. They used two datasets for their experiments, namely, Pinellas County Sheriff's Office (PSCO) and Labeled Faces in the Wild (LFW) dataset. Their results show that the commercial face matcher outperformed the component-based for Rank Order clustering but they could not run the commercial face matcher on K-means or Spectral clustering because the feature vectors are not provided by the commercial product. They also show that Rank Order clustering performs better than K-means and Spectral clustering.

3 Methodology

As shown in Fig. 1, the general methodology of clustering faces consists of four main stages. The first step is to acquire a face dataset from any appropriate source which may be a video surveillance camera. The datasets we used for this purpose are mentioned in Sect. 4. After attaining the image collection, the next step is to preprocess and filter the images to concentrate only on faces of people. Then, feature extraction techniques are applied on the processed faces to get a feature vector of each image/face. The last step is to cluster the extracted feature vectors. More detail on each step is explained below.

3.1 Preprocessing

Face preprocessing of input images involves three major steps as depicted in Fig. 2.

1. **Detect Face Region:** The first step is to apply face detection over the image in order to get the face region of a person. By applying face detection, we eliminate extra details in the image and focus only on the face of the person. We have used dlib's implementation [11] of the Histograms of Oriented Gradients (HOG) face detection method as described in [6]. The output of this method is a face image of size 96×96 pixels.
2. **Face Landmark Detection:** After the face region is extracted, we compute face landmarks as shown in Fig. 2. We have used dlib's implementation of face pose estimation as presented in [10]. In their work, they have made an ensemble of

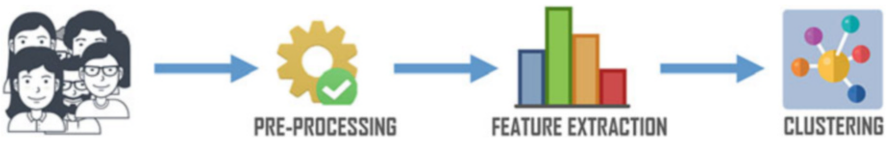

Fig. 1 Face clustering overall methodology

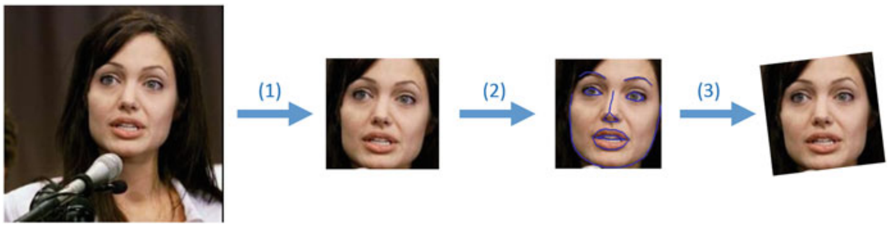

Fig. 2 Preprocessing steps example

regression trees to estimate landmark positions of a face from an image. They achieved high quality and fast predictions. The output from this method is 128 points that represent head pose.

3. **Face Alignment:** The last step performed is to align the head position straight with no rotation while keeping same eye, nose, and mouth position for all images. This step is important so that all faces are properly aligned because a slight variation in face alignment would be enough to trigger a false positive match with another person in the dataset. A face is aligned by using landmarks detected to put the eyes, mouth, and nose at a similar location for every image so that the features extracted for every face will have almost the same face position. This is done by doing affine transformation of faces with the help of landmarks to normalize and align faces at the same position.

3.2 Feature Extraction

After completing the preprocessing stage, face images become ready to extract their features. Extracting features of an image corresponds to building a feature vector which represents its important pixel information. These feature vectors are to be used in the clustering process. There are several feature extraction techniques that can be applied to the face. Currently, the top feature extraction technique is the one based on convolutional neural networks developed by Google's FaceNet [22]. In their work, they use up to 200 million private images of people to train a deep neural network to learn a feature vector of a face image and map it to a compact Euclidean space. Using this method, the similarity measure between two faces is simply the squared L2 distance between the two images.

In our analysis, we chose the following image feature extraction techniques. Then, we study the effect of using each of these feature extraction techniques with the clustering models.

- **SIFT** [15]: Scale-invariant feature transform (SIFT) method was developed by D. Lowe in 2004. SIFT extracts key-points of an image and then computes its descriptors. The algorithm to detect key-points involves four major steps: Scale-

space extreme detection, key-point localization, orientation assignment, and key-point descriptor generation. Scale-space is found using an approximate Laplacian of Gaussian (LoG) with difference of Gaussian.
- **SURF** [4]: Speeded-Up Robust Features (SURF) method came out in 2006 as a speeded-up version of the SIFT algorithm. It does its speedup by using approximation algorithms to improve every step of the SIFT algorithm.
- **BRISK** [14]: Binary Robust Invariant Scalable Key points (BRISK) was developed in 2011 to make feature extraction effective and faster than previous methods such as SIFT and SURF. BRISK samples pattern out of concentric rings and then apply Gaussian smoothing. Building the descriptor is done by performing intensity comparisons.
- **DAISY** [23]: This method was developed in 2010. It depends on histograms of gradients like SIFT for key-point descriptor, but also uses a Gaussian weighting and circularly symmetrical kernel.
- **KAZE** [2]: Developed in 2012, this method analyzes and describes an image by operating in a nonlinear-scale space. The nonlinear-scale space is built efficiently by means of Additive Operator Splitting (AOS) schemes, which are stable for any step size and could be parallelized.
- **LBPH** [1]: The Local Binary Patterns Histograms (LBPH) feature extraction method can be described in the following steps:

  – Extract local features from images: This is done by not considering the whole image as a high-dimensional vector, instead describe only local features of a face. Features extracted this way will have low dimensions.
  – Summarize the local structure in an image by comparing each pixel with its neighborhood.
  – Take a pixel as center and threshold its neighbors accordingly.
  – Divide the LBP image into m local regions and extract a histogram from each. The corresponding feature vector of the fa
    ce is obtained by concatenating local histograms. These histograms are called Local Binary Patterns Histograms and the feature vectors of all images have the same size (size of the histogram).

- **OpenFace** [3]: The last feature extraction technique is OpenFace's implementation of FaceNet from Google. FaceNet yields the highest accuracy reported so far; the model and the data used in training remain private. For this purpose, OpenFace target was to implement the same neural network model of FaceNet and train it with 500k images from public datasets.

All these feature extraction methods, except for LBPH and OpenFace, identify local features in an image and calculate its descriptor as its feature vector. This local feature property of images would lead to feature vectors of different sizes. However, to classify faces, all images should have feature vectors of the same size. To overcome this problem, we follow the feature extraction process proposed in [20] and highlighted in Fig. 3. This feature extraction process has three main components.

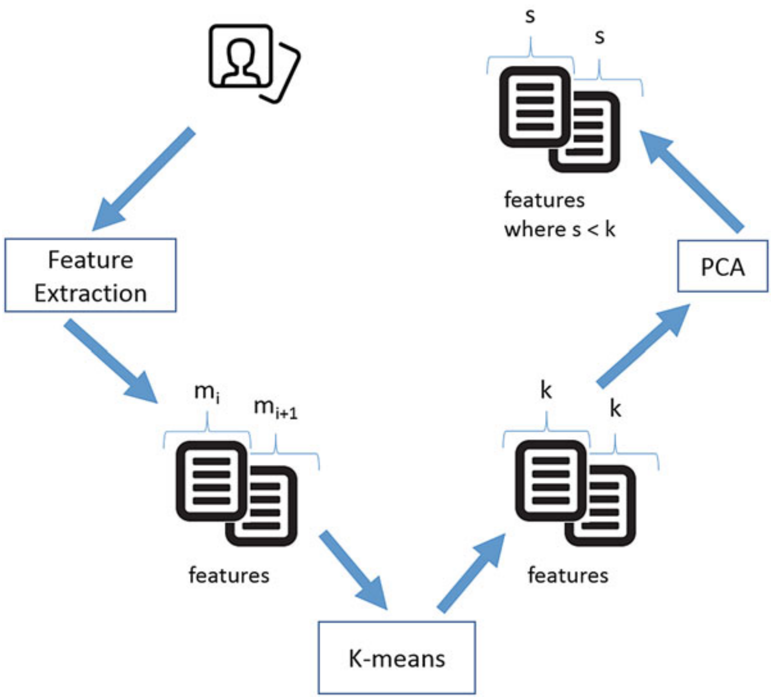

Fig. 3 Feature extraction process

1. **Image Feature Extraction:** The first step of the feature extraction process is to get as input face images and apply one of the general feature extraction methods described above to get corresponding feature vectors. These feature vectors may vary in size across different images.
2. ***K*-means Clustering:** After having the feature vectors of all face images, the next step is to map them into corresponding feature vectors of equal length. This is achieved by clustering the features using K-means to obtain K bins (where K is set to 200). As a result, every feature from the original feature vectors will be assigned a label of its cluster, in the range 1–200. These labels are used to build for each image a feature vector of fixed size K. This is done by iterating over original features of every image and increasing the ith entry of its new feature vector where i is the label of a given original feature.
3. **Principle Component Analysis:** The last step in the general feature extraction process is to reduce the dimensionality of the feature space produced based on K-means as described in step 2. Here, principal component analysis (PCA) is applied on the new feature vectors and leads to s features per image. PCA is a statistical method where given a set of feature vectors of possibly correlated variables, it converts correlated variables into a set of linearly uncorrelated

variables called principal components. This will eliminate unnecessary features from the feature space and will lead to more efficient clustering of the actual face images.

3.3 Applying Clustering Techniques

The last step in the process is to apply clustering algorithms on the produced feature vectors such that images of each person end up in a separate cluster. We have used in this study three of the clustering algorithms described in the literature, namely, **K-means** [7], **Spectral clustering** [18], and **Rank-Order clustering** [26].

K-means and Spectral clustering are the most widely used clustering algorithms. Both algorithms require specifying the number of clusters as an input parameter. This requires knowing in advance the number of people who appear in the video surveillance system, which is a serious restriction in several applications, for example, forensic investigation. Moreover, K-means suffer because the final result highly depends on the initial seeds of the clusters. This makes it difficult to handle clusters with varying density, size, and shape. Spectral clustering, on the other hand, can handle nonuniform distribution of data, but its complexity is high and usually performs poorly with noisy data [26]. Noise may come from the detection of faces in the various frames and will badly affect the performance of the clustering algorithm.

Rank-Order clustering method successfully tackles the problems associated with K-means and Spectral clustering. This method checks neighborhood of a face to determine its cluster. The method defines a new distance measure based on the dissimilarity in the neighborhood structure. Zhu et al. also claimed that their algorithm can handle nonuniform data distribution and it is robust to noise [26]. The algorithm has three major steps:

1. **Initialize Clusters:** Each face image forms a cluster on its own.
2. **Candidate Merging:** Compute the Rank Order distance between every pair of clusters C_i and C_j. If it is less than a threshold t, then mark the two clusters as a candidate merging pair.
3. **Transitive Merge:** Transitively merge all candidate merging pairs, then update the distance between clusters and loop back to the second step until no further merging is possible.

4 Datasets

We have used four datasets to evaluate the feature extraction techniques and the clustering algorithms described in the previous section. Each dataset has a different set of images to cluster, and each has its own challenges exhibited for face recognition.

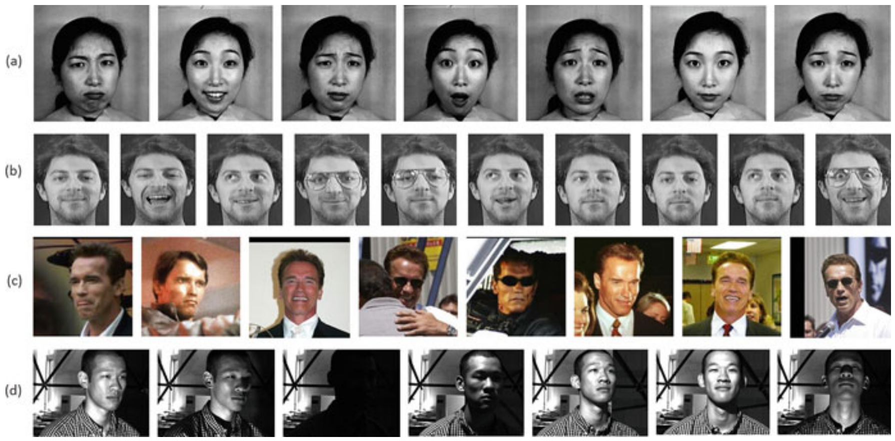

Fig. 4 (**a**) presents the JAFFE dataset, as seen all the images are taken in a controlled environment with the difference being the expression shown by the female. (**b**) presents the AT&T dataset where the face images are also taken in a controlled lab environment with difference in facial features and expressions for each person. (**c**) presents the LFW dataset; this dataset has a collection for each person images from the Web. As seen the images are not related to a scene and the image for a person is taken in different time frames (old/young). It presents a unique challenge to face recognition, as also other people can interfere in the image as shown in the sample. (**d**) presents the YaleB dataset, the dataset is taken in a controlled environment but the challenge here is with the illumination of each different image such that some images are unrecognizable even for humans

4.1 JAFFE Dataset

The Japanese Female Facial Expression (JAFFE) [16] database contains 213 images posted by 10 Japanese females. This dataset challenges face clustering by showing different facial expressions for each female in the dataset, these include happy, sad, angry, disgust, fear, surprise, and neutral. Examples from the JAFFE dataset are shown in Fig. 4a.

4.2 AT&T Face Dataset

The AT&T dataset [21] contains a set of faces collected at the University of Cambridge. The dataset contains 400 face images of 40 distinct persons, 10 images per person. The challenge for this dataset is to recognize people where each image was captured at different times with varying lightning scenes, different facial expressions, different facial details (glasses/no glasses), and different face alignments. An example of the dataset is shown in Fig. 4b.

4.3 LFW Dataset

The Labeled Faces in the Wild (LFW) dataset [25] contains 13,233 images of faces collected from the Web, representing 5749 persons. As shown in Fig. 4c, faces in this dataset are very challenging for face recognition algorithms as they were captured in the wild with varying conditions; the size of the clusters is varying in size and density because 1680 of the pictured persons have two or more distinct photos in the dataset, some have tens of images, and others have only one image. Given these challenges, this dataset could be classified as the hardest used in this study.

4.4 Extended YaleB Dataset

The Extended Yale Face Database B (Yale B) [13] is the largest dataset used in this study with 16,128 gray-scale images of 28 individuals. Every person has nine poses, where each pose has 65 images with a different facial expression or configuration. A sample of the dataset is shown in Fig. 4d.

5 Experiments and Results

In this section, we evaluate the performance of different clustering algorithms using several feature extraction methods on the four datasets mentioned in the previous section. We first present results of the preprocessing step, then we show performance of the feature extraction techniques and how this affects the clustering algorithms based on running time. Finally, we show the performance of the clustering algorithms for every feature extraction technique based on clustering accuracy. All experiments were run on a single machine with Intel Core i5-2400 CPU @ 3.1 GHz with 8 GB of RAM.

5.1 Face Preprocessing Results

Recall that the second step of the methodology described in Sect. 3.1 is to detect the face region in an image using an HOG descriptor for face detection. Table 1 reports for each dataset the original number of images the dataset has against the number of faces detected from our HOG face detector. All images from JAFFE dataset were successfully detected. We were able to apply preprocessing steps on them. This is expected with this dataset because it was captured in a controlled environment where the difference between the images is just facial expressions. However, face detection accuracy decreased drastically for AT&T and

Table 1 Preprocessing face detection accuracy

	Original	Detected	Percentage
JAFFE	213	213	100
AT&T	400	300	75
YaleB	16,380	90,371	55.17
LFW	13,233	13,176	99.54

YaleB datasets, scoring 75% and 55%, respectively. Even though both datasets were captured in a controlled environment, the HOG detector failed to identify faces with low illumination where their features can be hardly seen. This is why almost just 55% of the YaleB dataset has been detected; almost half the images of this dataset have low illumination. As for the LFW dataset which includes faces captured in an uncontrolled environment as shown in Fig. 4, we were able to detect almost the whole dataset with 99.54% detection accuracy. Having excellent accuracy in detecting the LFW dataset is very important because these images were taken in the wild and many different applications such as video surveillance systems capture images in a similar environment.

5.2 *Feature Extraction Runtime*

In this section, we report the running time results of the feature extraction process. It is very important to study the time required to extract the features and cluster the images. This is true because there are time-critical applications where time is an essential factor in deciding on the method to use and on acceptable clustering results.

As mentioned in Sect. 3.2, first each of **SIFT**, **SURF**, **KAZE**, **DAISY**, and **BRISK** extracts features of a face, then K-means and PCA are applied on the result to assign equal number of features for all images. To apply **LBPH** and **OpenFace**, we just have to get features from the given image.

Table 2 reports the run time for extracting features by the different extraction techniques for the listed datasets. Table 2 includes the following columns. "Extract Feature" time is common to all techniques and refers to the total time needed to extract features. "K-means" and "PCA" reveal the time needed by the feature extraction techniques. "Total" is the total time required by the features extraction techniques, K-means and PCA. As shown in Table 2, as the dataset size increases, the run time for the extraction techniques increases. From these results, it can be seen that SURF and KAZE are the fastest methods to extract features from images, closely followed by LBPH, and then OpenFace, while other methods take significantly more time to extract features for datasets. The difference margin is quite clear in case of **LFW** dataset where LBPH, SURF, DAISY, and OpenFace completed the process between 6 and 118 min, while KAZE needed almost 20 min, SIFT needed almost an hour and a half, and BRISK completed in around 2 and half hours. As detailed in the table, BRISK took so much time because of the extraction technique it uses, where the method needed considerable time to process a single

Table 2 Feature extraction run time

	Extract features	K-means	PCA	Total
(a) JAFFE				
SIFT	0:00:13	0:00:23	0:00:01	0:00:37
SURF	0:00:02	0:00:01	0:00:01	0:00:05
KAZE	0:00:19	0:00:04	0:00:01	0:00:24
DAISY	0:00:03	0:00:02	0:00:01	0:00:06
BRISK	0:02:22	0:00:03	0:00:01	0:02:26
LBPH	0:00:05	–	–	0:00:05
OpenFace	0:00:09	–	–	0:00:09
(b) AT&T				
SIFT	0:00:19	0:00:55	0:00:02	0:01:15
SURF	0:00:04	0:00:02	0:00:01	0:00:07
KAZE	0:00:26	0:00:04	0:00:02	0:00:32
DAISY	0:00:05	0:00:03	0:00:01	0:00:09
BRISK	0:03:19	0:00:02	0:00:01	0:03:22
LBPH	0:00:08	–	–	0:00:08
OpenFace	0:00:19	–	–	0:00:19
(c) YaleB				
SIFT	0:07:51	0:43:37	0:00:15	0:51:43
SURF	0:02:16	0:00:49	0:00:15	0:03:20
KAZE	0:12:43	0:01:44	0:00:14	0:14:41
DAISY	0:02:22	0:01:44	0:00:12	0:04:18
BRISK	1:41:21	0:08:02	0:00:16	1:49:39
LBPH	0:03:30	–	–	0:03:30
OpenFace	0:07:19	–	–	0:07:19
(d) LFW				
SIFT	0:12:37	1:13:25	0:00:22	1:26:14
SURF	0:06:10	0:01:14	0:00:20	0:07:44
KAZE	0:17:30	0:02:27	0:00:20	0:20:17
DAISY	0:03:30	0:02:38	0:00:17	0:06:25
BRISK	2:29:25	0:06:41	0:00:23	2:36:29
LBPH	0:07:04	–	–	0:07:04
OpenFace	0:11:52	–	–	0:11:52

The table reports related to a dataset, a comparison of the run time of the three different clustering techniques used in this study. The results are shown in the format of H:MM:ss, where H is hour, M is minutes, and s is seconds. Values marked as "–" indicate that the clustering algorithm did not finish in a matter of running for 1 day

image and get its features. While SIFT is not so slow in the extraction process, it slows down when it moves to the K-means step to produce clusters. The reason behind this is that SIFT generates a huge feature vector describing one image. So, applying K-means on all features of every image requires a lot of time.

5.3 Clustering Results

In this section, we first show a study similar to that discussed in the previous section. Here, we analyze clustering time for different feature extraction techniques and clustering tools. As mentioned earlier, the importance of a clustering algorithm should compensate between run time and clustering accuracy.

Clustering run time is reported in Table 3. Run time is not reported for some clustering techniques, especially on large datasets. This is because we show only results for clustering algorithms that finished in at most 1 day time.

For the first three datasets described in Sect. 5.1, namely JAFFE and AT&T, corresponding results for all the feature extraction techniques are shown in Table 3a–c; these three datasets are the smallest in size in terms of the number of face images. Clustering run time for these three datasets is very fast; it is almost identical for K-means and Spectral clustering, taking almost a second each for all feature extraction techniques. On the other hand, the results show that Rank Order clustering is significantly slower than the other two clustering methods.

As reported in Table 3a for the JAFFE dataset, the performance of Rank Order clustering ranges between 12 and 18 s for all feature extraction techniques except for the LBPH method. However, it took 49 s to complete the clustering for the LBPH extraction technique. This is because the number of features generated by LBPH is larger than that of the others. Actually, LBPH constructs a feature histogram for every region in an image. The difference between the performance of LBPH and other methods can be clearly seen in Table 3b. To finish the clustering process, the other methods needed almost a minute, while LBPH took 2 min.

Concerning the two datasets, LFW and YaleB, K-means finished successfully on all the feature extraction techniques. Spectral clustering terminated successfully on the YaleB dataset but failed on LFW. On the other hand, Rank Order clustering couldn't finish running for both YaleB and LFW datasets. LBPH was again the slowest taking more than 3 h to complete on the LFW dataset, while running time of the other feature extraction techniques like K-means ranged from 1 to 2 min.

Concerning clustering accuracy, our aim is to determine the best feature extraction technique used for face recognition and the best performing clustering technique based on the extracted features. We evaluated the accuracy of the clustering results based on the confusion matrix of the set of class labels predicted by the clustering algorithm for which true values are known from the dataset. To calculate the adjacency matrix, we use external validity indices which were designed to measure the similarity between two partitions (predicted labels vs true labels). This method's confusion matrix as described in [17] represents the count of pairs of points based on whether they belong to the same cluster or not by considering the two partitions. For each pair in the predicted partition, we check whether these pairs have the same label or not and based on that populate the four entries in the confusion matrix, that is, true positive, true negative, false positive, and false negative counts.

Table 3 Clustering run time

	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
(a) JAFFE							
K-means	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01
Spectral	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01
Rank order	0:00:17	0:00:15	0:00:12	0:00:18	0:00:17	0:00:49	0:00:12
(b) AT&T							
K-means	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:02	0:00:01
Spectral	0:00:02	0:00:02	0:00:02	0:00:02	0:00:02	0:00:03	0:00:03
Rank order	0:01:03	0:01:24	0:01:06	0:01:00	0:01:14	0:02:10	0:00:57
(c) YaleB							
K-means	0:00:03	0:00:02	0:00:02	0:00:02	0:00:02	0:01:36	0:00:02
Spectral	3:13:32	3:04:31	3:03:43	4:44:03	3:03:20	5:09:30	4:11:15
Rank order	–	–	–	–	–	–	–
(d) LFW							
K-means	0:01:40	0:01:03	0:01:36	0:00:23	0:01:13	3:07:41	0:02:22
Spectral	–	–	–	–	–	–	–
Rank order	–	–	–	–	–	–	–

The table reports results of a dataset, comparing run time of the three clustering techniques used in this study . The results are shown in the format of H:MM:ss where H is hour, M is minutes, and s is seconds. Values marked as "–" indicate that the clustering algorithm did not finish in a matter of running for 1 day

After getting the confusion matrix, we calculate precision and recall by considering images in each cluster produced by the clustering algorithm and compare them with the corresponding ground truth. A given cluster C may contain some face images which are indeed members of C based on the ground truth and some other face images which should have not been included in C. Precision is the proportion of face images that were correctly classified as members of C, that is, the number of correct faces in C divided by all faces in C. On the other hand, recall considers face images in the ground truth to determine their proportion correctly classified in C, that is, it is the number of face images correctly classified in C divided by the number of all face images which should have been classified in C according to the ground truth. We also calculate F-measure which is a summary statistic that combines precision and recall as given in Eq. (1).

$$F = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{1}$$

Clustering accuracy results are presented in Table 4 for all the datasets and clustering algorithms applied on the feature extraction techniques. Figure 5 shows how clustering accuracy of Rank Order clustering varies for different threshold values.

Table 4 Clustering accuracy

	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
(a) JAFFE							
K-means	**78.87%**	74.64%	68.3%	58.02%	55.46%	65.4%	78.46%
Spectral	73.88%	70.07%	75.6%	55.5%	50.9%	**80.16%**	63.28%
Rank order	64.68% (22)	66.4% (26)	74.14% (22)	63.59% (26)	40.64% (21)	71.61% (25)	**82.14% (21)**
(b) AT&T							
K-means	56.05%	35.16%	40.91%	48.1%	23.71%	37.55%	**83.83%**
Spectral	50.33%	34.62%	35.82%	39.65%	18.59%	35.53%	**77.45%**
Rank order	52.15% (24)	41.03% (19)	44.82% (16)	49.76% (13)	26.22% (15)	57.22% (11)	**94.78% (19)**
(c) YaleB							
K-means	6.63%	4.73%	5.5%	5.49%	4.72%	8.46%	**65.84%**
Spectral	6.66%	4.83%	5.53%	6.64%	4.29%	6.65%	**75.11%**
Rank order	–	–	–	–	–	–	–
(d) LFW							
K-means	0.16%	0.01%	0.16%	0.17%	0.05%	0.24%	**1.41%**
Spectral	–	–	–	–	–	–	–
Rank order	–	–	–	–	–	–	–

The table reports results of a dataset, comparing run time of the three clustering techniques used in this study. The results are shown in the format of H:MM:ss where H is hour, M is minutes, and s is seconds. Values marked as "–" indicate that the clustering algorithm did not finish in a matter of running for 1 day

Table 4 reports accuracy results obtained by applying the different clustering algorithms on each feature extraction technique mentioned above. Values of best clustering accuracy for each clustering algorithm are shown in bold font. Table 4a shows the results obtained for JAFFE dataset. The best result in this table is obtained by using Rank Order clustering with the OpenFace feature extraction technique (82%). This is followed by using Spectral clustering on the LBPH method (80%). We can conclude from this table that the OpenFace technique has on average the highest accuracy across the three different clustering techniques. The second best-performing method is LBPH, while BRISK performed on average the lowest. This conclusion is further supported in Table 4b where the difference margin becomes clear as the dataset size increases.

The best clustering results belong to OpenFace using Rank Order clustering (94.78%). In this table and all following tables, it is possible to notice that the best results for K-means, Spectral, and Rank Order clustering have been obtained using the OpenFace technique. Another conclusion which could be noticed from this table is that Rank Order clustering gave on average the best results compared to the other clustering algorithms.

Table 4c shows clustering results for the YaleB dataset which has almost 9000 images after preprocessing. In the table, we miss the values of Rank Order

Clustering which gave the best results in the previous table. This is because Rank Order clustering could not finish in 1 day. The results show a big gap when using OpenFace compared to the other feature extraction techniques, reaching 60%.

Spectral clustering with OpenFace reached 75% while the best result shown by the other methods is LBPH (8%). Table 4d reports accuracy results for the LFW dataset where only K-means finished in a day. OpenFace reported best results; its accuracy is just 1.41%, while accuracy for the other techniques ranges from 0.01% to 0.24%. This is because the LFW dataset has imbalanced cluster sizes, and algorithms like K-means would fail when applied on this kind of dataset.

Figure 5 shows Rank Order clustering results in terms of precision, recall, and F-measure for JAFFE, ATT&T, and ATT&T Filtered image sets. This figure shows that having low threshold will result in high precision almost 100%, while recall is low. And while the threshold of the clustering increases, precision starts to decrease, recall gets higher, and F-measure always goes up to a certain threshold then back down with the margin of precision and recall getting high.

This can be explained as follows, for a low threshold, most images will be merged together to belong to the same cluster. Having high precision means most retrieved images have the same label. Low recall means a smaller percentage of images from a target cluster have been retrieved. As the threshold increases, few images will be considered to belong to the same cluster. The number of clusters will increase such that person B will have a cluster with some noise, that is, recall will increase, while some of person A images will end up in other clusters (due to false positives) leading to lower precision.

6 Conclusions

We have performed a study on different feature extraction techniques to determine which one is better to use in Face clustering. In addition to feature extraction techniques, we also used three clustering algorithms to see which clustering algorithm performs the best on features extracted by the feature extraction techniques.

For this purpose, we have used four datasets, each with its own challenges in the face recognition problem and with varying sizes. We did our experiments on a single machine and noted down the results based on both run time and clustering accuracy. From the experiments and results, we concluded that the best clustering algorithm is Rank Order clustering, but due to its time complexity we could not manage to run it for large datasets. As for time complexity, K-means run time is massively better than Rank Order clustering and Spectral clustering. Spectral clustering is even faster than Rank Order clustering. Concerning the best feature extraction technique, it was OpenFace that performed the best when used with Rank Order clustering. Not only accuracy levels were great by good margins, also feature extraction time was acceptable and in a good range compared to the other techniques.

As for future work, we would like to enhance Rank Order clustering by using approximation methods to construct a K nearest neighbor (K-NN) graph instead of

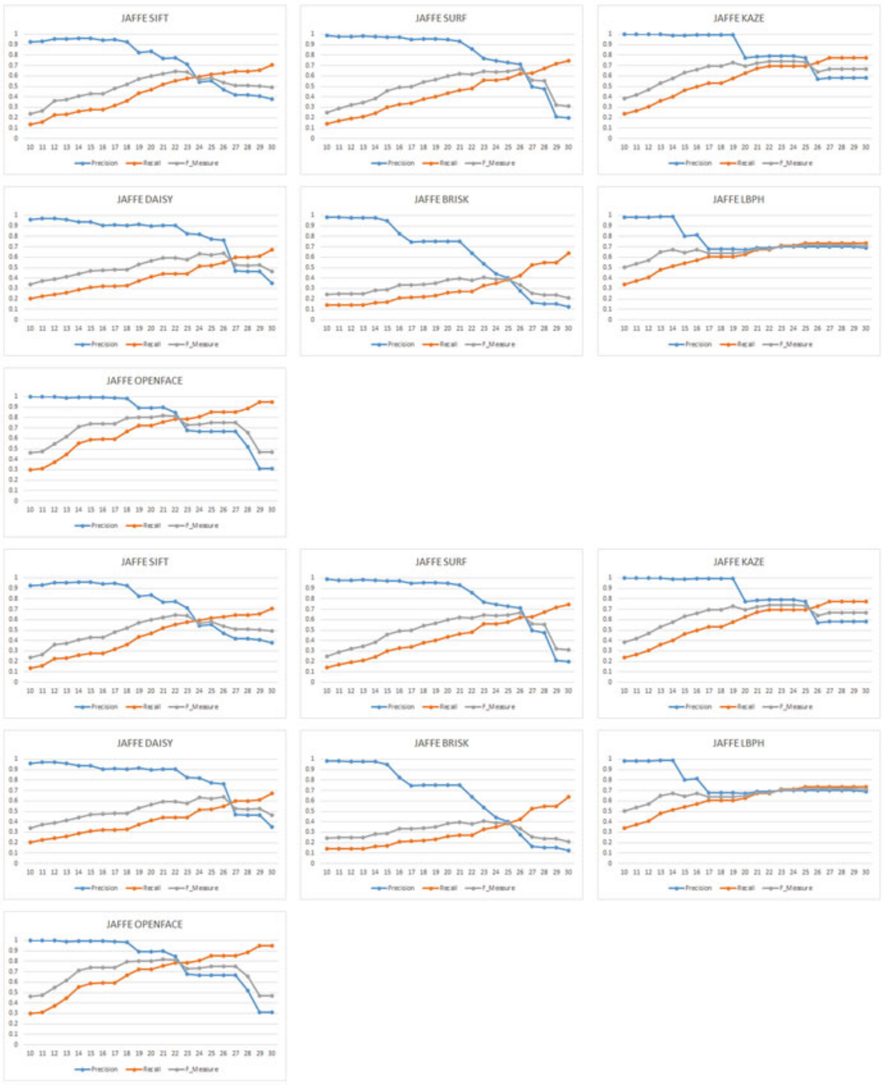

Fig. 5 Plots of all the rankorder results with varying thresholds

calculating the distance between a given image and all other images in the same dataset. Also we want to use more enhanced hardware to perform parallel execution of this procedure, thus making it faster. This way, it would become possible to faster produce clustering results for LFW and YaleB datasets.

References

1. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: application to face recognition. IEEE Trans. Pattern Anal. Mach. Intell. **28**(12), 2037–2041 (2006)
2. Alcantarilla, P.F., Bartoli, A., Davison, A.J.: Kaze features. In: Computer Vision–ECCV 2012, pp. 214–227. Springer, Berlin (2012)
3. Amos, B., Ludwiczuk, B., Satyanarayanan, M.: Openface: a general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science (2016)
4. Bay, H., Tuytelaars, T., Gool, L.V.: Surf: speeded up robust features. In: Computer Vision–ECCV 2006, pp. 404–417. Springer, Berlin (2006)
5. Bruce, V., Young, A.: Understanding face recognition. Br. J. Psychol. **77**(3), 305–327 (1986)
6. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp. 886–893. IEEE, Washington (2005)
7. Hartigan, J.A., Wong, M.A.: Algorithm as 136: a k-means clustering algorithm. J. R. Stat. Soc. Ser. C (Appl. Stat.) **28**(1), 100–108 (1979)
8. Heisele, B., Ho, P., Poggio, T.: Face recognition with support vector machines: global versus component-based approach. In: Proceedings of the Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001, vol. 2, pp. 688–694. IEEE, Washington (2001)
9. Ho, J., Yang, M.-H., Lim, J., Lee, K.-C., Kriegman, D.: Clustering appearances of objects under varying illumination conditions. In: Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003, vol. 1, pp. I–11. IEEE, Washington (2003)
10. Kazemi, V., Sullivan, J.: One millisecond face alignment with an ensemble of regression trees. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1867–1874 (2014)
11. King, D.E.: Dlib-ml: a machine learning toolkit. J. Mach. Learn. Res. **10**(July), 1755–1758 (2009)
12. Klontz, J.C., Jain, A.K.: A case study of automated face recognition: the boston marathon bombings suspects. Computer **46**(11), 91–94 (2013)
13. Lee, K.-C., Ho, J., Kriegman, D.J.: Acquiring linear subspaces for face recognition under variable lighting. IEEE Trans. Pattern Anal. Mach. Intell. **27**(5), 684–698 (2005)
14. Leutenegger, S., Chli, M., Siegwart, R.Y.: Brisk: binary robust invariant scalable keypoints. In: 2011 IEEE International Conference on Computer Vision, pp. 2548–2555. IEEE, Washington (2011)
15. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
16. Lyons, M.J., Akamatsu, S., Kamachi, M., Gyoba, J., Budynek, J.: The Japanese female facial expression (jaffe) database (1998)
17. Milligan, G.W., Cooper, M.C.: A study of the comparability of external criteria for hierarchical cluster analysis. Multivar. Behav. Res. **21**(4), 441–458 (1986)
18. Ng, A.Y., Jordan, M.I., Weiss, Y., et al.: On spectral clustering: analysis and an algorithm. Adv. Neural Inf. Process. Syst. **2**, 849–856 (2002)
19. Otto, C., Klare, B., Jain, A.K.: An efficient approach for clustering face images. In: 2015 International Conference on Biometrics, pp. 243–250. IEEE, Washington (2015)
20. Puttemans, S., Howse, J., Hua, Q., Sinha, U.: Opencv 3 Blueprints: Expand Your Knowledge of Computer Vision by Building Amazing Projects with Opencv 3. Packt Publishing, Birmingham (2015)
21. Samaria, F.S., Harter, A.C.: Parameterisation of a stochastic model for human face identification. In: Proceedings of the Second IEEE Workshop on Applications of Computer Vision, 1994, pp. 138–142. IEEE, Washington (1994)

22. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 815–823 (2015)
23. Tola, E., Lepetit, V., Fua, P.: Daisy: an efficient dense descriptor applied to wide-baseline stereo. IEEE Trans. Pattern Anal. Mach. Intell. **32**(5), 815–830 (2010)
24. Zhao, M., Teo, Y.W., Liu, S., Chua, T.-S., Jain, R.: Automatic person annotation of family photo album. In: International Conference on Image and Video Retrieval, pp. 163–172. Springer, Berlin (2006)
25. Zhu, X., Ramanan, D.: Face detection, pose estimation, and landmark localization in the wild. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2879–2886. IEEE, Washington (2012)
26. Zhu, C., Wen, F., Sun, J.: A rank-order distance based clustering algorithm for face tagging. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition, pp. 481–488. IEEE, Washington (2011)