Novel hybrid object-based non-parametric clustering approach for grouping similar objects in specific visual domains

Kuru, Kaya and Khan, Wasiq

Novel hybrid object-based non-parametric clustering approach for grouping similar objects in specific visual domains

Kaya Kuru[a,*], Wasiq Khan[b]

[a]School of Engineering, University of Central Lancashire, Preston, PR1 2HE, UK
[b]School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University, Manchester, M15 6BH, UK

Abstract

Current widely employed clustering approaches may not yield satisfactory results with regard to the characteristics and distribution of datasets and number of clusters to be sought, especially for visual domains in multidimensional space. This study establishes a novel clustering methodology using a pairwise similarity matrix, *Clustering Visual Objects in Pairwise Similarity Matrix (CVOIPSM)*, for grouping similar objects in specific visual domains. A dimensionality reduction and feature extraction technique, along with a distance measuring method and a newly established algorithm, *Clustering in Pairwise Similarity Matrix (CIPSM)*, are combined to develop the CVOIPSM methodology. CIPSM utilizes both *Rk-means* and an *agglomerative, contractible, expandable* (ACE) technique to calculate a membership degree based on maximizing inter-class similarity and minimizing intra-class similarity. CVOIPSM has been tested on several datasets, with average success rates on downsized subsamples between 87.5% and 97.75% and between 81% and 87% on the larger datasets. The difference in the success rates for small and large datasets is not statistically significant (p>0.01). Moreover, this method automatically determines the likely number of clusters without any user dictation. The empirical results and the statistical significance test on these results ensure that CVOIPSM performs effectively and efficiently on specific visual domains, disclosing the interrelated patterns of similarities among objects.

Keywords: Cluster analysis, visual object detection, feature extraction, image processing, pairwise similarity comparison, non-parametric clustering

1. Introduction

Cluster analysis divides a set of nearby objects into categories in which objects are more homogeneous than in other categories, based on their features and intrinsic similarities (i.e., without referring to pre-trained datasets). Clustering is of increasing importance owing to the quickly growing number of large databases in every discipline, most of which are in multidimensional space and beyond human comprehension. Clustering algorithms are applied in a wide variety of domains and in a remarkable number of different disciplines – such as astronomy, medicine and genetics, biology and zoology, marketing, and geography – that require similarities to be distinguished from dissimilarities. Furthermore, we live in a digital world in which the number of visual datasets is increasing logarithmically and thus, the demand for better clustering techniques specific to visual datasets has risen considerably. Researchers are often more interested in grouping visual datasets based on the specific objects they contain, instead of on whole images. Therefore, new methodologies should be established to handle the particularities of working with rapidly growing datasets, i.e., first, clustering a set of objects into the desired number of groups without supervised dictation and second, correctly accommodating objects in these groups. Our methodology is presented in phases and its merits are demonstrated for sample visual objects belonging to different databases whose characteristics differ in several respects in order to quantify the results.

*Corresponding author
Email address: `kkuru@uclan.ac.uk; Address: Fylde Rd, CT Building, CM 225, School of Engineering, Preston, Lancashire PR1 2HE, UK; tel:+44 (0) 1772 89 5156` (Kaya Kuru)

Broadly, our method combines conceptual understanding of a novel hybrid methodology using a dimensionality reduction technique for feature extraction and selection and a distance measuring technique for calculating the similarities among objects are merged with a new method, *Clustering in Pairwise Similarity Matrix (CIPSM)*, for unsupervised partitioning of objects. The notation used throughout the paper for the proposed hybrid methodology is *Clustering Visual Objects in Pairwise Similarity Matrix (CVOIPSM)*. The methodology is presented, along with a general framework and a roadmap for clustering objects in visual domains, and its implementation is separated into two components: the first acquires features and the distances/similarities between these features and the second clusters objects using these similarities. A variety of techniques is required for good object detection, feature selection, and distance measurement among objects in different visual datasets. Thus, these steps are separated from the clustering component to allow users to cluster datasets whose features and distances are acquired using different methods. Therefore, the user can choose other techniques for the first part based on the characteristics of the visual domain being studied. Moreover, the clustering component, CIPSM, is general and can be performed alone for unsupervised clustering of not only visual datasets, but also any dataset. This methodology is tested on four datasets from three databases. Clustering success is evaluated using an external quality measure technique that compares the clusters produced by our methodology to known classes already in the related dataset domains.

The structure of this paper is as follows: Section 2 discusses related studies, Section 3 unveils the conceptual framework of the proposed methodology, Section 4 presents the datasets and the study design for evaluating the proposed methodology, the results are explained in Section 5, and finally, Sections 6 and 7 discuss and conclude the findings in this study.

2. Related literature

In this section, we briefly analyze related subjects to provide an overview of the general concepts and techniques that have so far been deployed in image and object analysis. Our evaluation of related studies formed the basis for our study.

2.1. Object detection in images

The specific objects or regions of interest (ROI) in an image have to be roughly distinguished from other objects and acquired in order to perform object-based clustering. Several techniques have been used to segment specific objects in images. JSEG uses color quantization and spatial segmentation to divide images into regions; a combination of color and texture features is widely used for image segmentation and discerning different regions in an image [1]. This technique is successfully employed for general texture-based image clustering. However, it is less successful in detecting specific visual objects in images, and consequently, for cluster analysis based on specific objects in images.

To this end, object-based image retrieval is employed for acquiring specific regions from images; examples of this technique can be found in many studies, such as [2], [3], and [4]. Some of the widely used automatic object detection techniques are template matching, scale-invariant feature transform (SIFT), speeded-up robust features (SURF), features from accelerated segment test (FAST), binary robust independent elementary features (BRIEF), oriented FAST and rotated BRIEF (ORB), maximally stable extremal regions (MSER), binary robust invariant scalable keypoints (BRISK). Deep convolutional networks are being used for object detection as well [5]. Most of these techniques give a similarity value regarding the specified number of most important keypoints and this value is utilized to decide if there is a similarity between two objects given a threshold value. In addition, a pairwise similarity table can be established using the similarity values acquired from these techniques for visual domains as well. Haar Cascade files are most commonly employed for detecting an object in an image. A Haar Cascade file can be trained on a few hundred samples of a particular object (e.g., a leaf or a face) – which is a time consuming process – or pre-trained Haar Cascade files in the public domain can be deployed. In particular, using Haar Cascade files makes it easier to detect more than one similar object in an image easily and quickly. In most of these object detection techniques, a reference object is compared to the objects in images to find similar objects using a resemblance threshold value. Likewise, we implement region-based object detection in our study, as explained in Section 3, to discern the objects in images. More specifically, we employ Haar Cascade files where possible, a Harris algorithm and local similar neighborhood points to detect the objects in our visual domains, as explained in Section 3.1.1. Cropping objects in images for further analysis after designating their ROIs using automatic object detection can be implemented, with or without background removal, as explained in Section 3.1.1.

2.2. *Feature extraction and selection, and distance measurement*

Some of the feature extraction and selection techniques for visual objects are kernel PCA, independent component analysis, probability density estimation, local feature analysis, elastic graph matching (EGM), multi-linear analysis, kernel discriminant analysis, Gabor wavelet (GW), Fisher's linear discriminant analysis (FLDA), and support vector machines. In particular, EGM, FLDA, GW, and PCA have been widely employed to extract features from an object in an image region [6]. The high accuracy of these methods for extracting features and subsequently revealing patterns has been demonstrated in many studies. For instance, these methods can perform feature extraction and selection with accuracy rates of up to 96% [6]. With PCA, good success rates can be obtained by detecting patterns from images captured in ideal environments particularly with good illumination – or by employing several image processing techniques to enhance images before feature extraction. In addition, it is computationally efficient compared with other similar methods [7] because dimensionality reduction can be performed easily to accelerate the calculations. Thus, we employ PCA to extract the most important features from visual objects, because of its extensive and successful application to many datasets. In other words, we use PCA for dimensionality reduction by removing less important information (e.g., noise and redundant datasets) before applying our clustering approach, CIPSM. Interested readers are referred to our previous study for more information about how to implement PCA [6].

Some of the well-known approaches for measuring the distance between two points in a features dataset are Bayesian decision theory, multiple similarity, city block, subspace, angular separation, Pearson correlation, and Mahalanobis, Minkowski, Canberra, Chebychev, and Euclidean distances [8]. The Mahalanobis- and Euclidean-based distance measurement techniques are the most widely used of these approaches [8]. We tested these two methods on the features in our visual datasets to determine the better one to use, and found that the Euclidean-based technique outperformed the Mahalanobis one. Thus, this matching technique was selected for our study. Interested readers can find more information about measuring the distance between the features of two visual objects in Calva's study [9].

2.3. *Cluster analysis*

Cluster analysis is a very broad and wide-ranging field that cannot be covered completely in this section. Therefore, we provide a brief summary of the general concepts of current clustering schemes (especially their shortcomings with regard to cluster analysis of visual objects). Several well-known clustering algorithms are examined in-depth by Bishop [10], Witten [11] and Everitt [12] in their books, as well as in other journal publications. In general terms, centroid-based (e.g., k-means, XMeans, FarthestFirst, FilteredClusterer and c-means), hierarchical (e.g., cobweb, CLINK, SLINK, CURE, and BIRCH), incremental, density-based (e.g., DBSCAN and OPTICS), and probability-based (e.g., EM and Bayesian clustering) schemes predominate clustering methods.

The performance of centroid-based clustering schemes depends on the number of clusters specified [13], initial centroid selection, and the number of iterations. A shortcoming of this scheme is that the number of clusters (i.e., k = n [14], *n* singleton clusters [15]) must be dictated beforehand, and the algorithms execute clustering calculations based on this pre-specified number; while the cluster centroids change, the members of these clusters relocate based on the changing centroids. However, the desired number of clusters can be significantly different from the number of clusters dictated by the user, especially for large datasets. In this case, some objects can be allocated to the wrong clusters if the number of clusters specified is less than the desired number. The other drawback of this scheme is that empty clusters can occur if no objects are allocated to a cluster during the membership assignment phase. These methods suffer more when there are outliers that dramatically affect the cluster centroids over many iterations. It should be emphasized that centroid-based clustering is not a particularly sophisticated approach for cluster analysis of visual objects [10, 16].

Hierarchical clustering schemes consist of a series of partitions, which may run from a single cluster containing all individuals to *n* clusters each containing a single individual. Hierarchical clustering techniques may be subdivided into agglomerative methods, which proceed by a series of successive fusions (bottom-up) of the *n* individuals into groups, and divisive methods (top-down), which separate the *n* individuals successively into finer groupings [12]. One of the drawbacks from which hierarchical schemes suffer is that divisions or fusions, once made, are irrevocable; when an agglomerative algorithm joins two individuals they cannot subsequently be separated, and when a divisive algorithm decomposes individuals, they cannot later be joined [12]. Every division or fusion affects the structure of all previous clusters significantly regarding other clusters and distances between them. This merge or split decision, if poorly chosen at any step, may lead to low-quality clusters [17, 18]. Hierarchical clustering is generally employed in document clustering [19].

Incremental clustering works by processing data objects one at a time, incrementally assigning data objects to their respective clusters using a threshold value [19]. It typically uses much less space than methods that store a complete dataset [20]. Density-based schemes partition datasets based on areas of higher density to detect cluster borders without requiring the number of clusters in advance: the greater the density differences, the better the clustering. Although they may be successful for certain types of datasets, these methods do not work effectively on many computationally complex real-life datasets, which tend to contain small density differences among the visual objects to be clustered. Probability-based clustering schemes employ a statistical model of the data using iterative methods. This scheme does not work well for high dimensional datasets owing to the large number of required iterations.

Hierarchical and incremental clustering schemes generate an explicit knowledge structure that describes the clustering in a way that can be visualized and reasoned about [11]. Users are more eager to see more explicit similar clustering methods to employ on their datasets for meeting their needs, because this facilitates further investigation and comparison with previous and emerging methods [21]. In most clustering approaches, an iterative process is performed in which either the iteration continues until a predefined iteration count is reached or there are no changes in the clusters [22], which is often inefficient and can fail to achieve desired results depending on the iterative count and dataset characteristics. To summarize, the clustering schemes mentioned above are not infallible or robust, and their advantages and disadvantages depend on the characteristics and distribution of datasets on which they are run. Therefore, they are not suitable for all types of datasets, and consequently, mixtures of clustering techniques have been proposed in many studies [10, 23]. There are many other clustering techniques available and each of them may result in a different grouping of datasets and may not be effective for each type of dataset. Moreover, these clustering techniques may not scale to large datasets because of their computational time [24] and low accuracy. Therefore, many cluster analysis techniques are being developed for specific practical problems [25, 26], such as finding classes of genes that have similar functions, grouping information on the Internet for different specific queries, and clustering biomedical data [27, 28]. Likewise, new clustering approaches specific to visual domains in high dimensional space are required in order to produce better results.

To this end, many studies are specific to cluster analysis on image datasets [29], [30], focusing on the object shapes, joints, and localization [31]; colors [1, 32]; and texture or image segmentation [1, 33] through which image regions are discriminated. Likewise, Karthikeyan [24] studies image clustering using content-based image retrieval (CBIR) on whole images. He utilizes hue saturation value (HSV) color space and color histograms to acquire and cluster color similarities. The images are then clustered according to their content and dominant colors, not the objects they contain. Hence, the background of these images, rather than the objects in them, dominates the groupings. Therefore, CBIR is not successful for clustering specific objects in visual domains. Chen [34] deploys a hierarchy of clusters based on image semantics using detected objects in images and the semantic relationship between them. He uses a bag-of-semantics model (i.e., a set of meaningful descriptors) derived from the image's object relation network. Images are then clustered based on several objects they contain and their relationships.

We have not encountered a comprehensive object-based cluster analysis study in specific visual domains in the literature. It is clear from the studies in cluster analysis that none of the clustering methods can be judged to be best in all circumstances; particular methods yield better accuracy rates for particular types of datasets [12]. It is proven that hybrid clustering algorithms produce more efficient results than other algorithms [35]. Thus, the methodology presented in this study incorporates a new technique into several well-known techniques to help cluster objects, particularly visual objects, better than the current off-the-shelf methods emphasized by some studies [36, 37].

3. Methodology

The components of our methodology, *CVOIPSM*, which embraces several new methods along with well-known object clustering techniques, are explained in detail in the following section. Fig. 1 briefly articulates the overall design of the approach. There are two main parts of the methodology: *object detection and PSM (pairwise similarity matrix) establishment* (Fig. 1, section A.1) and *CIPSM* (Fig. 1, A.2). CIPSM itself is composed of two parts: *Rk-means* (Fig. 1, section A.2.1) and *ACE* (Fig. 1, section A.2.2).

The implementation of CVOIPSM is separated into two main parts. Feature extraction of objects and determining similarities between them based on these features are performed using PCA and Euclidean distance respectively in the first part. These methods are explained briefly in the following subsections. We present a framework for applying several common approaches toward visual datasets to cluster analysis for specific objects in these visual domains.

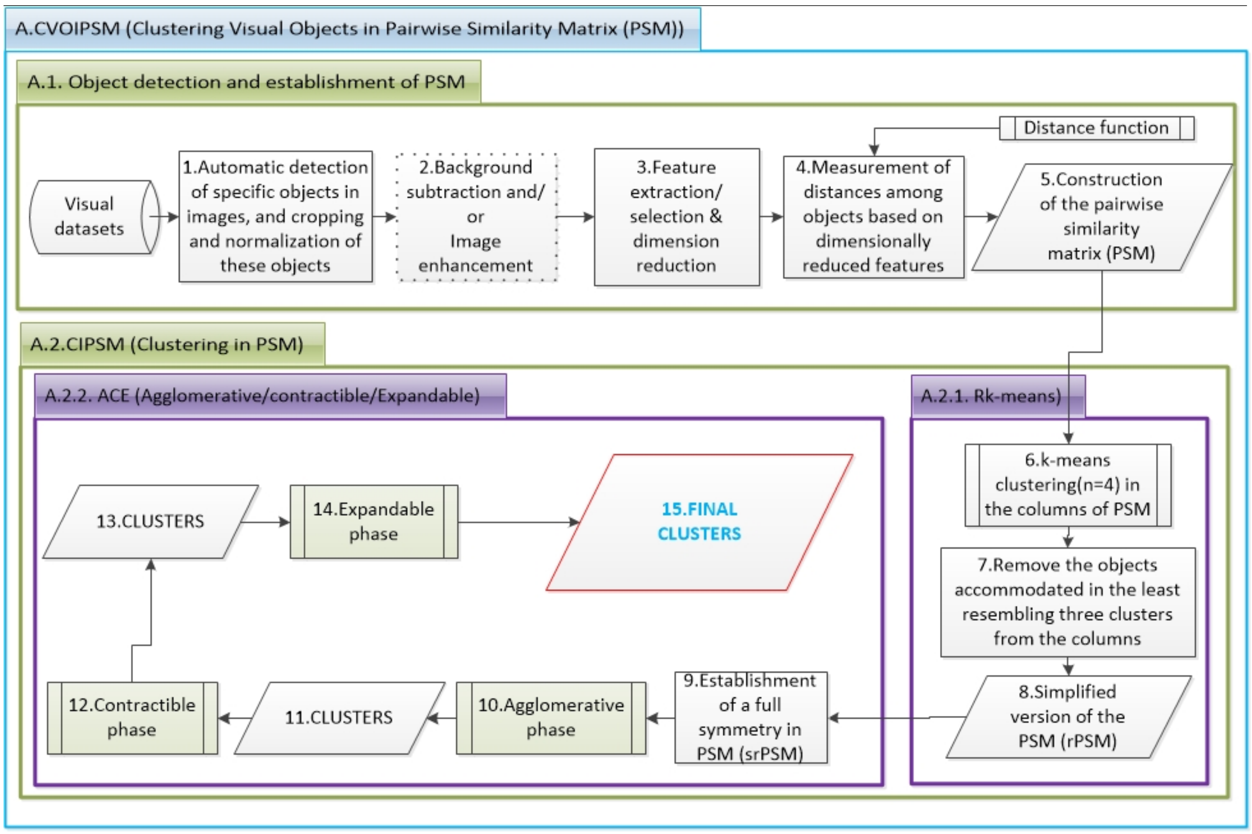

A.CVOIPSM (Clustering Visual Objects in Pairwise Similarity Matrix (PSM))
A.1. Object detection and establishment of PSM
Distance function
Visual datasets
1.Automatic detection of specific objects in images, and cropping and normalization of these objects
2.Background subtraction and/ or Image enhancement
3.Feature extraction/ selection & dimension reduction
4.Measurement of distances among objects based on dimensionally reduced features
5.Construction of the pairwise similarity matrix (PSM)
A.2.CIPSM (Clustering in PSM)
A.2.2. ACE (Agglomerative/contractible/Expandable)
A.2.1. Rk-means)
6.k-means clustering(n=4) in the columns of PSM
7.Remove the objects accommodated in the least resembling three clusters from the columns
8.Simplified version of the PSM (rPSM)
9.Establishment of a full symmetry in PSM (srPSM)
10.Agglomerative phase
11.CLUSTERS
12.Contractible phase
13.CLUSTERS
14.Expandable phase
15.FINAL CLUSTERS

Figure 1: Abstract presentation of CVOIPSM with two parts: 1) object detection and PSM establishment and 2) CIPSM. CIPSM has two sub parts as well: *Rk-means* and *ACE*. The functions incorporated in these main and sub parts are presented in detailed boxes of these parts. The part, A.2.1 is illustrated in Fig. 6. The part, A.2.2 (particularly the boxes 10, 12 and 14) is delineated in Fig. 7 in detail.

Interested readers can consult our previous study [6] for several basic image processing techniques (e.g., cropping, interpolation or extrapolation, median filtering, and histogram equalization) for the preparation of image datasets. We discuss how PCA and Euclidean distance measurements work on objects acquired from images to obtain pairwise comparisons in a pairwise similarity matrix (PSM) in our previous study as well [6]. In the first part, in brief, the dataset is prepared automatically for further analysis, which requires only the memory space needed to store the PSM, $O(n^2)$. In the second part, space is required to store both the reduced PSM, which is explained in subsection 3.2, and the group relationships of emerging clusters, which is less than $O(n^2)$. Therefore, the total space complexity of CVOIPSM is $O(n^2)$.

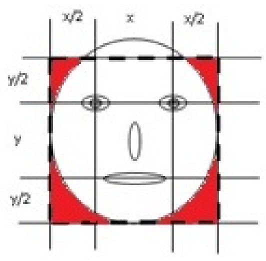


Figure 2: Cropping a face object using a face template. The red sections at the corners indicate the background.

3.1. Object detection & PSM establishment

Fig. 1, section A.1, depicts the first part of the methodology. Its implementation is explained in subsections 3.1.1, 3.1.2, and 3.1.3.

3.1.1. Object detection, normalization, and background subtraction

The object detection, normalization, and background subtraction phases are depicted in Fig. 1, A.1, numbers 1 and 2. Acquiring specific objects in images requires specific techniques based on the objects' characteristics. In other words, there is no "best technique" that can be employed for every type of image and all objects in those images. Therefore, we employ two different techniques to acquire objects from images based on the characteristics of our datasets, one

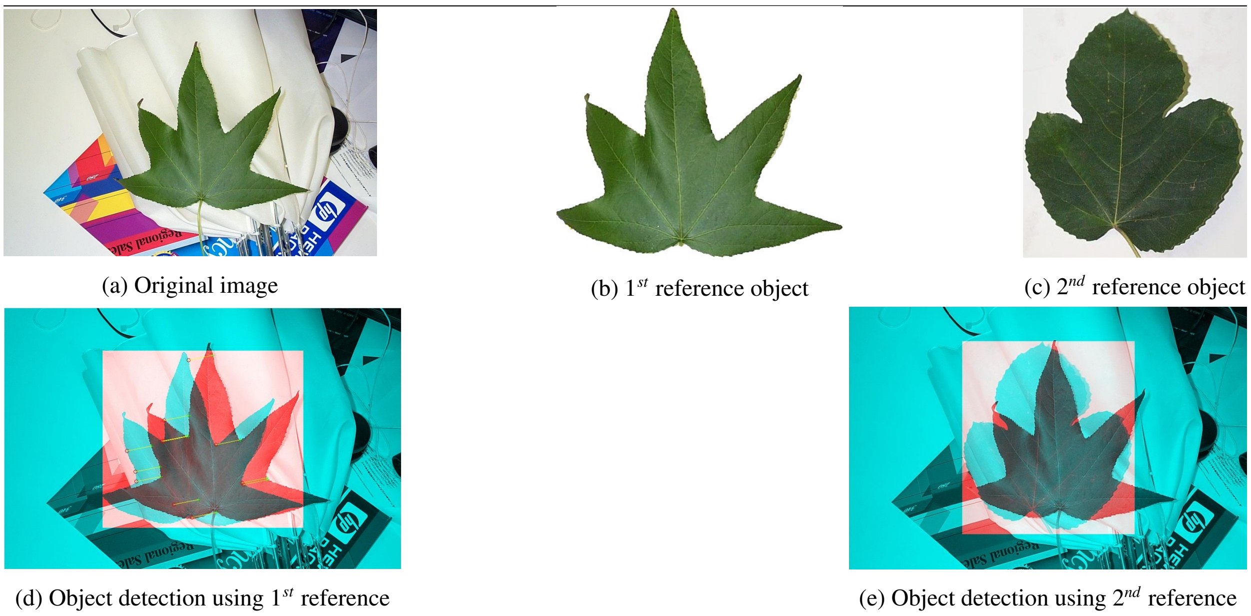

(a) Original image (b) 1^{st} reference object (c) 2^{nd} reference object (d) Object detection using 1^{st} reference (e) Object detection using 2^{nd} reference

Figure 3: Designation a leaf object in an image using a similar reference leaf object.

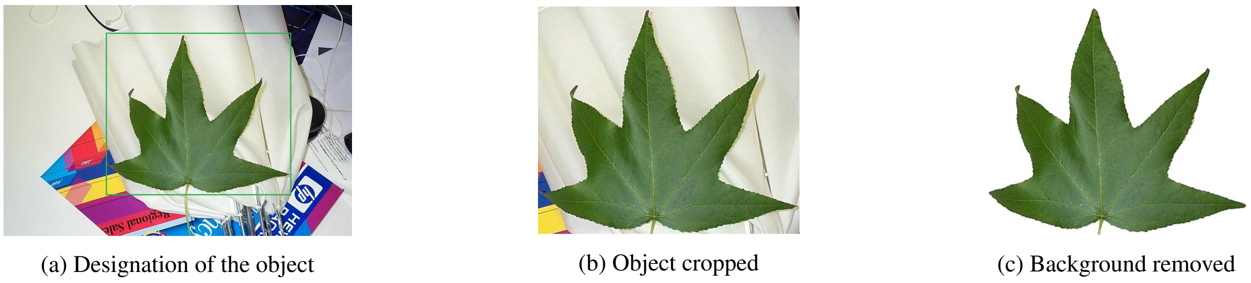

(a) Designation of the object (b) Object cropped (c) Background removed

Figure 4: Conceptual basis of leaf object cropping and background removing in an image.

of which contains faces and the other of which contains leaves. The detection of face and leaf objects in images is illustrated in Figures 2, 3, and 4. Images with faces are cropped to include just faces – with the forehead, eyes, cheeks and mouth – using a template, as delineated in Fig. 2. We obtain the leaf objects in images using a different technique, employing color-based region segmentation using a reference leaf (e.g., Figures 3b and 3c). First, the most similar areas of the leaf objects are determined by corresponding interest points between the images and the reference leaf object with the Harris algorithm and local similar neighborhood points, as depicted by the yellow lines in Figure 3d. This approach can be used to easily designate ROIs even with a less similar reference object (e.g., Fig. 3c), as depicted in 3e. Specific objects are cropped to remove unrelated sections after specifying ROIs (Fig. 4a) using color-based region segmentation, as displayed in Fig. 4b. Better features can be acquired after removing the background, as shown in Fig. 4c, which is analyzed in Section 4 with results presented in Section 5.

The sizes of the objects cropped or the background removed depends on the resolution, image size, and cropped objects occupying these images. As a result, feature comparison for further processing is not possible between objects of different sizes. Therefore, the cropping stage is followed by normalizing the objects using interpolation and extrapolation methods. All objects are initially cropped and an average mapping size is calculated. Then, all cropped objects are scaled to this new specified size; the objects whose height and width are less than the average size are extrapolated and the others are interpolated. Thus, each cropped object is mapped to a specified width and height.

The cropped objects are standardized by employing several image enhancement methods, such as median filtering and histogram equalization, to ensure a standard brightness and contrast and to remove illumination variations; these methods are explained in more detail in one of our previous studies [6]. Hence, dark or low contrast cropped objects are enhanced so that better features representing them can be acquired.

3.1.2. Feature extraction and selection & dimensionality reduction

The feature extraction and selection and dimensionality reduction phases are depicted in Fig. 1, A.1-3. The time and space complexity of image datasets is high; thus, it is imperative to reduce their dimensionality and acquire the most important common features for clustering these visual datasets better and faster. PCA is performed to extract the most important features from cropped, interpolated, and extrapolated objects. With this technique, a set of visual objects is transformed from a high dimensional space into a lower dimensional space using several principal component (PC) features (e.g., eigenvectors and eigenvalues). The first PC delineates the direction along which the dataset has maximum variance, and thus is considered the most significant feature in the dataset. Each following PC in turn has the next greatest variance and the next most significant feature. Rows of acquired 8-bit grayscale images are entered into a column vector (I) in order to implement PCA, which runs on vectors instead of images. The vector representing an average object, namely, the mean object (ψ), is acquired using the formula in Eq: $\psi = (I_1 + I_2 + ... + I_m)/m)$ where I indicates the objects in the column vector and m represents the number of visual objects in the dataset. Then, the unique features different from the average object vector are acquired by subtracting the mean vector from each object vector ($\theta_i = I_i - \psi$). A covariance matrix is further calculated using these unique features to generate a subspace of reduced dimensionality. Eigenvectors are generated using the covariance matrix (C) as $C = AA^T$ for real space and, $C = A^T A$ for subspace where $A = [\theta_1, \theta_2, \theta_3, ...\theta_m]$. Single value decomposition is performed to acquire the most significant eigenvectors with the equation $A = UDV^T$, where D corresponds to the diagonal (σ_i singular values of A), U corresponds to the eigenvectors of AA^T for real space, and V corresponds to the eigenvectors of $A^T A$ in subspace. The most significant k eigenvectors with the largest eigenvalues are selected to run on subspace to reduce noise in the data, as well as to significantly reduce the calculation burden. In other words, the eigenvectors correspond to the directions of the new coordinate axes and the eigenvalues correspond to the variance in each corresponding eigenvector. The eigenvectors with the largest eigenvalues are the most dominant PCs of the dataset; the eigenvectors with the lowest eigenvalues have little information about the characteristics of the datasets, and thus, removing these eigenvectors does not greatly affect our feature data, but considerably simplifies it without much information loss. In this manner, the objects in the dataset are symbolized by a reduced k number of eigenvectors. Weight vectors are described as $[w_1, w_2, w_3...w_k]$, where w_1 corresponds to the contribution of the first eigenvector to regenerate the objects. Each object can be reproduced from the sum of the weighted sum of these k eigenvectors and the mean object by means of vector space. Distance calculations are carried out between the weight vectors of the input object and those of the other objects in the dataset, and are described in the following subsection.

$$PSM = \left\{ \begin{array}{c|cccccc} & o_1 & o_2 & o_3 & o_4 & ... & o_n \\ \hline o_1 & 1.00 & S_{(o_1,o_2)} & S_{(o_1,o_3)} & S_{(o_1,o_4)} & ... & (S_{o_1,o_n}) \\ o_2 & S_{(o_2,o_1)} & 1.00 & S_{(o_2,o_3)} & S_{(o_2,o_4)} & ... & S_{(o_2,o_n)} \\ o_3 & S_{(o_3,o_1)} & S_{(o_3,o_2)} & 1.00 & S_{(o_3,o_4)} & ... & S_{(o_3,o_n)} \\ o_4 & S_{(o_4,o_1)} & S_{(o_4,o_2)} & S_{(o_4,o_3)} & 1.00 & ... & S_{(o_4,o_n)} \\ ... & ... & ... & ... & ... & ... & \\ o_n & S_{(o_n,o_1)} & S_{(o_n,o_2)} & S_{(o_n,o_3)} & S_{(o_n,o_4)} & ... & 1.00 \end{array} \right\}$$

Figure 5: PSM: the columns and rows indicate the objects and the cells indicate the similarity between intersecting objects. For instance, $S_{(O_1,O_2)}$ shows the similarity between objects 1 and 2. Orthogonal cells have a value of 1.00, since the similarity between two identical objects is 100%.

3.1.3. Distance calculations between objects & PSM construction

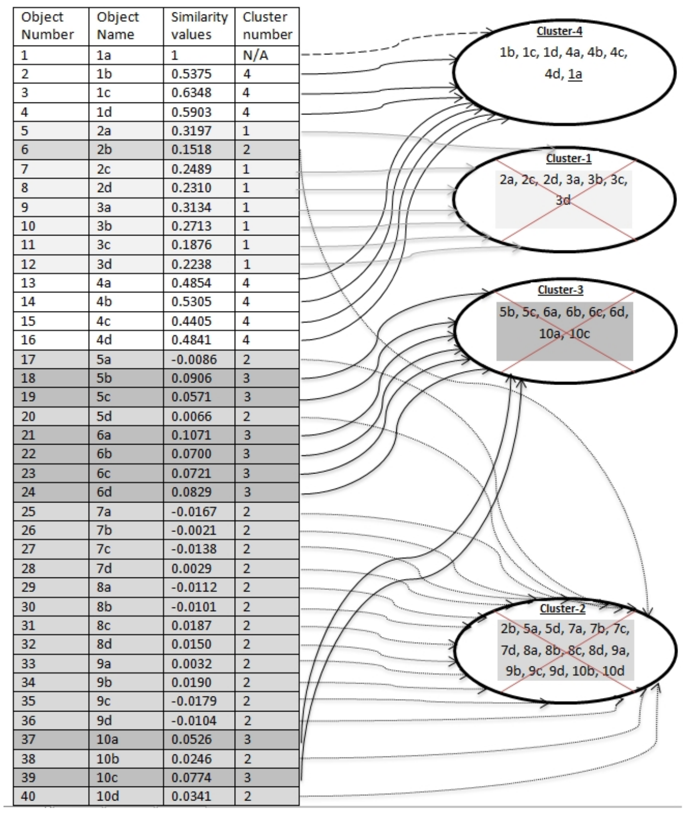
The distance calculation and PSM construction phases are depicted in Fig. 1, A.1-4 and A.1-5. Euclidean distance measurement is employed to acquire pairwise distances between all objects by considering the acquired features mentioned in subsection 3.1.2. The similarity values of each object to the other objects in the dataset are calculated and a pairwise similarity matrix (PSM) is established that comprises all the pairwise similarity values between the objects in the dataset. Euclidean distance measurement results in the determination of how different an object is from the other objects by means of a total sum of the differences between the weight vectors of the input image and those for other objects. All PCs of the input object are represented as a weight vector, $[w_1, w_2, w_3...w_k]$, and these are compared to the weight vectors of the other objects in the dataset. Eventually, the value acquired in each comparison corresponds to how different two objects are. The calculation formula for acquiring the distances between the input object and other objects is depicted in Eq. 1 [6] where D is

$$D_1^{(m-1)} \{D_1, D_2 ... D_{(m-1)}\} = \left\{ {}_1^{(m-1)} \left(\frac{\sum_{n=1}^{k} \left\| I_{t_{w_n}} - I_{t_{m_n}} \right\|^2}{k} \right) \right\} \tag{1}$$

the column array that holds distances between the input object and the other objects, *m* is the number of objects, $I_{t_{wn}}$ represents the constant weight values in the weight vectors of the input object with respect to all other objects, whereas $I_{t_{mn}}$ represents the weight values of other objects in the dataset, and k is the number of the values in a weight vector. The equation above is run for each D vector from 1 to m. In D vectors, the acquired distance values (m - 1, except the input object itself, whose distance equals zero), each of which represents the distance between the input object and the other objects in the dataset, are mapped between 0 and 1. Then, these values are subtracted from the value of 1 to determine how similar two objects are, rather than how different they are. All these values are placed in a PSM whose junction points of columns and rows are the similarity value of two objects. A PSM that has two identical sections separated diagonally from the leftmost upper cell to the rightmost lower cell is illustrated in Fig. 5 (Table A.5). The properties of PSM are explained with more examples in the following sections.

3.2. *Cluster analysis (CIPSM)*

The second component of CVOIPSM employs a new method, CIPSM, in which objects that most highly resemble each other are brought together while dissimilar objects in emerging clusters are singled out and each object is assigned to a specific nearest group in expanding and contracting groups by following a novel pattern. CIPSM combines Rk-means and ACE methods, as depicted in Fig. 1, A.2.1 and A.2.2. The CIPSM method pseudo code is presented in Algorithm 1. First, Rk-means, where $k = 4$, is run on a PSM and the least similar objects are removed from the PSM columns to reduce computation cost and to reveal a similarity pattern among objects. In other words, Rk-means is carried out to help reveal the interrelated patterns among objects, as well as to alleviate the cost of further calculations. Rk-means works similarly to the k-means method (or Lloyd's algorithm), but in a new understanding which is explained in subsection 3.2.1. The second part of the CIPSM approach, ACE, executes the final object clustering, discussed in subsection 3.2.2 with figures, tables, and pseudo codes. The executable files used in this subsection are included in the supplementary materials for users who would like to test the CIPSM method for their datasets as well as ours, which can also be found in the supplementary materials.



Object Number	Object Name	Similarity values	Cluster number
1	1a	1	N/A
2	1b	0.5375	4
3	1c	0.6348	4
4	1d	0.5903	4
5	2a	0.3197	1
6	2b	0.1518	2
7	2c	0.2489	1
8	2d	0.2310	1
9	3a	0.3134	1
10	3b	0.2713	1
11	3c	0.1876	1
12	3d	0.2238	1
13	4a	0.4854	4
14	4b	0.5305	4
15	4c	0.4405	4
16	4d	0.4841	4
17	5a	-0.0086	2
18	5b	0.0906	3
19	5c	0.0571	3
20	5d	0.0066	2
21	6a	0.1071	3
22	6b	0.0700	3
23	6c	0.0721	3
24	6d	0.0829	3
25	7a	-0.0167	2
26	7b	-0.0021	2
27	7c	-0.0138	2
28	7d	0.0029	2
29	8a	-0.0112	2
30	8b	-0.0101	2
31	8c	0.0187	2
32	8d	0.0150	2
33	9a	0.0032	2
34	9b	0.0190	2
35	9c	-0.0179	2
36	9d	-0.0104	2
37	10a	0.0526	3
38	10b	0.0246	2
39	10c	0.0774	3
40	10d	0.0341	2

Cluster-4
1b, 1c, 1d, 4a, 4b, 4c, 4d, 1a
Cluster-1
2a, 2c, 2d, 3a, 3b, 3c, 3d
Cluster-3
5b, 5c, 6a, 6b, 6c, 6d, 10a, 10c
Cluster-2
2b, 5a, 5d, 7a, 7b, 7c, 7d, 8a, 8b, 8c, 8d, 9a, 9b, 9c, 9d, 10b, 10d

Figure 6: Rk-means method: Rk-means runs on the first column of Table A.5 by excluding the cell with a value of 1. Four clusters, numbered 1-4, are established. The method compares the values of the first objects in each cluster to find the cluster that has the largest similarity value: object, 1b, with a value of 0.5375, has the largest value when compared to the values of objects 2a, 5b, and 2b in other clusters. Thus, the objects in cluster 4 are retained in their cells, while the other objects in clusters 1-3 are removed from their cells. Finally, the object with a value of 1, 1a, is added to the remaining cluster, cluster 4. The reason for excluding the object with a value of 1 is that this value dramatically effects the centers of the clusters in such a way that it can be set alone in a cluster with all other cells removed from the column, which is not a desired result.

3.2.1. *Reducing PSM using Rk-means*

The PSM reduction phase using Rk-means is depicted in Fig. 1, A.2.1, numbers 6, 7, and 8. Rk-means regards each column in the PSM separately for clustering and in four independent groups with respect to similarity values using k-means. Then, the objects in the three groups with the lowest similarity values are removed from the columns and only the objects in the group that has the highest similarity values remain in their cells. Our tests on several datasets substantiate that the four is the ideal number of groups for clustering objects to obtain one group of the most similar objects. An example employing Rk-means on the first column of Table A.5 is delineated in Fig. 6. In this example, the similarity values are clustered in four groups and only cluster 4, which has the most similar objects (1b, 1c, 1d, 4a, 4b, 4c and 1a) remains in the column; cells pointing out objects in other clusters (1, 2 and 3) are set to null. The 80% reduction in search space increases the efficiency of the ACE method, which is run following Rk-means. An example of a reduced PSM (rPSM) on which Rk-means is run is presented in Table A.5, with the remaining objects in columns and rows corresponding to white cells. The execution of Rk-means is explained in the CIPSM user manual in the supplementary materials.

3.2.2. Agglomerative, contractible, expandable (ACE) cluster analysis

The agglomerative, contractible, expandable (ACE) phases of cluster analysis are depicted in Fig. 1, A.2.2, numbers 9-15. The ACE method (the second component of Algorithm 1) is performed on the remaining cells in the rPSM to achieve the final clustering using the implicit interrelated similarity patterns among the objects. In a broader sense, ACE unveils these patterns of objects using the cell values of the rPSM, independently of object position. ACE reveals a degree of membership based on maximizing inter-class similarity minimizing the intra-class similarity. In other words, the similarity within a group is strengthened and the dissimilarity between the groups is made stronger after initial cluster assignment. ACE is a three-phase clustering technique. Each attribute of ACE – agglomerative, contractible, and expandable – inspects the others to find and remedy any weak memberships within established clusters; then, it converges to a well-separated solution. In most clustering algorithms, overfitting occurs as a result of the uncertainty in datasets. In this case, the most appropriate way to avoid overfitting and assign these objects to clusters is to find the minimum distance (maximum similarity) from the overfitted objects to the candidate clusters rather than using a threshold value, as in most clustering approaches, such as incremental clustering. In this way, the overfitted objects are incorporated into the closest clusters while they are removed from other clusters. To avoid overfitting, clustering is performed so that similar images are brought together and each object is assigned to a specific group. With this method, clusters are singled out beginning from the leftmost and then from the rightmost columns of the srPSM consecutively, and cells in the srPSM are set to null as new clusters are established; subsequent clusters are searched through the cells of the srPSM until all the cells are set to null. The conceptual framework of the algorithm is outlined in Fig. 7 and the phases are explained in the following paragraphs in detail with figures and tables.

First, ACE establishes an srPSM (i.e., the outcome of Rk-means) by placing values in cells whose diagonally symmetric counterparts have values. Our method works on two similar sections of the symmetric srPSM. Interested readers should refer to the second part of Algorithm 1 (– > Establish a diagonally symmetric rPSM) to visualize the symmetrical assignment. If n objects to be clustered are defined by a set O: $O = O_1, O_2, O_3, ..., O_N$ where N is the number of objects to be clustered and O_i is the representation of object i. The srPSM elements can be defined as S_{ij} = Similarity[O_{O_i}, O_{O_j}] where $S_{ij} = S_{ji}$. Thus, the srPSM is N x N in size. The matrix has a unit value of 1 in diagonal entries, Sij = 1, where i = j because these entries compare an object to itself. The cells denoted as (O_{O_i}, O_{O_j}) and their symmetric cells, (O_{O_j}, O_{O_i}), which have the same similarity value based on the symmetrical positioning of cells. The aim of establishing the srPSM is to gain back valuable similarities that are already set to null in the rPSM during the execution of Rk-means. With full symmetry, the ACE method results in the same outcome when referencing the columns or rows of the srPSM. Following srPSM formation, two sets of clusters are formed independently, one of which is acquired by detecting the patterns from the leftmost column and the other from the rightmost column of the srPSM, as shown in Algorithm 2. With the ACE method, the properties of the srPSM are exploited and objects are assigned to clusters in which their similarity values are greatest rather than changing centroids, as is implemented in several well-known cluster analysis methods. The conceptual basis of the agglomerative aspects of cluster formation is presented in Table 1 for the first cluster of the Tarrlab female face dataset. In this table, especially in step 13, the interdependencies in the srPSM represent a kind of selection criterion desirable for clustering. The ACE method finds the most likely set of clusters using the steps explained in Table 1. The distance from one object to the others plays a major role for assembling objects to form two sets of clusters. The method chooses as many candidate clusters as possible in terms of the agglomerative attribute. Within this phase, ACE treats each object as a singleton cluster and merges the closest singleton clusters to establish new clusters. The executable file, smLeftRight_cont.exe, can be run to visualize this property, as explained in the CIPSM user manual in the supplementary materials.

The outcome files of clusters obtained from the left and right sides of the srPSM for the Tarrlab female face dataset

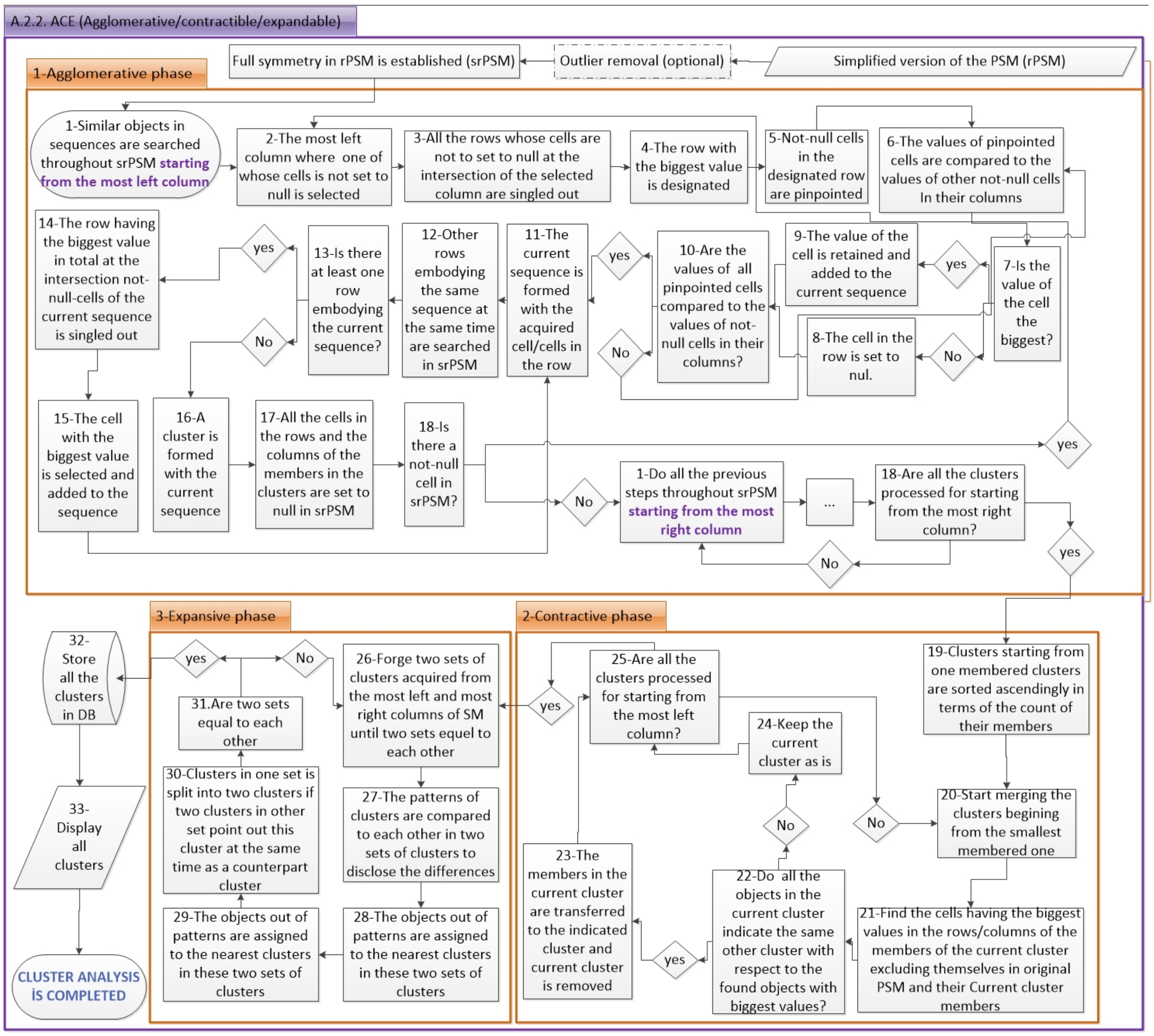

A.2.2. ACE (Agglomerative/contractible/expandable)
Simplified version of the PSM (rPSM)
Outlier removal (optional)
Full symmetry in rPSM is established (srPSM)
1-Agglomerative phase
1-Similar objects in sequences are searched throughout srPSM starting from the most left column
2-The most left column where one of whose cells is not set to null is selected
3-All the rows whose cells are not to set to null at the intersection of the selected column are singled out
4-The row with the biggest value is designated
5-Not-null cells in the designated row are pinpointed
6-The values of pinpointed cells are compared to the values of other not-null cells In their columns
7-Is the value of the cell the biggest?
yes
No
8-The cell in the row is set to nul.
9-The value of the cell is retained and added to the current sequence
10-Are the values of all pinpointed cells compared to the values of not-null cells in their columns?
yes
No
11-The current sequence is formed with the acquired cell/cells in the row
12-Other rows embodying the same sequence at the same time are searched in srPSM
13-Is there at least one row embodying the current sequence?
yes
No
14-The row having the biggest value in total at the intersection not-null-cells of the current sequence is singled out
15-The cell with the biggest value is selected and added to the sequence
16-A cluster is formed with the current sequence
17-All the cells in the rows and the columns of the members in the clusters are set to null in srPSM
18-Is there a not-null cell in srPSM?
yes
No
1-Do all the previous steps throughout srPSM starting from the most right column
...
18-Are all the clusters processed for starting from the most right column?
No
yes
3-Expansive phase
2-Contractive phase
19-Clusters starting from one membered clusters are sorted ascendingly in terms of the count of their members
20-Start merging the clusters begining from the smallest membered one
21-Find the cells having the biggest values in the rows/columns of the members of the current cluster excluding themselves in original PSM and their Current cluster members
22-Do all the objects in the current cluster indicate the same other cluster with respect to the found objects with biggest values?
yes
No
23-The members in the current cluster are transferred to the indicated cluster and current cluster is removed
24-Keep the current cluster as is
25-Are all the clusters processed for starting from the most left column?
yes
No
26-Forge two sets of clusters acquired from the most left and most right columns of SM until two sets equel to each other
27-The patterns of clusters are compared to each other in two sets of clusters to disclose the differences
28-The objects out of patterns are assigned to the nearest clusters in these two sets of clusters
29-The objects out of patterns are assigned to the nearest clusters in these two sets of clusters
30-Clusters in one set is split into two clusters if two clusters in other set point out this cluster at the same time as a counterpart cluster
31.Are two sets equal to each other
yes
No
32-Store all the clusters in DB
33-Display all clusters
CLUSTER ANALYSIS İS COMPLETED

Figure 7: Conceptual framework of the CIPSM method: Employing Rk-means on the PSM to simplify it and obtain a reduced similarity matrix is the first step of the CIPSM method, which is not shown in this figure. Clustering begins from the left of the rPSM. A fully symmetric rPSM (srPSM) is constructed, which is shown at the beginning of Algorithm 1. An example of a fully symmetric table is Table A.6. Then, clusters are created starting from the leftmost and rightmost columns of the srPSM depicted in Algorithm 2 (steps 1-18). A conceptual understanding for creating the first cluster from the leftmost column of the srPSM is presented in Table 1 and an example of the clusters created from the left and right of the srPSM is presented in Tables A.7 and A.8. Some of the clusters are merged together from single-member clusters if they are close to each other, which is depicted in Algorithm 3 (steps 20-26). An example of merging clusters obtained from the left and right of the srPSM is presented in Tables A.9 and A.10. The customization of these two tables is presented in Table A.11. The two sets of final clusters acquired from the leftmost and rightmost of the srPSM are forged together until the two sets are equal to each other, which is shown in Algorithm 4 (step 27). An example of the forged cluster sets obtained from the left and right of the srPSM is presented in Tables A.12 and A.13. The customization of these two tables is presented in Table A.14.

Table 1: Cluster establishment using the agglomerative attribute of the ACE method (the initial steps in the procedure, find_clusters_in_matrix(ascending), in Algorithm 2). The first cluster starting from the left of the PSM for Tarrlab female face objects in Table A.5 is established step-by-step in a conceptual framework.

Stages	Instructions	Specification and calculations	Outcomes of calculations
1	Clustering begins from the leftmost column of the srPSM.		
2	The leftmost column that has cells with a non-null value is specified.	1a	
3	All the rows whose cells have a non-null value in the specified column are singled out.	1a, 1b, 1c, 1d, 4a, 4b, 4c, 4d	
4	The row with the largest value of the non-null cells in the leftmost column from the previous step is determined.	$1a_{1.00} > 1c_{0.63} > 1d_{0.59} > 1b_{0.54} > 4b_{0.53} > 4a_{0.49} > 4d_{0.48} > 4c_{0.44}$	$1a_{1.00}$
5	Non-null cells in the row specified in the previous step are revealed.	1a, 1b, 1c, 1d, 4a, 4b, 4c, 4d	
6	The values of the cells designated in the previous step are compared to the values of non-null cells throughout their columns. The values in these cell are retained if they are the largest in value, otherwise they are deleted from the sequence, as well as from the srPSM.	1)for the column $1a_{1.00}$: $1a_{1.00} > 1c_{0.63} > 1d_{0.59} > 1b_{0.54} > 4b_{0.53} > 4a_{0.49} > 4d_{0.48} > 4c_{0.44}$; 2)for the column $1b_{0.54}$: $1a_{0.54} < 1b_{1.00}$; 3)for the column $1c_{0.63}$: $1a_{0.63} < 1c_{1.00}$; 4)for the column $1d_{0.59}$: $1a_{0.59} < 1d_{1.00}$;5)for the column $4a_{0.49}$: $1a_{0.49} < 4a_{1.00}$; 7)for the column $4b_{0.53}$: $1a_{0.53} < 4b_{1.00}$; 7)for the column $4c_{0.44}$: $1a_{0.44} < 4c_{1.00}$; 8)for the column $4d_{0.48}$: $1a_{0.48} < 4d_{1.00}$	1)1a:retained; 2)1b:removed; 3)1c:removed; 4)1d:removed; 5)4a:removed; 6)4b:removed; 7)4c:removed; 8)4d:removed
7	The current sequence is formed with the remaining cells and the other cells are set to null in the PSM and removed from the sequence.	Sequence: $1a$	
8	Other rows containing the sequence acquired in the previous step are searched for in the srPSM.	1b, 1c, 1d, 4a, 4b, 4c and 4d have values in the columns of the sequence (1a)	1b, 1c, 1d, 4a, 4b, 4c, 4d
9	Of the current sequence, the row having the largest value at the intersecting non-null cells is singled out.	$1c_{0.63} > 1d_{0.59} > 1b_{0.54} > 4b_{0.53} > 4a_{0.49} > 4d_{0.48} > 4c_{0.44}$	$1c_{0.63}$
10	The non-null cells in the row designated in the previous step are compared to those of the other cells in their columns. The object in a cell is added to the current sequence if it has the largest comparative, otherwise it is set to null in the srPSM.	The non-null cells in the row, 1c, following the sequence, 1a, are 1b, 1c, and 1d. The cell at the intersection column of 1c has the biggest value (1.00). Thus, 1c is added to the sequence, while the other cells in the row (1b, 1d) are set to null.	Add 1c to the previous sequence (1a)
11	The remaining values in the previous step are added to the sequence.	Sequence: $1a, 1c$	
12	Other rows containing the same sequence acquired in the previous step are searched for in the PSM (repeating step 8 above).	Rows 1b and 1d have values in the columns of the sequence (1a, 1c)	1b, 1d
13	The row having the biggest value in the non-null intersecting cells of the current sequence is singled out (repeating step 9 above).	-For 1b: $1a_{0.54} + 1c_{0.55} = 1.09$; -For 1d: $1a_{0.59} + 1c_{0.65} = 1.24$; 1d has the biggest value in total.	$1d_{1.24}$
14	The biggest value at the row selected in the previous step following the current sequence is specified.	The cell that has the largest value following the sequence ($1a, 1c$) in row 1d is 1.00 at the intersection with column 1d. The other cells in row 1d (1b, 4a, 4b, 4d) are set to null, except for the sequence (1a, 1c, 1d).	Add 1d to the previous sequence (1a, 1c)
15	The remaining objects in the previous step are added to the sequence.	Sequence: $1a, 1c, 1d$	
16	Other rows containing the same sequence acquired in the previous step are searched for in the PSM (repeating step 8 above).	Row 1b has values in the columns of the sequence (1a, 1c, 1d)	1b
17	The row with the largest value in the intersecting non-null cells of the current sequence is singled out (repeating step 9 above).	For 1b: $1a_{0.54} + 1c_{0.55} + 1d_{0.44} = 1.53$; 1b has the largest value, as it is the only object in the calculation.	$1b_{1.53}$
18	The largest value in the row selected in the previous step following the current sequence is specified.	The biggest value following the sequence $1a, 1c, 1d$ in row 1b is 1.00 at the intersecting cell of column 1b. The other cells in row 1b (none) are set to null except for the sequence (1a, 1c, 1d, 1b).	Add 1b to the previous sequence (1a, 1c, 1d)
19	The remaining objects in the previous step are added to the sequence.	Sequence: $1a, 1c, 1d, 1b$	
20	Other rows containing the same sequence acquired in the previous step are searched.	No row has the same objects in the columns of the sequence (1a, 1c, 1d, 1d).	Assign the sequence as a cluster.
21	There is no other row embodying the previous sequence that turns out to be a cluster. Set all the cells in the rows and columns of the members in the currently established cluster in the srPSM to null.	-All the cells in the rows and columns with the object names 1a, 1b, 1c, and 1d are set to null; Establish the cluster: the cluster with its column object names 1a, 1b, 1c, 1d are set to 1 in a randomly selected empty row of a table that is as big as the PSM since the cluster count can be, at most, the count of objects to be clustered.	Cluster: $1a, 1b, 1c, 1d$

are presented in Tables A.7 and A.8, respectively. The number of clusters before applying the contractive attribute is 11 for the left side and 10 for the right. The contractive attribute proceeds inversely, such that the members in clusters are evaluated and some clusters are merged, starting from single-member clusters depending on their similarity to ensure the coherency of members in candidate clusters, as shown in Algorithm 3. First, objects in single-member clusters are assigned to the nearest cluster based on the principle in cluster analysis that an object should be grouped with at least another object [11]. The nearest cluster for an object in a single-member cluster is determined using the cell that has the largest value in the same row/column in which this object is located. Thus, these objects in single-member clusters are assigned to clusters whose members have the largest value in the same rows/columns as these objects. By this phase, all single-member clusters are merged with other clusters. Second, for clusters with more than one member, the similarity/proximity values of objects within the cluster play a major role in merging some of these clusters. Thus, some of the objects in emerging clusters are added to the nearest clusters with similar objects without taking account of cluster centroids. Each member of any cluster with two or more members is assigned to some other target cluster if and only if each and every member of this cluster points to the same target cluster, as it is defined for single-member clusters; otherwise, it is accepted as a cluster in its current state. In other words, some emerging clusters are merged if each and every object in the cluster shows similarity to another cluster at the same time. The implementation to run for constructing two sets of clusters, including the merging phase mentioned above, with an executable file, smLeftRight_cont.exe, is explained in the CIPSM user manual in the supplementary materials.

An example of the outcome of the executable file, smLeftRight[1], for the left and right files is presented in Tables A.9 and A.10. The number of clusters for the left file is 9, two less compared to Table A.7 and 10 for the right file, the same as in Table A.8, meaning that there is no merging of clusters obtained from the right of the srPSM. As a result of the contractive attribute of the method, the members of two previous clusters are merged into clusters obtained from the left of srPSM using the distance between the two closest data points in the clusters: the one-member cluster, 24, that contains the object 6d is assigned to cluster 23, and the two-member cluster, 38, that contains objects 10a and 10b, is merged with cluster 33. For the one-member cluster, 24, the algorithm searches through the column of 6d to find the largest similarity value in the original TableA.5. It finds that object 6b has the largest similarity value, 0.60. 6b is in cluster 23, as can be seen in Table A.7; thus, 6d is transported to row 23 to form a cluster with the objects, 5c, 6a, 6b, 6c, and 6d, as can be seen in Table A.9. For the two-member cluster, 38, the algorithm searches through the columns of 10a and 10b to find the largest similarity values in the original TableA.5; note that the algorithm does not take the value into consideration in the intersection of 10b for the search of 10a and vice versa 10a for the search of 10b, because they are already in the same cluster. During this search, the algorithm finds that object 10d has the largest similarity value, 0.63, to 10a; 10d is in cluster 33. The largest similarity value to object 10b is 0.55, for object 10d; 10d is in cluster 33, as can be seen in Table A.7. Both objects in cluster 38, 10a and 10b, designate cluster 33 as a target at the same time. Therefore, 10a and 10b are transported into row 33 to form a cluster with the objects in this row, as can be seen in Table A.9. Thereby, the number of clusters, 11, becomes 9 after the contractive attribute of the ACE method. For the clusters obtained from the right of the srPSM, there is no one-member cluster in Table A.8, but there is a two-member cluster in row 25, with objects 7a and 7b. For the two-member cluster 25, the algorithm searches through the columns of 7a and 7b to find the biggest similarity values in the original TableA.5; note that the algorithm does not take the value of 7b into consideration during the search of 7a and vice versa, because they are already in the same cluster. During this search, the algorithm finds that object 7a is most similar to 7c, with a similarity value of 0.49; 7c is in cluster 27. The largest similarity value for object 7b is 0.62, for 7d; 7d is in cluster 28, as can be seen in Table A.8. Objects 7a and 7b, in cluster 25, designate different clusters, namely 27 and 28, as targets. Therefore, objects 7a and 7b keep their position as cluster 25. Likewise, other clusters with more members are examined for merging, however, the possibility of merging decreases as the number of members in clusters increases.

In the expansive attribute of the algorithm, which finalizes the number of clusters and the elements in each, the clusters in the two sets are paired and the members of these two cluster sets are forged together to yield one set of clusters based on similar clusters in the two sets and the degree of coherence of the elements in these clusters, as shown in Algorithm 4. Bad decisions from previous phases are processed and correction schemes are employed in this phase. That is to say, an object remains in a cluster where the other members in the same cluster are closer to it

[1]The executable file smLeftRight embraces both the agglomerative and contractible features of the ACE technique; thus, there is no need to perform the executable file smLeftRight_cont.exe, which includes the agglomerative portion of the ACE method. This executable file was created to visualize the agglomerative attribute of the method, how it works, and what the result is after the agglomerative phase.

than they are to the other objects in other clusters if this object appears in a cluster in the first set different from its counterpart cluster in the second set; otherwise, this object is transported into the other cluster, one of whose members is more close to it. Thus, members assigned to clusters may be relocated to group them with the correct objects nearby. First, the patterns of clusters are compared across the two cluster sets to determine the differences; objects are assigned to the nearest clusters in these two sets of clusters and some new clusters are created to equalize the number of clusters in the two sets. A cluster in one set is split into two clusters if two clusters in the other set point to this cluster at the same time as a counterpart cluster; in other words, the most similar cluster containing the most similar objects. These processes last until the two sets equal each other in cluster number and the objects in these clusters. Forging two sets of clusters into one is performed with an executable file, smFinalResult.exe, as explained in the CIPSM user manual in the supplementary materials.

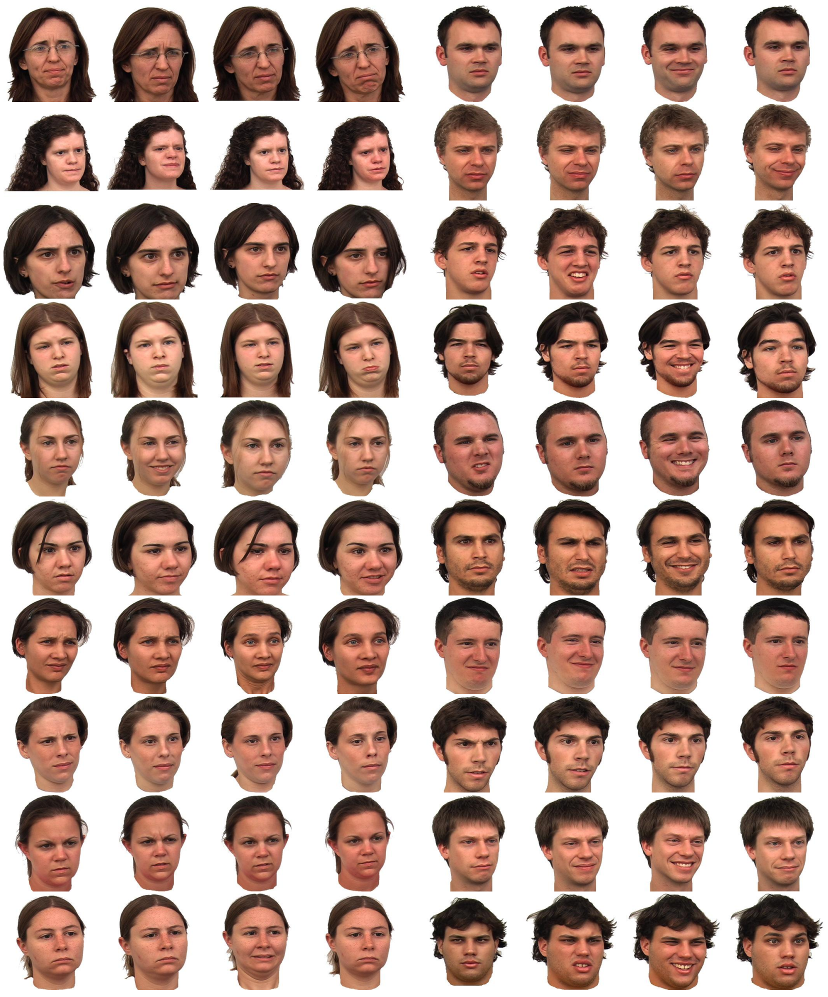


Figure 8: Frontal faces belonging to 10 Caucasian females (left) and males (right) from the Tarrlab database, 4 for each person. Each image is numbered such that a1-a4 refer to the images in the first row, whereas j1-j4 are used to refer to images in the last row for both females and males.

Output files after the ACE method is finalized are cluster_left_result.csv, obtained from the left side of the srPSM and cluster_right_result.csv, obtained from the right side of the srPSM. Clusters presented in these files are same but in different rows, since the method runs until the two sets are equal to each other in terms of columns. An example of forging two cluster sets, one obtained from the left and the other obtained from the right of the srPSM, is depicted in Table A.11 for the Tarrlab female face dataset in Fig. 8. There are two cluster sets in this table. These two sets are customized by removing the empty rows in Tables A.9 and A.10. The clusters in different rows and the objects in these clusters obtained from the left and right of the srPSM are not same in terms of the columns and groupings. The differences in these two sets refer to the counterpart clusters in the other. A counterpart cluster may harbor more objects than its counterpart cluster. For instance, the counterpart cluster of 1 in the first set is cluster 2 in the second set. Cluster 2 in the second set has three objects from cluster 1 in the first set. The first object, 1a, in cluster 1 in the first set is assigned as a different object. Likewise, for the second set, cluster 1 points out cluster 4 in the first set as its counterpart cluster; 1a in cluster 1 in the second set is assigned as a different object. All the differences are mapped in dark grey cells. These marked objects are assigned to their nearest clusters. The nearest cluster for an object is determined using the cell that has the biggest value in the same row/column where this object is positioned in the original PSM TableA.5.

For instance, object 1a points to object, 1c, with the largest value of 0.63. Object 1a is already in cluster 1, in which 1c is located, so this object remains in the same cluster. Otherwise, it would be transported to the cluster in which 1c is located. Similarly, the other dark grey objects signed as different are searched in TableA.5 in the two sets. As a result, for the first set, 3c points out object 3b, with a value of 0.48, and retains its cluster; 5b points to 5a, with a value of 0.53, and retains its current cluster; 5c points to 5b, with a value of 0.48, and is transported to cluster 8, in which 5b is located. By the same token, 5d, 7c, 7d, 9d remain in their clusters and 8c is transported to cluster 8. The objects transported to new clusters are marked with a check sign for both sets in Table A.11. This phase is repeated until there is no transportation. Our experiments on several different visual datasets show that the objects of the clusters in the two sets can converge to their stationary transported points, as depicted in Table A.11 in several steps, usually at the first step without iteration, because the same original PSM is referenced to decide on object transportation. The expansive attribute of the ACE method is finalized after this moment, where there is no transportation, but differences remain between the two sets. As can be seen in this example, the number of clusters is different in the two sets (i.e., the number of clusters is 9 for the first set and 10 for the second): the differences are all transported to some other empty row to form a new cluster, as seen in row 40 in Table A.12. Consequentially, the two sets turn into the exact same sets. To equalize the number of clusters for these two sets after previous transportations, the objects signed as different would be 5b, 5c, and 5d in cluster 8 in the first set and these objects are transported to any other empty row to form a new cluster, by which these two sets become the same in terms of the number of clusters and objects in these clusters, as shown in Table A.14.

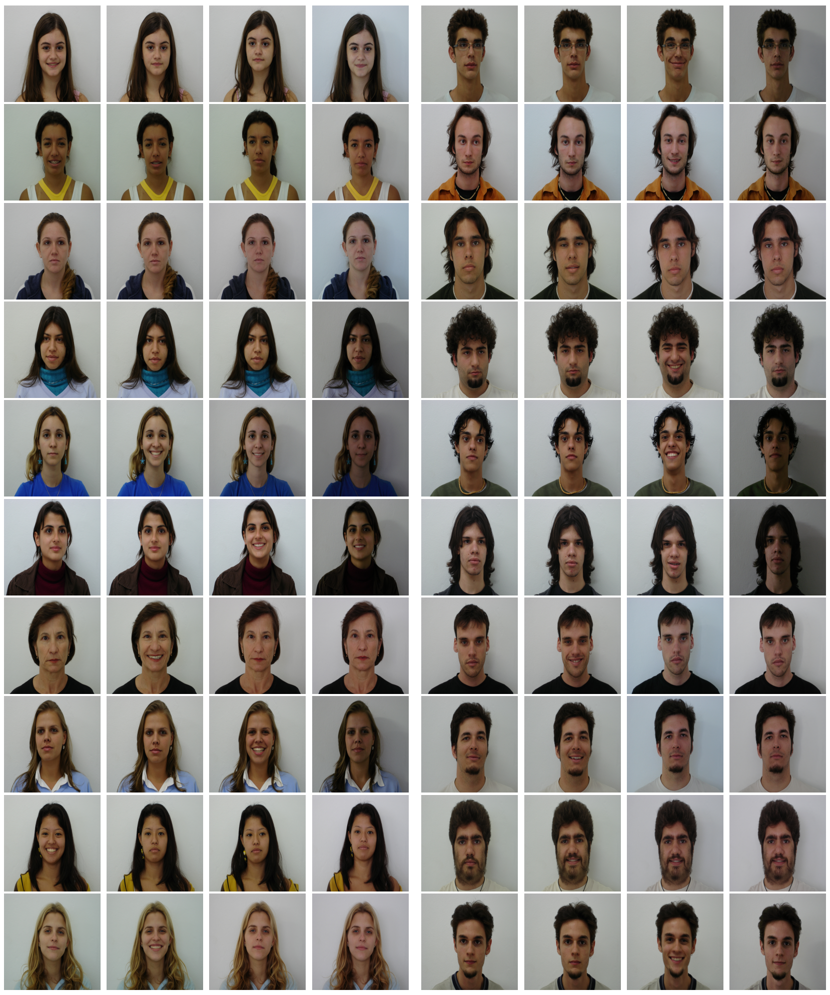


Figure 9: Frontal faces belonging to 10 females (left) and males (right) from the FEI database.

The idea behind the mixture of agglomerative, contractible and expandable attributes is to correct previous wrong decisions, so that high-quality clustering can be achieved. Clustering terminates and the final clustering result is reached at number 15 in Fig. 1, after the ACE phases are completed. ACE converges to a well-partitioned solution from a weaker grouping to a stronger clustering following these phases employed in the ACE technique.

4. Datasets & experimental study design

In order to test the efficacy of our proposed method, we have carried out a set of experiments on visual datasets: three face datasets [38] – the Tarrlab, FEI, and Caltech sets – and one leaf object dataset from Caltech [39]. Each of these four datasets is differentiated from the others by various distinctive characteristics to quantify comparative results. The Tarrlab face dataset contains multiple images for 200 individuals of different races with consistent lighting, multiple views, and real emotions. The choice of this database is primarily motivated by both the high number of image samples and quality of images it contains with 250 x 250 pixels and *no background noise*. However, many individuals in this database are disguised with wigs and sunglasses. These images were excluded from our study, because, analyzing of individuals together with disguised samples is out of the scope of this study. In addition, many faces are turned left or right with profile rotation of up to about 180°. Frontal faces are captured based on the template in Fig. 2 automatically by the application depicted in Fig. 10. Profile faces can also be explored separately as explained in the user manual that is in the supplementary materials of the manuscript. In the literature, these two subjects have been studied in different context based on the different characteristics of their features (i.e., feature extraction algorithms give exactly different features that cannot be compared to find a similarity between a profile sample and a frontal sample), and profile and frontal faces are not examined together at the same time because it would be neither reasonable nor fair to compare exactly different features by targeting the same group. The total number of the remaining frontal faces in this database is 508 for 88 individuals. The FEI face database contains a set of face images, 14 each of 200 individuals, with a total of 2800 images. All images have *a white homogeneous background* with profile rotation of up to about 180°, 640 x 480 pixels. The images were captured under *different light variation*. After excluding face images of side profiles, the number of the remaining images of the front-view of faces that we can use in this database is 800, 4 images each for 200 individuals. The face images are in these three datasets are under different *lighting, facial expression, and background* conditions. We chose the Caltech leaves dataset because this dataset consists of 186 images with 896 x 592 pixels *against distinctive different backgrounds*. The features of the objects in this dataset are completely different from those of the face objects in other three datasets, which helps to better quantify the CVOIPSM methodology presented in this study in terms of its application to different kinds of object sets.

CVOIPSM has been tested using both small and the largest possible datasets in the databases mentioned above. The downsized datasets are used to explain the methodology explicitly and to better compare the results, whereas the large datasets are deployed to explore CVOIPSM's performance for big datasets. First, regarding the denominator of the three face datasets, we use almost all the individuals in the smallest dataset, the Caltech face dataset, and we use the same number images from the other two face datasets to better compare the results and to aid readers in understanding and implementing CVOIPSM. In this sense, we incorporated 80 images representing 20 individuals, 4 for each individual, into our study from each of these face datasets, totaling 240 images. The data that we wish to cluster consist of two sets of images from each of these datasets: a set of female and a set of male face images depicted in Figures 8, 9, and 11. In addition, our methodology is run on the Caltech leaves dataset, using 100 images in 10 groups to represent the downsized datasets, as displayed in Fig. 12. We have also applied CVOIPSM to the largest possible sets from these databases, whose results are presented in Section 5. In this way, we aim to evaluate the efficiency

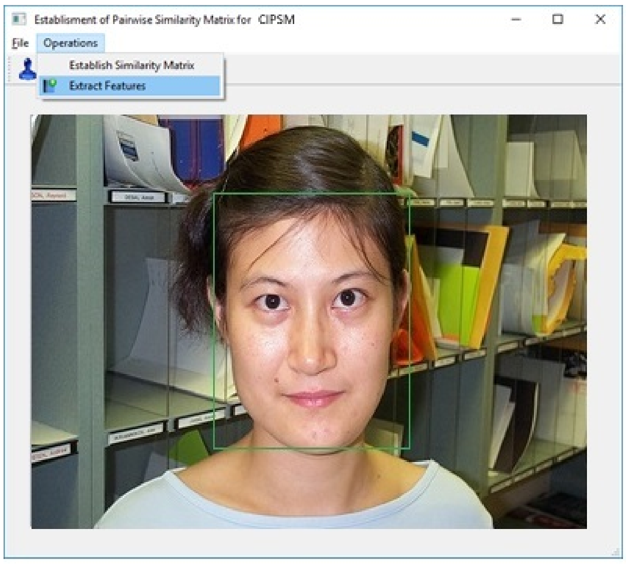

Establisment of Pairwise Similarity Matrix for CIPSM
File
Operations
Establish Similarity Matrix
Extract Features

Figure 10: The first component of CVOIPSM: First, the "Train All Databases" function implements the steps mentioned in subsection3.1. A PSM is generated by employing the "Establish Similarity Matrix" function, which is a learning phase on the pair matching task in which all objects acquired from the images in one directory are put into the application in a changing comparing set using leave-one-out cross-validation.

and viability of our methodology based on accuracy results acquired from both small and large datasets.

A user-friendly interface was implemented in the C++ programming language to carry out the methods described in subsection 3.1. A screenshot of the interface for PSM acquisition is depicted in Fig. 10. The application (i.e., smApplication) is in the supplementary materials of the manuscript with a user manual. With these implementations, objects in images in large datasets can be acquired in several minutes; a PSM can also be established in several minutes based on the similarities among object features. The sample images are cropped and converted to grayscale, and histogram equalization and median filtering are performed. Following image normalization to a fixed size using interpolation and extrapolation methods, PCA is conducted to extract the essential features from acquired objects. The eigenfaces and the mean image belonging to the Tarrlab female dataset obtained from the interface in Fig. 10 are displayed in Fig. 13 as an example. Euclidean distance is employed to acquire the similarities between the objects by comparing their features. The acquired PSM for the Tarrlab female face dataset is displayed in Table A.5. Other PSMs for downsized datasets used in these experiments are mentioned in the supplementary materials [2]. PSMs are generated using the interface (Fig. 10); the implementation of the first component of CVOIPSM is presented in the supplementary materials for users who would like to test their own datasets. A PSM is created for the Caltech leaves dataset by applying PCA on the leaf objects acquired using the approach mentioned in Section 3 (Figures 3 and 4) following image processing steps similar to those above. PSMs are constructed using cropped objects both with and without background removal (Figures 4c and 4b, respectively). The background removal technique is specified in Fig. 1, A.1-2. In this way, we aim to highlight both the statistical significance of background removal before feature acquisition for cluster analysis, and consequently the imperative action of obtaining better features to result in better visual object clustering with CIPSM. The better the features, the better the CIPSM clustering results.

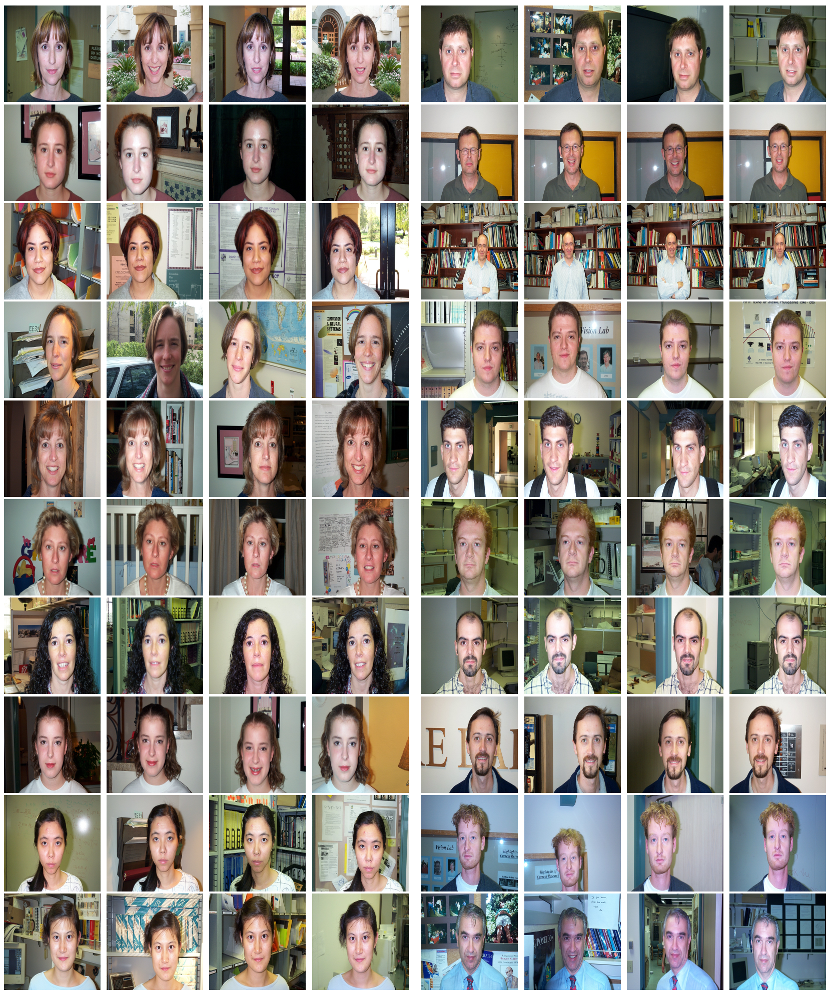


Figure 11: Frontal faces belonging to 10 females (left) and males (right) from the Caltech database.

The second component is performed to cluster the visual objects using PSMs. In this part, Rk-means, where k=4, is employed in the PSM columns to remove the least similar objects in three clusters while the objects in one

[2]as smTarFemale, smTarMale, smFEIFemale, smFEIMale, smCalFemale, and smCalMale, in .csv format

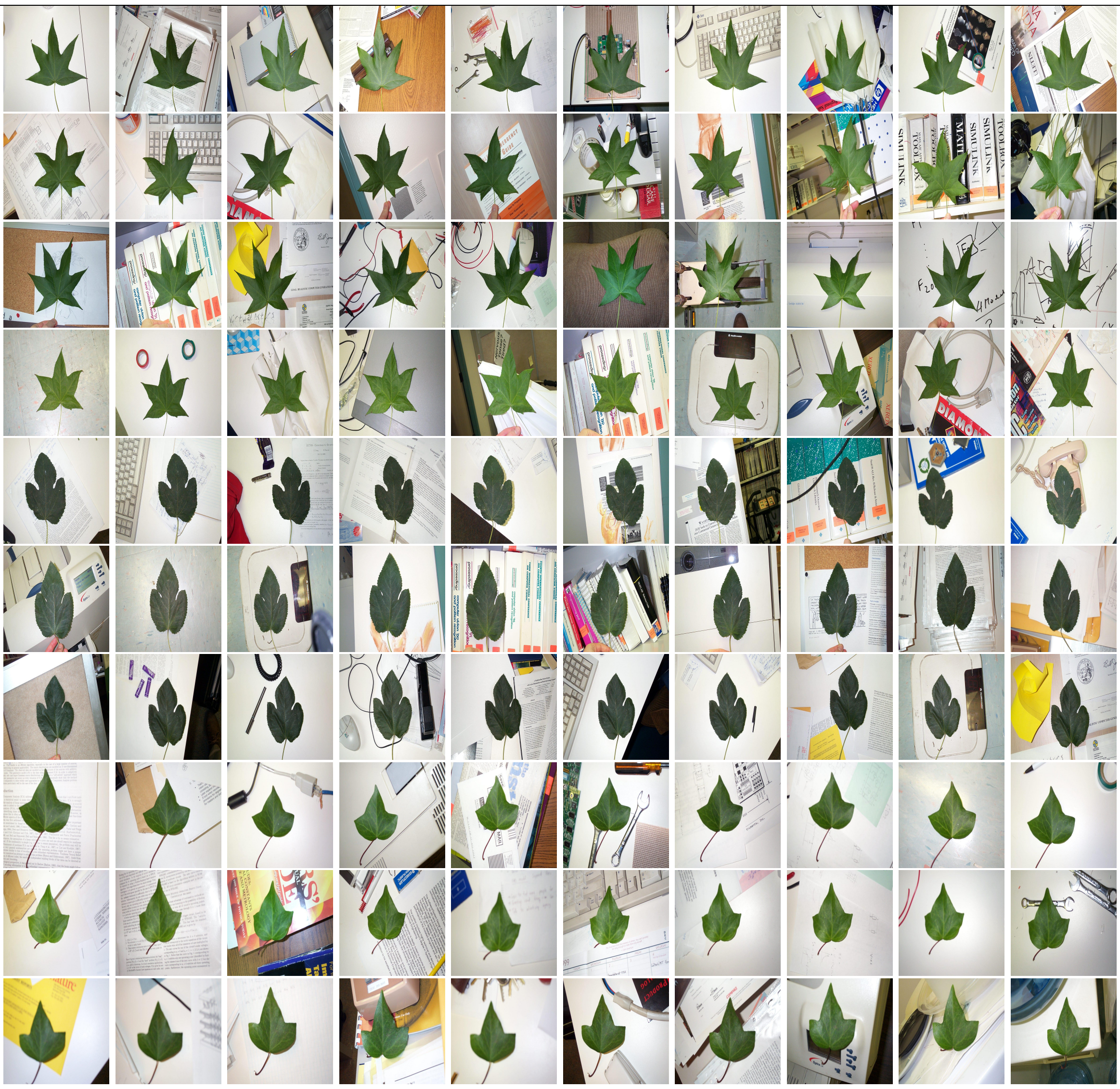
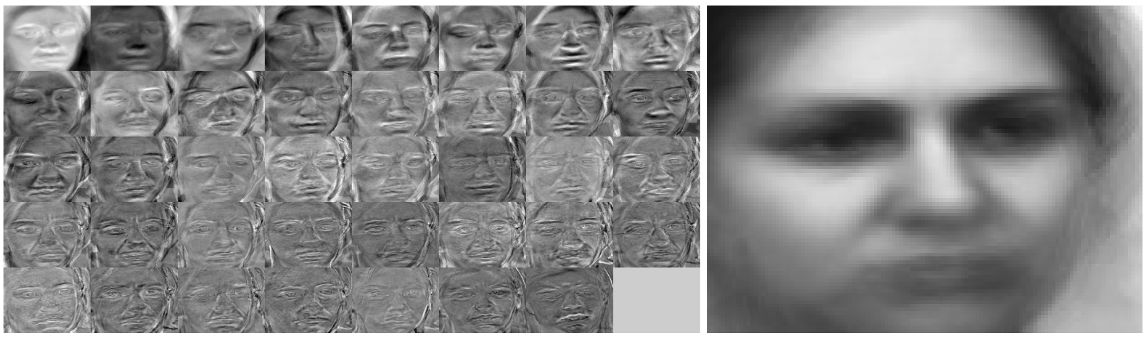


Figure 12: Leaves dataset from the Caltech database. Each image is numbered such that a1-a10 refer to the images in the first row, whereas j1-j10 refer to the images in the last row.



(a) Eigenfaces (b) Mean face

Figure 13: (a) Eigenfaces of 10 females that comprise 40 frontal faces from the Tarrlab dataset (Figures 8), (b) mean face(ψ).

cluster are retained. The grey cells in Table A.5 are removed from the PSM; the remaining cells are displayed in white. However, uncertainty in cluster assignment for most objects is evident; practically, a query point may belong to different clusters with different membership values ranging from 0 to 1, as can easily be seen in Table A.5. The files for the downsized datasets used in this study after Rk-means is performed are in the supplementary materials [3]. About 20% of the cells remain in the rPSM after Rk-means is run. Hence, the calculation costs are alleviated and finally, the remaining rPSMs can be analyzed by the ACE method after establishing symmetry in rPSMs (srPSM), as mentioned in Section 3.2. In-depth analysis is carried out with the srPSM and the exact clustering employs the ACE method, by which the most similar images are brought together and each image is assigned to a specific group. Eventually, clusters are established using the srPSM pattern using the values in the cells without dictating a fixed number of clusters beforehand. The output files specified as full_matrix, cluster_left, and cluster_right that are the outputs of smLeftRight.exe are included in the supplementary materials [4]. The output files, cluster_left_result.csv and cluster_right_result.csv, after smFinalResult.exe is run are in the supplementary materials as well [5].

Furthermore, we tested our method on the largest possible datasets in the databases: 88 individuals with 508 frontal faces, ranging from 4 to 9 per individual, for the Tarrlab database; 200 individuals, with 4 images per individual, for a total of 800 frontal face images for the FEI database; and 13 types of leaf objects ranging from 10 to 18 examples each, for a total of 186 images from the Caltech leaves database. We had already used all the individuals in the Caltech face database as a denominator of other datasets on the downsized dataset previously. Therefore, we could not test our method on larger datasets for this face dataset because of the limited number of images. The results obtained from the tests were evaluated and validated using the datasets that are already classified in the databases. In other words, we compared the obtained results to externally known information about the correct number of clusters and correct object clustering.

5. Results

We measured the accuracy of the clustering results based on the known labels (i.e., desired results) of classes in the databases. The final cluster outcomes of CIPSM for the downsized Tarrlab, FEI, and Caltech datasets are displayed in Table 2 without background removal and in Table 3 with background removal.

The clustering success rates achieved without background removal (Fig. 4b) were 97.5%, 98.75%, and 80% for the Tarrlab, FEI, and Caltech face datasets respectively and 81% for the Caltech leaves dataset, for an average of 89.31%. In addition, the number of clusters (cl) generated were 18, 20, 17, and 10 for the Tarrlab, FEI, and Caltech faces and Caltech leaves datasets, respectively, where the desired number of clusters (cl_d) for face datasets is cl_d = 20 and cl_d = 10 for the leaf dataset. Conversely, clustering success rates with background removal (Fig. 4b) were 97.5%, 98.75%, and 87.5% for the Tarrlab, FEI, and Caltech face datasets respectively and 92% for the Caltech leaves dataset, averaging 93.94% have been achieved and the number of generated clusters (cl) are 18, 20, 20, and 10, respectively. There is no difference in accuracy with and without background removal for the Tarrlab and FEI datasets, whose backgrounds do not change significantly, whereas there is a considerable difference in accuracy for the Caltech datasets, whose backgrounds change considerably, between the two cases. The results of downsized datasets are presented in Fig. 14.

The success rates on the largest possible datasets with background removal were 81% for the Tarrlab (n = 508), 83% for the FEI (n = 800), and 87.5% for the Caltech leaves dataset (n = 186). The number of clusters for the Tarrlab face, FEI face, and Caltech leaves datasets were cl = 73, cl = 183 and cl = 11, respectively, where cl_d = 88, cl_d = 200, and cl_d = 13. Accuracy on both downsized and full datasets with background removal are summarized in Fig. 15.

[3]as TarFem_Rk_Means, TarMale_Rk_Means, FEIFem_Rk_Means, FEIMale_Rk_Means, CalFem_Rk_Means, and CalMale_Rk_Means, in .csv format.

[4]as TarFem_full_matrix, TarFem_cluster_left, TarFem_cluster_right, TarMale_full_matrix, TarMale_cluster_left, TarMale_cluster_right, FEIFem_full_matrix, FEIFem_cluster_left, FEIFem_cluster_right, FEIMale_full_matrix, FEIMale_cluster_left, FEIMale_cluster_right, CalFem_full_matrix, CalFem_cluster_left, CalFem_cluster_right, CalMale_full_matrix, CalMale_cluster_left, and CalMale_cluster_right in .csv format.

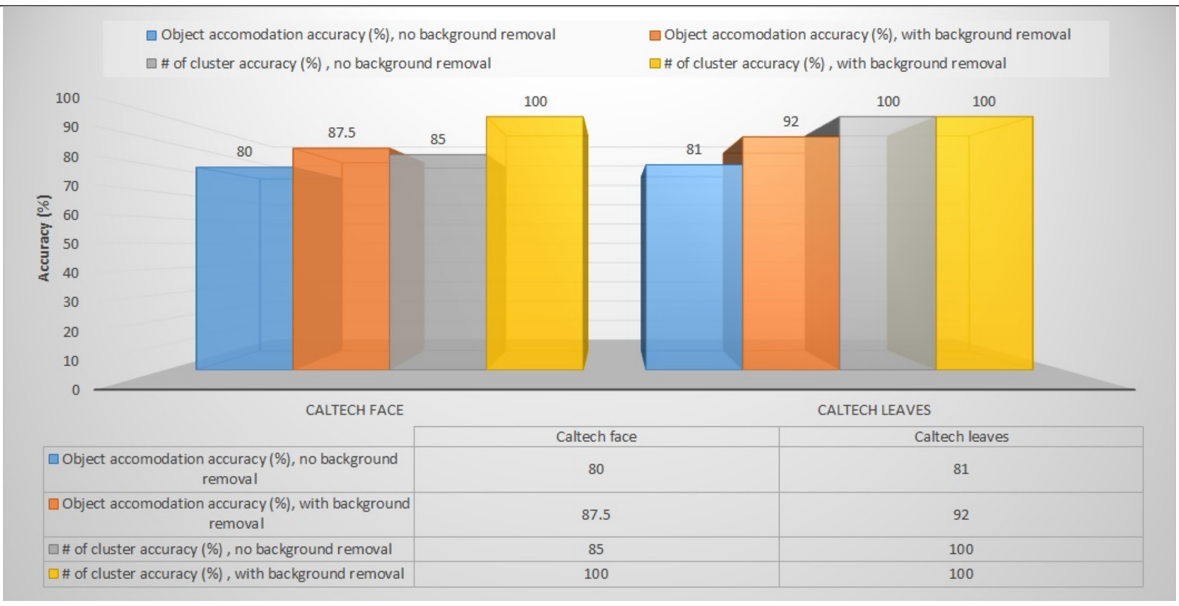
[5]as TarFem_cluster_left_result, TarFem_cluster_right_result, TarMale_cluster_left_result, TarMale_cluster_right_result, FEIFem_cluster_left_result, FEIFem_cluster_right_result, FEIMale_cluster_left_result, FEIMale_cluster_right_result, CalFem_cluster_left_result, CalFem_cluster_right_result, CalMale_cluster_left_result, and CalMale_cluster_right_result, in .csv format.

Table 2: Final clustering of downsized datasets acquired using CVOIPSM: *No background subtraction is applied to the acquired objects*. Objects represented by bold letters are not clustered correctly. Clusters for females (left), males (right), and leaves (far right). Two images (e1 and i1) are grouped in the wrong clusters for the Tarrlab female images, which corresponds to a 95% success rate. All images are grouped correctly for the Tarrlab males, 100%, if you exclude clusters 2 and 8, in both of which two people are grouped together. The mean success rate for females and males in the Tarrlab dataset was 97.5%. The desired number of clusters, 10, was correctly generated for females; but the number of clusters for males was 8, lacking 2 clusters in total. Likewise, the success rate for FEI females was 97.5% with 10 desired clusters, whereas it was 100% with 10 desired clusters for FEI males, averaging 98.75%; accuracy was 72.5% with 8 clusters for Caltech females, whereas it was 87.5% with 9 clusters for Caltech males, averaging 80%. The clustering of Caltech leaves resulted in 81% accuracy with 10 clusters.

Cluster #	Tarrlab females	Tarrlab males	FEI females	FEI males	Caltech females	Caltech males	Caltech leaves
1	a1,a2,a3,a4	a1,a2,a3,a4	a1,a2,a3,a4	a1,a2,a3,a4	a1,a2,a3,a4,f1,f2,f3,f4	a1,a2,a3,a4,j1,j2,j3,j4,**h3**,**i2**,**c4**	a1,a2,a4,a5,a6,a7,a9,a10,**d4**,**d8**
2	b1,b2,b3,b3	b1,b2,b3,b4,i1,i2,i3,i4	b1,b2,b3,b3	b1,b2,b3,b3	b1,b4	b1,b2,b3,b4	b1,b2,b3,b4,b5,b6,b7,b8,b9,b10
3	c1,c2,c3,c4	c1,c2,c3,c4	c1,c2,c3,c4	c1,c2,c3,c4		c1,c2,c3	c1,c2,c3,c4,c5,c6,c7,c8,c9,c10
4	d1,d2,d3,d4	d1,d2,d3,d4	d1,d2,d3,d4	d1,d2,d3,d4	d1,d2,d3	d2,d3,d4,**h1**	d1,d2,d3,d5,d6,d7,d9,d10,**a3**,**a8**
5	e2,e3,e4	e1,e2,e3,e4	e1,e2,e3,e4,**f1**	e1,e2,e3,e4	e1,e3	e1,e2,e3,e4	e1,e2,e3,e4,e5,e7,e9,**f6**
6	f1,f2,f3,f4	f1,f2,f3,f4	f2,f3,f4	f1,f2,f3,f4		f1,f2,f3,f4	f1,f2,f3,f5,f7,f8,f9,f10,**e6**,**e10**,**g4**,**g10**
7	g1,g2,g3,g4	g1,g2,g3,g4	g1,g2,g3,g4	g1,g2,g3,g4	g1,g2,g4,**c1**,**e2**,**e4**	g1,g2,g3,g4	g1,g2,g3,g5,g6,g7,g8,g9,**f4**,**e8**
8	h1,h2,h3,h4	h1,h2,h3,h4,j1,j2,j3,j4	h1,h2,h3,h4	h1,h2,h3,h4	h2,h3,h4,**b3**,**c2**,**d4**	h2,h4,**d1**	h1,h2,h3,h7,h8,h9,**i3**,**i9**,**j6**,**j8**
9	i2,i3,i4		i1,i2,i3,i4	i1,i2,i3,i4	i1,i2,i3,i4,**c3**,**c4**,**g3**,**h1**	i1,i3,i4	i1,i2,i4,i5,i6,i7,i8,i10,**h5**,**h6**
10	j1,j2,j3,j4,**e1**,**i1**		j1,j2,j3,j4	j1,j2,j3,j4	j1,j2,j3,j4,**b2**		j1,j2,j3,j4,j5,j7,j9,j10,**h4**,**h10**

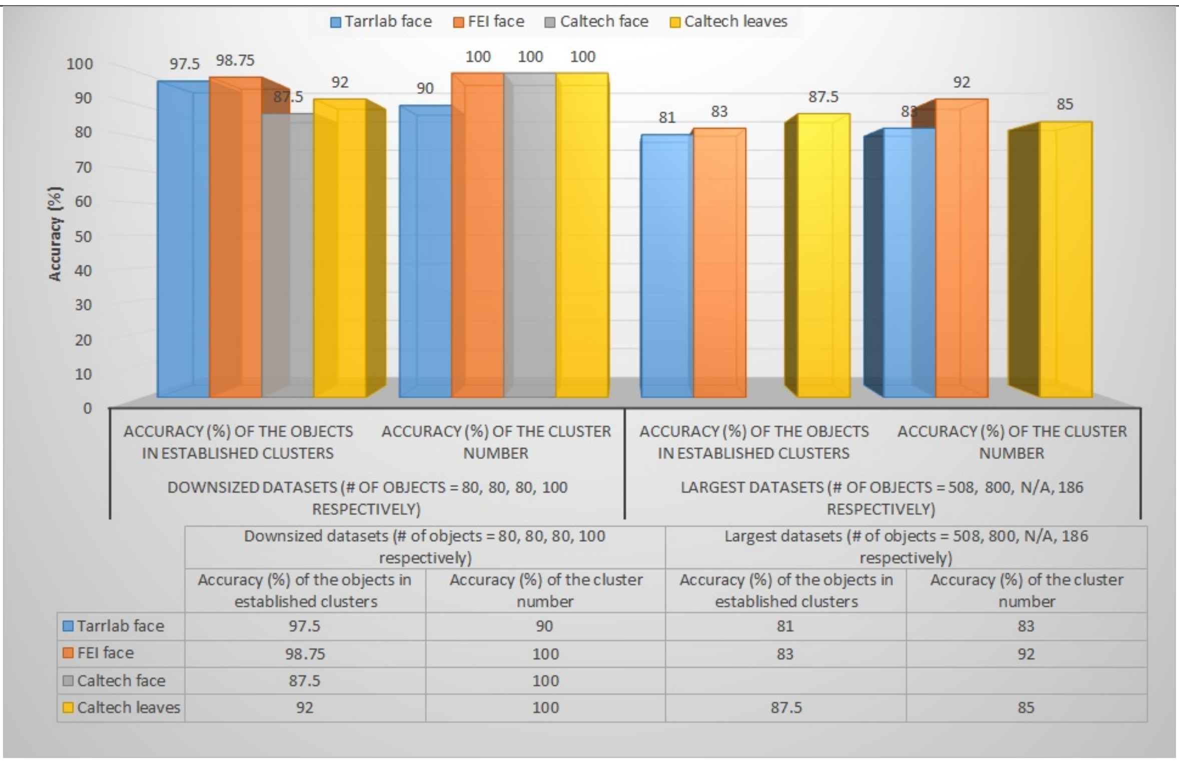
Table 3: Final clustering of downsized datasets acquired using CVOIPSM. Objects represented by bold letters are not clustered correctly. *Background subtraction is applied to the acquired objects*. Clusters are of females (left), males (right), and leaves (far right) datasets. The mean success rate for females and males from the Tarrlab dataset was 97.5%. The desired number of clusters, 10, was generated correctly for females; but for males, CVOIPSM produced two fewer clusters, for a total of 8. Likewise, the success rate for the FEI female dataset was 97.5% with 10 desired clusters, whereas it was 100% with 10 desired clusters for the FEI male dataset, averaging 98.75%; it was 85% with 10 clusters for Caltech females, and 90% with 10 clusters for Caltech males, averaging 87.5%. Clustering of the Caltech leaves database resulted in 92% accuracy with 10 clusters.

Cluster #	Tarrlab females	Tarrlab males	FEI females	FEI males	Caltech females	Caltech males	Caltech leaves
1	a1,a2,a3,a4	a1,a2,a3,a4	a1,a2,a3,a4	a1,a2,a3,a4	a1,a2,a3,a4	a1,a2,a3,a4,**h3**,**i2**	a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,**d4**,**d8**
2	b1,b2,b3,b3	b1,b2,b3,b4,i1,i2,i3,i4	b1,b2,b3,b3	b1,b2,b3,b3	b1,b3,b4	b1,b2,b3,b4	b1,b2,b3,b4,b5,b6,b7,b8,b9,b10
3	c1,c2,c3,c4	c1,c2,c3,c4	c1,c2,c3,c4	c1,c2,c3,c4	c3,c4	c1,c2,c3,c4	c1,c2,c3,c4,c5,c6,c7,c8,c9,c10
4	d1,d2,d3,d4	d1,d2,d3,d4	d1,d2,d3,d4	d1,d2,d3,d4	d1,d2,d3	d2,d3,d4,**h1**	d1,d2,d3,d5,d6,d7,d9,d10
5	e2,e3,e4	e1,e2,e3,e4	e1,e2,e3,e4,**f1**	e1,e2,e3,e4	e1,e2,e3,e4	e1,e2,e3,e4	e1,e2,e3,e4,e5,e7,e9,e10
6	f1,f2,f3,f4	f1,f2,f3,f4	f2,f3,f4	f1,f2,f3,f4	f1,f2,f3,f4	f1,f2,f3,f4	f1,f2,f3,f5,f6,f7,f8,f9,f10,**e6**
7	g1,g2,g3,g4	g1,g2,g3,g4	g1,g2,g3,g4	g1,g2,g3,g4	g1,g2,g4,**c1**	g1,g2,g3,g4	g1,g2,g3,g4,g5,g6,g7,g8,g9,**f4**,**e8**
8	h1,h2,h3,h4	h1,h2,h3,h4,j1,j2,j3,j4	h1,h2,h3,h4	h1,h2,h3,h4	h2,h3,h4,**c2**,**d4**	h2,h4,**d1**	h1,h2,h3,h5,h7,h8,h9,h10,**i3**,**j8**
9	i2,i3,i4		i1,i2,i3,i4	i1,i2,i3,i4	i1,i2,i3,i4,**g3**,**h1**	i1,i3,i4	i1,i2,i4,i5,i6,i7,i8,i9,i10,**h6**
10	j1,j2,j3,j4,**e1**,**i1**		j1,j2,j3,j4	j1,j2,j3,j4	j1,j2,j3,j4,**b2**	j1,j2,j3,j4	j1,j2,j3,j4,j5,j6,j7,j9,j10,**h4**



	Caltech face	Caltech leaves
🟦 Object accomodation accuracy (%), no background removal	80	81
🟧 Object accomodation accuracy (%), with background removal	87.5	92
⬜ # of cluster accuracy (%) , no background removal	85	100
🟨 # of cluster accuracy (%) , with background removal	100	100

Figure 14: Clustering accuracy of Caltech objects and number of clusters with and without background removal. This figure mainly shows the importance of background subtraction from specific objects (ROIs) in increasing accuracy in cluster analysis of visual domains.



	Downsized datasets (# of objects = 80, 80, 80, 100 respectively)		Largest datasets (# of objects = 508, 800, N/A, 186 respectively)	
	Accuracy (%) of the objects in established clusters	Accuracy (%) of the cluster number	Accuracy (%) of the objects in established clusters	Accuracy (%) of the cluster number
🟦 Tarrlab face	97.5	90	81	83
🟧 FEI face	98.75	100	83	92
⬜ Caltech face	87.5	100		
🟨 Caltech leaves	92	100	87.5	85

Figure 15: Clustering accuracy and # of clusters of small and largest datasets after applying a background removal technique.

6. Discussion

In this study, the unsupervised grouping of objects in visual datasets into similar partitions is examined using a hybrid non-parametric clustering methodology that does not require the desired number of clusters to be known in advance, especially for large datasets. The presented methodology combines several well-known and new methods into one coherent hybrid procedure for the unsupervised grouping of similar objects. The major contribution of our study is the establishment of a clustering method to partition similar objects for both small and large datasets that provides a framework for clustering specific objects in visual domains. Our framework may be viewed as a three-level unsupervised approach: first, we acquire features with PCA and employ Euclidean distance to obtain similarities among the objects acquired from images; second, we use an Rk-means method on these similarity values both to remove the objects with the lowest similarities and to obtain the most similar objects; third, the remaining similarity values are compared in order to assign objects to their nearest clusters.

In this study, we aim to counter the deficiencies of the current widely used clustering schemes mentioned in Section 2. Both agglomerative and divisive hierarchical algorithms are static. They never undo what has been done previously, which means that objects that are committed to a cluster in early stages cannot move to another cluster. In other words, once a cluster is split or two clusters are merged, the split objects will never come together in one cluster or the merged objects will be never in the same cluster, no matter whether the split or the merge was the correct action. In practice, however, some splits or merges may not be correct, and there may be a need to rearrange the partitions. This problem causes inaccurate clustering, especially for poorly separated datasets [35]. However, in CVOIPSM, the ACE method corrects previous wrong decisions so that high-quality clustering can be achieved. In addition, one of the most difficult decisions is how many clusters there should be. For most current clustering methods, such as centroid-based (e.g., k-means) clustering, the number of clusters is dictated by the user beforehand and data points are assigned to one of those fixed *k* clusters using changing centroids throughout an iterative process. Everitt [12] suggests that the largest number of clusters should be dictated (unless external information from the subject matter suggests a suitable choice) where different stopping rules suggest different numbers of clusters. However, one of the advantages of our clustering approach is that there is no need to know the number of clusters in advance, which significantly helps to overcome the misclassification problem, especially for large datasets. In our approach, the desired number of clusters is calculated by the methodology that analyzes features in a manner that uses the intra-/interrelated patterns of similarities among distinct features belonging to objects. The main problem of inter-class overlapping and overfitting, which is an important issue in cluster analysis, is managed effectively with CVOIPSM.

The success of CVOIPSM depends on the quality of the features initially acquired, as presented in Section 3.1. Better feature extraction and selection algorithms would increase the clustering accuracy rate of CIPSM, as explained in Section 3.2. CVOIPSM is tested on datasets with and without background removal for the objects detected in images. Background removal was shown to increase cluster analysis accuracy, particularly where there is a large variation in object backgrounds. For example, object clustering accuracy is increased from 80% to 87.5% for Caltech face objects and from 81% to 92% for Caltech leaf objects. Similarly, cluster number accuracy is increased from 85% to 100% for Caltech face objects, as shown in Fig. 14. However, the success rate is same for the FEI datasets, in which there is large variation in image lighting, before and after background removal. It should be noted that first, there is no large background difference for these objects and second, histogram equalization is performed on cropped objects to enhance illumination variation; thus, better features are acquired despite significant variation in illumination. Our data are normally distributed; a statistical Z-test analysis was carried out to compare the difference in success rates with and without background removal. The null hypothesis, that there is no significant difference in CVOIPSM success rates with and without background removal ($\mu = \mu_0$), can be rejected, with $p<0.01$. As a result, we can conclude that a background noise removal technique should be applied to objects that have intensive background noise to acquire better features representing the objects in ROIs. The more distinctive the features for any dataset, the better the results acquired are, especially when specifying cluster number.

The success rates of CVOIPSM decreases moderately when datasets are much larger, as presented in Fig. 15. However, we have analyzed the significance of these differences, namely, 14%, 15%, and 4.5% for the Tarrlab, FEI, and Caltech leaves databases, respectively. Our data are normally distributed; therefore, a statistical Z-test analysis has been performed to compare the differences between the success rates on the downsized and full datasets. The null hypothesis, that there is no significant difference in the success rates between two datasets ($\mu = \mu_0$), is supported by the results of the Z-test, with $p>0.01$. Thus, we can conclude that CVOIPSM can perform well even for large datasets.

We do not employ any outlier removal parameter before clustering for our datasets in which objects are similar to each other and there is no outlier. However, the outliers in datasets can be excluded using optional "-O thresholdValue" parameter as explained in the user manual in the supplementary materials. If the similarity values of an object with respect to every object in the dataset are less than the entered threshold similarity value, it is accepted as an outlier, meaning that there is no object to be grouped together.

There are many studies [40, 41, 42] that explore the viability of off-the-shelf clustering techniques on a variety of domains. For instance, Etienne [42] analyses the capability of off-the-shelf techniques on determining the desired number of clusters in 21 selected benchmark datasets that contain different numbers of objects and clusters using 17 clustering techniques; these datasets cannot be clearly divided into desired non-overlapping groups by traditional clustering algorithms. The average success rates of these techniques in terms of all datasets range from around 10% to 60% with a mean absolute difference $\pm\sigma$ ranging from 4.52 ± 5.05 to 2.19 ± 2.50, which is far below the results obtained in our study; species in these datasets are often assigned to wrong groups through the clustering process of 17 clustering techniques. It can be concluded from the findings of these studies that off-the-shelf clustering techniques suffer from the need to specify the desired number of clusters in addition to assigning samples to correct groups.

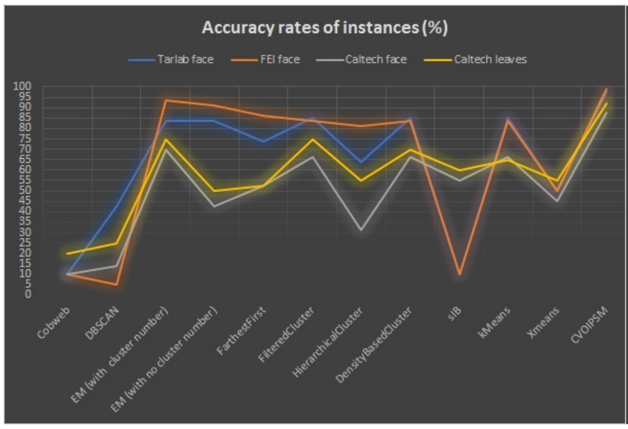
Furthermore, our datasets were tested with the most widely-used clustering techniques using the popular Waikato Environment for Knowledge Analysis (WEKA) application suite [6]. These techniques are cobweb, DBSCAN, EM (with cluster number), EM (without cluster number), FarthestFirst, FilteredCluster, HierarchicalCluster, Density-BasedCluster, sIB, kMeans and XMeans. The number of clusters is required to be defined by the user before processing these techniques, except cobweb, DBSCAN and our technique CVOIPSM. EM works either with or without a pre-defined cluster number. The detailed results obtained from the Weka tool are in the supplementary materials of the manuscript. The summary of these results is presented in Table 4 and accuracy rates of instances and cluster numbers are depicted in Fig. 16 based on the results presented in Table 4. Our technique, CVOIPSM outperforms the

[6]WEKA developed at the University of Waikato is freely available from http://www.cs.waikato.ac.nz/ml/weka/.

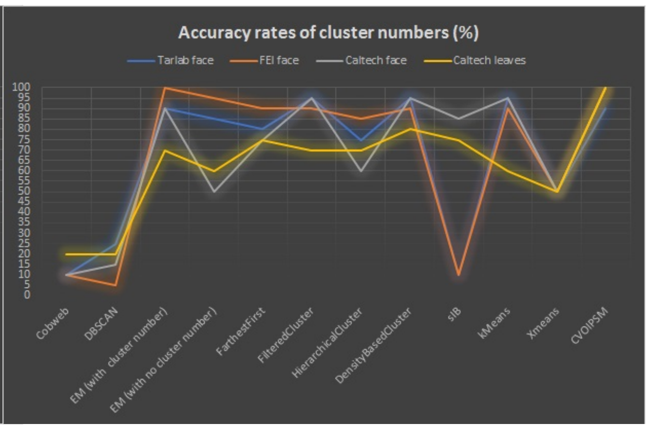
other techniques in four datasets regarding the instance accuracy and it outperforms these techniques for two datasets (i.e., Caltech face, Caltech leaves) regarding the cluster number, but not for Tarlab dataset in which FilteredCluster, HierarchicalCluster and kMeans (i.e., 95%) are better than CVOIPSM (i.e., 90%) and it is same with the EM (with cluster number) (i.e., 100%). However, we would like to emphasize that a cluster number (C_n=20) is required to be defined for these techniques whereas it is not required for CVOIPSM, and moreover, accuracy rates of placing instances for these techniques are smaller than CVOIPSM (i.e., 66.25% for FilteredCluster, HierarchicalCluster and kMeans < 87.50% and 93.75% for EM (with cluster number) < 98.75%). The techniques without pre-defined cluster numbers, namely Cobweb, DBSCAN performs poorly regarding the accuracy rates of cluster numbers obtained (i.e., <25%), which reduces the accuracy rates of placing instances into correct groups as well. EM (without cluster number) outperforms these two techniques (i.e., Cobweb and DBSCAN) with the accuracy rates (i.e., 85%, 95%, 50% and 60% respectively for our datasets) regarding finding the desired number of clusters. In conclusion, the proposed method, CVOIPSM performs similarly for all dataset in high accuracy rates regarding both the number of clusters and placing the instances in the desired groups as it can be noticed in Fig. 16.

Table 4: The results of the widely-used clustering techniques applied on our datasets: the accuracy rates of several techniques that outperform better than the other techniques are highligted in gray background and the best technique is highlighted in red color. The number of clusters is required to be defined by the user before processing these techniques, except cobweb, DBSCAN and our technique CVOIPSM. EM works either with or without a pre-defined cluster number. Instances may be grouped in a number of clusters different from the pre-defined cluster number.

	Prior values			Tarlab face				FEI face				Caltech face				Prior values			Caltech leaves			
Clustering techniques	Total instances	Total cluster number	Defined cluster number	Correct instances	Instance accuracy(%)	Obtained cluster number	Cluster number accuracy(%)	Correct instances	Instance accuracy(%)	Obtained cluster number	Cluster number accuracy(%)	Correct instances	Instance accuracy(%)	Obtained cluster number	Cluster number accuracy(%)	Total instances	Total cluster number	Defined cluster number	Correct instances	Instance accuracy(%)	Obtained cluster number	Cluster number accuracy(%)
Cobweb	80	20	N/A	8	10	2	10	8	10	2	10	8	10	2	10	100	10	N/A	20	20	2	20
DBSCAN	80	20	N/A	32	42.50	5	25	4	5	1	5	12	14	3	15	100	10	N/A	25	25	2	20
EM (with cluster number)	80	20	20	67	83.75	18	90	75	93.75	20	100	56	70	18	90	100	10	10	75	75	7	70
EM (with no cluster number)	80	20	N/A	67	83.75	17	85	73	91.25	21	95	34	42.50	10	50	100	10	N/A	50	50	6	60
FarthestFirst	80	20	20	59	73.75	16	80	69	86.25	18	90	42	52.5	15	75	100	10	10	52	52	7	70
FilteredCluster	80	20	20	68	85	19	95	67	83.75	18	90	53	66.25	19	95	100	10	10	75	75	7	70
HierarchicalCluster	80	20	20	51	63.75	15	75	55	81.25	17	85	25	31.25	12	60	100	10	10	55	55	7	70
DensityBasedCluster	80	20	20	68	85	19	95	67	83.75	18	90	53	66.25	19	95	100	10	10	70	70	8	80
sIB	80	20	20	8	10	19	10	8	10	2	10	44	55	17	85	100	10	10	60	60	7	70
kMeans	80	20	20	68	85	19	95	67	83.75	18	90	53	66.25	19	95	100	10	10	65	65	6	60
Xmeans	80	20	10/30	40	50	10	50	40	50	10	50	36	45	10	50	100	10	5/15	55	55	5	50
CVOIPSM	80	20	N/A	78	97.50	18	90	79	98.75	20	100	70	87.50	20	100	100	10	N/A	92	92	10	100



(a) Accuracy rates of instances in clusters.



(b) Accuracy rates of cluster numbers.

Figure 16: Clustering accuracy of several widely-used techniques along with our technique CVOIPSM based on Table 4.

7. Conclusions and future work

This study is one of the only comprehensive studies in the literature on clustering objects in visual domains and may provide new insights into cluster analysis. We explore the idea of clustering similar sets of objects in visual domains. In particular, we present a state-of-the-art framework that successfully clusters objects. The methodology that we present runs effectively by revealing interrelated patterns of similarities between objects. The merits of our approach, CVOIPSM, are substantiated by applying it to small and large datasets in several visual domains. We have performed a variety of experiments showing that our approach results in unsupervised grouping of similar objects with high accuracy for both small and large datasets, placing objects into correct groups and correctly determining the number of clusters without prior knowledge. The technique outperforms the well-known off-the-shelf techniques with respect to same datasets. The second part of the CVOIPSM methodology, CIPSM, is of crucial importance for the final and successful clustering of objects, and can be employed for unsupervised clustering on any kind of dataset whose similarity features can be acquired using other methods. Interested readers can find our executable files in the supplementary materials and can employ our technique on their datasets using their pairwise similarity matrices obtained from any dataset.

To summarize, our approach allows visual datasets to speak for themselves without using prior references or knowledge. The methodology proposed in this study leads to improvements in cluster analysis and is expected to influence the direction of unsupervised group discovery. The three-level unsupervised data analysis approach presented in this study is general and has the potential for broad application and, more specifically, may be applicable to essentially any type of data on which PCA or other feature extraction methods are run, including market research, medical, molecular, and genetic data. Our future work will explore our clustering technique on many other datasets.

Acknowledgements

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) (grant number B.14.2.TBT.0.06.01-219-5-123). We are grateful to Mahesan Niranjan, University of Southampton, for helpful suggestions during the early stages of this work. The study could not be realized without his guidance. The study has been improved technically thanks to the reviewers and attendees of 9th International Conference on Hybrid Artificial Intelligence Systems [43]. We would like to thank Caltech University for letting us use their visual datasets [38, 39].

References

[1] Deng, Y., Alto, P., Manjunath, B.S.: Unsupervised segmentation of color-texture regions in images and video. IEEE Transactions on Pattern Analysis and Machine Intelligence **23**(8) (2001) 800–810

[2] Liu, Y., Zhang, D., Lu, G.: Region-based image retrieval with high-level semantics using decision tree learning. Pattern Recognition **41**(8) (2008) 2554–2570

[3] Blaschke, T.: Object based image analysis for remote sensing. ISPRS Journal of Photogrammetry and Remote Sensing **65**(1) (2010) 2–16

[4] Blaschke, T., Hay, G.J., Kelly, M., Lang, S., Hofmann, P., Addink, E., Feitosa, R.Q., van der Meer, F., van der Werff, H., van Coillie, F., Tiede, D.: Geographic object-based image analysis towards a new paradigm. ISPRS Journal of Photogrammetry and Remote Sensing **87**(1) (2014) 180–191

[5] Estivill-Castro, V.: Spatial pyramid pooling in deep convolutional networks for visual recognition. ACM SIGKDD Explorations Newsletter **4**(1) (2002) 65–75

[6] Kuru, K., Niranjan, M., Tunca, Y., Osvank, E., Azim, T.: Biomedical visual data analysis to build an intelligent diagnostic decision support system in medical genetics. Artificial Intelligence in Medicine **62**(2) (2014) 105–118

[7] Iancu, C., Corcoran, P., Costache, G.: A review of face recognition techniques for in-camera applications. in signals, circuits and systems. Signals, Circuits and Systems **1, 1**(1) (2007) 1–4

[8] Kapoor, S., Khanna, S., Bhatia, R.: Facial gesture recognition using correlation and mahalanobis distance. International Journal of Computer Science and Information Security **7**(2) (2010) 267–272

[9] Calvo, A.R., Ruiz, F.T., Rurainsky, J., Eisert, P.: 2d-3d mixed face recognition schemes. In Delac, K., Grgic, M., Bartlett, M.S., eds.: Recent Advances in Face Recognition. In-Teh, Croatia (2008) 125–148

[10] Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York, NY, USA (2006)

[11] Witten, I.H., Frank, E., Hall, M.A.: Data mining: practical machine learning tools and techniques. 3rd edition edn. Elsevier, Berlington, USA (2011)

[12] Everitt, B.S., Landau, S., Leese, M.: Cluster Analysis. 5 edn. Wiley, West Sussex, UK (2011)

[13] Fauzdar, P., Kindri, S.: Comparitive analysis of k means and fuzzy c means algorithm. International Journal of Engineering Research and Technology **2**(6) (2013) 2088–2095

[14] Kim, T.K., Cipolla, R.: Mcboost: Multiple classifier boosting for perceptual co-clustering of images and visual features. In Koller, D., Schuurmans, D., Bengio, Y., Bottou, L., eds.: NIPS, Curran Associates, Inc. (2008) 841–856

[15] Antonopoulos, P., Nikolaidis, N., Pitas, I.: Hierarchical face clustering using sift image features. In JGuan, L., Hirota, K., eds.: Computational intelligence in image and signal processing. IEEE, Honolulu, HI (2007)

[16] Chandhok, C.: Color image segmentation usng k-means clustering. International Journal of VLSI and Signal Processing Applications **2**(3) (2012) 241–245

[17] Balcan, M.F., Liang, Y., Gupta, P.: Robust hierarchical clustering. Journal of Machine Learning Research **15** (2014) 4011–4051

[18] Rani, Y., Rohil, H.: A study of hierarchical clustering algorithm. International Journal of Information and Computation Technology **3**(1) (2013) 1225–1232

[19] Liu, Y., Ouyang, Y., Xiong, Z.: Incremental clustering using information bottleneck theory. International Journal of Pattern Recognition and Artificial Intelligence **25**(05) (2011) 695–712

[20] Ackerman, M., Dasgupta, S.: Incremental clustering: The case for extra clusters. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q., eds.: Advances in Neural Information Processing Systems 27, Curran Associates, Inc. (2014) 307–315

[21] Estivill-Castro, V.: Why so many clustering algorithms: a position paper. ACM SIGKDD Explorations Newsletter **4**(1) (2002) 65–75

[22] Yang, Z., Fang, J., Chittuluru, J., Asturias, F.J., , Penczek, P.A.: Iterative stable alignment and clustering of 2d transmission electron microscope images. Structure **20**(2) (2012) 237–247

[23] Godec, M., Sternig, S., Roth, P.M., Bischof, H.: Context-driven clustering by multi-class classification in an active learning framework. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. (2010) 19–24

[24] Karthikeyan, M., Aruna, P.: Probability based document clustering and image clustering using content-based image retrieval. Applied Soft Computing **13** (2012) 959–966

[25] Lahmiri, S.: The effect of mother wavelet and subband choce on clusterng of images database. International Journal of Computer Science, Engineering and Applications **1**(5) (2011) 13–20

[26] Elizabeth, C.B., Devi, K.U.K.: Spectral clustering of images in luv color space by spatial-color pixel classification. International Journal of Computer Applications **3**(9) (2010) 1–5

[27] Liu, Y., Wan, X.: Information bottleneck based incremental fuzzy clustering for large biomedical data. Journal of Biomedical Informatics **62**(05) (2016) 48–58

[28] Hamilton, N.A., Teasdale, R.D.: Visualizing and clustering high throughput sub-cellular localization imaging. BMC Bioinformatics **9**(81) (2008) 1–12

[29] Robotka, Z., Zemplni, A., Hajas, C., Seres, C., Balzs, S.: Genetic algorithms and grid technologies in clustering, an example: Clustering of images. Quality and Reliability Engineering International **24**(6) (2008) 693–703

[30] Cho, W.H., Na, I.S., Choi, J.Y., Lee, T.H.: Automatic classification for various images collections using two stages clustering method. Open Journal of Applied Sciences **3** (2013) 47–52

[31] Chandrashekar, A., Torresani, L., Granger, R.: Learning what is where from unlabeled images: joint localization and clustering of foreground objects. Machine Learning **94**(2) (2014) 261–279

[32] Silakari, S., Motwani, M., Maheshwari, M.: Color image clustering using block truncation algorithm. International Journal of Computer Science Issues **4**(2) (2009) 31–35

[33] Mekapothula, S.K., Kumar, V.J.: An efficient algorithm for segmentation using fuzzy local information c-means clustering. nternational Journal of Computer Science and Network Security **12**(10) (2012) 139–149

[34] Chen, N., Prasanna, V.K.: Semantic image clustering using object relation network. In Hu, S.M., Martin, R.R., eds.: Computational Visual Media. Volume 7633. Springer, Berlin Heidelberg (2012) 59–66

[35] Srinivas, M., Mohan, C.K.: Efficient clustering approach using incremental and hierarchical clustering methods. In: The 2010 International Joint Conference on Neural Networks (IJCNN), IEEE (July 2010) 1–7

[36] Wozniak, M., Graa, M., Corchado, E.: A survey of multiple classifier systems as hybrid systems. Information Fusion **16** (2014) 3–17

[37] Corchado, E., Wozniak, M., Abraham, A.: Recent trends in intelligent data analysis. Neurocomputing **126** (2014) 1–2

[38] Markus Weber: Frontal face dataset. `http://www.vision.caltech.edu/Image_Datasets/faces/faces.tar` (1999) Online; accessed 2 March 2013.

[39] Markus Weber: Caltech leaves dataset. `http://www.vision.caltech.edu/Image_Datasets/leaves/leaves.tar` (1999) Online; accessed 29 January 2015.

[40] Fang, Y., Wang, J.: Selection of the number of clusters via the bootstrap method. Comput. Stat. Data Anal. **56** (2012) 468–477

[41] Arbelaitz, O., Gurrutxaga, I., Muguerza, J.: An extensive comparative study of cluster validity indices. Pattern Recognition **46** (2013) 243–256

[42] Lord, E., Willems, M., Lapointe, F.J., Makarenkov, V.: Using the stability of objects to determine the number of clusters in datasets. Information Sciences **393** (2017) 29–46

[43] Kuru, K.: A novel hybrid clustering approach for unsupervised grouping of similar objects. In: 9th International Conference on Hybrid Artificial Intelligence Systems, Springer (2014) 642–53

Appendix A. Outcome of CIPSM method applied on Tarlab datasets

Table A.5: The similarity table for Tarrlab female images in Fig. 8 and the remaining similarity values after Rk-means is employed: grey cells are removed from the PSM for further processes. The value of 1.00 is placed for the orthogonal cells having null values since the similarity between two identical object indicates 100% resemblance (the value of 1.00 indicates 100% whereas the value of 0 indicates 0% similarity between objects in PSM; all similarity values are mapped between 0 and 1).

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1a	1.00	0.54	0.63	0.59	0.32	0.15	0.25	0.23	0.31	0.27	0.19	0.22	0.49	0.53	0.44	0.48	-0.01	0.09	0.06	0.01	0.11	0.07	0.07	0.08	-0.02	0.00	-0.01	0.00	-0.01	-0.01	0.02	0.02	0.00	0.02	-0.02	-0.01	0.05	0.02	0.08	0.03
1b	0.54	1.00	0.55	0.44	0.28	0.10	0.22	0.18	0.39	0.34	0.29	0.33	0.32	0.35	0.32	0.32	-0.02	0.09	0.06	0.00	0.12	0.12	0.10	0.16	-0.02	0.02	-0.01	0.06	0.05	0.05	0.05	0.06	0.03	0.02	-0.04	-0.08	0.02	0.01	0.06	0.01
1c	0.63	0.55	1.00	0.65	0.34	0.14	0.24	0.22	0.37	0.31	0.24	0.26	0.44	0.46	0.40	0.45	0.01	0.12	0.07	0.01	0.12	0.11	0.07	0.13	0.01	0.03	0.01	0.05	0.04	0.03	0.06	0.04	0.03	0.02	-0.04	-0.01	0.07	0.05	0.07	0.03
1d	0.59	0.44	0.65	1.00	0.24	0.09	0.17	0.16	0.34	0.29	0.20	0.19	0.48	0.46	0.34	0.49	0.01	0.08	0.06	0.01	0.12	0.09	0.04	0.10	0.03	0.04	0.02	0.05	0.04	0.03	0.05	0.05	0.04	0.02	-0.02	0.02	0.05	0.02	0.04	0.03
2a	0.32	0.28	0.34	0.24	1.00	0.43	0.64	0.57	0.13	0.09	0.07	0.12	0.24	0.28	0.37	0.26	0.11	0.14	0.09	0.12	0.01	0.01	0.01	-0.01	-0.04	0.00	-0.01	0.01	-0.03	-0.03	0.01	-0.01	0.06	0.06	0.03	0.09	0.19	0.20	0.18	0.16
2b	0.15	0.10	0.14	0.09	0.43	1.00	0.52	0.55	0.03	0.02	-0.01	0.07	0.15	0.18	0.26	0.14	0.12	0.13	0.09	0.13	-0.04	-0.02	-0.03	-0.03	0.00	0.02	0.01	0.01	-0.05	-0.06	-0.02	-0.04	0.07	0.06	0.05	0.15	0.20	0.22	0.19	0.17
2c	0.25	0.22	0.24	0.17	0.64	0.52	1.00	0.69	0.09	0.05	0.03	0.11	0.19	0.23	0.31	0.20	0.12	0.15	0.10	0.15	-0.01	0.00	-0.01	-0.03	-0.05	0.00	-0.01	0.00	-0.04	-0.05	-0.01	-0.01	0.07	0.07	0.05	0.12	0.22	0.23	0.19	0.19
2d	0.23	0.18	0.22	0.16	0.57	0.55	0.69	1.00	0.06	0.03	0.01	0.09	0.21	0.24	0.33	0.21	0.12	0.14	0.10	0.14	-0.01	-0.02	-0.02	-0.05	-0.05	-0.03	-0.02	-0.01	-0.04	-0.05	-0.02	-0.02	0.06	0.06	0.05	0.12	0.23	0.22	0.18	0.18
3a	0.31	0.39	0.37	0.34	0.13	0.03	0.09	0.06	1.00	0.68	0.45	0.53	0.26	0.25	0.19	0.25	0.07	0.14	0.20	0.08	0.26	0.31	0.20	0.33	0.10	0.16	0.12	0.20	0.18	0.20	0.19	0.24	0.13	0.09	0.05	0.03	0.10	0.10	0.07	0.09
3b	0.27	0.34	0.31	0.29	0.09	0.02	0.05	0.03	0.68	1.00	0.48	0.58	0.22	0.21	0.15	0.20	0.06	0.13	0.20	0.06	0.26	0.29	0.22	0.34	0.13	0.17	0.14	0.22	0.19	0.21	0.19	0.22	0.12	0.09	0.06	0.03	0.07	0.07	0.07	0.07
3c	0.19	0.29	0.24	0.20	0.07	-0.01	0.03	0.01	0.45	0.48	1.00	0.48	0.13	0.13	0.11	0.12	0.16	0.24	0.31	0.18	0.37	0.36	0.31	0.42	0.22	0.27	0.23	0.33	0.28	0.31	0.29	0.31	0.22	0.19	0.14	0.10	0.13	0.12	0.16	0.17
3d	0.22	0.33	0.26	0.19	0.12	0.07	0.11	0.09	0.53	0.58	0.48	1.00	0.16	0.18	0.17	0.14	0.09	0.16	0.23	0.11	0.25	0.28	0.22	0.33	0.09	0.15	0.13	0.23	0.22	0.23	0.21	0.26	0.12	0.11	0.07	0.03	0.12	0.12	0.11	0.12
4a	0.49	0.32	0.44	0.48	0.24	0.15	0.19	0.21	0.26	0.22	0.13	0.16	1.00	0.73	0.56	0.78	-0.01	0.05	0.03	0.00	0.09	0.05	0.01	0.05	-0.02	0.00	-0.01	0.00	-0.02	-0.02	0.00	0.02	0.00	-0.01	-0.04	0.02	0.04	0.00	0.03	0.02
4b	0.53	0.35	0.46	0.46	0.28	0.18	0.23	0.24	0.25	0.21	0.13	0.18	0.73	1.00	0.63	0.72	0.00	0.07	0.04	0.01	0.08	0.04	0.02	0.04	-0.04	-0.02	-0.03	-0.02	-0.04	-0.04	-0.01	0.00	-0.01	0.00	-0.03	0.02	0.05	0.01	0.05	0.03
4c	0.44	0.32	0.40	0.34	0.37	0.26	0.31	0.33	0.19	0.15	0.11	0.17	0.56	0.63	1.00	0.55	0.03	0.11	0.09	0.05	0.07	0.04	0.04	0.01	-0.06	-0.05	-0.04	-0.03	-0.05	-0.06	-0.03	-0.01	0.00	0.01	-0.02	0.02	0.09	0.06	0.10	0.06
4d	0.48	0.32	0.45	0.49	0.26	0.14	0.20	0.21	0.25	0.20	0.12	0.14	0.78	0.72	0.55	1.00	0.00	0.06	0.02	0.01	0.09	0.05	0.00	0.04	-0.02	-0.01	-0.01	-0.01	-0.03	-0.03	0.00	0.02	0.01	0.00	-0.03	0.03	0.06	0.02	0.04	0.03
5a	-0.01	-0.02	0.01	0.01	0.11	0.12	0.12	0.12	0.07	0.06	0.16	0.09	-0.01	0.00	0.03	0.00	1.00	0.53	0.47	0.74	0.26	0.27	0.21	0.20	0.26	0.36	0.34	0.34	0.25	0.31	0.33	0.33	0.45	0.35	0.38	0.46	0.46	0.43	0.45	0.54
5b	0.09	0.09	0.12	0.08	0.14	0.13	0.15	0.14	0.14	0.13	0.24	0.16	0.05	0.07	0.11	0.06	0.53	1.00	0.48	0.52	0.29	0.35	0.26	0.30	0.28	0.34	0.32	0.37	0.28	0.32	0.35	0.31	0.42	0.35	0.31	0.35	0.39	0.37	0.50	0.43
5c	0.06	0.06	0.07	0.06	0.09	0.09	0.10	0.10	0.20	0.20	0.31	0.23	0.03	0.04	0.09	0.02	0.47	0.48	1.00	0.47	0.42	0.43	0.39	0.35	0.22	0.30	0.31	0.36	0.31	0.34	0.32	0.38	0.44	0.34	0.32	0.34	0.37	0.37	0.38	0.43
5d	0.01	0.00	0.01	0.01	0.12	0.13	0.15	0.14	0.08	0.06	0.18	0.11	0.00	0.01	0.05	0.01	0.74	0.52	0.47	1.00	0.26	0.26	0.21	0.20	0.26	0.35	0.34	0.34	0.26	0.31	0.31	0.33	0.45	0.37	0.39	0.45	0.46	0.43	0.47	0.55
6a	0.11	0.12	0.12	0.12	0.01	-0.04	-0.01	-0.01	0.26	0.26	0.37	0.25	0.09	0.08	0.07	0.09	0.26	0.29	0.42	0.26	1.00	0.46	0.48	0.37	0.15	0.20	0.22	0.25	0.24	0.26	0.23	0.31	0.29	0.23	0.19	0.18	0.18	0.17	0.19	0.21
6b	0.07	0.12	0.11	0.09	0.01	-0.02	0.00	-0.02	0.31	0.29	0.36	0.28	0.05	0.04	0.04	0.05	0.27	0.35	0.43	0.26	0.46	1.00	0.39	0.60	0.20	0.29	0.24	0.36	0.33	0.35	0.30	0.38	0.34	0.25	0.19	0.20	0.22	0.24	0.22	0.25
6c	0.07	0.10	0.07	0.04	0.01	-0.03	-0.01	-0.02	0.20	0.22	0.31	0.22	0.01	0.02	0.04	0.00	0.21	0.26	0.39	0.21	0.48	0.39	1.00	0.33	0.10	0.14	0.16	0.19	0.18	0.19	0.18	0.21	0.22	0.20	0.17	0.12	0.13	0.13	0.15	0.16
6d	0.08	0.16	0.13	0.10	-0.01	-0.03	-0.03	-0.05	0.33	0.34	0.42	0.33	0.05	0.04	0.01	0.04	0.20	0.30	0.35	0.20	0.37	0.60	0.33	1.00	0.24	0.31	0.25	0.39	0.35	0.36	0.30	0.34	0.28	0.20	0.15	0.15	0.15	0.16	0.19	0.18
7a	-0.02	-0.02	0.01	0.03	-0.04	0.00	-0.05	-0.05	0.10	0.13	0.22	0.09	-0.02	-0.04	-0.06	-0.02	0.26	0.28	0.22	0.26	0.15	0.20	0.10	0.24	1.00	0.64	0.49	0.49	0.35	0.39	0.42	0.32	0.32	0.32	0.32	0.33	0.17	0.15	0.28	0.23
7b	0.00	0.02	0.03	0.04	0.00	0.02	0.00	-0.03	0.16	0.17	0.27	0.15	0.00	-0.02	-0.05	-0.01	0.36	0.34	0.30	0.35	0.20	0.29	0.14	0.31	0.64	1.00	0.58	0.62	0.47	0.53	0.53	0.45	0.43	0.40	0.39	0.39	0.25	0.24	0.34	0.32
7c	-0.01	-0.01	0.01	0.02	-0.01	0.01	-0.01	-0.02	0.12	0.14	0.23	0.13	-0.01	-0.03	-0.04	-0.01	0.34	0.32	0.31	0.34	0.22	0.24	0.16	0.25	0.49	0.58	1.00	0.55	0.40	0.47	0.51	0.44	0.39	0.47	0.49	0.40	0.25	0.23	0.35	0.32
7d	0.00	0.06	0.05	0.05	0.01	0.01	0.00	-0.01	0.20	0.22	0.33	0.23	0.00	-0.02	-0.03	-0.01	0.34	0.37	0.36	0.34	0.25	0.36	0.19	0.39	0.49	0.62	0.55	1.00	0.55	0.60	0.52	0.53	0.46	0.40	0.36	0.32	0.28	0.28	0.34	0.35
8a	-0.01	0.05	0.04	0.04	-0.03	-0.05	-0.04	-0.04	0.18	0.19	0.28	0.22	-0.02	-0.04	-0.05	-0.03	0.25	0.28	0.31	0.26	0.24	0.33	0.18	0.35	0.35	0.47	0.40	0.55	1.00	0.68	0.54	0.59	0.42	0.40	0.34	0.25	0.21	0.23	0.26	0.25
8b	-0.01	0.05	0.03	0.03	-0.03	-0.06	-0.05	-0.05	0.20	0.21	0.31	0.23	-0.02	-0.04	-0.06	-0.03	0.31	0.32	0.34	0.31	0.26	0.35	0.19	0.36	0.39	0.53	0.47	0.60	0.68	1.00	0.63	0.66	0.43	0.42	0.36	0.29	0.24	0.24	0.28	0.29
8c	0.02	0.05	0.06	0.05	0.01	-0.02	-0.01	-0.02	0.19	0.19	0.29	0.21	0.00	-0.01	-0.03	0.00	0.33	0.35	0.32	0.31	0.23	0.30	0.18	0.30	0.42	0.53	0.51	0.52	0.54	0.63	1.00	0.56	0.40	0.46	0.41	0.33	0.26	0.24	0.32	0.32
8d	0.02	0.06	0.04	0.05	-0.01	-0.04	-0.01	-0.02	0.24	0.22	0.31	0.26	0.02	0.00	-0.01	0.02	0.33	0.31	0.38	0.33	0.31	0.38	0.21	0.34	0.32	0.45	0.44	0.53	0.59	0.66	0.56	1.00	0.41	0.39	0.35	0.30	0.28	0.27	0.28	0.32
9a	0.00	0.03	0.03	0.04	0.06	0.07	0.07	0.06	0.13	0.12	0.22	0.12	0.00	-0.01	0.00	0.01	0.45	0.42	0.44	0.45	0.29	0.34	0.22	0.28	0.32	0.43	0.39	0.46	0.42	0.43	0.40	0.41	1.00	0.50	0.43	0.44	0.41	0.42	0.41	0.46
9b	0.02	0.02	0.02	0.02	0.06	0.06	0.07	0.06	0.09	0.09	0.19	0.11	-0.01	0.00	0.01	0.00	0.35	0.35	0.34	0.37	0.23	0.25	0.20	0.20	0.32	0.40	0.47	0.40	0.40	0.42	0.46	0.39	0.50	1.00	0.63	0.42	0.34	0.32	0.43	0.41
9c	-0.02	-0.04	-0.04	-0.02	0.03	0.05	0.05	0.05	0.05	0.06	0.14	0.07	-0.04	-0.03	-0.02	-0.03	0.38	0.31	0.32	0.39	0.19	0.19	0.17	0.15	0.32	0.39	0.49	0.36	0.34	0.36	0.41	0.35	0.43	0.63	1.00	0.47	0.32	0.29	0.42	0.40
9d	-0.01	-0.08	-0.01	0.02	0.09	0.15	0.12	0.12	0.03	0.03	0.10	0.03	0.02	0.02	0.02	0.03	0.46	0.35	0.34	0.45	0.18	0.20	0.12	0.15	0.33	0.39	0.40	0.32	0.25	0.29	0.33	0.30	0.44	0.42	0.47	1.00	0.40	0.37	0.42	0.44
10a	0.05	0.02	0.07	0.05	0.19	0.20	0.22	0.23	0.10	0.07	0.13	0.12	0.04	0.05	0.09	0.06	0.46	0.39	0.37	0.46	0.18	0.22	0.13	0.15	0.17	0.25	0.25	0.28	0.21	0.24	0.26	0.28	0.41	0.34	0.32	0.40	1.00	0.68	0.51	0.63
10b	0.02	0.01	0.05	0.02	0.20	0.22	0.23	0.22	0.10	0.07	0.12	0.12	0.00	0.01	0.06	0.02	0.43	0.37	0.37	0.43	0.17	0.24	0.13	0.16	0.15	0.24	0.23	0.28	0.23	0.24	0.24	0.27	0.42	0.32	0.29	0.37	0.68	1.00	0.45	0.55
10c	0.08	0.06	0.07	0.04	0.18	0.19	0.19	0.18	0.07	0.07	0.16	0.11	0.03	0.05	0.10	0.04	0.45	0.50	0.38	0.47	0.19	0.22	0.15	0.19	0.28	0.34	0.35	0.34	0.26	0.28	0.32	0.28	0.41	0.43	0.42	0.42	0.51	0.45	1.00	0.58
10d	0.03	0.01	0.03	0.03	0.16	0.17	0.19	0.18	0.09	0.07	0.17	0.12	0.02	0.03	0.06	0.03	0.54	0.43	0.43	0.55	0.21	0.25	0.16	0.18	0.23	0.32	0.32	0.35	0.25	0.29	0.32	0.32	0.46	0.41	0.40	0.44	0.63	0.55	0.58	1.00

Table A.6: Full symmetric table of Tarrlab female images in Fig. 8: the cells in this table should be compared to the white cells in Table 1 to see how the symmetry is constructed: for instance, the values of 0.49, 0.53, 0.44 and 0.48 in the column, 1a, and in the intersection rows of 4a, 4b, 4c and 4d respectively are placed in the row 1a and in the intersection columns of 4a, 4b, 4c and 4d respectively as shown in grey cells where the cells denoted as (O_{I_i}, O_{I_j}) and their symmetric cells denoted as (O_{I_j}, O_{I_i}), in PSM indicate the similarity between the objects of I_i and I_j that have the same similarity value for the sake of symmetrical positioning of cells.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1a	1.00	0.54	0.63	0.59	0	0	0	0	0	0	0	0	0.49	0.53	0.44	0.48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1b	0.54	1.00	0.55	0.44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1c	0.63	0.55	1.00	0.65	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1d	0.59	0.44	0.65	1.00	0	0	0	0	0	0	0	0	0.48	0.46	0	0.49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2a	0	0	0	0	1.00	0.43	0.64	0.57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2b	0	0	0	0	0.43	1.00	0.52	0.55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2c	0	0	0	0	0.64	0.52	1.00	0.69	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2d	0	0	0	0	0.57	0.55	0.69	1.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3a	0	0	0	0	0	0	0	0	1.00	0.68	0.45	0.53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3b	0	0	0	0	0	0	0	0	0.68	1.00	0.48	0.58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3c	0	0	0	0	0	0	0	0	0.45	0.48	1.00	0.48	0	0	0	0	0	0	0	0	0.37	0.36	0.31	0.42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3d	0	0	0	0	0	0	0	0	0.53	0.58	0.48	1.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4a	0.49	0	0	0.48	0	0	0	0	0	0	0	0	1.00	0.73	0.56	0.78	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4b	0.53	0	0	0.46	0	0	0	0	0	0	0	0	0.73	1.00	0.63	0.72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4c	0.44	0	0	0	0	0	0	0	0	0	0	0	0.56	0.63	1.00	0.55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4d	0.48	0	0	0.49	0	0	0	0	0	0	0	0	0.78	0.72	0.55	1.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.00	0.53	0.47	0.74	0	0	0	0	0	0	0	0	0	0	0	0	0.45	0	0	0.46	0.46	0.43	0.45	0.54
5b	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.53	1.00	0.48	0.52	0	0	0	0	0	0	0	0	0	0	0	0	0.42	0	0	0	0	0	0.50	0.43
5c	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.47	0.48	1.00	0.47	0.42	0.43	0.39	0	0	0	0	0	0	0	0	0	0.44	0	0	0	0	0	0	0.43
5d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.74	0.52	0.47	1.00	0	0	0	0	0	0	0	0	0	0	0	0	0.45	0	0	0.45	0	0	0.47	0.55
6a	0	0	0	0	0	0	0	0	0	0	0.37	0	0	0	0	0	0	0	0.42	0	1.00	0.46	0.48	0.37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6b	0	0	0	0	0	0	0	0	0	0	0.36	0	0	0	0	0	0	0	0.43	0	0.46	1.00	0.39	0.60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6c	0	0	0	0	0	0	0	0	0	0	0.31	0	0	0	0	0	0	0	0.39	0	0.48	0.39	1.00	0.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6d	0	0	0	0	0	0	0	0	0	0	0.42	0	0	0	0	0	0	0	0	0	0.37	0.60	0.33	1.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.00	0.64	0.49	0.49	0	0	0.42	0	0	0	0	0	0	0	0	0
7b	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.64	1.00	0.58	0.62	0	0.53	0.53	0	0.43	0	0	0.39	0	0	0	0
7c	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.49	0.58	1.00	0.55	0	0.47	0.51	0.44	0.39	0.47	0.49	0.40	0	0	0	0
7d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.49	0.62	0.55	1.00	0.55	0.60	0.52	0.53	0.46	0	0	0	0	0	0	0
8a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.55	1.00	0.68	0.54	0.59	0.42	0	0	0	0	0	0	0
8b	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.53	0.47	0.60	0.68	1.00	0.63	0.66	0.43	0	0	0	0	0	0	0
8c	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.42	0.53	0.51	0.52	0.54	0.63	1.00	0.56	0.40	0.46	0	0	0	0	0	0
8d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.44	0.53	0.59	0.66	0.56	1.00	0.41	0	0	0	0	0	0	0
9a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.45	0.42	0.44	0.45	0	0	0	0	0	0.43	0.39	0.46	0.42	0.43	0.40	0.41	1.00	0.50	0.43	0.44	0.41	0.42	0.41	0.46
9b	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.47	0	0	0	0.46	0	0.50	1.00	0.63	0.42	0	0	0.43	0
9c	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.49	0	0	0	0	0	0.43	0.63	1.00	0.47	0	0	0.42	0
9d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.46	0	0	0.45	0	0	0	0	0	0.39	0.40	0	0	0	0	0	0.44	0.42	0.47	1.00	0.40	0.37	0.42	0.44
10a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.41	0	0	0.40	1.00	0.68	0.51	0.63
10b	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.42	0	0	0.37	0.68	1.00	0.45	0.55
10c	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.45	0.50	0	0.47	0	0	0	0	0	0	0	0	0	0	0	0	0.41	0.43	0.42	0.42	0.51	0.45	1.00	0.58
10d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.54	0.43	0.43	0.55	0	0	0	0	0	0	0	0	0	0	0	0	0.46	0	0	0.44	0.63	0.55	0.58	1.00

Table A.7: Clusters obtained from the left of PSM for Tarrlab female images in Fig. 8 before merging, after the agglomerative attribute of ACE method is performed: the number of clusters is 11 in the rows of 2, 6, 11, 15, 23, 24, 31, 32, 33, 36 and 38 where the cells in the columns are equal to the value of 1; for instance, the first cluster in row 2 consists of 1a, 1b, 1c and 1d objects.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A.8: Clusters obtained from the right of PSM for Tarrlab female images in Fig. 8 before merging, after the agglomerative attribute of ACE method is performed: the number of clusters is 10 in the rows of 1, 2, 6, 9, 11, 19, 25, 27, 28, and 36 where the cells in the columns are equal to the value of 1; for instance, the first cluster in row 1 consists of 1a, 4a, 4b, 4c and 4d objects.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A.9: Clusters obtained from the left of PSM for Tarrlab female images in Fig. 8 after merging, after the agglomerative and contractible attributes of ACE method are performed: the number of clusters is 9 in the rows of 2, 6, 11, 15, 23, 31, 32, 33 and 36 where the cells in the columns are equal to the value of 1; for instance, the first cluster in row 2 consists of 1a, 1b, 1c and 1d objects.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A.10: Clusters obtained from the right of PSM for Tarrlab female images in Fig. 8 after merging, after the agglomerative and contractible attributes of ACE method are performed: the number of clusters is 10 in the rows of 1, 2, 6, 9, 11, 19, 25, 27, 28 and 36 where the cells in the columns are equal to the value of 1; for instance, the first cluster in row 1 consists 1a, 4a, 4b, 4c and 4d objects.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A.11: Customized clusters obtained from the left and right of PSM for Tarrlab female images in Fig. 8 after the agglomerative and contractible attributes of ACE method are performed: these two sets are customized by removing the empty rows in Tables A.9 and A.10. The clusters in different rows and the objects in these clusters obtained from the left (first set below) and right (second set below) of PSM are not same in terms of the columns and grouping together nearby. The differences are disclosed in these two sets referring to the counterpart clusters in one another. A counterpart cluster harbours more same objects than the other clusters. For instance, the counterpart cluster of 1 in the first set is the cluster 2 in second set. The cluster 2 in second set has the three same objects of the cluster 1 in the first set. That's why, first object, 1a, in the cluster 1 in the first set is signed as a different object. Likewise, for the second set, the cluster 1 points to the cluster 4 in the first set as a counterpart cluster and 1a in the cluster 1 in the second set is signed as a different object. All the differences are mapped in dark grey cells. objects are assigned to their nearest clusters. The nearest cluster for an object is determined using the cell that has the biggest value in the same row/column where this object is positioned in the original TableA.5. For instance, the object, 1a, points to the object, 1c, with the biggest value of 0,63 and 1a is already in the cluster 1 where the 1c is located; so, this object retains in the same cluster. Otherwise, it would be transported to the cluster where 1c is located. Likewise, the other dark grey objects are searched in TableA.5. As a result, 3c points to the object 3b with a value of 0.48 and 3c retains in its cluster; 5b points to 5a with a value of 0.53 and it retains in its current cluster; 5c points to 5b with a value of 0.48 and it is transported to the cluster 8, where 5b is located. Similarly, 5d, 7c and 7d retain, 8c is transported into the cluster 7 and 9d retains for the first set. The transported object to their new clusters are marked with check sign for both sets below. The expansive attribute of ACE method is realized after there is no transportation, but still there are differences between two sets which means that the number of clusters are different in two sets as it can be seen in this example (i.e, the number of clusters is 9 for the first set and 10 for the other one): the differences are all transported to some other empty row as seen at the row 40 in Table A.12 to form a new cluster, and consequentially the two sets turn into exact same sets as shown in Table A.14. To equalize the number of clusters for these two sets after previous transportations, the objects signed as different would be 5b, 5c and 5d in the cluster 8 in the first set and these objects are transported to any other empty row to form a new cluster by which these two sets turn out to be same in terms of the number of clusters and objects in these clusters.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	1	1	1	1																																				
2					1	1	1	1																																
3									1	1	1	1																												
4													1	1	1	1																								
5																			1		1	1	1	1																
6																									1	1	1	1			1									
7																													1	1	✓	1								
8																	1	1	✓	1													1				1	1	1	1
9																																		1	1	1				

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	1												1	1	1	1																								
2	✓	1	1	1																																				
3					1	1	1	1																																
4									1	1	✓	1																												
5											1										1	1	1	1																
6																		1	1	1																				
7																									1	1	✓	✓												
8																											1							1	1	✓				
9																												1	1	1	1	1								
10																	1																1			1	1	1	1	1

Table A.12: Final clusters obtained from the left of PSM for Tarrlab female images in Fig. 8, after the agglomerative, contractible and expandable attributes of ACE method are performed: established clusters are assigned into the randomly selected empty rows of a table that is as big as PSM since the count of the clusters can be as many as the count of objects to be cluster at most. The number of clusters is 10 in the rows of 2, 6, 11, 15, 23, 31, 32, 33, 36 and 40 where the cells in the columns are equal to the value of 1; for instance, the first cluster in row 2 consists of 1a, 1b, 1c and 1d objects.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A.13: Final clusters obtained from the right of PSM images for Tarrlab female images in Fig. 8, after the agglomerative, contractible, expandable attributes of ACE method are performed: established clusters are assigned into the randomly selected empty rows of a table that is as big as PSM since the count of the clusters can be as many as the count of objects to be cluster at most. The number of clusters is 10 in the rows of 1, 2, 6, 9, 11, 19, 25, 27, 28 and 36 where the cells in the columns are equal to the value of 1; for instance, the first cluster in row 1 consists of 4a, 4b, 4c and 4d objects.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A.14: Customized final clusters obtained from the left and right of PSM for Tarrlab female images in Fig. 8, after the agglomerative, contractible and expandable attributes of ACE method are performed: these two tables are customized by removing the empty rows in Tables A.12 and A.13. The clusters in different rows and the objects in these clusters obtained from the left and right of PSM are same in terms of the columns and grouping together nearby.

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1	1	1	1	1																																				
2					1	1	1	1																																
3									1	1	1	1																												
4													1	1	1	1																								
5																					1	1	1	1																
6																									1	1	1	1												
7																													1	1	1	1								
8																	1																1				1	1	1	1
9																																		1	1	1				
10																		1	1	1																				

	1a	1b	1c	1d	2a	2b	2c	2d	3a	3b	3c	3d	4a	4b	4c	4d	5a	5b	5c	5d	6a	6b	6c	6d	7a	7b	7c	7d	8a	8b	8c	8d	9a	9b	9c	9d	10a	10b	10c	10d
1													1	1	1	1																								
2	1	1	1	1																																				
3					1	1	1	1																																
4									1	1	1	1																												
5																					1	1	1	1																
6																		1	1	1																				
7																									1	1	1	1												
8																																		1	1	1				
9																													1	1	1	1								
10																	1																1				1	1	1	1

Appendix B. CIPSM Pseudo codes

```
Data: Pairwise similarity matrix (PSM) acquired following PCA and Eucledean techniques are employed. PSM in csv format
Result: PSM, $S_{ij}$, is simplified after Rk-means method is run; many cells, about 80%, are set to null
>> Rk-means.exe -df Similarity_Matrix.csv -k 4 -r 40 -c 40
− >where -df is the PSM in csv format on which Rk-means is executed, -k is desired number of clusters, -r is the number of rows in PSM and -c is the number of columns in PSM;
Data: Reduces pairwise similarity matrix (rPSM) acquired following Rk-means algorithm. PSM processed by Rk_means in csv format (main_matrix)
Result: An agglomerative, contractible, expandable (ACE) algorithm. Clusters are constructed while the cells of the symmetric rPSM (srPSM), $S_{ij}$, are being
        set to null regarding the objects in emerging clusters
− >Variables;
int *embodyingRows= new int[rows]; int * r_one =new int[rows]; int *emptyRow=new int[rows]; int * r_two =new int[rows]; double val=0; int sR=0; int sC=0;double biggestVal=0;int colNumber=0; int totalCount=0; double *cmax=new double[cols];int *sequence=new int[cols]; flag isThereEmbodying=false; int *numberofclusters=new int[rows];int *cindex=new int[cols];int reference_row=0; int** cluster_matrix = new int*[rows];
− >Create an empty PSM as big as PSM to sign clusters;
foreach $I_j$ do
    cluster_matrix[$I_j$] = new int[cols];
end
− >initialize cluster_matrix;
foreach $I_i$ do
    foreach $I_j$ do
        cluster_matrix[$I_i$][$I_j$]=0;
    end
end
− >Establish a full symmetric rPSM (srPSM) with respect to diagonal cells;
foreach $I_i$ do
    foreach $I_j$ do
        if main_matrix[$I_i$][$I_j$]is null then
            main_matrix[$I_i$][$I_j$] = main_matrix[$I_j$][$I_i$];
        end
        if main_matrix[$I_j$][$I_i$] is null then
            main_matrix[$I_j$][$I_i$] = main_matrix[$I_i$][$I_j$];
        end
    end
end
− >Set diagonal entries to 1, $S_{ij}$ = 1, where i = j which means similarity between two identical objects is 100%;
foreach $I_i$ do
    j=i; main_matrix[$I_i$][$I_j$] =1;
end
− >Run the procedures;
− >Clusters are generated starting from the leftmost column of srPSM;
call find_clusters_in_matrix(ascending);
− >Clusters are generated starting from the rightmost column of srPSM;
call find_clusters_in_matrix(descending);
− >Generation of left clusters is completed;
call merge_clusters (cluster_matrix_left);
− >Generation of right clusters is completed;
call merge_clusters (cluster_matrix_right);
− >Final clusters are obtained;
call forge_two_sets_clusters_into_one_set (merged_cluster_matrix_left, merged_cluster_matrix_right);
```

Algorithm 1: Pseudo code of the CIPSM (clustering in pairwise similarity matrix) method for final clustering including Rk-means and ACE methods: *i* corresponds to the columns and j corresponds to the rows of srPSM.

```
PROCEDURE find_clusters_in_matrix(start_position):
while not similarity matrix, $S_{ij}$, empty do
    − >Sequences starting from the leftmost/rightmost column are found. The leftmost/rightmost column where one of whose cells is not null is picked up.
    ($I_n$) in $S_{ij}$;
    foreach $I_i$ start_position do
        foreach $I_j$ do
            if main_matrix[$I_i$][$I_j$]is null then
                − >Clustering starts from the leftmost of srPSM; sC: the object from which first and following clusters are formed;
                sC = $I_i$; break;
            end
        end
    end
    − >The object that has the biggest value in sC column is found to specify the row for the next step;
    foreach $I_j$ do
        if main_matrix[sC][$I_i$] not null and main_matrix[sC][$I_i$] >val then
            sR = $I_i$; val = main_matrix[sC][$I_i$];
        end
    end
    − >The maximum values are found for each column for further steps;
    foreach $I_i$ do
        cmax[$I_i$]=-1000; /*initilize the maximum value*/;
        foreach $I_j$ do
            if main_matrix[$I_i$][$I_j$] not null and cmax[$I_i$] < main_matrix[$I_i$][$I_j$] then
                cmax[$I_i$]= main_matrix[$I_i$][$I_j$];
            end
        end
    end
    − > The not-null values in the cells of the row are compared to the cells in their columns to be singled out if they are the biggest; sign the cell as a
    member of the current cluster, otherwise set the cell to null value;
    foreach $I_i$ do
        if main_matrix[$I_i$][sR] not null and main_matrix[$I_i$][sR] < cmax[$I_i$] then
            main_matrix[$I_i$][sR]=null; sequence[$I_i$]= 0;
        end
        else
            sequence[$I_i$]= 1; /* The members of the sequence are signed with a value of 1 in terms of columns as a candidate cluster*/;
        end
    end
    if func_rows_embodying_sequence is true then
        call find_biggest_in_total_in_embodying_rows; double biggestVal = -1000; /* Initilializa tion to single out the biggest value in the selected row, bR,
        different from the values of the cells in current sequence (i.e.,sequence[Ii])*/;
        foreach $I_i$ do
            if sequence[$I_i$]<>1and biggestVal<main_matrix[$I_i$][bR] then
                biggestVal= main_matrix[$I_i$][bR]; bC = $I_i$; /*column name is signed to be a member of the current sequence*/;
            end
        end
        foreach $I_i$ and $I_i$= bC do
            sequence[$I_i$]= 1; /*A new sequence expanding with nem members*/;
        end
        foreach $I_j$ do
            embodyingRows[$I_j$]=0; /* reset the array of embodying rows to establish a another sequence*/;
        end
        isThereEmbodying = false; /* reset the control parameter*/;
    end
    else
        call clear_cluster_members_in_sm;
        call establish_cluster(start_position);
    end
end
END
```

Algorithm 2: Pseudo code of the procedure in CIPSM method for establishing initial clusters starting from either the leftmost or rightmost columns of srPSM.

```
PROCEDURE merge_clusters(cluster_matrix):
− >Firstly, members in one_membered clusters are assigned to their nearest clusters;
foreach $I_j$ do
    numberofclusters[$I_j$]=0; /*Initialize the values of the number of members in clusters*/;
end
foreach $I_j$ do
    foreach $I_i$ do
        if cluster_matrix[$I_i$][ $I_j$]=1 then
            numberofclusters[$I_j$]++; /*The count of the members in clusters is specified*/;
        end
    end
end
foreach $I_j$ do
    cmax[$I_i$]=-1000;/*Initialize the cells values*/;
end
foreach $I_j$ do
    if numberofclusters[$I_j$]=1 then
        biggestVal = -1000; colNumber=0; /*Initialize the other parameters*/;
        foreach $I_i$ do
            if main_matrix[$I_i$][$I_j$]> biggestVal then
                − >the biggest value from one-membered clusters is found and column number is specified for this value
                if cluster_matrix[$I_i$][$I_j$]<>1 then
                    biggestVal= main_matrix[$I_i$][$I_j$]; colNumber=$I_i$;
                end
            end
        end
    end
    cmax[$I_j$]= biggestVal; cindex[$I_i$]= colNumber;
end
foreach $I_j$ do
    rowNumber = 0; if numberofclusters[$I_j$]=1 then
        foreach $I_j$ do
            if cluster_matrix[$I_j$][cindex[$I_j$]]=1 then
                rowNumber = $I_j$;
            end
        end
        foreach $I_i$ do
            if cluster_matrix[$I_i$][ $I_j$]=1 then
                cluster_matrix[$I_i$][rowNumber]= cluster_matrix[$I_i$][ $I_j$]; cluster_matrix[$I_i$][ $I_j$]=0;
            end
        end
    end
end
− >Secondly, members starting from two_membered clusters to other more-membered clusters are assigned to their nearest clusters if and only if each member
in these cluster points to the same cluster similar to the assignment of one_membered clusters mentioned above
END
```

Algorithm 3: Pseudo code employed in CIPSM method and in forge_two_sets_clusters_into_one_set (merged_cluster_matrix_left,merged_cluster_matrix_right) for merging pre-established clusters that point to each other as similar starting from one-membered to more-membered clusters.

```
PROCEDURE forge_two_sets_clusters_into_one_set(merged_cluster_matrix_left,merged_cluster_matrix_right):
− >The two sets of final clusters acquired from the leftmost and rightmost of the srPSM are forged together until the two sets are equal to each other. Firstly, left matrix is examined referring to the right matrix;
while merged_cluster_matrix_left = merged_cluster_matrix_right do
    foreach $I_x$ do
        /*# of embodying rows are disclosed for each row where x = j*/;
        foreach $I_j$ do
            foreach $I_i$ do
                if merged_cluster_matrix_right [$I_j$][$I_x$]=1 then
                    if merged_cluster_matrix_right[$I_j$][ $I_x$]= merged_cluster_matrix_right [$I_j$][$I_j$] then
                        embodyingRows[$I_j$]=++; /*The cells matching in the same rows are counted with regard to the columns*/;
                    end
                end
            end
        end
        foreach $I_j$ do
            /*The biggest values of embodyingRows[$I_j$] cells in the rows are revealed*/;
            if embodyingRows[Ij] > reference_row then
                reference_row = $I_j$;
            end
        end
        − >The cells matched with a value of 20 along with not matching cells with a value of 1 are singled out and counted;
        foreach $I_j$ do
            r_one[$I_j$] = 0; r_two[$I_j$] = 0; /*Former counted values are set to 0 */;
        end
        foreach $I_j$ do
            foreach $I_i$ do
                if merged_cluster_matrix_left [$I_i$][$I_j$]=1 then
                    if merged_cluster_matrix_right [$I_i$][$I_x$]=1 then
                        r_one[$I_i$]=++;
                    end
                end
                if merged_cluster_matrix_left [$I_i$][$I_j$]=20 then
                    r_two[$I_i$]=++;
                end
            end
        end
        − >The values marked with a value of 20 are turned into the value of 30 If a reference row is picked out more than once for not losing the former marked values (a row in left cluster matrix might hold the biggest value of matching, embodyingRows[$I_j$], with regard to more than one row in right cluster matrix);
        if r_two [reference_row] > 0 then
            if r_one [reference_row]>= r_two [reference_row] then
                foreach $I_i$ do
                    if merged_cluster_matrix_left [$I_i$][ reference_row] =20] then
                        merged_cluster_matrix_left [$I_i$][ reference_row] =30;
                    end
                end
            end
        end
        − >The cells matched in the reference row are picked out and the value of 1s in the cells is turned into another value such as 20 to mark them;
        foreach $I_i$ do
            if merged_cluster_matrix_right [$I_i$][ $I_x$] =1 then
                if merged_cluster_matrix_left [$I_i$][ reference_row] =1 then
                    merged_cluster_matrix_left [$I_i$][ reference_row] =20;
                end
            end
        end
        − >The remaining cells with a value of 1 and not in the embodying cells are transported to a new row to mark a new cluster if their number is >= to the number of embodying cells having a value of 20;
        if r_two [reference_row] >0 then
            if r_one [reference_row]>= r_two [reference_row] then
                foreach $I_j$ do
                    /*Find first row to carry out them */;
                    totalCount = 0;
                    foreach $I_i$ do
                        if merged_cluster_matrix_left [$I_i$][$I_j$]! =0 then
                            totalCount ++;
                        end
                    end
                    if totalCount>0 then
                        emptyRow = $I_j$;
                    end
                end
            end
        end
        − >Transport the cells with a value of 1 to the specified reference row;
        foreach $I_j$ do
            if merged_cluster_matrix_left [$I_i$][reference_row] =1 then
                merged_cluster_matrix_left [$I_i$][emptyRow] =1; merged_cluster_matrix_left [$I_i$][reference_row] =0;
            end
            if merged_cluster_matrix_left [$I_i$][reference_row] =30 then
                merged_cluster_matrix_left [$I_i$][emptyRow] =30; merged_cluster_matrix_left [$I_i$][reference_row] =0;
            end
        end
        − >The cells in one-membered clusters are transported to their nearest clusters;
        call merge_clusters (cluster_matrix);
        − >All the values in the cells different from 0 are replaced with a value of 1 to process the left matrix;
        foreach $I_j$ do
            foreach $I_i$ do
                if merged_cluster_matrix_left [$I_i$][$I_j$]! =0 then
                    merged_cluster_matrix_left [$I_i$][$I_j$]=1;
                end
            end
        end
        − >All the steps above are executed for the merged_cluster_matrix_right using the modified merged_cluster_matrix_left and these steps are executed until left and right matrices equal to each other. Finally, the rows that are not set to null indicate the clusters regarding the columns.
    end
end
END
```

Algorithm 4: Pseudo code in CIPSM method for forging two sets of clusters (one acquired from the leftmost and the other one acquired from the rightmost of srPSM) into one.

```
FUNCTION func_rows_embodying_sequence:
− >Find other rows that embody the current sequence;
/*different from the row having the current sequence*/;
foreach $I_i$ and $I_j$ <> sR do
    embodyingRows[$I_j$]=1; /* first accept the row as it embodies beforehand */;
    foreach $I_i$ do
        if sequence[$I_i$]= 1 and $O(I_j)$ is null then
            embodyingRows[$I_i$]=0; /* it doesn't embody; so, remove the row */;
            break;
        end
    end
    if embodyingRows[$I_j$]=1 then
        isThereEmbodying = true;
    end
end
if isThereEmbodying is true then
    return 1;
end
else
    return 0
end
END
```

Algorithm 5: Pseudo-code of the function in the procedure, *find_clusters_in_matrix(start_position)*, for deciding to go further to either assign new members to the current sequence or construct a new cluster with the objects in the current sequence.

```
PROCEDURE find_biggest_in_total_in_embodying_rows:
totalVal = 0; preVal = 0; bR=0;
foreach $I_j$ do
    foreach $I_i$ do
        if embodyingRows[$I_j$]=1 and sequence[$I_i$]= 1 then
            totalVal = + main_matrix[$I_i$][$I_j$];
        end
    end
end
if totalValue > preVal then
    bR = $I_j$; /* The row having the biggest value in total is picked out*/;
end
preVal = totalVal;
END
```

Algorithm 6: Pseudo-code of the procedure in the procedure, *find_clusters_in_matrix(start_position)*, for finding the most valuable embodying sequence.

```
PROCEDURE clear_cluster_members_in_sm:
− >clear the columns and rows of srPSM as final clusters are formed;
foreach $I_i$ do
    if sequence[$I_i$]= 1 then
        foreach $I_j$ do
            main_matrix[$I_i$][$I_j$] = null;
        end
    end
end
foreach $I_j$ do
    /*set the cells to null value regarding the rows*/;
    if sequence[$I_j$]= 1 then
        foreach $I_j$ do
            main_matrix[$I_i$][$I_j$] = null;
        end
    end
end
END
```

Algorithm 7: Pseudo-code of the procedure in the procedure, *find_clusters_in_matrix(start_position)*, for deleting the members of the current established cluster from srPSM.

```
PROCEDURE establish_cluster(start_position):
if start_position = ascending then
    start_position=left;
end
else
    start_position=right;
end
foreach $I_j$ do
    foreach $I_j$ do
        if cluster_matrix[$I_i$][$I_j$]=0 then
            cmR = $I_j$;
        end
        else
            cmR= null; break;
        end
    end
    if cmR <> null then
        break;
    end
end
foreach $I_i$ do
    /*establish a new cluster with the members in the current final sequence*/;
    if sequence[$I_i$]= 1 then
        cluster_matrix[$I_i$][cmR]=1 /*the members of the established cluster are put in the first emty row*/;
        cluster_matrix+"_"+(start_position)[$I_i$][cmR]=1; /*either cluster_matrix_right or cluster_matrix_left is created*/;
        sequence[$I_i$] = 0; /*The cells equal to 1 are assigned to 0 to form new cluster members*/;
    end
end
END
```

Algorithm 8: Pseudo-code of the procedure in the procedure, *find_clusters_in_matrix(start_position)*, for establishing the current cluster: *i* corresponds to the columns and j corresponds to the rows of srPSM.