Journal of Computers Vol. 31 No. 1, 2020, pp. 148-165
doi:10.3966/199115992020023101012

Multi-object Cosegmentation Using Density-based Clustering

Tzu-Chiang Wang, I-Cheng Chang*, Chun-Man Lin

Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien 974, Taiwan
icchang@mail.ndhu.edu.tw

Received 29 September 2019; Revised 19 October 2019; Accepted 29 October 2019

Abstract. Cosegmentation is one of the interesting and popular topics in computer vision. The goal of cosegmentation is to extract the common foreground objects from an image set with minimum additional information. The existing cosegmentation algorithms could be classified into two categories. One is to extract one kind of foreground objects in the image set under unsupervised approaches; the other one is to find different kinds of common foreground objects in the image set under supervised approaches of which the number of kinds should be predefined. In this paper, we propose an unsupervised cosegmentation method for multiple foreground objects, which need not preset the number of object kinds. Moreover, most of the existing cosegmentation algorithms assume that the common foreground objects should appear in all images of the image set. However, if the foreground object only appears in a few images, the object is often misclassified. Our proposed algorithm can segment different kinds of common objects and have a higher segmentation rate for some foreground objects not appearing in all images. In the proposed work, an image is considered as the combination of several objects, and each object is composed of object elements. The image set could be decomposed into lots of object elements, and then object elements with similar features could be clustered into one sub-object class representing one part of an object. According to the class distribution of elements, common objects are extracted by the selection criteria. The concept of independent object elements is also proposed to increase the segmentation rate. In the experimental results, we demonstrate that the proposed approach could get better segmentation results compared with other methods.

Keywords: cosegmentation algorithm, density-based clustering algorithm, multi-class cosegmentation algorithm, multiple object segmentation, unsupervised clustering algorithm

1 Introduction

Cosegmentation becomes a popular topic in recent years. The objective of cosegmentation is to automatically segment common objects among input images with minimal additional information provided. It can be used in many applications which need manual labeling. According to the number of kinds of foreground objects, the cosegmentation algorithms could be grouped into two categories. One is to segment one kind of common objects in an image set. The features of different images are extracted and used to segment out common objects. Most of these algorithms were unsupervised. However, these algorithms could not deal with multiple kinds of common foreground objects. To deal with this problem, the other approaches used the concept of multiple objects classes to find the different kinds of common objects in an image set. They classified the parts of each image into multiple objects classes by using clustering algorithms or multiple object models, and produced the segmented result objects. However, most of these algorithms are not fully unsupervised.

* Corresponding Author

In the work, we consider an image as a composition of several objects, and each object is composed of several object elements. Then an image is decomposed into a number of object elements. Those object elements with similar features could be grouped into one object element cluster. The object elements belonging to the same cluster may appear in an image or several images, and some object elements located on the same image could form a sub-object in an image. A combination of several image elements in an image from different sub-object classes forms one object. We adopt a clustering algorithm to get the object element clusters. However, it is not easy to select the appropriate number in advance since the number of object element clusters depends on the input image set. In the work we adopt an unsupervised clustering method to classify these sub-object classes. Besides, when some object elements only appear in a few images, they are possible to be misclassified. For the above consideration, DBSCAN [1] is adopted in the work. DBSCAN is an unsupervised clustering algorithm and generally used in data mining. The advantage is that DBSCAN can discover the clusters with arbitrary shape, and need not set the number of clusters. A cluster is considered to be located at the high-density space in a feature space. If the density around an object is low, this object would be considered as a noise object. The number of clusters of each image set could be automatically found, and independent object elements could be prevented from being falsely classified. The existing cosegmentation algorithms assume that the foreground objects should appear in all images in the image set. For some image sets, the foreground objects may appear in a few images. Their corresponding object elements are possible to be classified to the wrong classes. Therefore, we propose a selection criteria to handle this issue. If the objects satisfy the selection criteria, they would be selected as one kind of foreground objects, even if they do not appear in all images. Chang and Wang [27] is the previous work of our approach.

The rest of this paper is organized as follows. Section 2 presents the related works, and our proposed algorithm is described in Section 3. Two kinds of experiments are shown in Section 4. One is for an image set with the same kind of objects; the other is for an image set with multiple kinds of objects. The experimental results demonstrate that our algorithm can achieve better segmentation accuracy. Section 5 draws the conclusions.

2 Related Work

Since human labor is usually needed in segmentation, some automatic segmentation algorithms were developed to handle the problem. Rother et al. [2] first proposed a cosegmentation algorithm that segmented out the common object from two images. This algorithm adopted the concept of GrabCut [3] that transformed the foreground segmentation problem into an energy minimization problem of a Markov Random field (MRF). They presented a cosegmentation MRF model by composing two image graphs and a global term. A foreground model is adopted to segment the common object from these two images and corrected the results based on the global term. Mukherjee et al. [4] extended the cosegmentation energy model to L2-norm model by modifying the global term, and used Pseudo-Boolean optimization to solve the energy minimization problem. They improved segmentation accuracy and reduced the computation speed. In the same year, Hochbaum and Singh [5] presented a reward model to modify the global term. They used the max-flow algorithm to solve the energy minimization problem. These algorithms could only work on two input images.

The cosegmentation algorithms can be classified into two categories according to their models. One category ([6-8]) developed the method based on the MRF energy model (Hochbaum and Singh [5]). Because the MRF model of [4] was originally designed to be applied to two input images, the computing load is large if the number of input images increases. Many algorithms worked on the modification of the MRF model. Chu et al. [6] presented an algorithm to deal with multiple objects in multiple images by integrating the confidence term and locality term. They used the idea of common pattern discovery to build the confidence map which is applied to find the regions where the similar objects are located and then used segmentation algorithm to segment the common objects on each image. The value of confidence map described the occurrence possibility of an object. Mukherjee et al. [7] presented a modified MRF model with a new global term, which could deal with the multiple images containing objects of different scales. The proposed energy model is decomposed into sub-modules, which could be solved by using Quadratic Pseudo-boolean function. Chang et al. [8] assumed that the similar objects may be located on the saliency regions. The co-saliency map is constructed by comparing the saliency region of each image, and the foreground model is learned by the co-saliency map. The original image is

transformed to an image graph of super-pixels to reduce the computation time.

The other category ([9-12]) considers the cosegmentation problem as a clustering problem. These algorithms find the groups formed by similar objects through the clustering process. Joulin et al. [9] proposed an energy model which includes two terms: one term maintained the spatial consistency of a single image by spectral clustering, and the other found common objects from different images by using discriminative clustering. This model was one of few models that could deal with multiple images at that time. Kim et al. [12] used a hierarchical clustering to solve the cosegmentation problem. A hierarchical super-pixels structure was used to represent an image, where each layer of the structure is an image of super-pixel with different numbers. Two terms were used to find the common objects among several images: the intra-image connection describes the relationship between the super-pixels on different layers, and the inter-image connection describes the super-pixels on the coarsest layers of two different images. Kim et al. [10] dealt with the cosegmentation problem by using multiple object classes. They transformed all images into the super-pixels for construction of the image graphs. Then those super-pixels were clustered by anisotropic diffusion clustering. Finally, they got similar objects by choosing some of the clusters. Their model could solve the images with higher complexity. Joulin et al. [11] also used multiple objects classes to handle the cosegmentation problem. They considered that the foreground and background were composed of several different objects and presented an energy model for multiple classes by modifying the original energy model for two classes. The energy model was optimized by the expectation minimization (EM) algorithm.

Besides the above studies, Batra et al. [13] presented an interactive cosegmentation system. Their system allows users to specify the foreground. Users select one image from the image set and label the foreground and background region. The system would learn the foreground model to segment out the foreground from the other images. Besides, they also built a standard database, iCoseg. This database that contains 38 groups and pixel-wise hand-annotated ground truth has been a comparison standard database. Kim and Xing [14] presented an interactive cosegmentation which could deal with multiple foregrounds. According to users' labeling, they learned the multiple foreground models by using GMM and SPM models. Vicente et al. [15] presented an object cosegmentation algorithm to reduce the segmentation error by a training process. The objects were extracted by using the automatic object segmentation algorithm, and two kinds of object features were adopted for training an image classifier. Collins et al. [16] presented an algorithm that used Random Walker algorithm instead of the MRF models. The cosegmentation model also considered two issues of multiple images and different objects scales. The model could allow the nonparametric representation of the foregrounds.

DBSCAN [1] is a density-based clustering algorithm that can search the clusters with arbitrary shape along with outliers. It is a full searching approach, so the computation loading is a critical issue when the size of input data increases. Some algorithms ([17-19]) focused on the improvement of computation speed. El-Sonbaty et al. [17] added the CLARANS (Cluster Large Applications based upon Randomized Search) for reducing the searching number of data points. Liu [18] sorted the data points of the database and searched the neighbors of each data point for reducing the computation. Viswanath and Pinkesh [19] presented a hybrid algorithm that adopted two types of prototypes: one is at the coarse level to reduce the consuming time, and the other is at the finer level to decrease the deviation of the results.

Recently, some studies applied the technique of deep learning to perform cosegmentation. Li et al. [20] proposed a new CNN-based method to solve the problem of object class co-segmentation, which jointly detects and segments the objects of a semantic class from a pair of images. Chen et al. [21] presented an attention-based deep object co-segmentation model by using a semantic attention learner. Three different architectures of attention learners were proposed in the work. Mukherjee et al. [22] treated object cosegmentation as a clustering problem and used semantic segmentation to segment the similar images based on a deep Siamese network. This work showed that deep features could provide more discriminative features. Hsu et al. [23] presented an unsupervised a CNN-based method, which has a better performance than supervised methods in the experimental results.

3 Cosegmentation Algorithm

Cosegmentation is considered as a problem of multi-label region classification along with noise in this work. Firstly, we segment all images into lots of object elements using a super-pixel algorithm, and then extract the feature vector for each object element. Secondly, we apply the clustering algorithm to get the

sub-object classes. Finally, we select the sub-object clusters that satisfy the selection criteria for cosegmentation results.

3.1 Cell Extraction

In this section, we transform the pixel-based images to cell-based ones (super-pixels). One object in the image is composed of several object elements. It is not a good choice to find the sub-object classes based on the pixel-based image because pixels do not have enough features and a large number of pixels would cause extensive computation loading. A cell is a region which consists of a group of similar pixels. Using the cell as the basic unit has two advantages. A cell owns more features than one pixel, and the number of cells is much smaller than the number of pixels. It would reduce the computation time. In this work, we select Normalized cut [24] to transform images from pixels to cells.

Normalized cut considers an image as a graph, and pixels as nodes. It checks the affinities (similarities) between nodes and their neighbors. If the affinity is strong, the node and the neighbor would be grouped into the same region. If the affinity is weak, the node and the neighbor would be grouped into different regions. If some pixels are labeled as A, the affinities of the edges between A nodes are strong. The affinities between B nodes are also strong. The affinities of edges between the A nodes and B nodes are weak. The measure of normalized cut is defined as:

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)} \tag{1}$$

Where $cut(A, B)$ is the sum of all weight of similarity between A nodes and B nodes. It is defined as:

$$cut(A,B) = \sum_{i \in A, j \in B} w_{ij} \tag{2}$$

Fig. 1. The result of Normalized cut

Where w_{ij} is the weight of similarity between two pixels i and j. Fig. 1 shows an example of the Normalized cut results where each region depicts one cell.

3.2 Clustering Process

After transforming the images into cell maps, we extract color and texture features for each cell. The feature vector of the *n-th* cell in image is represented as $r_{n,i} = (color_{n,i}, texture_{n,i})$, where $color_{n,i}$ is the color histogram and $texture_{n,i}$ is the average SIFT descriptor [25] of the *n-th* cell of image i. We select HS (Hue-Saturation) color histogram model to represent the color distribution, where hue and saturation of the model are divided into 10 levels to creature 10×10 bins. SIFT feature has robustness for rotation and scale. The SIFT descriptors are composed of the orientation of the neighbor pixels. All the SIFT descriptors within a cell are averaged to get the texture feature.

The cells of the image set are grouped into different sub-object classes. Those cells without sufficient similar cells are considered as independent cells and ignored since they cannot form sub-object classes. Fig. 2 shows the processing steps. Each image is transformed into a number of cells, and all the cells from the image set are collected as a global cell set. Different from the early multi-class approaches, we do not set the number of classes in advance. Our approach classifies the cells of the global set into sub-object classes. Our approach adopts DBSCAN [1] as our clustering kernel. DBSCAN clusters objects in feature space according to the object density distribution. If the number of objects within the range (*Eps*) is larger than the number of minimum objects (*MinPts*), those object points would form an object cluster.

If not, those objects would be considered as noise objects. It satisfies the requirement because the noise objects could be considered as independent cells in our system. DBSCAN can find sub-object classes without setting the number of sub-object classes in advance.

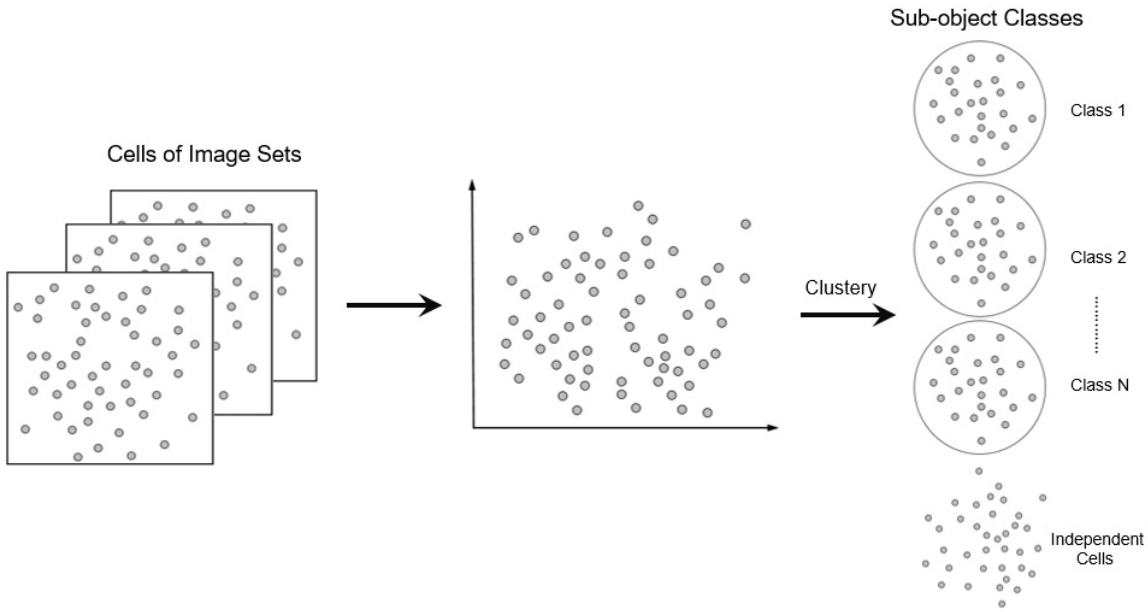

Fig. 2. The process of class clustering

In DBSCAN, some terms would be defined:

- Eps-neighbor: If the distance of neighbor is smaller than Eps, the neighbor is called Eps-neighbor.
- Directly density-reachable: If the count of Eps-neighbors of a given object p is larger than the minimum number *MinPts*, then the relation of the given object p to each Eps-neighbor is called directly density-reachable.
- Density-reachable: If an object p and an object q can be linked by a sequence of objects which are directly density-reachable in the direction from object p to object q, the relation of object p to object q is called density-reachable.
- Density-connected: If an object r could connect to object p and q with density-reachable, this relation between object p and q is called density-connected.
- Cluster: If the relation between each pair of the objects within a non-empty set is density-reachable and density-connected, this subset is called a cluster.
- Noises: If the objects are not belonging to any subset, these objects are called noise objects.

Our approach collects the objects with the relations of density-reachable and density connected. If there is no new object belonging to the cluster, it would search for the Eps-neighbors of the next object. After all the objects are visited, we could get several clusters and noise objects. We use spatial distance to represents the similarity between two cells. The spatial distance between two cells is defined as follows:

$$Eps(r_{n_1,i_1}, r_{n_2,i_2}) = \lambda_{color} Eps_{color}(color_{n_1,i_1}, color_{n_2,i_2}) + \lambda_{sift} Eps_{texture}(texture_{n_1,i_1}, texture_{n_2,i_2}) \quad \textbf{(3)}$$

Where Eps_{color} is spatial distance of color features, and $Eps_{texture}$ is the spatial distance of texture features between two cells. The parameters λ_{color} and λ_{sift} are the weight of color and SIFT vector. The range of λ_{color} and λ_{sift} is [0, 1], and $\lambda_{color} + \lambda_{sift} = 1$. $Eps_{color}(\cdot, \cdot)$ is defined as:

$$Eps_{color}(color_{n_1,i_1}, color_{n_2,i_2}) = \sqrt{1 - \rho(color_{n_1,i_1}, color_{n_2,i_2})} \quad \textbf{(4)}$$

Where the Bhattacharyya coefficient ρ is defined as follows:

$$\rho(color_{n_1,i_1}, color_{n_2,i_2}) = \sum_{j=1}^{100} \sqrt{\frac{1}{total(color_{n_1,i_1}) \times total(color_{n_2,i_2})} color_{n_1,i_1}(j) \times color_{n_2,i_2}(j)} \quad \textbf{(5)}$$

Where $color(j)$ is the *j-th* color histogram and $total(color)$ is the total number of pixels in the cell $s_{n,i}$. $Eps_{\text{texture}}(\cdot,\cdot)$ is defined as the similarity of texture features between two cells as follows.

$$Eps_{texture}(texture_{n_1,i_1}, texture_{n_2,i_2}) = \sqrt{\frac{1}{128}\sum_{j=1}^{128}(sift_{n_1,i_1}(j) - sift_{n_2,i_2}(j))^2} \quad \textbf{(6)}$$

Algorithm 1. Clustering (*S*, ε, *MinPts*)

```
1.  cid =0; //Each cluster is given an identifier cid
2.  For each cell r_a in region R do
3.   if r_a is not marked as "seen cell" then
4.     Mark r_a as "seen cell";
5.     Find similar cells N of r_a by {r_b ∈ R | Eps(r_a, r_b) < ε, r_b ∈ R, a ≠ b};
6.     if Count(N) < MinPts then
7.       Mark r_a as "independent cell";
8.     else
9.       cid = cid + 1;
10.      Mark each cell of N with cid;
11.    For each cell r_b ∈ N and r_b is not marked as "seen cell" do
12.      Mark r_b as "seen cell";
13.      Find similar cells N' of r_b by {r_c ∈ R | Eps(r_b, r_c) < ε, r_c ∈ R, b ≠ c};
14.      If Count(N') > MinPts then
15.        Mark each cell of N' with cid;
16.              If any cell of N' is marked as "independent cell" then remove that
17.              mark cid;
18.      end if
19.    end for
20.   end if
21. end for
22. Output all cells of S along with cid or "independent cell" mark
```

Where $sift_{n,i}(j)$ is the *j-th* SIFT descriptor in cell $s_{n,i}$. Algorithm 1 shows clustering procedure.

3.3 Cluster Selection

In this section, we de-project the cells of sub-object classes back to images to get the distribution of cells on images and then select the sub-object classes that satisfy the selection criteria. If the count of cells of some sub-object class is large enough to form a cluster; however, these cells only distribute in a few images. Therefore, they are not recognized to form common objects and are rejected. Selection criteria are used to filter out such classes.

After density-based clustering, we can get the number of clusters *K*. These clusters may appear in a few images or most of the images. To determine which clusters satisfy the selection criteria, we get the distribution of cells of each cluster. Firstly, We build up the histogram NOR_k that represents the number of cells of each image in the cluster C_k. According to histogram NOR_k, we define $COI_k(i)$ that represents whether the cells of cluster C_k appear in image *i*.

$$COI_k(i) = \begin{cases} 1 & NOR_k(i) > \alpha \\ 0 & others \end{cases} \quad \textbf{(7)}$$

Where $NOR_k(i)$ is the number of cells of image i in cluster C_k. α is the threshold to determine whether the cluster C_k satisfies the requirement of a single image. Then, we can get the total number NOI_k of the images, which is the sum of $COI_k(i)$ for cluster k. A rate P_k is defined to describe the ratio between the count of images containing the cells of cluster k and the count of total images.

$$P_k = \frac{NOI_k}{|I|} \tag{8}$$

Where $|I|$ is the total number of images in the image set. $Check_k$ is a Boolean parameter representing whether cluster C_k is recognized as a sub-object class or not.

$$Check_k = \begin{cases} 1 & P_k > \beta \\ 0 & others \end{cases} \tag{9}$$

According to this filtering process, we can filter the unexpected clusters and retain the cosegmentation results. By this filtering, our algorithm could deal with the image set that some objects only appear in a few iamges.

Fig. 3 is the image results mapping from the feature points in feature space. Fig. 3(a), Fig. 3(b), Fig. 3 (c), Fig. 3(d) are the original images, and Fig. 3(e), Fig. 3(f), Fig. 3(g), Fig. 3(h) are the results of which the colors depict different sub-object clusters. The gray regions means those regions are independent cells because their count number is not large enough. We also observe that the blue regions only exist in Fig. 3(e), Fig. 3(h). Those regions can form a cluster, but they do not appear in all of the images.

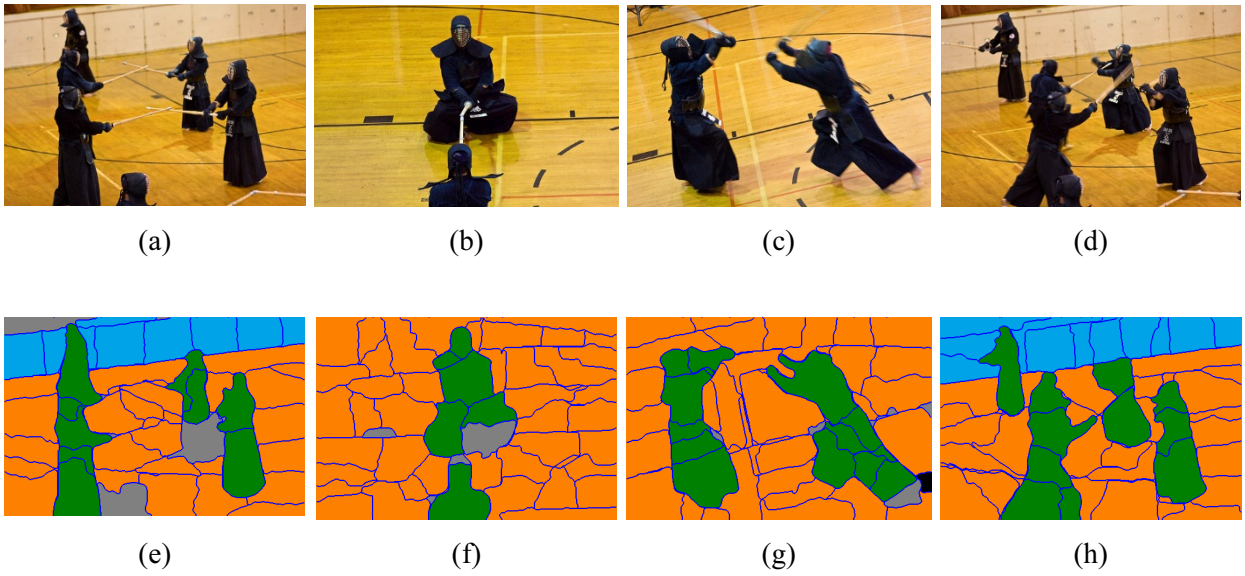

Fig. 3. Original and cosegmention result images

4 Experimental Results

In this section, we evaluate our algorithm with two experiments according to the types of test image sets: (1) Image set including the same kind of foreground objects: we use the image sets from the databases, iCoseg and MSRC-v2. (2) Combined image set including several kinds of foreground objects: our approach is extended to deal with the image sets which have different kind of foreground objects. Our algorithm is coded in C++ and operated on Acer M4610 with CPU i5-2320.

4.1 An Image Set with the Same Kind of Objects

In this section, we evaluate the performance of the proposed approach on two databases: iCoseg and MSRC. We only choose some classes from MSRC because the foreground objects in the same MSRC class have a lot of variation with color and texture. The complexity of DBSCAN is $O(n^2)$, where n is the total count of cells. We get the results of [10-11] by using their released codes. The parameters of our algorithm include the spatial distance $\varepsilon \in (0.22, 0.39)$, the image filtering parameter $\alpha = 2$, and the cluster filtering threshold $\beta = 0.6$.

When evaluating the segmentation accuracy, we choose the labeled foreground object as the ground truth. Since the images in MSRC database have multiple labels, we choose the main object category as ground truth of foreground objects, and other categories as the background [11]. Segmentation accuracy is measured by the intersection-over-union score which is a standard in PASCAL challenges and defined as $\frac{1}{|I|}\sum_{i \in I} \frac{GT_i \cap R_i}{GT_i \cup R_i}$, where GT_i is the ground truth, and R_i is the extracted object which is the combination of several cells belonging to the same or different sub-object classes in image *i*. An object is usually segmented into several cells belonging to the same class. But for some objects, the texture or color feature of segmented parts are so different that these parts may be classified into different class.

Fig. 4. Experimental results. The first row: original images; the second row: results of [11]; the third row: our results; the fourth row: labels for comparison

Fig. 4 shows the results of our method and [11]. The same color means the same class. The gray-color cell is the independent cells. In our result images, the main object was segmented into two parts because their colors are different. The ground-truth of object regions are also shown in the figure.

Table 1. Results on iCoseg and MSRC

Image #	Dataset	Class	Ours	[11]	[10]
30	MSRC	Bike	**45.9**	43.3	29.9
30		Tree	**68.9**	67	60
30		Face	52	**70.5**	33.2
5		Brown Bear	66.1	**75.6**	40.4
7		Stonehenge	80.6	**86.3**	64.4
11		Skating	**72.5**	64	51.1
11		Ferrari	**69.2**	65.2	60.5
11		Helicopter	**78.2**	59.7	13.4
12	iCoseg	Statue of Liberty	**94.5**	72.4	50.4
17		Monk	73.1	**77.6**	71.3
18		Panda	**73.6**	55.9	39.4
25		Baseball Player	**65.1**	62.2	51.1
30		Kite Colt	**61.1**	47.9	20.7
30		Kendo	87.6	77.8	**88.7**
41		Kite Panda	57.2	57.8	**66.2**

Table 1 gives a quantitative comparison with [10] and [11], where the best results are shown in bold. Our algorithm achieves the best performance on 9 out of 15 object classes. Moreover, our approach is an unsupervised approach with fewer parameters.

The performance of our approach is better in the class of Statue of Liberty. Most of the images in the class include three objects: a statue of Liberty, the base, and the sky (Fig. 5). Some images include only two objects since the base does not appear in all images. Some images with two objects would not be segmented well by [11] because some regions will be misclassified to the wrong cluster when the class number is set to be 3.

Fig. 5. The segmentation results of the Statue of Liberty. The first row: original images; the second row: the results of our method; the third row: the results of [11]

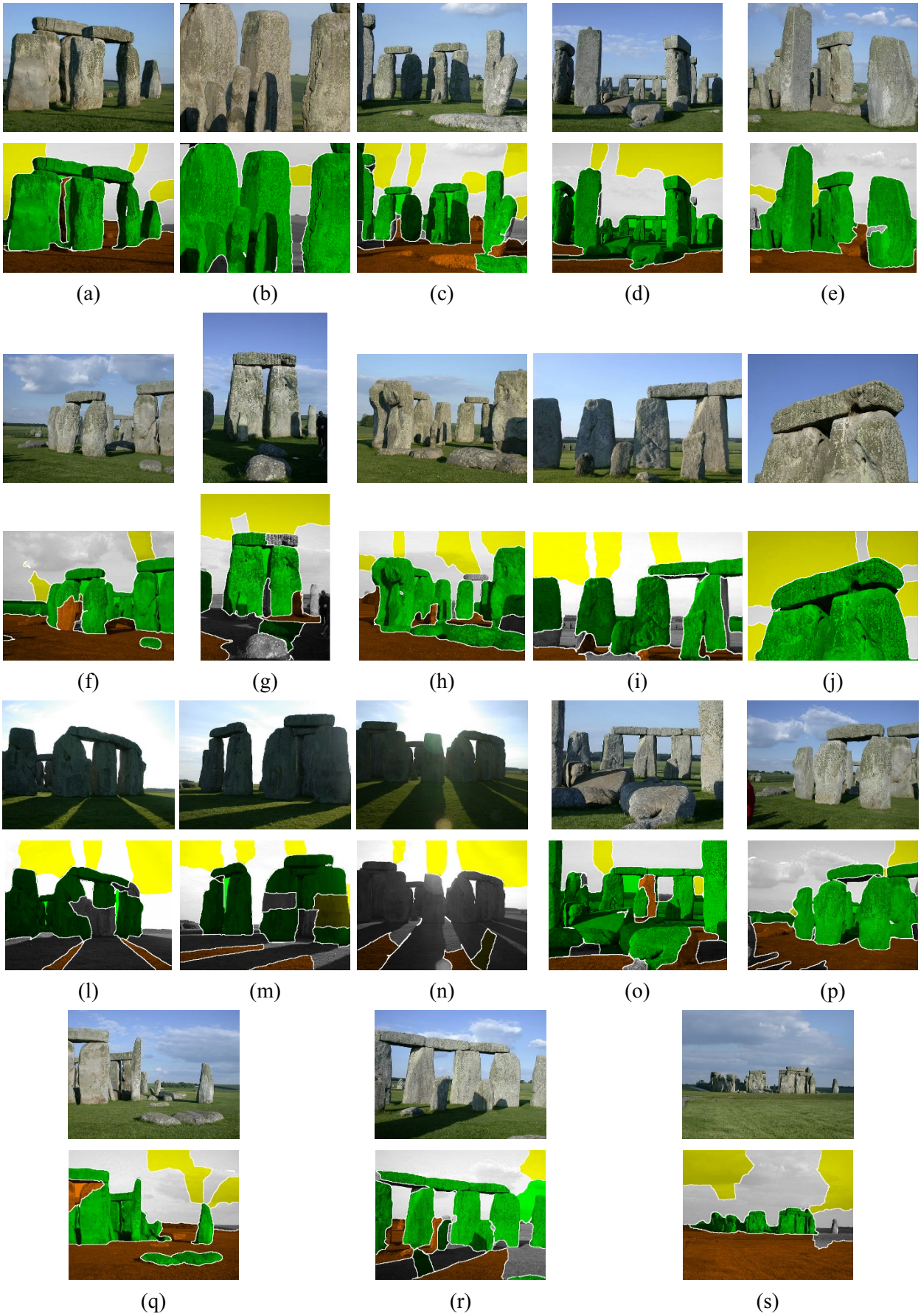

Fig. 6. The results of the class of Stonehenge. The upper row is the original images and the lower row is our results

Fig. 6. shows the results of our algorithm for the class of Stonehenge. Overall speaking, most of the segmentation results are well. Segmentation errors usually occur in the images with improper exposure, such as Fig. 10(l), Fig. 10(m), Fig. 10(n), since the color of the stone hedge is very different from that on the other images. Therefore, these regions are determined to be independent objects.

Fig. 7. Some other results of the proposed algorithm. The upper rows are original images, and lower rows are our results

Fig. 7 shows some other results. Fig. 7(a) depicts that the cars are well-segmented from the image set; besides, the floor in the first two images are clustered to the same class, and the background of the third image is recognized as the cell belonging to a different class. We can see that some classes do not appear in all images at each image set. Fig. 7(b) is another example. The men and floor are successfully segmented from the image set. Note the wall in the second image is recognized as the region belonging to another class. Since the proposed algorithm is unsupervised, it can find a suitable number of classes to get the appropriate results.

4.2 An Image Set with Different Kind of Objects

This section shows the segmented results for the image sets containing different kinds of foreground objects. We evaluate our algorithm with three experiments. Firstly, the test image set includes the images from one category plus one image from others. Our algorithm can avoid that the cells of the images belonging to different categories being misclassified into the same object classes. Secondly, the test image set includes the images from two different categories of iCoseg database. We can segment the forground cells of different categories to different classes. Finally, we select images from three categories of iCoseg database to form the test image set.

In the first experiments, we select nine images from iCoseg kendo category and one image from the baseball player category to form an image set (Fig. 8). The baseball player image is a noise image which could affect the cosegmentaion results. Our algorithm can avoid misclassification of the baseball player and recognize it as the noise image as shown in Fig. 9(j). The grey appearance means that the cells from Fig. 9(j) are independent cells. Fig. 10 shows the results of [11]. Some background parts are classified as the class of the foreground objects (Fig. 10(g) to Fig. 10(i)), and the baseball player is labeled as the kendo player (Fig. 10(j)). [11] classified the foregrounds into the same class, even these foregrounds are not similar.

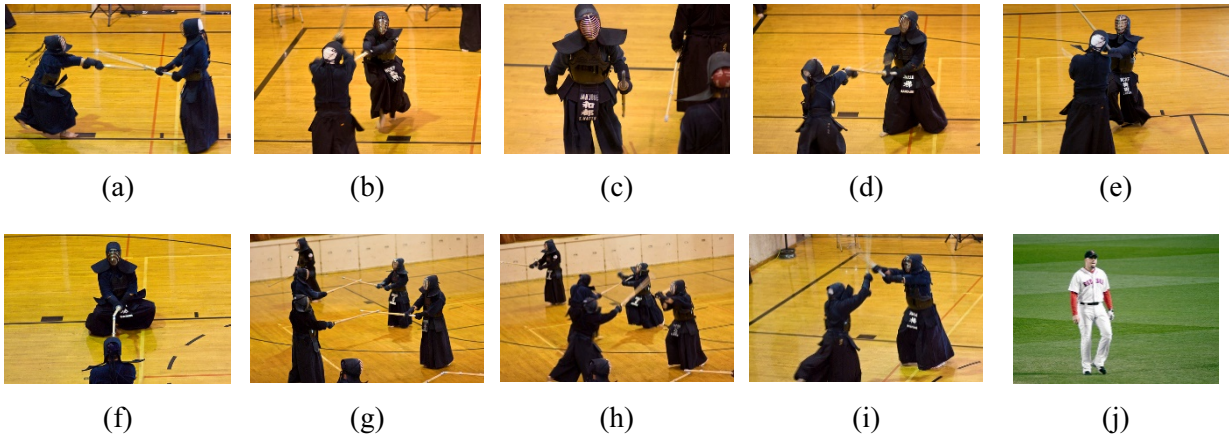

Fig. 8. Test input image set

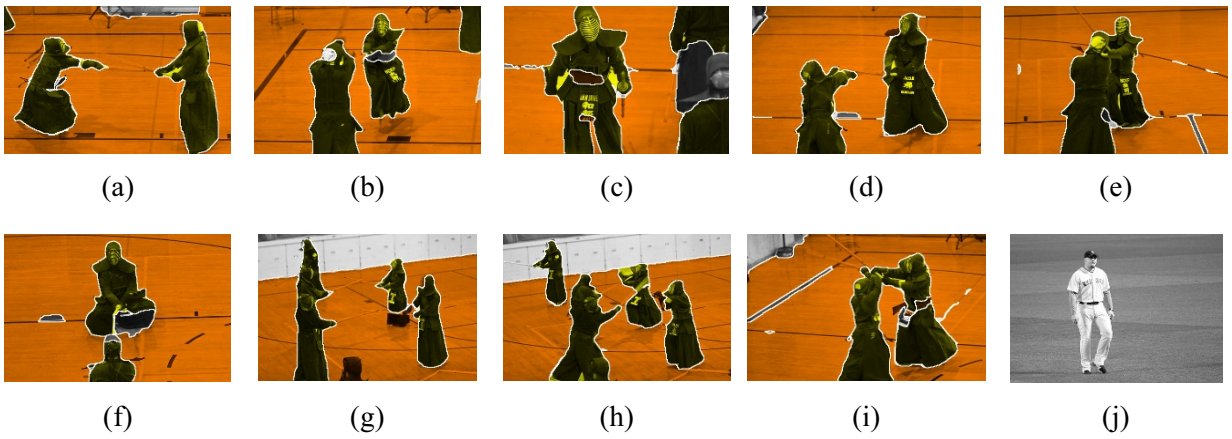

Fig. 9. Our cosegmentation results

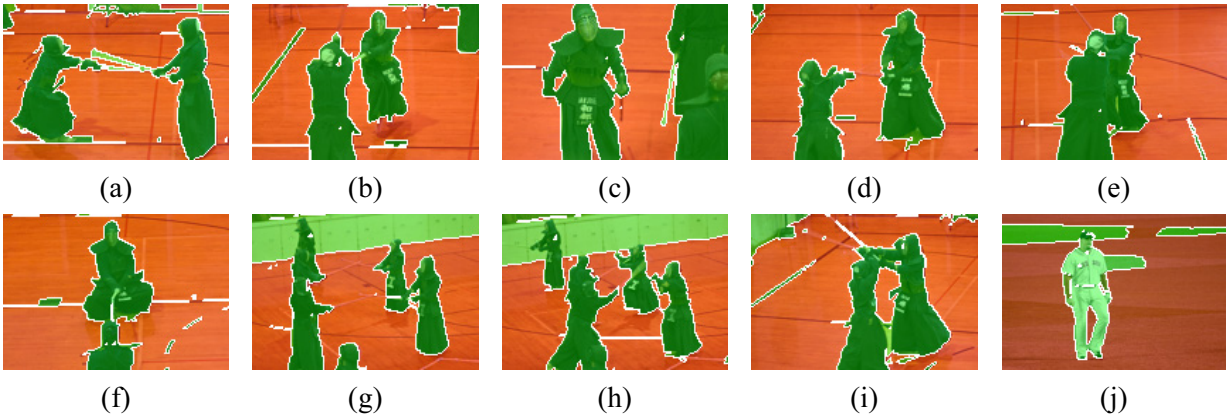

Fig. 10. The cosegmentation results of [11] with two classes

In the second experiment, the image set includes 5 images from Ferrari category and 5 images from Stonehenge category (Fig. 11). Fig. 12 shows the segmentation results of our method. In Fig. 12(a) to Fig. 12(e), most of the cells corresponding to Ferrari are segmented out and labeled with blue color, and the cells corresponding to Stonehenge are labeled with orange color in Fig. 12(f) to Fig. 12(j). It is observed that the foreground objects are well-segmented; furthermore, the results would be more complete if the additional merging process is applied. Fig. 13 shows the results of [11] with class number is 3. Because each image should be segmented into three classes, more segmentation errors occur in Stonehenge result images. In Fig. 14, the number of classes is set to be the same as ours. It is found that many cells of the same objects are falsely classified into different object classes if the class number is not properly selected.

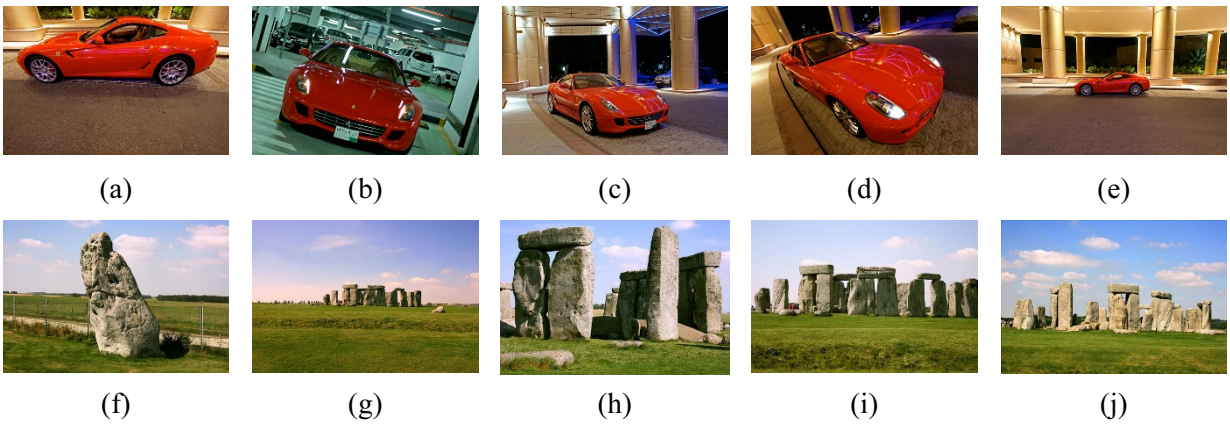

Fig. 11. The image set includes two kinds of foreground objects

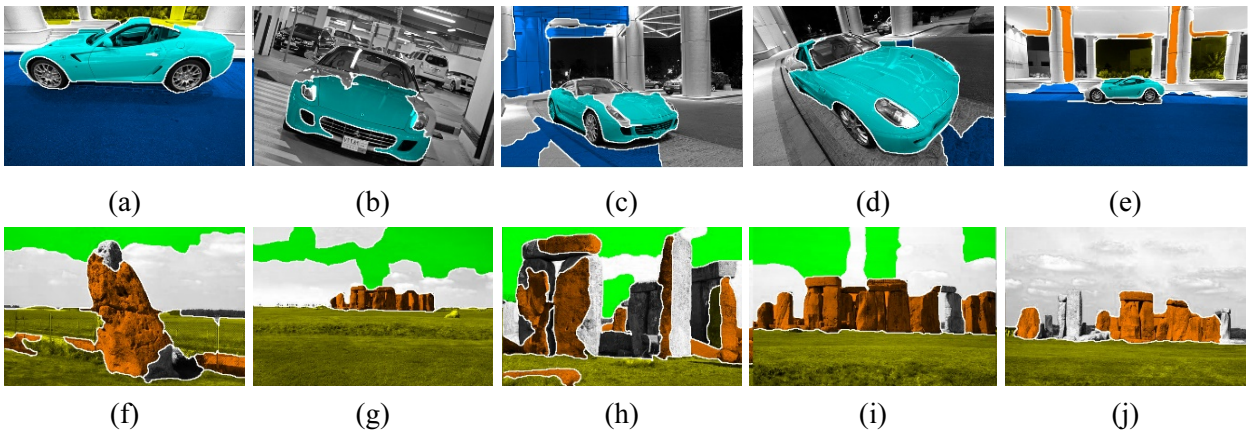

Fig. 12. The result images of our method

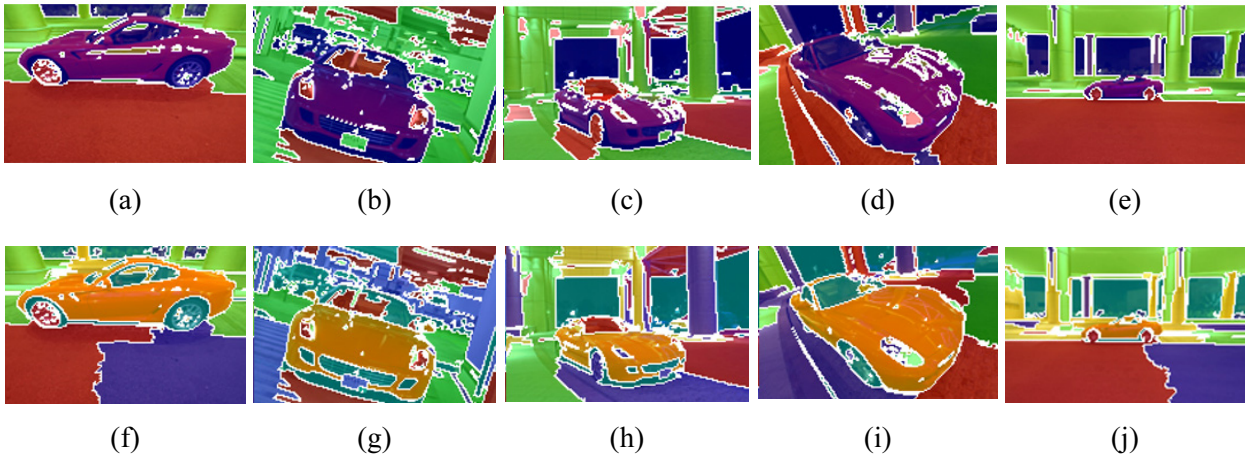

Fig. 13. The result images of [11] and the class number is set as 3

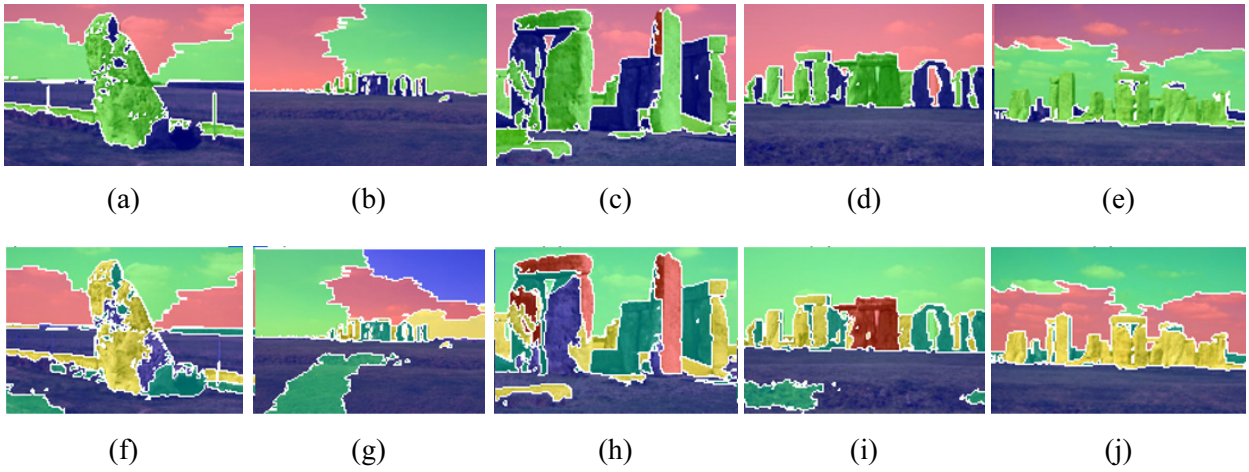

Fig. 14. The result images of [11] and the class number is set as the same as ours

The image set for the third experiment includes three kinds of foreground objects. The image set is composed of 5 images from the State of Liberty category, 5 images from Agra Taj Mahal category, and 5 images from Cheetah category (Fig. 15). Our algorithm can differentiate the independent cells and similar cells (Fig. 16) and most of the foreground objects are well labeled. Skin parts are classified into different classes because the texture distributions of some parts are different from others; however, we can merge them together to form more complete skin areas. We also evaluate the performance of [11] by segmenting these images as the class number is 3 (Fig. 17). It is observed that the results could be good if the class number is selected properly, but it is not easy to get a proper class number. Fig. 18 shows the results of [11] when the number of classes is set to be the same as ours. Most of the segmented foreground objects are broken in these result images.

5 Conclusions

We propose an unsupervised framework to deal with the cosegmentation problem. We first transform pixel-based images to cell-based ones, and then the DBSCAN is applied to cluster the cells into sub-object clusters. Several sub-objects can form a whole object, and finally the common objects are identified. Our method can detect the independent cells and filter out the unexpected clusters. We demonstrate the performance of our algorithm in the experiment results which show our algorithm could get better segmentation results. Besides the image sets with the single kind of objects, our method can handle the image sets with several kinds of objects.

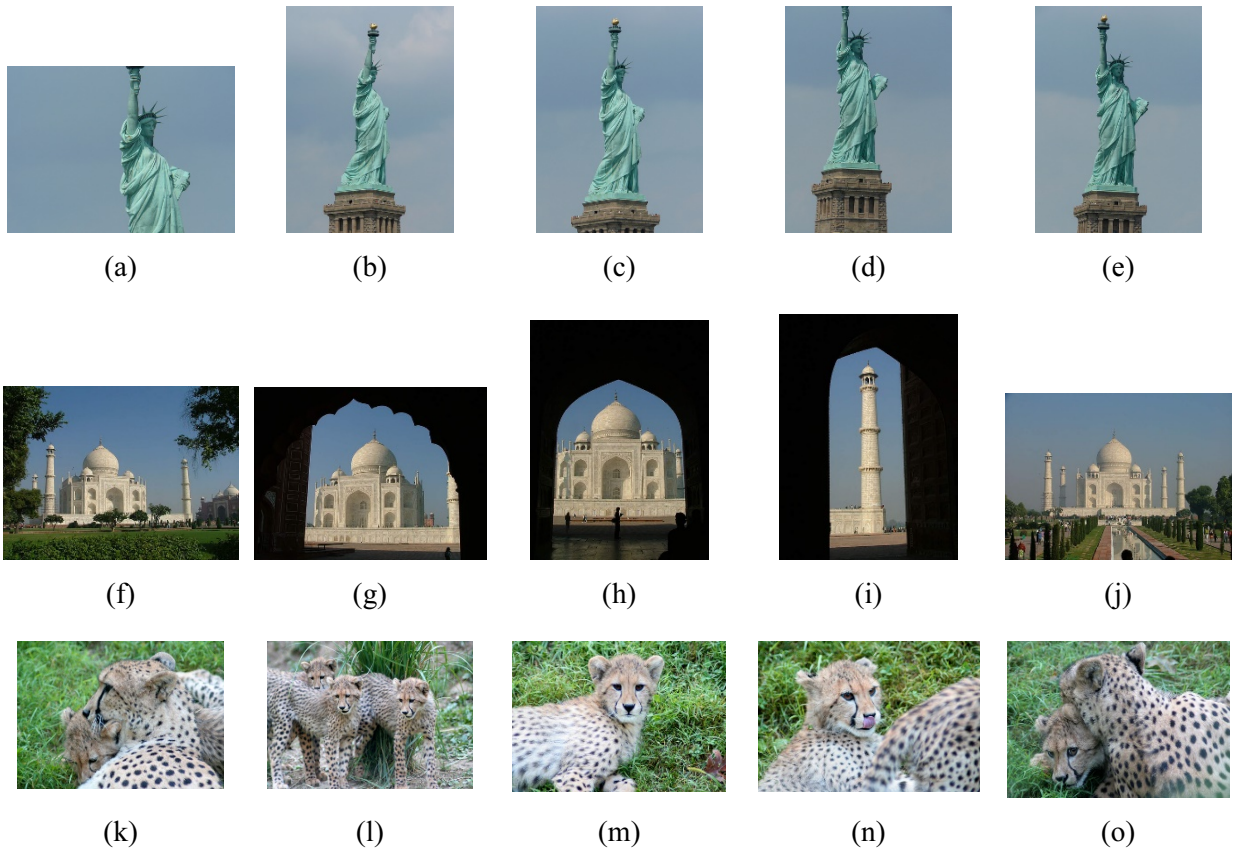

Fig. 15. The image set contains three kinds of foreground objects

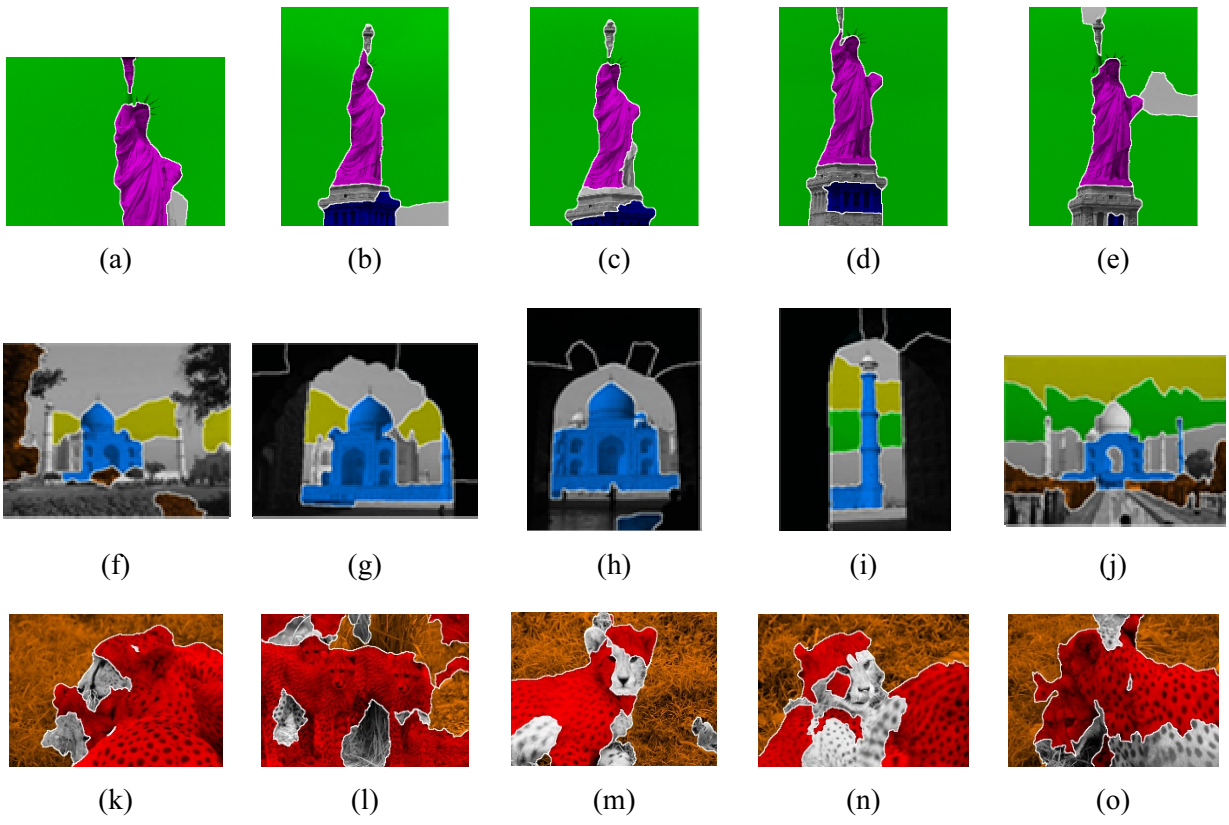

Fig. 16. The result images of our method

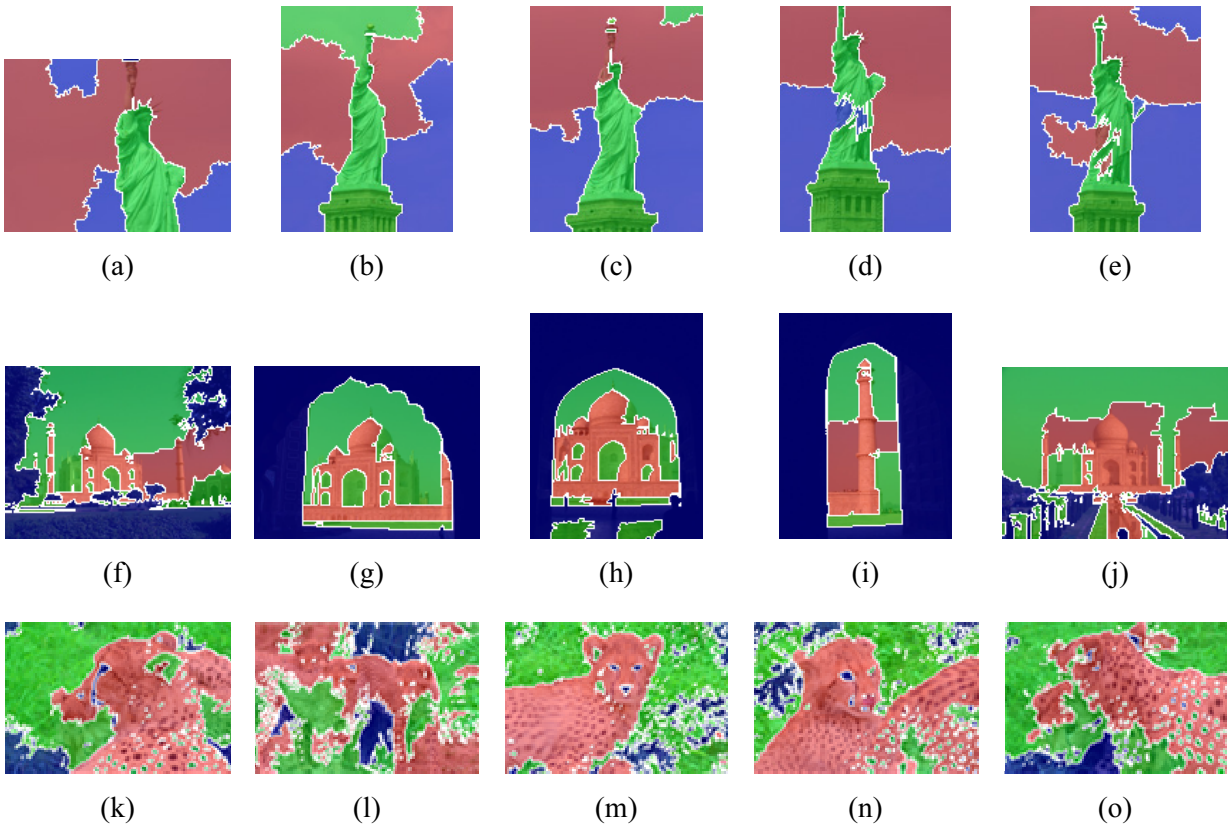

Fig. 17. The result images of [11] and the class number is set as 3

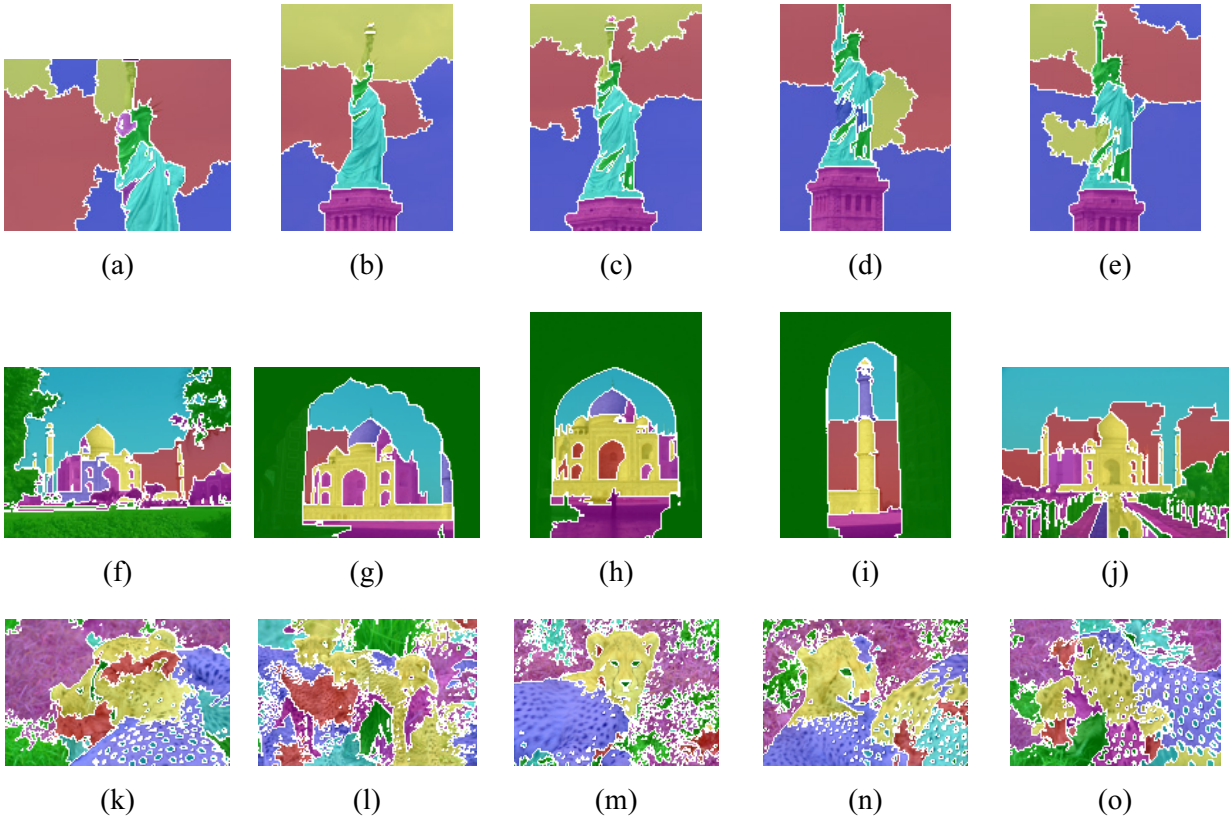

Fig. 18. The result images of [11] and the class number is set to be the same as ours

Our algorithm is hard to identify a complete common object if the parts of the object have a big difference in color or texture. In the future, we will try to merge the sub-objects into a complete object through analyzing the relationship between these sub-objects. Moreover, we also try to improve the clustering algorithm to reduce the error rate.

Acknowledgments

This work was supported by Ministry of Science and Technology, Taiwan, ROC, under Grant No. 106-2221-E-259-020-MY2 and 108-2218-E-259-001.

References

[1] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proc. Knowledge Discovery and Datamining, 1996.

[2] C. Rother, V. Kolmogorov, T. Minka, A. Blake, Cosegmentation of image pairs by histogram matching– incorporating a global constraint into MRFs, in: Proc. 2006 Conference on Computer Vision and Pattern Recognition, 2006.

[3] C. Rother, V. Kolmogorov, A. Blake, Grabcut: interactive foreground extraction using iterated graph cuts, in: Proc. 2004 SIGGRAPH, 2004.

[4] L. Mukherjee, V. Singh, C.R. Dyer, Half-integrality based algorithms for cosegmentation of image, in: Proc. 2009 Conference on Computer Vision and Pattern Recognition, 2009.

[5] D.S. Hochbaum, V. Singh, An efficient algorithm for co-segmentation, in: Proc. 2009 Conference on Computer Vision and Pattern Recognition, 2009.

[6] W.S. Chu, C.P. Chen, C.S. Chen, MOMI-cosegmentation: simultaneous segmentation of multiple objects among multiple images, in: Proc. 2010 Asian Conference on Computer Vision, 2010.

[7] L. Mukherjee, V. Singh, J. Peng, Scale Invariant Cosegmentation for image groups, in: Proc. 2011 Conference on Computer Vision and Pattern Recognition, 2011.

[8] K.Y. Chang, T.L. Liu, S.H. Lai, From co-saliency to co-segmentation: an efficient and fully unsupervised energy minimization model, in: Proc. 2011 Conference on Computer Vision and Pattern Recognition, 2011.

[9] A. Joulin, F. Bach, J. Ponce, Discriminative clustering for image co-segmentation, in: Proc. 2010 Conference on Computer Vision and Pattern Recognition, 2010.

[10] G. Kim, E.P. Xing, L. Fei-Fei, T. Kanade, Distributed cosegmentation via submodular optimization on anisotropic diffusion, in: Proc. 2011 IEEE International Conference on Computer Vision, 2011.

[11] A. Joulin, F. bach, J. Ponce, Multi-class cosegmentation, in: Proc. 2012 Conference on Computer Vision and Pattern Recognition, 2012.

[12] E. Kim, H. Li, X. Huang, A hierarchical image clustering cosegmentation framework, in: Proc. 2012 Conference on Computer Vision and Pattern Recognition, 2012.

[13] D. Batra, A. Kowdle, D. Parikh, J. Luo, T. Chen, Icoseg: interactive co-segmentation with intelligent scribble guidance, in: Proc. 2010 Conference on Computer Vision and Pattern Recognition, 2010.

[14] G. Kim, E.P. Xing, On multiple foreground cosegmentation, in: Proc. 2012 Conference on Computer Vision and Pattern Recognition, 2012.

[15] S. Vicente, C. Rother, V. Kolmogorov, Object cosegmentation, in: Proc. 2011 Conference on Computer Vision and Pattern Recognition, 2011.

[16] M.D. Collins, J. Xu, L. Grady, V. Singh, Random walks based multi-image segmentation: quasiconvexity results and GPU-based solutions, in: Proc. 2012 Conference on Computer Vision and Pattern Recognition, 2012.

[17] Y. El-Sonbaty, M.A. Ismail, M. Farouk, An efficient density based clustering algorithm for large databases, in: Proc. 2004 IEEE International Conference on Tools with Artificial Intelligence, 2004.

[18] B. Liu, A fast density-based clustering algorithm for large databases, in: Proc. 2006 International conference on Machine Learning and Cybernetics, 2006.

[19] P. Viswanath, R. Pinkesh, l-DBSCAN: a fast hybrid density based clustering method, in: Proc. 2006 International Conference on Pattern Recognition, 2006.

[20] W. Li, O.H. Jafari, C. Rother, Deep object cosegmentation, in: Proc. 2018 Asian Conference on Computer Vision, 2018.

[21] H. Chen, Y. Huang, H. Nakayama, Semantic aware attention based deep object co-segmentation, in: Proc. 2018 Asian Conference on Computer Vision, 2018.

[22] P. Mukherjee, B. Lall, S. Lattupally, Object cosegmentation using deep siamese network, in: Proc. 2018 International Conference on Pattern Recognition and Artificial Intelligence, 2018.

[23] K. Hsu, Y. Lin, Y. Chuang, Co-attention CNNs for unsupervised object co-segmentation, in: Proc. 2018 International Joint Conference on Artificial Intelligence, 2018.

[24] J. Shi, J. Malik, Normalized cuts and image segmentation, in: Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000.

[25] G. Lowe, Object recognition from local scale-invariant features, in: Proc. 1999 IEEE International Conference on Computer Vision, 1999.

[26] G. Mori, Superpixel code, http://www2.cs.sfu.ca/~mori/research/superpixels/.

[27] I.C. Chang, T.C. Wang, Unsupervised Multi-class cosegmentation, in: Proc. 2019 International Conference on Ubi-media Computing, 2019.