MVA'98 IAPR Workshop on Machine Vision Applications, Nov. 17-19, 1998, Makuhari, Chiba, Japan

3—2

Clustering of Learning Images and Generation of Multiple Prototypes for Object Recognition

Jin JIA
Graduate School of Science and Engineering
Shizuoka University, Japan

Keiichi ABE *
Department of Computer Science
Shizuoka University, Japan

Abstract

In this paper, we propose two methods of clustering learning images to generate prototypes automatically for object recognition. One is for clustering views of a single object and the other is for clustering different objects observed in a similar direction which belong to a same object class. In both two cases, we first group all learning images into several clusters by considering appearance features and spatial relations between these features, then we construct a prototype in each cluster.

1 Introduction

The ultimate goal of our study is to investigate how to develop a "generic" object recognition system, which can recognize, for instance, *any* chair or *any* car, etc. In considering applications of object recognition, the first problem to be solved is how to construct models. In order to develop a generic recognition system, we want to construct a model by which as more objects as possible can be recognized. Especially, when the model is constructed automatically based on learning objects, we want the model can be used to recognize those objects which did not appear in the learning object set, that is, we want to give the recognition system so-called "generalization" capability. However, the methods which recognize objects by finding correspondence between appearance features of each learning object and unseen scenes (like the ones using affine transformation[1] or CAD based methods[2]) often fail in doing this. In these approaches, for recognizing more objects, a large number of features, even entire learning objects might be stored in the memory. When the number of learning objects increases, however, the memory for storing them and the time for recognizing will also increase.

If the task of a machine vision application is to detect an object from an unseen scene, we do not have to record all characteristic information associating with each learning object. We may record common features in all learning objects only. The generated model which contains common features of all learning objects can be regarded as a prototype. However, most objects, even if they belong to a same object class, can not be described by only one prototype because of a large variety of appearance features and spatial relations among the features. In order to solve the problem of how many prototypes are needed to "cover" all learning objects, we may divide all learning objects into several clusters, so that the objects in the same cluster have similar appearance features and similar partial relations between these features. The benefit of the clustering approach is at least two-fold. The first is to reduce cost of recognition. The second is that even if an object in an unseen scene does not belong to the learning object set, if the object is similar to some learning objects, it can also be "covered" by the prototype generated based on the learning objects.

In this paper, we propose two methods of clustering images. One is about clustering views of one object, the other is about clustering different objects which belong to a same object class. The methods of prototype generation in clusters are also mentioned.

The rest of this paper is arranged as follows. We describe how to cluster views of one object in Section 2. The way of clustering views of different objects is described in Section 3. The experimental results of testing our methods are also illustrated in each section.

2 Prototype generation from views of one object

2.1 Clustering

Suppose an object is depicted by its views. The views are obtained by observing the object around it at a relatively same height and the views are arranged in order of the observing direction. All views are segmented first. We noticed that the shape of parts (they are segmented regions in each image) of an object and the relations between these parts do not change much if the object is observed from two near directions. Therefore, we can divide all views

*Address: Johoku 3-5-1 Hamamatsu, 432-8011 Japan. E-mail: {jia,abe}@cs.inf.shizuoka.ac.jp

into several clusters. Each cluster covers a range of observing directions. The clustering is implemented by considering both the change of appearance features of parts and the change of adjacency relations between parts.

Each segmentation result of a view can be described by a graph by considering adjacency relations among regions. By matching two graphs, we can find the correspondence of each part (region) in two views. Based on the matching result, first we define measures of change of appearance features of a part and of change of adjacency relations between two views. Then we cluster all views of an object as follows. Let the starting view of a cluster be V_m. First we test the following two conditions between V_m and the next view V_{m+1}. If both conditions are met, they are put in the same cluster. Pick up the successive views $V_{m+2}, \ldots$ and repeat the test. If both conditions hold between V_m and $V_{m+1}, \ldots, V_n$, but one or both of the conditions break for (V_m,V_{n+1}) pair, then we make a cluster with $V_m, V_{m+1}, \ldots, V_n$, leaving V_{n+1} as the starting view of another cluster.

Condition 1

$$\sum_{i=1}^{N} \Upsilon(diff(R_{m,i}, R_{j,i})) \times w_i \leq \alpha \qquad (1)$$

$$(j = m+1, m+2, \ldots)$$

where N is the number of matched regions in two views. $diff(\cdot,\cdot)$ is a measure of shape change we defined in our early paper[3]. $R_{m,i}$ denotes the contour of region i in V_m and $R_{j,i}$ denotes the contour of region matched with region $R_{m,i}$ in V_j. w_i is the weight for part i and α is a predefined threshold. In our experiments, we assign weights to regions according to their areas. Thus the larger region is given a higher weight. $\Upsilon(diff(R_{m,i}, R_{j,i}))$ is equal to 1 if $diff(R_{m,i}, R_{j,i})$ is larger than a threshold γ, 0 otherwise. This condition ensures that for two views to belong to one cluster, their parts must have similar shapes.

Condition 2

$$rd(V_m, V_j) \leq \beta \qquad (j = m+1, m+2, \ldots)$$

where β is a predefined threshold. $rd(\cdot,\cdot)$ is a measure of difference of adjacency relations between two views[3]. This condition ensures that the views in one cluster have similar adjacency relations among parts.

2.2 Prototype generation in each cluster

After making clusters by the method we mentioned above, a prototype is generated to represent all views in each cluster. In our study, a prototype is represented by a graph $G = \{R, E\}$. R is the node set and each node corresponds to a part in the prototype. E denotes the edge set in G. Edges represent relations between parts in the prototype. The prototype contains only those parts which exist in all views in the cluster. We use Planar Partial Curve Matching[4] method to generate appearance of parts in the prototype. In fact, for two matched parts in V_i and V_j, the mean shape of the two parts can be generated[3] by using PPCM method.

2.3 Experiment

We tested our clustering method on views of a chair. Eight views of a chair approximately each 45° apart in viewing directions are shown in Figure 1. The clustering yielded three clusters from them: $C_1 = \{V_1, V_2, V_3\}$, $C_2 = \{V_4, V_5, V_6\}$ and $C_3 = \{V_7, V_8\}$. The prototype generated in C_1 is shown in Figure 2(a). In the graph of the prototype, nodes correspond to common parts in V_1, V_2 and V_3; edges record adjacency relations among parts.

3 Prototype generation from different objects

3.1 Clustering

Clustering images of different objects is more difficult than clustering views of one object. We assume that all objects belong to a same object class and are viewed in similar directions. All images are segmented first.

The clustering of images of different objects is implemented in two steps. In the first step, we focus on adjacency relations existing in each object. Two connecting regions in an image make an adjacency relation. The basic idea in the first step is that the more similar adjacency relations exist in two objects, the higher is the possibility that these two objects belong to a same cluster.

We use distance between two adjacency relations to express the similarity. The less the distance, the more the similarity. For calculating the distance between adjacency relations, each adjacency relation is described by a vector $r_{i,j}(a_1, a_2, \ldots, a_m)$, where $r_{i,j}$ denotes relation j in object i, a_k is an attribute attached to the relation and m is the number of attributes. The specific attributes used in our experiment will be seen in Section 3.3.

The distance between two adjacency relations is the Euclidean distance between two vectors. Of course, each attribute in the vector should be normalized first.

Each object O_i can be regarded as a union of adjacency relations, i.e., $O_i = \{r_{i,j}\}$, $j = 1, \ldots, n_i$, where n_i is the number of total relations in O_i. The

distance between O_i and O_j is defined as follows.

$$S(O_i, O_j) = \frac{\#(O_i, O_j)}{n_i} + \frac{\#(O_i, O_j)}{n_j}$$

where $\#(O_i, O_j)$ is the number of *similar* relations in O_i and O_j. A *similar* relation pair is the one whose distance between two relations is less than a predefined threshold.

Once the distance between two different objects is defined, any clustering method based on the distance between objects can be used. We can get a primary clustering result after the first step.

In the first step, we considered each adjacency relation in objects separately and did not take the inter-relations among these adjacency into account. Therefore, we often can not get satisfiable clustering results. In the second step, the primary clustering result is revised based on the structure of each object.

In a primary cluster, each object can be represented as a graph. Each node in the graph corresponds to a region in the object. If two regions connect each other, then there is an edge between two nodes corresponding to these two regions. By matching two graphs, a distance between two objects can be defined. Because two graphs represent different objects, inexact graph matching method is prefered[5]. The distance is defined based on *stable matching*. We define stable matching as follows. First we match one graph to the other and then vice versa: a *stable matching* is a matched region pair which exist in both matching results. Then distance between two objects is defined as:

$$dis(O_i, O_j) = (1 - \frac{\#S}{\#T_i}) + (1 - \frac{\#S}{\#T_j})$$

where O_i and O_j are two objects, $\#S$ stands for number of stable matchings and $\#T_i$ and $\#T_j$ stand for total number of regions in O_i and O_j, respectively.

We can also define the distance between an object O_j to a cluster C_i that O_j belongs to as:

$$Dis(C_i, O_j) = \sum_{k=1}^{N_i - 1} \frac{dis(O_k, O_j)}{N_i - 1}, \quad O_k \in C_i, O_k \neq O_j$$

where N_i is the number of objects belonging to C_i.

Then the primary clustering result is revised in two steps: rejection and assignment. An object O_j is rejected from a cluster C_i if $Dis(C_i, O_j)$ is larger than a threshold δ. After rejection, the objects remaining in the same cluster are used to generate a prototype. (See Section 3.2). Then each rejected object O_p is assigned to a cluster C_i if the distance $dis(Pro_i, O_p)$ is less than δ, where Pro_i is the prototype generated in cluster C_i.

3.2 Generating a prototype in each cluster

After rejecting some objects from a cluster, the left ones are similar. In each cluster, a prototype can be generated. The prototype contains those parts which exist in all objects. The detail of the prototype generation can be found in our early paper[6]. We describe our method roughly here. The prototype generation is also based on graph matching. Like the way of generation a prototype from views of one object, by matching graphs of two different objects, we regard the stable matching parts as common parts existing in both objects.

Each prototype is represented by a graph $G = \{R, E\}$. Unlike the parts in the prototype generated in a cluster of different views of one object, which is the mean shape of two stable matching region pair, each node in the prototype of different objects may contain more than one part. Because each graph represents different object, even two stable matching regions may have very different appearances. The distance between the regions in a stable matching is calculated[3], and if the distance is less than a predefined threshold, the mean shape of these two regions is generated and is recorded in the prototype. Otherwise, both of these two regions are stored in the prototype with an "*OR*" relation. E is an edge set and records relations between nodes (i.e., regions).

3.3 Experiment

We tested our clustering method on real images. All learning images are shown in Figure 3. In the first step of clustering, we used two attributes to describe each adjacency relation: $r(Ratio, Angle)$. $Ratio$ is the ratio of areas of two regions in concern and equal to the ratio of area of bigger region to area of smaller region. $Angle$ is the angle between the line linking the centers of gravity of two regions and x-axis. We found 12 clusters in the first step: $\{O_1, O_3, O_{21}\}$,$\{O_2, O_7\}$,$\{O_4, O_{18}, O_{30}\}$, $\{O_5, O_{23}\}$,$\{O_6, O_8, O_{10}, O_{11}, O_{22}\}$,$\{O_8, O_{24}\}$, $\{O_{12}, O_{14}\}$,$\{O_{13}, O_{26}\}$,$\{O_{14}, O_{17}\}$,$\{O_{11}, O_{16}, O_{20}\}$, $\{O_{27}, O_{28}\}$. O_{25} and O_{28} have not been grouped with any others. After the second step, there are 7 clusters left, that is: $C_1 = \{O_1, O_3\}$, $C_2 = \{O_4, O_{18}\}$, $C_3 = \{O_5, O_{23}\}$, $C_4 = \{O_6, O_8, O_{10}, O_{11}, O_{19}, O_{22}\}$, $C_5 = \{O_7, O_8, O_{24}\}$, $C_6 = \{O_{12}, O_{15}\}$, $C_7 = \{O_{27}, O_{28}\}$. The rest of objects do not belong to any cluster. As an example, the prototype generated in cluster C_3 is shown in Figure 2(b).

4 Conclusion and further works

We proposed two clustering methods to generate multiple prototypes automatically from learning images for object recognition. The generated

prototypes are described by graphs. In the graphs, the appearance features of common parts existing in all learning objects and the relations between these parts are recorded. These information can be used to recognize objects in unseen scenes.

One of the further works is to try to combine two clustering methods proposed in this paper. Suppose multiple views of various objects in a class are given. First, we can obtain several prototypes for each object by using the clustering method described in Section 2.1. Then, choosing the prototypes of all objects in a similar observing direction and using the clustering method described in Section 3.1, we can generate the prototypes for images of multiple objects viewed in multiple directions.

Another further work is to test our methods in a practical object recognition system.

References

[1] R. Manmatha. A framework for recovering affine transforms using points, lines, or image brightness. In *Proc. of CVPR'94*, pages 141–146, 1994.

[2] F. Arman and J. K. Aggarwal. CAD-based vision: Object recognition in clustered range images using recognition strategies. *Image Understanding*, Vol.58:33–48, 1993.

[3] J. Jia and K. Abe. Learning and recognizing 3D objects by using partial planar curve matching method. In *Proc. of ACCV'98, Lecture Notes in Computer Science 1352*, pages 450–457, 1998.

[4] A. Pikaz and I. Dinstein. Matching of partially occluded planar curves. *Pattern Recognition*, 28(2):199–209, 1995.

[5] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*, volume 2. Addison-Wesley Publishing, 1993.

[6] J. Jia and K. Abe. Automatic generation of prototypes in 3D structural object recognition. In *Proc. of 14th Int'l Conf. on Pattern Recognition*, pages 697–700, 1998.

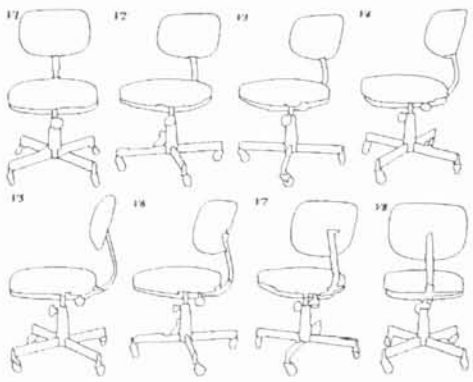

Figure 1: Eight views of an object.

Acknowledgement

This study is partly supported by the Telecommunications Advancement Foundation. The authors also thanks Dr.Takahiro Sugiyama for his helpful discussions.

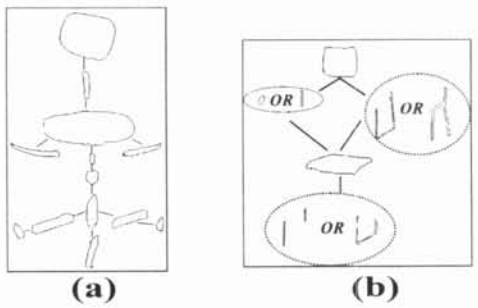

Figure 2: Examples of generated prototypes.
(a) A prototype generated from a single object.
(b) A prototype generated from multiple objects.

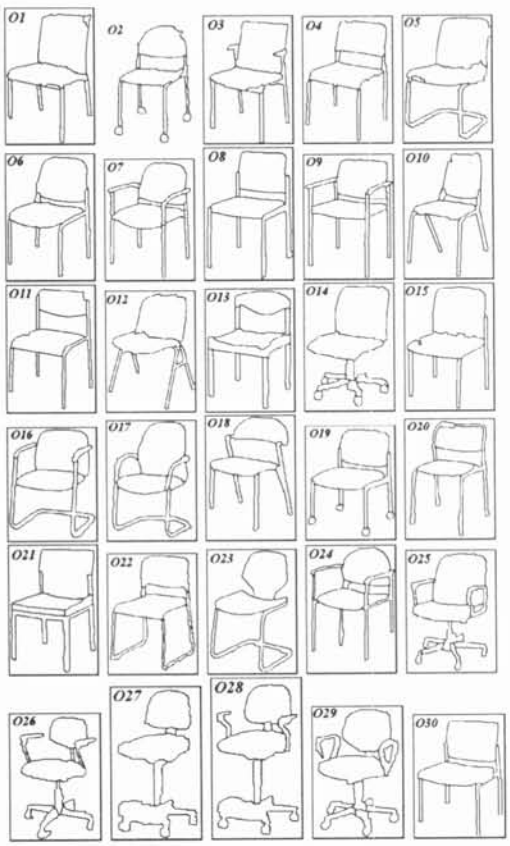

Figure 3: A set of training objects.