Learning 3D Object Recognition Models from 2D Images

Arthur R. Pope David G. Lowe

Department of Computer Science, University of British Columbia
Vancouver, B.C., Canada V6T 1Z2
Email: {pope,lowe}@cs.ubc.ca

Abstract

To recognize an object in an image one must have some internal model of how that object may appear. We show how to learn such a model from a series of training images depicting a class of objects. The model represents a 3D object by a set of characteristic views, each defining a probability distribution over variation in object appearance. Features identified in an image through perceptual organization are represented by a graph whose nodes include feature labels and numeric measurements. Image graphs are partitioned into characteristic views by an incremental conceptual clustering algorithm. A learning procedure generalizes multiple image graphs to form a characteristic view graph in which the numeric measurements are described by probability distributions. A matching procedure, using a similarity metric based on a non–parametric probability density estimator, compares image and characteristic view graphs to identify an instance of a modeled object in an image. We present experimental results from a system constructed to test this approach. The system is demonstrated learning to recognize partially occluded objects in images using shape cues.

1. Introduction

To recognize an object in an image one must have some expectation of how the object may appear. We are interested in acquiring models of objects from image data and using the acquired models for recognition and other tasks. We have in mind the following scenario. One presents to the learning system a series of example images, each depicting a particular object from some viewpoint. From those examples the system develops a model of the object's appearance. When presented with a test image, the system can identify and rank apparent instances of the object in that image. By learning models directly from images the system should be able to avoid the problem, faced by most current approaches to model–based recognition, of having to estimate actual appearance from an idealized model. In particular, the system can learn the actual range of variation in the object's appearance under normal viewing conditions. We can also expect this learning system to be more convenient to use than one that must first be supplied with its models in some encoded form.

An object may vary in appearance because of changes in camera position, lighting, or the object's shape. Further variation is possible due to noise and to differences among individual instances of the object. Accommodating such variation is a recurring issue in the design of a system that learns to recognize objects. The scheme used to represent image content must be stable so that whenever variation is small, changes in representation are likely to be small also. The scheme used to model an object's appearance must adequately describe the possible variation. The learning procedure must generalize enough to overcome insignificant variation, but not so much as to confuse dissimilar objects. And the procedure used to identify modeled objects in images must tolerate the likely range of mismatch between model and image.

In a previous paper [7], we considered the problem of learning a single characteristic view from a series of 2D training images. We are now extending that work by considering the problem of learning a set of characteristic views, automatically chosen to be sufficient for representing all aspects of a 3D object. Here we describe and demonstrate the procedures for learning and recognizing single characteristic views, and we describe the extensions now under development for learning covering sets of characteristic views.

A noteworthy feature of our approach is that it produces a model representing a probability distribution over variations in object appearance. Depending on the training images used to create it, the model can either represent a specific object with precision, or it can be a generalization encompassing an entire class of similar objects. Moreover, objects can be recognized as similar in general appearance, yet still be distinguished according to their detailed features.

2. Related research

Among published model learning approaches, one due to Connell and Brady [2] most closely resembles our own. They use graphs to represent the part/whole and adjacency relations among object regions described by smoothed local symmetries. An attribute of a region, such as its elongation or curvature, is encoded symbolically by the presence or absence of additional graph nodes. A structural learning procedure forms a model graph from multiple example graphs, most commonly by deleting any nodes not shared by all graphs. Similarity between two graphs is measured

by counting the nodes they do share. In our own approach we have placed greater importance on the need to represent probabilistic information. Rather than weight each feature of an object with equal importance, we weight each according to how likely it is to be seen with the object. Where Connell and Brady encode numeric attributes symbolically with predetermined categories, we keep them in numeric form and use them to estimate probability distributions. Our measure of graph similarity is based on this probabilistic information.

Segen [9] describes an approach where both the model and the features used to represent it are learned from images. Clusters of *n*–ary relations among distinguished points (e.g., curvature extrema) define features, and once a relation's cluster membership is determined its numeric attributes are discarded. Thus the clustering determines the extent of shape discrimination and generalization. In contrast, we include numerical attributes in the model and learn appropriate ranges for those attributes for each model feature.

In representing a model we use an attributed graph in which attribute values are characterized by probability distributions. This structure is similar to what Wong and others have called a *random graph*. He and McArthur [6] use the random graph to represent and to learn an object model from images in which the pose or feature correspondences have been provided. Attributes record feature positions in 3D space, and their distributions are modeled as Gaussian. Our approach does not require that correspondence or pose be given, and we make less restrictive assumptions about the form of attribute distributions so that our models can represent flexible objects and classes of similar objects.

3. Approach

The approach may be summarized as follows. Using shape features found in 2D training images, the system learns a model of a 3D object. The model is organized as a series of characteristic views that are formed automatically from the training images by a conceptual clustering algorithm. Each characteristic view represents a probability distribution over variations in object appearance throughout a range of viewpoints. The probability distribution, which is estimated using data collected from training images, describes the probability of various features being observed and the probability distribution of various numeric measurements. Recognizing an object in a test image, then, involves finding the most probable correspondence between features of the test image and those of some characteristic view. The form of the probability estimate allows the search for this optimal correspondence to be performed efficiently.

3.1 Image representation

Any image input to the system, whether for training or for recognition, is first represented as an attributed graph by a perceptual organization process. This *image graph* contains explicit information about the presence and spatial arrangement of significant features in the image. For our experiments, these features include line and circular arc segments, their junctions, parallel pairs of lines and arcs, closed regions, and ribbons. Features are intended to satisfy the viewpoint invariance and detection conditions described in [5], to represent explicitly all important appearance distinctions among objects of interest, and to be easily detected. All are found by a bottom–up, data–driven process of grouping and abstraction controlled by several thresholds and other parameters (like, for example, [1]).

Once detected in an image, a feature is represented by a *token* that records the type of feature plus its image location, orientation, and scale. Depending on the feature type, the token may include additional numerical attributes, such as the angle of a corner, that are always expressed in a manner invariant with respect to translation, rotation, and scaling in the image plane. The attributes associated with a token form its *attribute vector*.

All but the lowest–level tokens are created by grouping or abstracting others. An image graph records the hierarchy of tokens created from a single image: each node represents a token and the directed arcs represent composition and abstraction. The image graph deliberately retains information that may be redundant.

3.2 Model representation

A model comprises a series of characteristic views, each represented by a *characteristic view graph* (CV graph). Like an image graph, a CV graph has nodes that are feature tokens, and arcs that represent composition and abstraction relations among features. So that it can represent not just one appearance but a distribution of them, the CV graph contains additional statistical information. Each token has counts of the times that token was matched by an image token during training, plus lists of the attribute and position vectors of the matching image tokens. The overall graph has a count of the training examples used to create it. As described in the following section, this information is used to estimate the probability that a model feature will be observed in an image of the object, and the probability that is will have a particular set of attribute values.

3.3 Matching CV and image graphs

To identify an instance of an object in an image the system must find a match between a CV graph and an image graph. That match will likely be a partial one, with some image features not explained by the model (perhaps because there

are other objects in the scene) and some model features not found in the image (perhaps because of shadows or occlusion). However, the partial match should, in some sense, be an optimal one that jointly maximizes both the number of tokens matched and the quality or closeness of those matches. Our matching procedure is guided by a graph similarity measure that combines both of these factors, weighting each according to past matching experience as recorded by the CV graph. The measure combines empirical statistics using the rules of Bayesian probability theory to judge the likelihood that a given pairing of CV tokens with image tokens corresponds to an actual instance of the object in the image.

Since the graph similarity measure is described in detail in [7] we will only briefly summarize it here. It is essentially of the following form:

$$g(\text{match between CV graph and image graph}) = \sum_{\substack{\text{match between} \\ \text{CV node } i \text{ and} \\ \text{image node } j}} \log L\,(i \text{ matches } j) + \sum_{\substack{\text{unmatched} \\ \text{CV node } i}} \log L(i \text{ is left unmatched})$$

The log-likelihood of a match between an image node and a CV node ($\log L\,(i$ matches $j)$) is estimated from the image node's attribute vector plus the series of attribute vectors recorded in the CV node. The estimation problem is this: given a series of vectors $\mathbf{a}_k$, $1 \le k \le r$, drawn at random from an unknown distribution, estimate the probability of some vector $\mathbf{a}$ according to that distribution. This could be solved by assuming that the distribution is of some parameterized form (e.g., normal) and using the $\mathbf{a}_k$ to estimate those parameters (e.g., the mean and variance). However, the attribute vector distributions may be complex since they depend not only on sensor noise and measurement errors, but also systematic variations in object shape, lighting, and pose. Therefore we have chosen a non–parametric estimation method. In its simplest, form, this method estimates probability density by summing contributions from a series of overlapping kernels:

$$f(\mathbf{a}) = \frac{1}{r\,h^d} \sum_k K\left(\frac{\mathbf{a} - \mathbf{a}_k}{h}\right)$$

where d is the dimension of the vectors $\mathbf{a}_k$, h is a constant smoothing factor, and K is a kernel function. We use the Epanechnikov kernel because it has finite support and can be computed quickly:

$$K(\mathbf{x}) = \begin{cases} \frac{1}{2} c_d^{-1} (d+2)\left(1 - \mathbf{x}^{\mathrm{T}}\mathbf{x}\right) & \text{if } \mathbf{x}^{\mathrm{T}}\mathbf{x} < 1 \\ 0 & \text{otherwise} \end{cases}$$

where c_d is the volume of a d–dimensional sphere of unit radius. The smoothing factor, h, strikes a balance between the smoothness of the estimated distribution and its fidelity to the observations $\mathbf{a}_k$.

A match optimizing the metric g is found by depth–first search of the space of consistent matches. Although this search requires time exponential in the size of the CV graph, it remains practical because the search has a low branching factor. To speed the search we do the following:

- All terms are precomputed. Then any match pairs (i, j) for which $L(i$ matches $j) < L(i$ remains unmatched) are eliminated immediately since, in such cases, a better score is obtained with i left unmatched.
- Each CV token is ranked according to a measure of specificity favoring CV tokens with few match alternatives yet high probabilities of matching. More specific tokens are matched earlier in the search to minimize backtracking.
- A branch of the search tree is pruned whenever it is found that the branch cannot improve upon the best match already obtained. An upper bound on g within a branch is obtained by assuming that each unmatched CV token matches its best unmatched image token.

After the first match is struck between a CV token and an image token, each subsequent token match is found in either of two ways. First an attempt is made to match any unmatched CV tokens that are directly connected to the subgraph already matched. If no such matches can be found, we choose a CV token that is not directly connected and try to match it. Using token matches already adopted we predict the position of the unmatched token. This prediction employs the tokens' position vector series, which record the relative positions of those tokens as found in training images. Matched tokens that by their nature can be localized reliably (e.g., junctions) contribute to an estimated distribution of token location. Those whose orientation and scale can be measured reliably (e.g., parallel lines) contribute to distributions of token orientation and scale. Thus a distribution of positions is obtained for the unmatched token, and matching image tokens are sought at those positions.

3.4 Model learning procedure

The model learning procedure partitions the training images into clusters representing different characteristic views of the object. At the same time, within each cluster, it merges the training images into a single CV graph representing that cluster's characteristic view. Here we'll describe first the method of partitioning into clusters, and then the method of merging each cluster's contents into a single CV graph.

An incremental conceptual clustering algorithm is used to create clusters among the training images as they are obtained. Like, for example, COBWEB [3], the algorithm makes a series of local changes in cluster partitions according to a global measure of cluster quality while training examples are added incrementally. We use a measure of

cluster quality, based on Rissanen's minimum description length principle [8], that seeks to optimize both the complexity of the model and the degree to which the model accounts for the training images. With the clusters denoted by C_i, their corresponding CV graphs denoted by G_i, and the training images denoted by x_j, the description length measure to be minimized is of the following form:

$$D = \sum_i \text{description length of } G_i - \sum_i \sum_{x_j \in C_i} \log P\left(x_j \mid G_i\right)$$

Here $P(x_j \mid G_i)$ denotes the probability of observing training image x_j as an instance of the characteristic view denoted by cluster C_i and CV graph G_i. It is estimated in the same way that the graph similarity measure, g, is computed for an optimal match between x_j and G_i, using a non-parametric probability density estimator to estimate the probability associated with each node match.

Within each cluster, the training imags are merged into a single CV graph. An initial CV graph is formed from the first training image assigned to a cluster. When a subsequent training image is assigned to the same cluster, an optimal match is found between its image graph and the current CV graph. Each CV token that matches an image token is updated by adding the image token's attribute and position vectors to the series recorded with the CV token. New tokens are added to the CV graph for those image tokens that weren't matched by the model, and CV tokens that aren't matched often enough in training images are eventually dropped. The net effect of each training image is to extend the CV graph in some areas, prune it in others, and augment the series of attribute vectors it contains. This process is repeated with each training image to complete the CV graph.

3.5 Recognition procedure

Recognition of an object in a test image is achieved when a satisfactory match is found between a CV graph representing a characteristic view of the object, and the image graph derived from the test image. In the simplest form of recognition procedure, which we have adopted initially, each CV graph is matched against the image graph and the match that scores highest is taken as indicating the presence of the corresponding object.

Two important refinements of this procedure remain for further work. First, an indexing scheme should be used to rank CV graphs for matching. Indexing could, for example, employ groups of features, as in [4]. Secondly, a decision procedure is needed that will validate an optimal match, and either accept or reject it. The procedure will likely need more information than that provided by the graph similarity measure, g, which confounds the effects of both occlusion and shape variation. The validation might be performed by projecting edgels of the model into the image according to the mapping determined by the optimal match, and then checking the correspondence between projected and actual edgels.

4. Results

A system that learns a single, characteristic view has been implemented and tested. This "2D" system includes the perceptual organizing, graph matching, and CV graph learning components of the full "3D" system, which remains under development. Figures 1 and 2 show the 2D system learning a model for a characteristic view of a cup, and then recognizing that cup in a scene cluttered with other objects.

The 2D system shows good ability to generalize across objects of similar appearance while still discriminating among them on the basis of small distinctions in appearance. Generalization relies on finding many features that are similar among objects, while discrimination relies on finding some that differ. The learning procedure discovers the relative importance of each feature, and the extent to which its attributes may be expected to vary. An illustration of the system's generalization and discrimination performance may be found in [7].

5. References

[1] R. Bergevin and M.D. Levine, "Extraction of Line Drawing Features for Object Recognition," *Pattern Recognition* 25 (1992) 319–334.

[2] J.H. Connell and M. Brady, "Generating and Generalizing Models of Visual Objects," *Artificial Intelligence* 31 (1987) 159–183.

[3] D.H. Fisher, "Knowledge Acquisition Via Incremental Conceptual Clustering," *Machine Learning 2* (1987) 139-172.

[4] D.W. Jacobs, *Grouping for Recognition*, MIT AI Lab Technical Report 1177 (1989).

[5] D.G. Lowe, *Perceptual Organization and Visual Recognition* (Kluwer 1985).

[6] B.A. McArthur and A.K.C. Wong, "Random graph representation for 3-D object models," in *Proc. Model-Based Vision Development and Tools* (SPIE Vol. 1609, 1991) 229–238.

[7] A.R. Pope and D.G. Lowe, "Learning Object Recognition Models from Images," in: *Proc. ICCV* (1993) 296–301.

[8] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *Annals of Statistics* 11 (1983) 416–431.

[9] J. Segen, "Model Learning and Recognition of Nonrigid Objects" in: *Proc. CVPR* (1989) 597–602.