Learning to Describe and Efficiently Recognize Patterns and Objects in Scenes

Terry Caelli
Department of Computer Science, Curtin University of Technology
Perth, WA 6001, Australia
and
Walter F. Bischof
Department of Psychology, University of Alberta
Edmonton, Alberta T6G 2E9, Canada

Abstract

Machine learning has been applied to many problems related to scene interpretation. It has become clear from these studies that it is important to develop or choose learning procedures appropriate for the types of data models involved in a given problem formulation. In this paper, we focus on this issue of learning with respect to different data structures and consider, in particular, problems related to the learning of relational structures in visual data. Finally, we discuss problems related to rule evaluation in multi-object complex scenes and introduce some new techniques to solve them.

1 Introduction

To develop systems which can detect relatively complex patterns or objects in complex scenes requires efficient and robust techniques for describing patterns and searching for them in such data strucutures. Machine Learning (ML) provides methods for solving such problems.

The type of representation most frequently used in visual pattern/object recognition has been the *relational structure* (RS) where patterns are encoded as parts (graph vertices) and part relations (graph edges), both being described by a set of attributes or features. Such graph representations are limited in the sense that generalization in terms of, for example, new views or non-rigid transformations of objects are difficult to represent. Further, pattern recognition typically involves graph matching, with a computational complexity that exponentiates with the number of parts [1, 2] - although constraints are used to prune the search space, as has been explored by a number of observers (see, for example, [3, 4]). However, little attention has been paid to the *design of optimal search procedures for matching RS's* - particularly for the recognition of objects embedded in scenes. [3, 4]).

In contrast to the RS representation and associated constraint-based graph matching (tree search) methods, in attribute-indexed systems patterns and objects are encoded by rules of the form:

if { **attribute conditions exist** } then

0-7803-3280-6/96/$5.00 ©1996 IEEE

{class evidence weights }

where the rule condition is usually defined in terms of bounds on feature attribute values, and where rules instantiated by data activate weighted evidence for different pattern classes.

Although such systems allow generalizations from samples (in terms of attribute bounds), they only attain implicit learning of the RS, in so far as unary rules (rules related to part features) and binary rules (rules related to part relational features) are both activated to evidence patterns or objects: they are 'attribute-indexed' and not 'part-indexed'.

In the following sections we focus on the analysis of a new technique for the learning of structural relations, **Conditional Rule Generation** (CRG) which takes into account both attributes and part labels in the process of rule generation. It generates a tree of rules for classifying structural pattern descriptions that aims at "best" generalizations of the rule bounds with respect to rule length (the number of label-consistent parts and relations). The aim of this paper is to show how the technique can be used to solve problems involving the recognition of 2D patterns and 3D objects in complex visual scenes.

The Conditional Rule Generation Method. In CRG, rules are defined as clusters in conditional feature spaces which correspond to either unary(part) or binary(relation) feature attributes of the training data. The clusters are generated to satisfy two conditions: one, they should maximize the covering of samples from one class and, two, they should minimize the inclusion of samples from other classes. In our approach, such rules are generated through controlled decision tree expansion and cluster refinement as described below.

Cluster Tree Generation. Each pattern (a 2D sample pattern or a view of a 3D object) is composed of a number of parts (pattern components) where, in turn, each part $p_r, r = 1, ..., N$ is described by a set of unary features $\vec{u}(p_r)$, and pairs of parts (p_r, p_s) belonging to the same sample (but not necessarily all possible pairs) are described by a set of binary features $\vec{b}(p_r, p_s)$. Below, $S(p_r)$ denotes the sample (in 3D object recognition, a "view") a part p_r belongs to, $C(p_r)$ denotes the class (3D object recognition - object) $S(p_r)$ belongs to, and H_i refers to the information, or cluster entropy statistic:

$$H_i = -\sum_k q_{ik} \ln q_{ik} \qquad (1)$$

where q_{ik} defines the probability of elements of cluster i belonging to class k. We first construct the initial unary feature space for all parts over all samples and classes $U = \{\vec{u}(p_r), r = 1, .., N\}$ and partition this feature space into clusters U_i. In our approach, the initial clustering procedure is not critical. Clusters that are unique with respect to class membership (with entropy $H_i = 0$) provide a simple classification rule for some patterns (e.g. U_3 in Figure 1). However, each non-unique (unresolved) cluster U_i is further analyzed with respect to binary features by constructing the (conditional) binary feature space $UB_i = \{\vec{b}(p_r, p_s) \mid \vec{u}(p_r) \in U_i$ and $S(p_r) = S(p_s)\}$. This feature space is clustered with respect to binary features into clusters UB_{ij}. Again, clusters that are unique with respect to class membership provide classification rules for some objects (e.g. UB_{11} in Figure 1). Each non-unique cluster UB_{ij} is then analyzed with respect to unary features of the second part and the resulting feature space $UBU_{ij} = \{\vec{u}(p_s) \mid \vec{b}(p_r, p_s) \in UB_{ij}\}$ is clustered into clusters UBU_{ijk}. Again, unique clusters provide class classification rules for some objects (e.g. UBU_{121} in Figure 1), the other clusters have to be further analyzed, either by repeated conditional clustering involving additional parts at levels $UBUB$, $UBUBU$, etc. or through cluster refinement, as described below.

Each element of a cluster at some point in the cluster tree corresponds to a sequence $U_i - B_{ij} - U_j - B_{jk}...$ of unary and binary features associated with a non-cyclic sequence (path) of pattern parts - see [5] for more details.

In the current implementation of CRG, we have used a simple splitting-based clustering method to enable the generation of *disjoint* rules and to simplify the clustering procedure. Cluster trees are generated in a depth-first manner up to a maximum level of expansion. Clusters that remain unresolved at that level are split at a node and with respect to an attribute which minimizes the weighted entropy function:

$$H_P(T) = (n_1 H(P_1) + n_2 H(P_2))/(n_1 + n_2). \tag{2}$$

The cut point T_F that minimizes $H_P(T_F)$ is considered the best point for splitting cluster C along feature dimension F (see also [6]).

The rules generated by CRG are sufficient for classifying new pattern or pattern fragments, provided that they are sufficiently similar to patterns presented during training and provided that the patterns contain enough parts to instantiate rules. However, cluster trees and associated classification rules can also be used for partial rule instantiation - and so predict model projection and pose. A rule of length m (for example, a *UBUBU*-rule) is said to be partially instantiated by any shorter ($l < m$) sequence of unary and binary features (for example, a *UBU*-sequence). From the cluster tree shown in Figure 1, it is clear that a partial instantiation of rules (for example, to the *UB*-level) can lead to unique classification of certain pattern fragments (for example, those matched by the U_3 or UB_{11} rules, but it may also *reduce* classification uncertainty associated with other nodes in the cluster tree (for example, UB_{23}).

In summary, CRG has been specifically developed to enable the learning of patterns defined by (labeled) parts and their relations. The technique determines the type of inductive learning (attribute generalizations) that can be performed and the associated minimum length descriptors of shapes for recognition. Finally, since the method precompiles patterns as relational trees, the technique is ideally suited for the learning of patterns with variable complexity and their detection in scenes.

Applications to Scene Labeling. Of specific interest in this paper is the recognition of patterns embedded in complex scenes using the rules generated by CRG. For illustrative purposes we consider a 2D recognition problem though we have also applied the technique to 3D object recognition.

Here, training data consited of four classes of patterns with four training examples each (see Figure 2a). Each pattern is described by the unary features "length" and "orientation", and the binary features "distance of line centers" and "intersection angle". The line patterns are simplified versions of patterns found in geomagnetic data that are used to infer the presence of certain metals or minerals.

CRG was run with maximum rule length set to *maxlevel* = 5 (i.e. rules up to the form of *UBUBU*are being generated), and it produced 35 rules, 3 *U*-rules, 18 *UB*-rules, 2 *UBU*-rules, and 12 *UBUB*-rules.

At recognition time, a montage of patterns was presented (see Figure 2b), and the patterns were identified and classified as described below, producing the classification result shown in Figure 2c. Pattern identification and classification was achieved using the following rule evaluation steps:

1) Unary features are extracted for all scene parts (lines), and binary features are extracted for all adjacent scene parts, i.e. pairs whose center distance does not exceed a given limit.

2) Given the adjacency graph, all non-cyclic paths up to a certain length l are extracted, where $l \leq maxlevel$. These paths, termed *evidence chains* or, more simply, *chains*, constitute the basic units for pattern classification. A chain is denoted by $S =< p_i, p_j, ..., p_n >$ where each p_i denotes a pattern part. For some chains, all parts belong to a single learned pattern, but other chains are likely to cross the "boundary" between different patterns.

3) Each chain $S =< p_i, p_j, ..., p_n >$ is now classified using the classification rules produced by CRG. Depending on the unary and binary feature states, a chain may or may not instantiate one (or more) classification rules. In the former case, rule instantiation may be partial (with a non-unique evidence vector $\vec{E}(S)$), or complete (with $H[\vec{E}(S)] = 0$). As discussed above, the evidence vector for each rule instantiation is derived from the empirical class frequencies of the training examples.

4) The evidence vectors of all chains $< p_{i_1}, p_{j_1}, ..., p_n >$, $< p_{i_2}, p_{j_2}, ..., p_n >$, etc., terminating in p_n determine the classification of part p_n. Some of these evidence vectors may be mutually incompatible and others may be non-unique (through partial rule instantiation). Here, we have studied two ways of combining the evidence vectors, a winner-take-all solution and a relaxation labeling solution.

Implementation of the **winner-take-all** (WTA) solution is straightforward. The evidence vectors of all chains terminating in p_n are averaged to give $\vec{E}_{av}(p_n)$, and the most likely class label is enacted. However, the WTA solution does not take into account that, for a chain $S =< p_i, p_j, ..., p_n >$, the average evidence vectors $\vec{E}_{av}(p_i)$, $\vec{E}_{av}(p_j)$, ...,$\vec{E}_{av}(p_n)$ may be very different and possibly incompatible. If they are very different, it is plausible to assume that the chain S is "crossing" boundaries between different patterns/objects. In this case, the chain and its evidence vectors should be disregarded for the identification and classification of scene parts.

This is achieved in the **relaxation labeling** (RL) solution, where evidence vectors are weighted according to intra-chain compatibility. Specifically, the RL solution is given by

$$\vec{E}^{t+1}(p_i) = \Phi \left[\sum_{S=<p_i \cdots p_n>} \vec{E}^t(p_i) C(p_i, p_n) \right] \quad (3)$$

where $\vec{E}^t(p_i)$ corresponds to the evidence vector of p_i at iteration t, with $\vec{E}^0(p_i) = \vec{E}_{av}(p_i)$. $C(p_i, p_n)$ corresponds to the compatibility between parts p_i and p_n, and Φ is a non-linear transducer function defined by

$$\Phi(z) = \frac{1}{2} \left(1 + \frac{1 - exp[-(z - 0.5)/0.05]}{1 + exp[-(z - 0.5)/0.05]} \right). \quad (4)$$

Further, we have encoded the compatibility function in terms of the scalar product between the evidence vectors of parts p_i and p_n,

$$C(p_i, p_n) = \vec{E}(p_i) \cdot \vec{E}(p_n). \quad (5)$$

For identical evidence vectors $\vec{E}(p_i)$ and $\vec{E}(p_n)$, $C(p_i, p_n) = 1$, and for incompatible evidence vectors, for example $\vec{E}(p_i) = [1, 0, 0]$ and $\vec{E}(p_n) = [0, 1, 0]$, $C(p_i, p_n) = 0$.

Compatibility of evidence vectors is a weak constraint for updating the evidence vectors of each part and it may even have an adverse effect if the adjacency graph is complete. Much stronger constraints can be derived from, for example, the label-compatibilities between pattern parts, or from pose information in the case of 3D object recognition. The usefulness of such information is, however, pattern dependent and considered beyond the scope of the present paper. In any case, for the simple patterns shown in Figure 2, and the low connectivity of the adjacency graphs of the montages, the relaxation method outlined here proved to be sufficient to obtain perfect part labeling. The results obtained using this technique are shown in Figure 2c.

2 Discussion

CRG develops structural descriptions of patterns in the form of part-indexed decision trees which generalize over attribute bounds (see Figure 2). It can also be viewed as an automated technique for generating hash functions which incorporates relational hashing of different arities.

CRG shares with ID3 / C4.5 [7, 8], and related techniques, similar methods for the search and expansion of decision trees. However, these latter techniques were not designed to generate rules satisfying label compatibility between unary and binary predicates. CRG, on the other hand, is explicitly designed to develop rules for unique identification of classes with respect to their "structural" (i.e. linked unary and binary feature) representation.

Finally, CRG raises the question as to what really is a "structural description" of a pattern. CRG simply generates conditional rules that combine an attempt to generalize the pattern definitions in terms of feature bounds and to restrict the description lengths as much as possible. For complex and highly variable training patterns, CRG can generate a large number of rules which can be thought of as a set of *equivalent descriptions* of the pattern structure. It is possible to determine the more frequently occurring paths and associated feature bounds from the cluster tree, if the notion of "commonness" is deemed necessary for a structural description. However, this may not really be a meaningful definition of structure. Rather than producing a singular rule structure, a "structural description" is defined by a *set of rules* that CRG generates from a set of training patterns.

References

[1] R. E. Tarjan and A. E. Trojanowski. Finding a maximum independent set. *SIAM Journal of Computing*, 6:537–546, 1977.

[2] D. Ballard and C. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[3] W. E. L. Grimson. *Object Recognition by Computer*. MIT Press, Cambridge, MA, 1990.

[4] P. Flynn and A. K. Jain. 3D object recognition using invariant feature indexing of interpretation tables. *Computer Vision, Graphics, and Image Processing*, 55:119–129, 1992.

[5] W. F. Bischof and T. Caelli. Learning structural descriptions of patterns: A new technique for conditional clustering and rule generation. *Pattern Recognition*, 27:1231–1248, 1994.

[6] U. Fayyad and K. Irani. On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8:87–102, 1992.

[7] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[8] J. R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

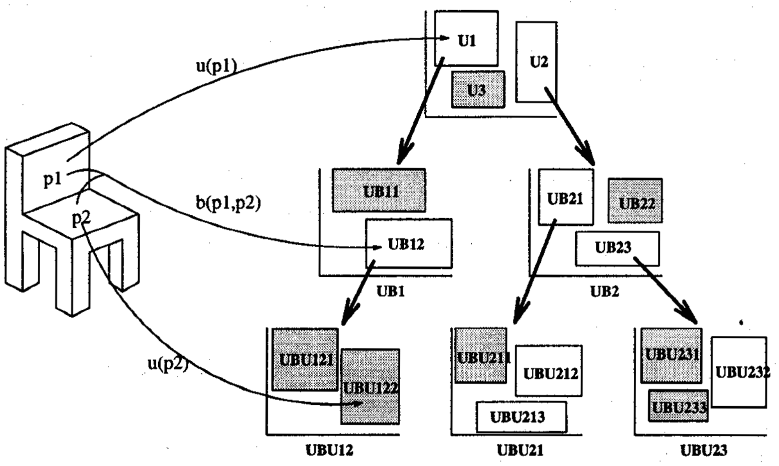

Figure 1: Conditional Rule Generation(CRG): The unresolved unary clusters ($U1$ and $U2$) - with element from more than one class - are expanded to the binary feature spaces $UB1$ and $UB2$, from where clustering and expansion continues until either all rules are resolved or the predetermined maximum rule length is reached, in which case rule splitting occurs. p_1, p_2 refer to image part labels while u, b refer to unary and binary attribute values, respectively.

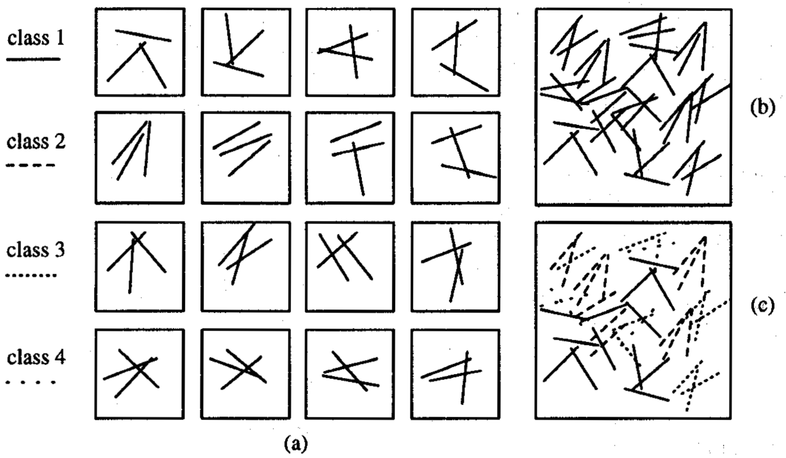

Figure 2: (a) Four classes of patterns with four training patterns (views) each. Lines are described by the unary features "line length" and "orientation"; pairs of lines by the binary distance between line centers and ntersection angle. (b) Montage of (slightly distorted) line triples. (c) Result of the pattern classification using the rules generated by CRG. Class labels for each line are shown on the right.