Learning How to Find Patterns or Objects in Complex Scenes

Walter F. Bischof[1] * and Terry Caelli[2]

[1] Department of Psychology, University of Alberta, Edmonton, Alberta T6G 2E9, Canada
[2] Department of Computer Science, Curtin University of Technology, Perth, WA 6001, Australia

Abstract. In this paper, we consider how machine learning can be used to help solve the problem of identifying objects or structures composed of parts as they occur in complex scenes. We first discuss an automatic conditional rule generation technique (CRG) that is designed to describe structures via part attributes and their relations. It does so by generating part-indexed decision trees where the branches define the types of pattern structures necessary to identify and to generalize from the different training examples. We then show how the resultant rules can be used for region labeling, and we examine grouping and constraint propagation techniques that are required for the identification of objects in complex scenes.

1 Introduction

Though the literature abounds with techniques for the recognition of *isolated* 2D patterns and 3D objects, the problem of efficiently detecting and recognizing such structures in complex scenes has not received as much attention. A number of authors have incorporated machine learning techniques to increase the robustness and efficiency to these methods, i.e. to improve their ability to generalize from training or known object data, and to improve their efficiency in searching scene data. For example, evidence-based methods have been used recently in 3D object recognition [4, 2] where object model views and parts are used to automatically generate rules (attribute bounds) which evidence different models. In such systems, generalizations to new views or to distortions are defined in terms of the rule attribute bounds and evidence weights. However, they typically have not encoded relational information and have not been adapted to recognition in complex scenes. This paper addresses both issues, the generation of rules encoding relational information and the application of rules to the interpretation of complex scenes.

* Supported by Grant OGP38521 from the Natural Sciences and Engineering Research Council of Canada.

2 Learning Structural Descriptions: CRG

Conditional Rule Generation (CRG) is a technique devised by the authors for learning structural descriptions of patterns as trees of hierarchically organized rules [1]. The rules are defined as clusters in conditional feature (attribute) spaces which correspond to either unary features of pattern parts or binary features of relation between parts. The clusters in a given attribute space are generated by splitting attributes (partitioning feature spaces) in such a way that each selected splitting operation creates new regions (defining rule bounds) which contribute better evidence for fewer classes. In our approach, such rules are generated conditionally through controlled decision tree expansion and a cluster refinement procedure.

More formally, each pattern sample (a 2D pattern or a view of a 3D object) is composed of a number of parts (pattern components). Each part $p_r, r = 1, \ldots, N$ is described by a set of unary features $\mathbf{u}(p_r)$, 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 $\mathbf{b}(p_r, p_s)$. Below, $S(p_r)$ denotes the sample to which a part p_r belongs to and H_i refers to the information, or cluster entropy statistic $H_i = -\sum_j q_{ij} \ln q_{ij}$ where q_{ij} defines the probability of elements of cluster i belonging to class j. We first construct the initial unary feature space for all parts over all samples and classes $U = \{\mathbf{u}(p_r), r = 1, .., N\}$ and partition this feature space into clusters U_i. Clusters that are unique with respect to class membership (with entropy $H_i = 0$) provide a simple classification rule for some patterns. Each non-unique (unresolved) cluster U_i is further analyzed with respect to binary features by constructing the (conditional) binary feature space $UB_i = \{\mathbf{b}(p_r, p_s) \mid \mathbf{u}(p_r) \in U_i \text{ and } S(p_r) = S(p_s)\}$. This feature space is partitioned 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. Each non-unique cluster UB_{ij} is then analyzed with respect to unary features of the second part and the resulting feature space $UBU_{ij} = \{\mathbf{u}(p_s) \mid \mathbf{b}(p_r, p_s) \in UB_{ij}\}$ is clustered into clusters UBU_{ijk}.

Uniqueness of pattern classification rules can be achieved either by repeated conditional clustering involving additional pattern parts or through cluster refinement. Refinement of a cluster C is achieved by finding the feature dimension F and the feature threshold T that minimizes the partition entropy $H_P(T)$:

$$H_P(T) = n_1 H(P_1) + n_2 H(P_2) \ . \tag{1}$$

where P_1 and P_2 denote the two partitions obtained using the threshold T. In addition, rather than splitting only leaf clusters, one can split the cluster tree at any level, and the cluster minimizing (1) is considered optimal for refining the cluster tree.

It should be noted that each feature space in the cluster tree corresponds to a standard decision tree [5]. CRG thus produces a tree of decision trees that is indexed by sequences of pattern parts, i.e. it is "part-indexed", whereas decision trees are purely "attribute-indexed". The dynamic expansion of cluster trees

constitutes a major advantage of CRG over decision trees: CRG can expand trees to the level optimized for a given data set whereas decision trees operate on fixed sets of features that have to be chosen a priori (see [1] for more details).

3 Scene Labeling and Recognition: SURE

Once CRG has generated rules from training samples, the problem of scene labeling reduces to that of instantiating rules in data, grouping labels and checking for their compatibilities. Indeed, the very purpose of the CRG method has been to "pre-compile" the types and number of parts, their attribute and relational attribute states that are necessary and sufficient for recognition. The problem remains, however, how to apply such rules to scenes composed of multiple objects. Of particular difficulty in such problems is the grouping of features or parts for rule evaluation. This problem has been studied by Grimson [3] and others in the context of model-based vision. Here, we discuss a solution in the context of a rule-based system that makes only weak and general assumptions about the structure of scene and objects. Our solution is based on the analysis of the relationships within (intra) and between (inter) instantiated rules. The solution method, termed SURE (Scene Understanding using Rule Evaluation), is based on the *sequential* evaluation of constraints described below.

Initial Rule Evaluation. The first stage in SURE involves direct activation of the CRG rules in a parallel, iterative deepening method. Starting from each scene part, all possible sequences of parts, termed *snakes*, are generated and classified using the CRG rules. Expansion of each snake $S = < s_1, s_2, \ldots, s_n >$ terminates if at least one of the following conditions occurs: 1) the part sequence $s_1, s_2, \ldots, s_n$ cannot be expanded without creating a cycle, 2) all CRG rules instantiated by S are completely resolved, or 3) the binary features $\mathbf{b}(s_n, s_{n+1})$ do not satisfy the features bounds of any CRG rule. If a snake S cannot be expanded, the evidence vectors of all rules instantiated by S are averaged to obtain the evidence vector $\mathbf{E}(S)$ of the snake S. Further, the set $\mathcal{S}_p$ of all snakes that start at p is used to obtain an initial evidence vector for part p:

$$\mathbf{E}(p) = \frac{1}{\#(\mathcal{S}_p)} \sum_{S \in \mathcal{S}_p} \mathbf{E}(S) \ . \tag{2}$$

where $\#(\mathcal{S})$ denotes the cardinality of the set $\mathcal{S}$. Classification of scene parts based on (2) has one major problem. Snakes that are contained completely within a single "object" are likely to be classified correctly, but snakes that "cross" two ore more objects are likely to be classified in an arbitrary way, and they therefore distort the classification in (2).

Snake Permutation Constraint. Every CRG rule encodes a set of model snakes $\{M_k = < m_{k1}, m_{k2}, \ldots, m_{kn} >, 1 \leq k \leq K\}$. When a snake $S = < s_1 s_2 \ldots s_n >$ instantiates such a rule each image part s_i indexes a set of model

parts $\mathcal{M}(s_i) = \{m_{ki}, 1 \leq k \leq K\}$. The snake permutation constraint is based on the assumption that rule instantiations are invariant to permutations, i.e. if two snakes are permutations of each other, for example $S_1 = < A, B, C >$ and $S_2 = < B, A, C >$, their parts must index the same set of model parts, independent of snakes and independent of instantiated rules.

Single Classification Constraint. The single classification constraint is based on the assumption that *at least one* snake among all snakes starting at a scene part does not cross an object boundary and that at least one instantiated rule indexes the correct model parts. Given this, if there is any scene part that initiates a single snake S_i and this snake instantiates a single classification rule then the model parts indexed by S_i can be used to constrain all snakes that touch S_i.

These two deterministic constraints are very powerful in terms of eliminating inconsistent (crossing) snakes. Their usefulness breaks down, however, for cases where the assumptions formulated earlier are not met for a given training and test data set.

Inter-snake Compatibility Analysis. The idea of the inter-snake compatibility analysis is as follows. The less compatible the evidence vector of a snake S_i is with the evidence vectors of all snakes that S_i touches, the more likely it is that S_i crosses an object boundary. In this case S_i is given a low weight in the computation of (2). More formally, let $S_i = < s_{i1}, s_{i2}, \ldots, s_{in_i} >$ and $S_j = < s_{j1} s_{j2} \ldots s_{jn_j} >$ be touching snakes, and let T_{ij} be the set of common parts, i.e. $T_{ij} = \{p \mid \exists k\ p = s_{ik} \text{ and } \exists l\ p = s_{jl}\}$ with $\#(T_{ij}) > 0$. The compatibility of S_i and S_j, $C(S_i, S_j)$ is defined as

$$C(S_i, S_j) = \frac{1}{\#(T_{ij})} \sum_{p \in T_{ij}} \frac{\#(\mathcal{M}(p|S_i) \cap \mathcal{M}(p|S_j))}{\#(\mathcal{M}(p|S_i) \cup \mathcal{M}(p|S_j))} \ . \tag{3}$$

The overall compatibility of a snake S_i is then defined with respect to the set $\mathcal{S}_T$ of snakes that touch S_i, i.e. $\mathcal{S}_T = \{S_j \mid \#(T_{ij}) > 0\}$:

$$w_{\text{inter}}(S_i) = \frac{1}{\#(\mathcal{S}_T)} \sum_{S \in \mathcal{S}_T} C(S_i, S) \ . \tag{4}$$

Using the inter-snake compatibility, the averaging of the evidence vectors in (2) changes to

$$\mathbf{E}(p) = \frac{\sum_{S \in \mathcal{S}_p} w_{\text{inter}}(S) \mathbf{E}(S)}{\sum_{S \in \mathcal{S}_p} w_{\text{inter}}(S)} \ , \tag{5}$$

where $\mathcal{S}_p$ is defined as in (2).

Intra-snake Compatibility Analysis. The last rule for detecting boundary-crossing snakes is based on the following idea. If a snake $S_i = < s_{i1}, s_{i2}, \ldots, s_{in} >$ does not cross boundaries of objects then the evidence vectors $\mathbf{E}(s_{i1})$, $\mathbf{E}(s_{i2})$, $\ldots$, $\mathbf{E}(s_{in})$ computed by (5) are likely to be similar, and dissimilarity of the evidence vectors suggests that S_i may be a "crossing" snake. Similarity of any pair of evidence vectors can be measured by their dot product, and similarity of all intra-snake evidence vectors is captured by the following measure

$$w_{\text{intra}}(S) = \frac{1}{n(n-1)} \sum_{k=1}^{n} \sum_{\substack{l \neq k \\ l=1}}^{n} \mathbf{E}(s_{ik}) \cdot \mathbf{E}(s_{il}) \ . \tag{6}$$

With the incorporation of the intra-snake compatibility analysis, the part evidence vectors are computed using the following iterative (relaxation) scheme:

$$\mathbf{E}^{(t+1)}(p) = \Phi \left[\frac{1}{Z} \sum_{S \in \mathcal{S}_p} w_{\text{inter}}(S) w_{\text{intra}}^{(t)}(S) \mathbf{E}(S) \right] \ , \tag{7}$$

where Z is a normalizing factor and Φ a logistic function. Iterative computation of (7) is required since recomputation of $\mathbf{E}(p)$ affects the intra-snake compatibility (6). As indicated above, the four rules presented in this Section are evaluated sequentially, and the final part classification is given by the iterative scheme (7).

4 An Example

Due to space limitations, we present a single example involving the recognition of 2D patterns in complex scenes. The line configurations are simplified versions of patterns found in geomagnetic images that are used to infer the presence of different precious metals. The training set consisted of four classes, corresponding, for example, to the presence of different types of metals, with four training patterns each, and each pattern consisted of three lines (see Fig. 1a). Patterns were described by the unary features "length" and "orientation", and the binary features "distance of line centers" and "intersection angle". CRG was run with maximum rule length set to the *UBUBU*-form, and it produced 35 rules, 3 *U*-rules, 18 *UB*-rules, 2 *UBU*-rules, and 12 *UBUB*-rules.

The SURE procedure was then run on the montage of patterns shown in Fig. 1b. Unary features were extracted for all scene parts (lines) and binary features were extracted for all neighboring scene parts, i.e. for pairs of lines whose center distance did not exceed a given limit. Results of the classification procedure are shown in Fig. 1c where 35 out of 42 scene parts are classified correctly.

5 Discussion

In the present paper, we have addressed two major issues. First, given that CRG represents structural descriptions in terms of sets of independent pattern snakes,

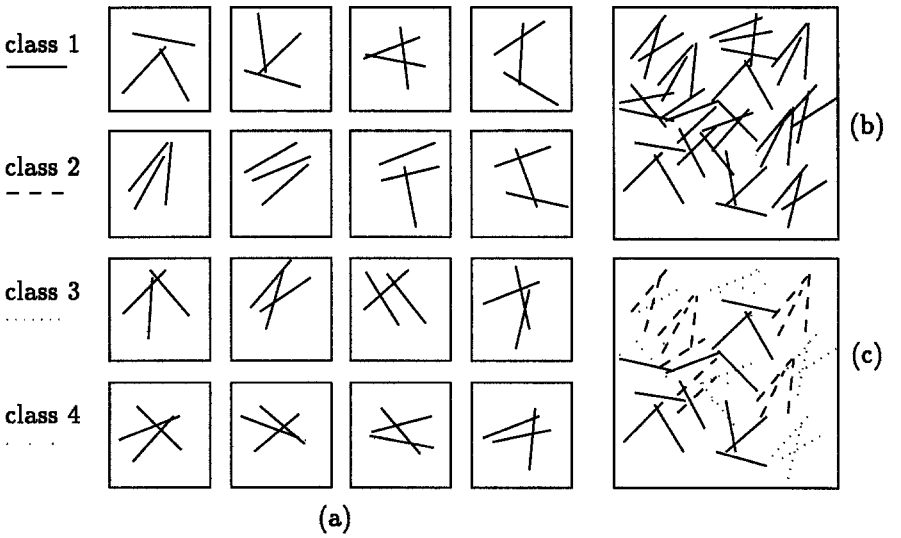

Fig. 1. (a) Four classes of patterns with four training patterns each. Each pattern is composed of three lines. (b) Montage of line triples. (c) Result of the pattern classification. Class labels for each line are shown on the left.

we have studied how interdependence of these snakes can be analyzed. Second, and more pertinent to this paper, we have studied how these interdependencies can be used to group pattern parts or image regions into groups that are likely to be associated with a single object.

References

1. Bischof, W. F., Caelli, T.: Learning structural descriptions of patterns: A new technique for conditional clustering and rule generation. Pattern Recognition **27** (1994) 689–698
2. Caelli, T., Dreier, A.: Variations on the evidenced-based object recognition theme. Pattern Recognition **27** (1994) 185–204
3. Grimson, W. E. L.: Object Recognition by Computer. Cambridge: MIT Press (1990)
4. Jain, A. K., Hoffman, D.: Evidence-based recognition of objects. IEEE Transactions on Pattern Analysis and Machine Intelligence **10** (1988) 783–802
5. Quinlan, J. R.: C4.5 Programs for Machine Learning. San Mateo: Morgan Kaufman (1993)