Learning Visual Recognition With Bayesian Networks

Justus H. Piater and Roderic A. Grupen

CMPSCI Technical Report 99-43

March 1999

Computer Science Department
Lederle Graduate Research Center
University of Massachusetts
Amherst, MA 01003-4601

{piater|grupen}@cs.umass.edu

Learning Visual Recognition With Bayesian Networks

Justus H. Piater and Roderic A. Grupen

E-mail: {piater|grupen}@cs.umass.edu

Abstract

Many realistic visual recognition tasks are "open" in the sense that the number and nature of the categories to be learned are not initially known, and there is no closed set of training images available to the system. One example are autonomous robotic agents that rely on visual sensors to guide their activity. We argue that open recognition tasks require incremental learning methods, and feature sets that are capable of expressing distinctions at any level of specificity or generality. We describe progress toward such a system that is based on an infinite combinatorial feature space. Feature primitives can be composed into increasingly complex and specific compound features. Distinctive features are learned incrementally, and are incorporated into a dynamically updated Bayesian network classifier. Experimental results illustrate the applicability and potential of our approach.

1. Introduction

During the past decade, considerable progress has been made in the area of machine recognition [10, 17, 9, 11]. Increasingly impressive recognition results are reported on large databases of various objects. While this success is truly remarkable, we observe that most work in this field shares certain characteristics: First, there exists a set of training images, which is complete at the outset and unchanged during the learning phase. In fact, many current recognition algorithms critically depend on the accessibility of the training set in its entirety. Second, most algorithms are based on descriptions of objects or classes in isolation, without regard to objects belonging to other classes, as opposed to descriptions of distinctions between objects. Thus, the properties of these descriptions largely predetermine the capabilities of the algorithm to generalize and to recognize minute distinctions. The assumption behind this design is that in terms of the description employed by the algorithm, all objects within a class are more similar to each other than to objects belonging to other classes.

Task domains that share these two characteristics we call *closed*. Are practically occurring recognition problems closed domains? We argue that many are not. For instance, a visually navigating mobile robot should be able to learn by itself what the distinctive landmarks are. If it is moved from one environment to another, one does not want to redesign the recognition algorithm – the same algorithm should be applicable in and adaptive to a variety of environments. In general, for robots that rely on visual and other sensory input to obtain state information about their environment to accomplish a task, the number and nature of distinct classes is usually not known a priori. As McCallum convincingly argued [8], it should be the *utility* or behavioral advantage to the agent that determines a sensory category, not some similarity metric in sensory space.

Thus, many realistic visual recognition problems constitute *open* task domains: To the agent, the number and nature of visual categories or object classes is not initially known. A given class may contain dissimilar objects, some of which may be very similar to – but distinguishable from – objects belonging to other classes. There is no fixed set of training images that perfectly describes the classes and is observable in its entirety. On the other hand, the environment may be cooperative in the sense that the learning agent has access to additional training views of given classes, e.g. by manipulating sample objects in front of the camera, turning the camera toward a previously seen landmark, or by means of an external teacher.

Most existing algorithms for visual recognition are not well suited for open task domains. While we cannot claim to have a full solution, this paper presents a framework for visual learning that constitutes a significant step toward this goal. It learns those features from an infinite feature space that prove useful for the task as it evolves over time. The feature space is introduced in the following section. Section 3 describes the proposed approach at a high level, and the following two sections present details of the recognition and feature discovery algorithms that are based on a Bayesian network. Experimental results are discussed in Section 6.

This work was supported in part by the National Science Foundation under grants CISE/CDA-9703217, IRI-9704530 and IRI-9503687, and by the Air Force Research Labs, IFTD (via DARPA) under grant F30602-97-2-0032.

2. Features

In order to learn distinctions at various levels of detail which are initially unknown, a very large feature space is required, along with a method of generating features from this space. To make the problem of finding useful features in an enormous feature space more tractable, we impose a partial order on this space that categorizes the features into various levels of structural complexity [1]. The underlying assumption is that structurally simple features are easier to discover and have less discriminative potential than complicated features, but are still useful for some aspects of the learning problem. Features are randomly sampled from the feature space, beginning at the lowest level of complexity. More sophisticated features are considered as required [1].

An obvious way to generate an infinite and partially ordered feature space is through combinatorics: *Primitive* features can be composed in various ways to yield *higher-order* features, which in turn can be composed. Any type of local image property can potentially serve as a primitive feature. In the context of an interactive vision system, this general framework may encompass three-dimensional or temporal cues in addition to conventional image properties.

Our system currently employs two types of primitive features [14]: (1) An *edgel* is given by the orientation of a step edge at a given point in the image. Edgels are extracted by convolving the image with two orthogonal oriented first-order Gaussian-derivative kernels. The orientation θ in the image is computed efficiently using the steerability property of these filters [4]. Intuitively, geometric combinations of edgels characterize aspects of shape. (2) A *texel* is a vector of responses of multiscale oriented Gaussian-derivative filters of various orders. At each scale and for each derivative d, a steerable basis consisting of $d + 1$ filter responses at specific orientations is computed [15]. Intuitively, a texel expresses local texture characteristics. Notably, both primitives can be steered to specific orientations. This property is used to achieve invariance with respect to image-plane rotation.

Two intuitively expressive types of feature compositions have so far been implemented: (1) *Geometric* relations are given by the relative angles and distances between the participating lower-order features (Figure 1). As long as these are rotation-invariant, so is their geometric composition. (2) *Co-presence* asserts the presence of the participating lower-order features without making any statement about their geometric or topological relationship.

Features of any type can be composed into a co-presence feature, while only primitive and geometric features can be composed into geometric features. Note that these two types of composition constitute two extremes along a continuum. One could conceivably define a composition that asserts relaxed geometric or topological relationships between its constituents.

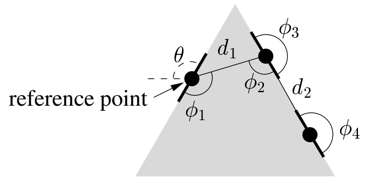

Figure 1. A geometric feature of order 3, composed of three primitives. The feature is defined by the angles ϕ and the distances d, and the orientation of this specific instance is denoted by θ. Each primitive is either an edgel or a texel.

Features are computed at various scales, generated by successively subsampling images by factor two. This achieves a certain degree of scale invariance. Moreover, many compositions of edgels are inherently tolerant to changes in scale. For example, the arrangement shown in Figure 1 applies equally to triangles of any size. Another desirable property of these features is that they do not rely on explicit contour extraction or segmentation. This avoids two difficult, open problems, and should provide robustness to various kinds of image degradation.

Our features constitute an interesting bridge between the two extremes of purely statistical, shape-less features [17, 9] on the one hand, and accurate 2-D or 3-D geometric models as used in the alignment methods [18] on the other hand. Our primitive features have about the same expressive power as Mel's corners and Gabor patches [9] and are similar to Cho and Dunn's "local properties" [2]. In contrast to these, our features can be composed into increasingly complex and specific descriptors of 2-D shape.

3. Paradigm

To avoid the practical difficulties involved in devising a truly open task, we instead employ a modified version of the conventional object recognition scenario. Note that we are not attempting to improve on state-of-the-art recognition systems for closed tasks. Our point is to demonstrate the feasibility and potential of an incremental learning system that relies on minimal prior information about the task.

Initially, the learning system knows nothing about the classes it will be required to learn. It has no features, but knows how to search in images for discriminative features from the feature space defined above. Its built-in classifier can be trained in a supervised fashion.

Training images are presented to the system one by one. If recognition fails, the system tries to discover distinctive features that facilitate recognition. Candidate features are evaluated on the basis of a temporary test set consisting of a small number of *evaluation images* that are accessible through interaction with the environment, as mentioned in the introduction. For the purposes of this paper, these evaluation images are randomly chosen from the training set.

This procedure is motivated by the following intuition. When humans learn to discriminate objects, they often evaluate feature hypotheses on the basis of a small number of current example views. As a result, the distinctive power of a hypothesized feature tends to be an optimistic estimate. This estimate is adjusted as more views are seen, which may even lead to the discovery that a feature is completely useless.

In order to make maximum use of the acquired information, our system retains all cases it ever encounters in an *instance list*. Each presentation of a training or evaluation image generates a new instance. An instance description contains the values of each feature that was observed, and a class designation of the image. In this way, all information obtained from an image is remembered for future use, without the need to store the images themselves.

4. A Bayesian network for recognition

Bayesian networks constitute a powerful and general framework for classification and inference. They possess a variety of attractive properties that are desirable for open-domain recognition problems. This section introduces a general Bayes net classifier model and shows how it is applied in our system.

In a Bayesian network, each node represents a random variable. The network structure specifies a set of conditional independence statements: The variable represented by a node is conditionally independent of its non-descendants in the graph, given the values of the variables represented by its parent nodes. In a typical classification scenario, the true class of an object is modeled as a discrete random variable, each state of which represents one possible class. The class of an object gives rise to observable features, which are represented by random variables whose distributions are conditioned on the class. Assuming that the features are conditionally independent given the class, the resulting Bayes net has the topology of a star, with arcs connecting the class node to each of the feature nodes. This is the well-known *naive Bayes classifier*. Given observed feature values, the class priors and conditional feature probabilities, the posterior class probabilities can be inferred by simple application of Bayes' Theorem.

If some features are not independent, corresponding arcs must be inserted between the appropriate feature nodes. For example, in Figure 2, Feature 3 may be a geometric composition with Feature 2, which is also in the feature set. Then, the presence of Feature 3 in an image implies the presence of Feature 2. Thus, in the Bayes net there is an arc from node 3 to node 2. An analogous argument holds for co-presence features, such as Feature 5 in Figure 2, which combines Features 3 and 4. To propagate evidence, more sophisticated mechanisms are needed than in the naive Bayes case. For the purposes of this paper, suffice it to say that after instantiation of some of the variables (nodes) with actually observed values, the net can be brought to *equilibrium* in which the probabilities and observations in the net are consistent. For more detail, the interested reader is referred to the literature on Bayesian networks [12, 6].

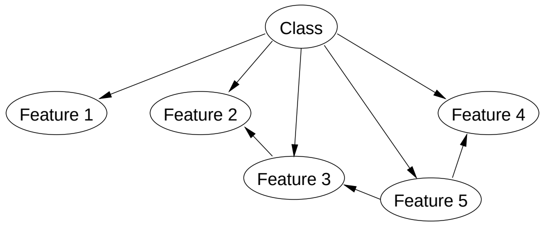

Figure 2. A Bayesian Network classifier with interdependent features.

Discretization of continuous variables Recall from Section 2 that the feature variables are continuous. However, most theory on belief propagation in Bayesian networks applies to discrete random variables only. One notable exception is the theory on conditional Gaussian distributions [7]. Unfortunately, current methods place restrictions on the joint probability model that limits their general applicability. Therefore, continuous distributions are usually discretized or *binned*. It is not obvious how to determine good cutpoints for binning. There are at least two objectives: First, it is desirable for many reasons to keep the number of bins small. Second, the bins should separate meaningful portions of the conditional distributions. Intuitively, bins are meaningful if different configurations of parent states give rise to different conditional probabilities of the individual bins.

To split a given continuous variable into meaningful bins, consider all pairs of conditions in turn, i.e. all pairs of parent state configurations. For each pair, generate a cutpoint that separates the two conditional distributions associated with the two conditions as well as possible. The cutpoint is chosen that maximizes the Kolmogorov-Smirnoff distance [19] between the distributions, which is the difference between the cumulative probabilities of this variable under the two conditions. This precisely serves our purpose of separating the instances of the two conditions as well as possible using a single cutpoint. Here, the cumulative distributions are estimated by counting the applicable instances in the instance list. To find candidate cutpoints, all instances are sorted by the respective feature value. Each midpoint between two adjacent values constitutes a candidate cutpoint.

For simplicity, the feature distributions were only conditioned on the class, not on other (derived) features. Thus, this procedure generated $\binom{n}{2}$ cutpoints for each feature variable, where n is the number of classes. To reduce the number of bins and increase robustness to small sample sizes, any bin that contained less than a fraction p of all applicable instances was removed by averaging the two enclosing split values. The parameter p controls the flexibility of the classifier to learn fine-grained distinctions using few features. If p is close to zero, the system quickly learns any training set, but tends to over-fit, and generalization is poor. If p is greater than 1/3, only even 2-bin splits are permitted, which is very restrictive. Most feature nodes will degenerate to single-bin nodes, which have no influence on the Bayes net. In practice, p should be kept at a moderately high value to discourage overfitting. Here, it was fixed at $p = 0.2$, which resulted in two or three bins for most features.

Probabilities and information The conditional probabilities in the Bayes net are easily estimated by counting instances in the instance list. The instance list may contain many unspecified values, since not every feature is evaluated for every image (see below). Some features may not have been evaluated at all for certain classes. In this case, the conditional probabilities are initialized uniformly, reflecting ignorance about how these classes affect a feature.

In a typical classification scenario, all feature variables are observed and their values entered into the corresponding nodes. The net is then brought to equilibrium, after which the posterior class distribution is available at the class node. Alternatively, one may observe the features one by one, compute the resulting class distribution, and stop the classification process on a given criterion, e.g. if the maximum posterior class probability exceeds a threshold. This can be desirable, for instance, if observing features is expensive, which is true for typical vision applications.

Clearly, the features should be observed in decreasing order of utility $U(f)$, conventionally defined as $U(f) = V(f)/S(f)$, where V is the value or merit of knowing the value of a feature, and S is the cost of observing this feature. For Bayes classifiers, a useful definition of the value V of a feature F is given by Shannon's mutual information (see e.g. [3]) between the class and feature variables C and F:

$$I(C,F) = \sum_c \sum_f P(c,f) \log \frac{P(c,f)}{P(c)P(f)}.$$

The summations range over all possible (discrete) values of the two random variables. Intuitively, $I(C,F)$ expresses the potential of an observation of F to reduce the uncertainty of C. In a Bayes net, the probabilities P are replaced by beliefs BEL, which are the corresponding (conditional) probabilities given all evidence currently entered into the network.

1. Compute $I(C,F)$ for all features F.
   If $I(C,F) = 0$ for all F, stop.
2. Observe the feature F that maximizes $I(C,F)$ and enter the value into the corresponding node.
3. Propagate beliefs to establish equilibrium; go to step 1.

Table 1. The procedure for recognizing an image.

Note that $BEL(c,f) = BEL(f|c)BEL(c)$. One can compute $BEL(f|c)$ by temporarily instantiating the class node C to value c, propagating beliefs through the network to establish equilibrium, and taking the resulting posterior probabilities at the feature nodes F as the $BEL(f|c)$ for each feature [12, 16]. The complete procedure for recognizing an image is summarized in Table 1.

5. Learning

During training, example images are presented to the system one by one. As the system classifies an image as outlined in Table 1, it notes the features that are queried, and stores their measured values along with the class designation in the instance list.

There are two reasons why recognition of an image can fail: (1) The image belongs to a class the system has never seen before, or (2) the posterior probability of the true class was lower than that of some other class(es). In the first case, a new state is added to the class node of the Bayes net. The conditional probability tables of all features are augmented accordingly, and are initialized to uniform probabilities for all conditions involving the new class. In either of the two cases, a sequence of steps is performed until correct recognition is achieved. These steps are now described in order.

Probability table update Using the instance list, all features are re-discretized, and the conditional probability tables are updated. This brings the classifier up-to-date with the current experience of the recognition system. As mentioned anecdotally in Section 3, this often results in a reduction of the conditional dependence of the distribution of a given feature variable on the class variable. If the discretization step results in a single bin, the feature and the associated node can be discarded.

Then, the updated classifier is run on the image again. If the image is still recognized incorrectly, the system notes the classes whose posterior probabilities exceed $1/|C|$, where $|C|$ is the number of classes. These are the *confused classes*, that are required for the next step.

Feature evaluation The system requests a fixed small number of evaluation images from each of the confused classes and measures the strength of each feature in each evaluation image. These data are added to the instance list, after which all features are re-discretized and the conditional probability tables updated. This step serves to evaluate existing features whose probabilities conditioned on the confused classes are still uniform or unreliable due to small sample sizes. Again, degenerate single-bin features can be eliminated. If recognition fails again, the system continues with the next and typically most expensive step.

Feature learning The system now attempts to find a new feature that facilitates correct recognition. This is done in the same way as in our earlier work [14] by sampling features from the image to be recognized. This sampling proceeds in stages: First, some number of new order-2 geometric edgel features and order-1 texels are generated by randomly choosing points from among the salient pixels in the image, and noting the two angles ϕ_i and the distance d, if applicable. To keep the features local, the distance between two sampled edgels is limited. Next, all existing geometric features (i.e. those previously learned and those just sampled) are augmented to higher-order geometric features. This is done by sampling a new primitive – edgel or texel – and noting the resulting ϕ_i and d with respect to the reference point of the parent feature. At the third stage, randomly chosen pairs of previously learned or newly sampled candidate features are composed into co-presence features. The latter two composition stages can be repeated several times.

After sampling a new feature, a corresponding node is added to the Bayes net. This involves adding an arc from the class node to the new feature node, and also arcs from the new node to any node representing an old feature of which the new feature is directly composed (see Section 4). Adding arcs involves updating the associated probability tables. This is done by measuring the strength of the new feature in each of the evaluation images, adding the results to the instance list, and re-initializing the conditional probability tables from this list.

Then, the recognition procedure is run on the image. If it is successful, the new feature remains in the Bayes net, and the feature learning procedure terminates successfully. Otherwise, the new feature node and the associated arcs are removed from the Bayes net, and a new feature is sampled. This procedure iterates until a usable feature has been found, or until a maximum number of new features has been sampled.

6. Experiments

To illustrate the operation of our system, we trained it on two simple supervised object recognition tasks, each containing example views of simple geometric objects. In one task, the database consisted of eight synthetic objects, each of which was rendered in high quality at 15 different views,

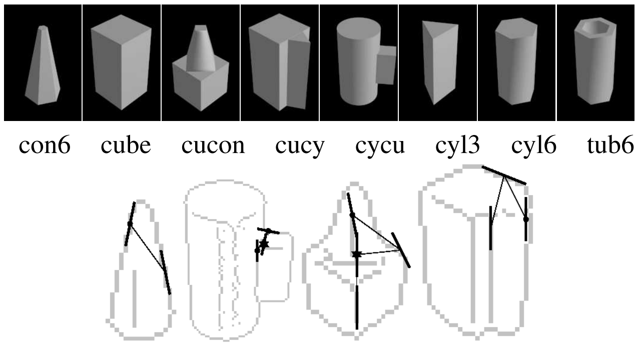

Figure 3. The synthetic-object task: Example views and examples of features learned.

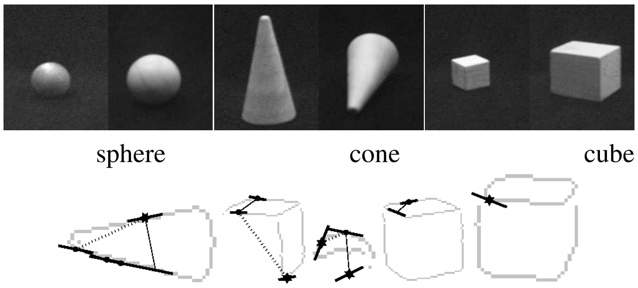

Figure 4. The real-object task: Example views and examples of features learned.

covering 40 horizontal and 20 vertical degrees of the viewing sphere (Figure 3). For the other task, low-quality images were taken of real geometric objects (Figure 4). There were 18 views of a sphere, 19 views of a cone in various positions, and 16 random views of a cube. The images of the class “sphere” included spheres of two different sizes, and the images of the class “cube” contained two cubes that differed in size.

The learning system was trained on each task as described above. The images of the training set were iteratively presented to the system in random order, until either the system had learned the training set perfectly, or until no feature was found during an entire cycle through the training set even though there were some misclassifications. New features were evaluated on the basis of 5 randomly chosen evaluation images per class. To learn a new feature, first up to 10 new features were sampled (texels and pairs of edgels). Then, the set of all pre-existing and new candidate features was augmented by one edgel or texel. Finally, up to the same number of co-presence features was generated. Tables 2 and 3 show the results obtained by 10-fold stratified cross-validation. In all test cases, the recognition procedure was allowed to use all available features.

	classification results:								sums:
	con6	cube	cucn	cucy	cycu	cyl3	cyl6	tub6	
con6	13					2			15
cube		12	1	2					15
cucon			15						15
cucy		2		11	1	1			15
cycu					13		2		15
cyl3	1					14			15
cyl6							13	2	15
tub6							1	14	15
sums:	14	14	16	13	14	17	16	16	120

Table 2. Confusion matrix summarizing the cross-validated test-set performance on the synthetic-object task. The overall proportion of correct recognitions was 0.88.

	classification results:			sums:
	sphere	cube	cone	
sphere	17		1	18
cube		13	3	16
cone			19	19
sums:	17	13	23	53

Table 3. Confusion matrix summarizing the cross-validated test-set performance on the real-object task. The overall proportion of correct recognitions was 0.93.

The synthetic objects were learned reasonably well (Table 2). Most of the confusions arose between objects that share most of their features (*cube-cucy*, *cycu-cyl6*, *cyl6-tub6*). To recognize an image, between 3 and 20 features were queried; typically around 8. Classification performance on the real objects was somewhat better. It took between 1 and 10 features to classify an image, usually 4 or fewer. On both data sets, the training set was always learned perfectly.

Due to the randomness of the algorithm and differing characteristics of the training images, the number of features learned and the number of iterations through the training set varied considerably between the individual folds of the cross-validation procedures, as detailed in Table 4.

It is interesting to follow the progression of the posterior class probabilities as the features are observed in sequence. Even in those cases that end with a misclassification, the correct class has the highest probability during most of the recognition procedure. Sometimes, however, the observation of a single additional feature – usually one with a very low information content $I(C, F)$ – changes the distribution drastically, and causes a misclassification. This was the case for most misclassifications. The reason is that the offending feature returned a value that was extremely

Fold:	1	2	3	4	5	6	7	8	9	10
Synthetic Objects:										
# iter.:	9	7	8	7	8	11	4	4	11	8
# feats.:	35	30	37	25	21	37	27	27	40	35
Accur.:	.81	.94	.94	.88	.88	.75	1.0	1.0	.63	.88
Real Objects:										
# iter.:	4	5	2	6	3	4	4	5	3	9
# feats:	6	8	3	10	6	6	6	11	6	11
Accur.:	1.0	.83	1.0	.83	1.0	.83	.80	1.0	1.0	1.0

Table 4. Characteristics of the synthetic- and real-object tasks, separated by folds of the cross-validation procedure. Shown are the numbers of iterations through the training set, the number of features learned, and the test-set accuracy achieved.

unlikely in conjunction with the evidence already entered in the net. This situation can be detected automatically in Bayesian networks. Whether detection of conflicting evidence can be exploited to increase classifier robustness is a topic of further investigation.

All of the Bayesian networks created by the synthetic-object task contained several dependencies between features. Is it worth modeling these dependencies, compared to the simplicity of a naive Bayes classifier? It is well known that ignoring the lack of independence among random variables can have a dramatic influence on the performance of Bayesian classifiers. We have conducted pilot experiments [13] that confirm this for our application. These experiments also indicate that simply discarding dependent features also degrades performance.

Figures 3 and 4 include some examples of features found during learning. The gray lines indicate the salient points used for sampling new features. Texels are marked by a small star, geometric relations by a solid line, and co-presence connections by a broken line.

7. Conclusions and future work

Adaptive, interactive agents – whether biological or artificial – benefit from learning those visual distinctions that turn out to be relevant for their tasks or behaviors. This learning process is inherently sequential, never complete, and unknown at the outset. We have presented a framework for progressive learning of such open visual recognition tasks. It is based on a combinatorial feature space of potentially infinite size. Our framework is general enough to incorporate any type of localized image property as feature primitives, and a variety of means for composing them into higher-order features. As long as the feature primitives are invariant to in-plane rotation, so are the compound features. Some degree of scale invariance is achieved by multiscale processing. Moreover, many geometric compositions of edgels are by themselves scale invariant.

A structural simple-to-complex partial ordering of the feature space facilitates relatively efficient feature search. While simple-to-complex feature sampling is not generally optimal with respect to any meaningful objective, this heuristic is intuitively pleasing in that it prefers simplicity over complexity. Assuming that most distinctions between object classes can be expressed in terms of low-order features, simple-to-complex sampling expends most effort in those areas of the feature space where success is most likely to occur.

Nevertheless, our current generate-and-test method for feature sampling is limited in that the search in feature space is essentially blind. It is only guided by the requirement that a new feature be present in the scene to be learned, and by simple locality heuristics. The identification of more focused search methods would lead to significant improvements in performance. Ideally, a system would learn heuristics or optimized systematic strategies for discovering useful features. This is an area of future research.

This work contributes a framework for recognition of images that is based on a dynamically updated Bayesian network. It replaces the simple classifier we adopted in our earlier work [14], and has several attractive properties. The network structure and the conditional probabilities that define the classifier are updated incrementally. We proposed a principled method to discretize continuous feature variables, which provides a parameter for tuning the capability of the classifier to fit highly variable data. Features can be queried sequentially as determined by an information-theoretic metric, while any hard decisions about classification can be deferred until a confidence threshold is attained.

Despite these advantages, Bayesian networks are not commonly used in computer vision applications (but see [16, 5]). Commonly encountered difficulties include the specification of the network structure and the conditional probability tables. For our application, we gave principled solutions to both problems based on known dependencies and an instance list. This list facilitates maximum reuse of acquired information. If the list is truncated by dropping the oldest instances, the learning system will smoothly adapt to a changing environment.

Acknowledgments

Our implementation employed the Hugin$^{\text{TM}}$ software for Bayesian networks. The database of synthetic objects was provided by Levente Tóth (levi@cis.plym.ac.uk) at http://www.cis.plym.ac.uk/cis/levi/UoP_CIS_3D_Archive/8obj_set.tar.

References

[1] Y. Amit, D. Geman, and K. Wilder. Joint induction of shape features and tree classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19:1300–1305, 1997.

[2] K. Cho and S. M. Dunn. Learning shape classes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(9):882–888, 1994.

[3] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.

[4] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(9):891–906, 1991.

[5] C. Jaynes, M. Marengoni, A. Hanson, and E. Riseman. Intelligent control for automatic model acquisition from aerial images. In *IASTED International Conference on Intelligent Systems and Control*, 1998.

[6] F. V. Jensen. *An introduction to Bayesian networks*. Springer, New York, May 1996.

[7] S. L. Lauritzen. Propagation of probabilities, means, and variances in mixed graphical models. *Journal of the American Statistical Association (Theory and Methods)*, 87(420):1098–1108, 1992.

[8] A. K. McCallum. *Reinforcement Learning with Selective Perception and Hidden State*. PhD thesis, Department of Computer Science, University of Rochester, Rochester, NY, 1995, revised 1996.

[9] B. W. Mel. Combining color, shape, and texture histogramming in a neurally-inspired approach to visual object recognition. *Neural Computation*, 9:777–804, 1997.

[10] H. Murase and S. K. Nayar. Visual learning and recognition of 3-D objects from appearance. *Int. J. Computer Vision*, 14:5–24, 1995.

[11] R. C. Nelson and A. Selinger. A cubist approach to object recognition. In *Int. Conf. on Computer Vision*, 1998.

[12] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[13] J. H. Piater. Visual object recognition using a Bayesian network. Stats 697J Class Project, 1998.

[14] J. H. Piater and R. A. Grupen. Toward learning visual discrimination strategies. In *Proc. Computer Vision and Pattern Recognition (CVPR '99)*, Ft. Collins, CO, June 1999. IEEE Computer Society.

[15] R. P. N. Rao and D. H. Ballard. An active vision architecture based on iconic representations. *Artificial Intelligence*, 78:461–505, 1995.

[16] R. D. Rimey and C. M. Brown. Task-oriented vision with multiple Bayes nets. In A. Blake and A. Yuille, editors, *Active Vision*, chapter 14, pages 217–236. MIT Press, 1992.

[17] B. Schiele and J. L. Crowley. Object recognition using multidimensional receptive field histograms. In *Fourth Europ. Conf. on Computer Vision*, Cambridge, UK, Apr. 1996.

[18] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(10):992–1006, 1991.

[19] P. E. Utgoff and J. A. Clouse. A Kolmogorov-Smirnoff metric for decision tree induction. Computer Science Technical Report 96-3, University of Massachusetts, Amherst, 1996.