Learning to Recognize Objects

Dan Roth Ming-Hsuan Yang Narendra Ahuja

Department of Computer Science and the Beckman Institute
University of Illinois at Urbana-Champaign
Urbana, IL 61801
danr@cs.uiuc.edu mhyang@vison.ai.uiuc.edu ahuja@vision.ai.uiuc.edu

Abstract

A learning account for the problem of object recognition is developed within the PAC (Probably Approximately Correct) model of learnability. The proposed approach makes no assumptions on the distribution of the observed objects, but quantifies success relative to its past experience. Most importantly, the success of learning an object representation is naturally tied to the ability to represent it as a function of some intermediate representations extracted from the image.

We evaluate this approach in a large scale experimental study in which the SNoW learning architecture is used to learn representations for the 100 objects in the Columbia Object Image Database (COIL-100). The SNoW-based method is shown to outperform other methods in terms of recognition rates; its performance degrades gracefully when the training data contains fewer views and in the presence of occlusion noise.

1 Introduction

The role of learning in computer vision research has become increasingly more significant in recent years. Statistical learning theory has had an influence on many applications ranging from classification and object recognition, grouping and segmentation, illumination modeling, scene reconstruction and others. The rising role of learning methods, made possible by significant improvements in computing power and storage, is largely motivated by the realization that explicit modeling of complex phenomena in a messy world cannot be done without a significant role of learning, both for model and knowledge acquisition, and to support generalization and avoid brittleness. Nevertheless, many statistical and probabilistic learning models require making explicit assumptions, e.g., on the distribution that governs the occurrences of instances in the world. For many visual inference problems such as recognition, categorization and detection, making these assumptions seems unrealistic.

This work develops a distribution free learning theory account to an archetypical visual recognition problem: object recognition. The problem is viewed as that of learning a representation of an object that, given a new image, is used to recognize the target object in it. The learning account is developed within the PAC (Probably Approximately Correct) model of learnability [16]. This framework allows us to (1) quantify success relative to the distribution of the observed objects, without making assumptions on the distribution. (That is, learnability guarantees that objects sampled from the same distribution as the one that governed the experience of the learner will be recognized correctly.) (2) study the theoretical limits of what can be learned from images in terms of the expressivity of the intermediate representation used by the learning process and (3) develop practical algorithmic solutions to the problem and exhibit their superiority over other methods.

Earlier works have discussed the possibility of identifying the theoretical limits of what can be learned from images [14] and found that learning in terms of the raw representation of the images is computationally intractable. Attempts to explain this focused the dependence of learnability on the representation of the object [4] but failed to provide a satisfactory explanation for it, or a practical solution. The approach developed here builds on suggestions made in [9] and relies heavily on the development of a feature efficient learning approach [10, 3]. That is, a learning process capable of learning in domains in which there is a very large number of potential features but any concept of interest actually depends on a fairly small number of those. At the heart of the learnability approach are two assumptions that we abstract as follows:

Representation: There exists a (possibly infinite) collection $\mathcal{M}$ of "explanations" such that an object can be represented as a simple function of polynomially many elements in $\mathcal{M}$.

Procedural: There exists a process that, given an image that is a positive example of the target object O generates, in polynomial time, "explanations" in $\mathcal{M}$ that are present in the image and such that, with high probability, at least one of them is in the representation of O.

Under these assumptions we prove that there exists an efficient algorithm that can learn a good representation of the object in the sense that, with high probability, it would make correct predictions on future images that contain (or do not contain) the object. Furthermore, we show that under these conditions, the learned representations are robust under realistic noise conditions. A significant non-assumption of our approach is that it has no prior knowledge on the distribution of images nor it is trying to estimate it. Section 2 describes this framework in details.

The framework developed here is very general. The *explanations* alluded to above can represent a variety of computational processes and information sources that operate on the image. They can depend on local properties of the image, the relative positions of primitives in the image, and even external information sources or context variables. Thus, the theoretical support given here applies also to an intermediate learning stage in a hierarchical process. The main assumptions of the framework are discussed in Sec. 3.

For this framework to contribute to a practical solution, and given our assumptions, there needs to be a computational approach that is able to learn in the presence of a large number of potential "explanations". The evaluation we provide for this framework relies on the SNoW learning architecture [3] that is used here in a large scale object recognition experiment. The SNoW system (available publicly at `http://L2R.cs.uiuc.edu/~cogcomp.html`) is described in Sec. 4, and the experimental study in Sec. 5.

2 Learning Framework

We study learnability within the standard PAC model [16] and the mistake-bound model [10]. In the learning scenario, we are given a concept class $\mathcal{C}$, a class of $\{0,1\}$-valued function over the instance space X with an associated complexity parameter n, and there is some unknown *target concept* $f_T \in \mathcal{C}$ that we are trying to learn. In the *mistake-bound* model, at each learning stage, an example $x \in X$ is presented; the learning algorithm is asked to predict $f_T(x)$ and is then told whether the prediction was correct. Each time the learning algorithm makes an incorrect prediction, we charge it one *mistake*. We say that $\mathcal{C}$ is *mistake-bound learnable* if there exists a polynomial-time prediction algorithm $\mathcal{A}$ (possibly randomized) that for all $f_T \in \mathcal{C}$ and any sequence of examples is guaranteed to make at most polynomially many (in n) mistakes. We say that $\mathcal{C}$ is *expected mistake-bound learnable* if there exists $\mathcal{A}$, as above, such that the expected number of mistakes it makes for all $f_T \in \mathcal{C}$ and *any* sequence of examples is at most polynomially many (in n). Note that the expectation is taken over the random choices made by $\mathcal{A}$; there is no probability distribution associated with the sequences. In learning an unknown target function $f_T \in \mathcal{C}$ in the PAC model, we assume that there is a fixed but arbitrary and *unknown* distribution D over the instance space X. The learning algorithm sees examples drawn independently according to D together with their labeling (positive/negative). Then it is required to predict the value of f_T on another example drawn according to D. Denote by $h(x)$ the prediction of the algorithm on the example $x \in X$. The error of the algorithm with respect to f_T and D is measured by $error(h) = Pr_{x \in D}\{f_T(x) \neq h(x)\}$.

We say that $\mathcal{C}$ is *PAC-learnable* if there exists a polynomial-time learning algorithm $\mathcal{A}$ and a polynomial $p(\cdot,\cdot,\cdot)$ such that for all $n \geq 1$, all target concepts $f_T \in \mathcal{C}$, all distribution D over X, and all $\epsilon > 0$ and $0 < \delta \leq 1$, such that if the algorithm $\mathcal{A}$ is given $p(n, 1/\epsilon, 1/\delta)$ examples, then with probability at least $1-\delta$, $\mathcal{A}$'s hypothesis, h, is such that $error(h) \leq \epsilon$. It can be shown that if a concept class $\mathcal{C}$ is learnable in the expected mistake-bound model (and thus in the mistake bound model) then it is PAC-learnable [6]. In practice, learning is evaluated on a training set. The hope that a classifier learned from a training set will perform well on previously unseen examples is based on the basic theorem of learning theory [16, 17] which, stated informally, guarantees that if the training data and the test data are sampled from the same distribution, good performance on large enough[1] training sample guarantees good performance on the test data (i.e., good "true" error).

2.1 Learning Scenario

Let $\mathcal{I}$ be an input space of images. Our goal is to learn a definition `apple`:$\mathcal{I} \rightarrow \{0,1\}$ that, when evaluated on a given image, outputs 1 when there is an apple in the image, and 0 otherwise. It is clear, though, that this target function is very complex in terms of the input space and, in particular, may depend on relational information and even quantified predicates. Many years of research in learning theory, however, have shown that efficient learnability of complex function is not feasible. In the learning scenario described here, therefore, learning will not take effect directly in terms of the raw input. Rather, we will learn the target definitions in terms of an intermediate representation that will be generated from the input image. This

[1] In this sense, the evaluation on a small training set done here is not as optimal as, say, a face detection study [18]. Note, however, that the theory implies that learning scales well with the size of the training data, as we show in Sec. 5.

will allow us to quantify learnability in terms of the expressivity of the intermediate representation as well as the function learned on top of it and, in particular, it would make explicit the requirements from an intermediate representation so that learning is possible.

Let $\mathcal{I}$ be the *instance space* (e.g., the space of all images). A *relation*[2] is a function $\chi : \mathcal{I} \to \{0, 1\}$. χ can be viewed as an indicator function over $\mathcal{I}$, defining the subset of those elements which are mapped to 1 by χ. A relation χ is *active* in $I \in \mathcal{I}$ if $\chi(I) = 1$.

Definition 2.1 *Let $\mathcal{X}$ be an enumerable collection of relations on $\mathcal{I}$. A relation-generation function (RGF) is a mapping $G : \mathcal{I} \to 2^{\mathcal{X}}$ that maps $I \in \mathcal{I}$ to a set of all elements in $\mathcal{X}$ that satisfy $\chi(I) = 1$. If there is no $\chi \in \mathcal{X}$ for which $\chi(I) = 1$, $G(I) = \phi$.*

RGFs can be thought as a way to define "kinds" of relations, or to parameterize over a large space of relations. Only when presented with an instance $I \in \mathcal{I}$, a concrete relation(s) is generated. For example, G may be the RGF that corresponds to all vertical edges of size 3 in an image. Given an image, a list of all these edges that are present in the image is produced.

We now present a *mistake-bound* algorithm for a class of functions that can be represented as DNF formulae over the space $\mathcal{X}$ of all relations. As indicated, this implies a PAC learning algorithm, but the proof is for the mistake-bound case is simpler. In Sec. 5 we will learn a more general function – a linear threshold function over conjunctions of relations in $\mathcal{X}$; the theoretical results can be expended to this case.

Definition 2.2 *Let $\mathcal{X}$ be a set of relations that can be generated by a set of RGFs. Let $\mathcal{M}$ be a collection of monomials (conjunctions) over the elements of $\mathcal{X}$, and $p(n)$, $q(n)$ and $g(n)$ be polynomials. Let $\mathcal{C}_{\mathcal{M}}$ be the class of all functions which are disjunctions of at most $p(n)$ monomials in $\mathcal{M}$. Following [9], we call $\mathcal{C}_{\mathcal{M}}$* polynomially explainable *if there exists an efficient (polynomial-time) algorithm $\mathcal{B}$ such that for every function $f \in \mathcal{C}_{\mathcal{M}}$, and every positive example of f as input, $\mathcal{B}$ outputs at most $q(n)$ monomials (not necessarily all of them are in $\mathcal{M}$) such that with probability at least $1/g(n)$ at least one of them appears in f (the probability is taken over the coin-flips of the (possibly probabilistic) algorithm $\mathcal{B}$).*

We note that, in principle, it is possible to abstract the generation of the conjunctions into the RGFs (Def. 2.1). However, we would like to emphasize the generation of conjunction over simple relations and the possibility of learning on top of it, given arguments in the literature of its effectiveness and potential biological plausibility [5, 15].

We emphasize that f itself is *not* given to the algorithm $\mathcal{B}$. Also note that a function f in the class $\mathcal{C}_{\mathcal{M}}$ may have several equivalent representations over $\mathcal{M}$.

Theorem 2.1 *If $\mathcal{C}_{\mathcal{M}}$ is polynomially explainable then $\mathcal{C}_{\mathcal{M}}$ is expected mistake-bound learnable. Furthermore, if $\mathcal{C}_{\mathcal{M}}$ is polynomially explainable by an algorithm $\mathcal{B}$ that always outputs at least one term of f (i.e., $g(n) \equiv 1$) then $\mathcal{C}_{\mathcal{M}}$ is mistake-bound learnable.*

Proof: [sketch] The algorithm is similar to an algorithm presented in [1] which learns a disjunction of terms in the infinite attribute model. The algorithm maintains an hypothesis h which is a disjunction of monomials. Initially h contains no monomials (i.e., $h \equiv FALSE$). Upon receiving an example e, the algorithm predicts $h(e)$; if the prediction is correct, h is not updated. Otherwise, upon a mistaken prediction, it proceeds as follows:

- If e is positive: execute $\mathcal{B}$ (the algorithm guaranteed by the assumption that $\mathcal{C}_{\mathcal{M}}$ is polynomially explainable) on the example e and add the monomials it outputs to h.
- If e is negative: remove from the hypothesis h all the monomials that are satisfied by e (there must be at least one).

The analysis of the algorithm is straight forward for the case $g(n) \equiv 1$, and more subtle in general. ■

The algorithm used in practice, in SNoW, is conceptually similar. The main difference is that the hypothesis h is a linear threshold function over elements in $\mathcal{M}$, and rather than dropping elements from it, their weight is updated. The details of this process (Sec. 4) are crucial for our approach to be computationally feasible for large scale domains and for robustness.

2.2 Robustness

Any realistic learning framework needs to support different kinds of noise in its input. Several kinds of noise have been studied in the literature in the context of PAC learning, and algorithms of the type we consider here have been shown to be robust to them. The most studied type of noise is that of classification noise [7] in which the examples are assumed to be given to the learning algorithm with labels that are flipped with some probability, smaller than 1/2. Learning in our framework can be shown to be robust

[2] In the machine learning literature a relation is sometimes called a feature.

to this kind of noise, as well as to a more realistic case of *attribute noise*, in which the the description of the input itself is corrupted to a certain degree. We believe that this is the type of noise that is more relevant in the current case. First, since learning is done in terms of the output of the RGFs, which may introduce some noise. Second, since attribute noise is related to *occlusion* noise that is important in object recognition. Specifically, attribute noise can be used to model the type of noise that usually occurs when other objects appear in the image, behind or in front of the target object. This is formalized using the notion of *domination*: Let f_1, f_2 be two concepts. We say that f_1 is *k-dominated* by f_2 if each f_1 example can be obtained from an f_2 example by flipping at most k of the active relations. In this case, f_2 *k-dominates* f_1. The labels of the examples, however, are generated according to the original concept, before the noise is introduced.

Theorem 2.2 *If a class $\mathcal{C}_{\mathcal{M}}$ is learnable by virtue of being polynomially explainable then it is learnable even if examples of the target class are cluttered by a k-domination attribute noise, for any constant k.*

Proof: [Sketch] The proof is an extension of the arguments in [11] regarding robustness to attribute noise, to the case of the infinite attribute model. ■

3 From Theory to Practice

Several issues need to be addressed in order to exhibit the practicality of our learning framework. The first is the availability of a variety of RGFs, capable of extracting from data evidence for the existence of primitive visual patterns under different conditions. In this work we illustrate the approach by utilizing simple edge detectors. The second issue is the composition of complex relations from primitive ones. This is crucial since it allows the representation of complex functions in terms of the relations by learning simple functional descriptions over their compositions. A language that supports composition of restricted families of conjunctions and can encode structural relations in images (e.g., above, to the left of...) is discussed in a companion paper. This work uses only general conjunctions and restricts only their size.

Finally, the issue of learnability which is the focus of this work. In learning situations in vision, the number of potential relations (features) that may affect each decision is very large but, typically, only a small number of them is actually relevant to a decision. Beyond correctness, a realistic learning approach needs therefore to be relation-efficient [10] in that its learning complexity (number of examples required for convergence) depends on the number of relevant relations and not the global number of relations in the domain. This can be phrases as the dependence of generalization quality on the number of examples observed. Also, since only a few of the many potential relations are active in any instance, the complexity of evaluating the learning hypothesis on an instance should depend on the number of active relations in the input rather than the total number in the domain. And, a learning approach should allow variable input size, since learning is in terms of relations that are generated from the image in a data-driven way, making it impossible, or impractical, for a learning approach to write explicitly, in advance, all possible "relations".

Given that, the learning approach used in this work is the one developed within the SNoW learning architecture [13, 3]. SNoW is specifically tailored for learning in domains in which the potential number of features taking part in decisions is very large, but may be unknown a priori, as in the infinite attribute learning model [1]. Specifically, as input, the algorithm receives labeled instances $< (x, l) >$, where an instance $x \in \{0,1\}^{\infty}$ is presented as a list of all the *active* relations in it and the label is a member of a discrete set of values (e.g., objects identifiers). Given a domain instance (an image) a set of pre-existing RGFs are evaluated on it and generate a collection of relations that are active in this image; these, in turn, may be composed to generate the elements of $\mathcal{M}$. A list of active elements in $\mathcal{M}$ is presented to the learning procedure and learning is done at this level.

4 The SNoW Architecture

The SNoW (Sparse Network of Winnows[3]) learning architecture is a sparse network of linear units over a common pre-defined or incrementally learned feature space. Nodes in the input layer of the network represent simple relations over the input instance and are being used as the input features. A linear unit is used for each *target node* and represents a relation or concept of interest over the input; in the current application, target nodes represent a definition of an object in terms of the relations (features) extracted from the 2D image input. An input instance is mapped into a set of features which are active in it; this representation is presented to the input layer of SNoW and propagates to the target nodes. Target nodes are linked via weighted edges to (some of) the input features.

Let $\mathcal{A}_t = \{i_1, \ldots, i_m\}$ be the set of features that are active in an example and are linked to the target node

[3] To winnow: to separate chaff from grain.

t. Then the linear unit corresponding to t is *active* iff

$$\sum_{i \in \mathcal{A}_t} w_i^t > \theta_t,$$

where w_i^t is the weight on the edge connecting the ith feature to the target node t, and θ_t is t's threshold.

Each SNoW *unit* includes a collection of subnetworks, one for each target relations but all using the same feature space. Here, we may have one unit with target subnetworks for all the target objects or we may define units to each have two competing target objects. A given example is treated autonomously by each target subnetwork; an example labeled t may be treated as a positive example by the subnetwork for t and as a negative example by the rest of the target nodes.

The learning policy is on-line and mistake-driven; several update rules can be used within SNoW. The most successful and the only one used in this work, is a variant of Littlestone's Winnow update rule [10], a multiplicative update rule tailored to the situation in which the set of input features is not known a priori, as in [1]. This mechanism is implemented via the sparse architecture of SNoW. That is, (1) input features are allocated in a data driven way – an input node for the relation i is allocated only if i was active in any input instance and (2) a link (i.e., a non-zero weight) exists between a target node t and i if and only if i was active in an example labeled t. One of the important properties of the sparse architecture is that the complexity of processing an example depends only on the number of features active in it, n_a, and is independent of the total number of features, n_t, observed over the life time of the system. This is important in domains in which the total number of features is very large, but only a small number of them is active in each example.

The Winnow update rule has, in addition to the threshold θ_t at the target t, two update parameters: a *promotion* parameter $\alpha > 1$ and a *demotion* parameter $0 < \beta < 1$. These are being used to update the current representation of the target t (the set of weights w_i^t) only when a mistake in prediction is made. Let $\mathcal{A}_t = \{i_1, \ldots, i_m\}$ be the set of active features that are linked to the target node t. If the algorithm predicts 0 (that is, $\sum_{i \in \mathcal{A}_t} w_i^t \leq \theta_t$) and the received label is 1, the active weights in the current example are *promoted* in a multiplicative fashion:

$$\forall i \in \mathcal{A}_t, w_i^t \leftarrow \alpha \cdot w_i^t.$$

If the algorithm predicts 1 ($\sum_{i \in \mathcal{A}_t} w_i^t > \theta_t$) and the received label is 0, the active weights in the current example are *demoted*:

$$\forall i \in \mathcal{A}_t, w_i^t \leftarrow \beta \cdot w_i^t.$$

All other weights are unchanged.

The key feature of the Winnow update rule is that the number of examples required to learn a linear function grows linearly with the number n_r of *relevant* features and only logarithmically with the total number of features. This is crucial in domains in which the number of potential features is vast, but a relatively small number of them is relevant. Moreover, in the sparse model, the generalization actually scales with a number of examples that scales with $O(n_r \log n_a)$. Winnow is known to learn efficiently (as a function of the margin in the data) any linear threshold function and to be robust in the presence of various kinds of noise and in cases where no linear-threshold function can make perfect classifications, while still maintaining its abovementioned dependence on the number of total and relevant attributes [11, 8].

Once target subnetworks have been learned and the network is being evaluated, a winner-take-all mechanism is employed to select the dominant active target node in the SNoW unit.

5 Experimental Evaluation

We use the Columbia Object Image Library (COIL-100) database in all the experiments below. COIL is available at http://www.cs.columbia.edu/CAVE. The COIL-100 dataset consists of color images of 100 objects where the images of the objects that were taken at pose intervals of 5°, i.e., 72 poses per object. The images were also normalized such that the larger of the two object dimensions (height and width) fits the image size of 128×128 pixels. Figure 1 shows the images of the 100 objects taken in frontal view, i.e., zero pose angle. The 32 highlighted objects in Figure 1 are considered more difficult to recognize in [12]; we use all 100 objects including these in our experiments. Each color image is converted to a gray-scale image of 32×32 pixels for our experiments.

5.1 Results Using Conjunction of Edges

Edge information contains significant visual cues for human perception and has the potential to provide more information than the previous representation and guarantee robustness. Edge-based representations can be used, for example, to obtain a hierarchical description of an object. While perceptual grouping has been applied successfully to many vision problems including object and face recognition, the grouping procedure is usually somewhat arbitrary. This work can this be viewed as a systematic method to learn representation of objects based on conjunctions of edges. For each image, a Canny edge detector [2] is first applied to extract edges. Let $I(x, y)$ represent the (x, y) pixel in an image I. Let $E(x, y)$

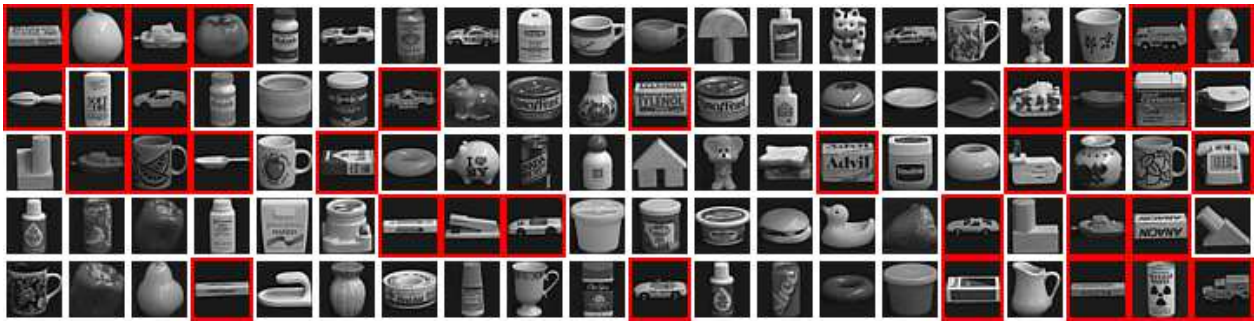

Figure 1: Columbia Object Image Library (COIL-100) consists of 100 objects of varying poses (5° apart). The objects are shown in row order where the highlighted ones are those considered more difficult to recognize in [12].

be the Canny edge map in which $E(x, y) = 1$ indicates the existence of an edge at $I(x, y)$. To prune extraneous small edge fragments and reduce the computation complexity we keep only edges with length above some threshold (e.g., 3 pixels). $\hat{E}$ is the resulting edge map after pruning. That is, the pixel $I(x, y)$ is considered to contain significant perceptual information to describe the object, when $\hat{E}(x, y) = 1$; otherwise, $\hat{E}(x, y) = 0$. For consistency we index an edge using its top left pixel. For each pixel we maintain up to two possible edges in the resulting $\hat{E}$ map, a vertical one and a horizontal one, denoted by

$$e =< (x, y), d >, d \in \{v, h\}.$$

Features are generated to represent conjunctions of size two of these edges. That is, features are elements of the cross product

$$\hat{E} \times \hat{E} = \{(e, e')|e \neq e'\}.$$

This representation thus constitutes a hierarchical representation of an object which encodes the spatial relationships of the edges in an object. Figure 2 illustrates the benefit of this encoding in object recognition. It shows two objects with very similar appearance for which the edge maps (where the minimum length is 3) are different. This difference grows when conjunctions of edges are used. Finally, we note that the number of potential features when using this representation is very large, but very few of them are active. For each 32×32 edge map, we first extract the horizontal and vertical edges whose length is at least 3 pixels. Therefore, we have two 32×32 edge maps in which the value of a pixel is 1 if it is on an edge. A vector for an object consists of the raster scans of these two maps, in which conjunctions of any two elements are computed. Thus, a feature vector of each object consists, in principle, of $\binom{2048}{2} = 2096128$ conjunctions, of which only an average of 1822 are active (present) in feature vectors of objects in the COIL-100 dataset. To reduce the computational complexity of the learning process, the feature vectors are further pruned so that only the 512 most frequently occurring features are retained in each instance representation. This corresponds to the process that produces elements in $\mathcal{M}$, alluded to in Def. 2.2. Note that the definition supports a randomized process, in this case, we use one based on frequency of feature occurrence.

Table 1 shows the performance of our method using conjunctions of edges to represent objects. We vary the number of views of an object (k) during training and use the rest of the views ($72 - k$) of an object for testing. The results indicate that conjunctions of edges provide useful information for object recognition and that SNoW is able to learn with it. The experimental results also exhibit that the superiority of the conjunctions based representation increases when the number of views per object is limited.

5.2 Results Using Pixel-Based Representation

To evaluate the SNoW approach with edge-based representation, we compare the experimental results of other classifiers. The pixel-based representation is used for the SNoW, Support Vector Machine (SVM) and nearest neighbor classifiers with each image is converted to a raster scan vector of intensity values.

The pixel-based representation in the SNoW approach consists of a set of Boolean features that encode the positions and intensity values of pixels. Let the (x, y) pixel of an image with width w and height h have intensity value $I(x, y)$ $(0 \leq I(x, y) \leq 255)$. This information is encoded as a feature whose index is $256 \times (y \times w + x) + I(x, y)$. This representation ensures that different points in the {`position` $\times$ `intensity`} space are mapped to different features. (That is, the feature indexed $256 \times (y \times w + x) + I(x, y)$ is *active* if and only if the intensity in position (x, y)

(a) object 65 (b) edge map (c) hor. edge (d) vert. edge (e) object 13 (f) edge map (g) hor. edge (h) vert. edge

Figure 2: Two objects with similar appearance (in terms of shape and intensity values) but very different edge maps. Note that some of the edges are blurred or missing because of aggressive downsampling (from 128×128 to 32×32 pixels).

Table 1: Experimental results of three classifiers using the 100 objects in the COIL-100 dataset

	# of views/object			
	36	18	8	4
	3600 tests	5400 tests	6400 tests	6800 tests
SNoW w/ conjunction of edges	**96.25%**	**94.13%**	**89.23%**	**88.28%**
SNoW w/ intensity values	95.81%	92.31%	85.13%	81.46%
Linear Support Vector Machine	96.03%	91.30%	84.80%	78.50%
Nearest Neighbor	98.50%	87.54%	79.52%	74.63%

is $I(x, y)$.) In our experiments images are normalizes so that $w = h = 32$. Note that although the number of potential features in our representation is $262,144$ ($32 \times 32 \times 256$), only 1024 of those are active (present) in each example, and it is plausible that many features will never be active. Indeed, in one of the experiments, only 13,805 of these features were ever active. Since the algorithm's complexity depends on the number of active features in an example rather than the total number of features, the sparseness also contributes to efficiency. Also notice that although this representation seems simplistic, the performance levels reached are surprisingly good.

Table 1 shows the recognition rates of the SNoW-based methods, the SVM-based method (using linear dot product for the kernel function), and the nearest neighbor classifier using the COIL-100 dataset. The important parameter here is that we vary the number of views of an object (k) during training and use the rest of the views ($72 - k$) of an object for testing.

The experimental results show that the SNoW-based method performs as well as the SVM-based method when many views of the objects are present during training and outperforms the SVM-based method when the numbers of views is limited. Although it is not surprising to see that the recognition rate decreases as the number of views available during training decreases, it is worth noticing that both SNoW and SVM are capable of recognizing 3D objects in the COIL-100 dataset with satisfactory performance if enough views (e.g., > 18) are provided and seem to be fairly robust even if only a limited number of views (e.g., 8 and 4) is used for training; the performance of both methods degrades gracefully. Overall, the SNoW-based method using conjunctions of edges clearly outperforms other classifiers.

5.3 Noise Model

To test whether the proposed learning framework is noise tolerant, we select a set of 10 objects[4] from the COIL-100 dataset and add in artificial occlusions for experiments. In the data set, each object has 36 images (10° apart) for training and the remaining 36 images for tests. The object images are occluded by a strip controlled by four parameters (α, p, l, g) where α denote the angle of the strip, p denote the percentage of occluded area, l denote the location of the center of the strip, and g denote the intensity values of the strip. Figure 3 shows some object images and the occluded object images for $\{\alpha, p, l, g\} = \{45°, 15\%, (16, 16), 0\}$.

[4]More specifically, the objects are selected from the set of objects on which the nearest neighbor classifier makes most mistakes, objects 8, 13, 23, 27, 31, 42, 65, 78, 80, 91.

This SNoW classifier is tested against this dataset us-

(a) (b)

Figure 3: Objects images with and without occlusion as well their edge maps.

ing the edge-based representation. Table 2 shows the experimental results with and without occlusions on this set of 10 objects with 36 views. The recognition performance degrades only slightly from 92.03% to 88.78%. Note that the objects are those on which the nearest neighbor classifier makes most mistakes.

Table 2: Experimental results of SNoW classifier on occluded images with 36 views per object

	Recognition rate w/o occlusion	Recognition rate w/ occlusion
SNoW	92.03%	88.78%

6 Summary and Conclusions

We proposed a learning framework for visual learning and evaluated it experimentally in the context of learning for object recognition. In this approach learnability can be rigorously studied without making assumptions on the distribution of the observed objects but, via the PAC model, the learned hypothesis' performance naturally depends on its prior experience.

An important feature of the approach is that learning is not studied directly in terms of the raw data but rather with respect to intermediate representations extracted from it and can thus be quantified in terms of the ability to generate expressive intermediate representations. It particular, it makes explicit the requirements from these representations to allow learnability. We believe that research in vision should concentrate on the study of these intermediate representations.

We have illustrated our approach in a large scale experimental study in which we use the SNoW learning architecture to learn representations for the 100 objects in COIL-100. Although it is clear that object recognition in isolation is not the ultimate goal, this study shows the potential of this computational approach as a basis for studying and supporting more realistic visual inferences.

Acknowledgments

Dan Roth is supported by NSF grant IIS-9801638. Ming-Hsuan Yang and Narendra Ahuja are supported through collaborative participation in the Advanced Displays and Interactive Displays Consortium sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAL01-96-2-0003.

References

[1] A. Blum. Learning boolean functions in an infinite attribute space. *Machine Learning*, 9(4):373–386, 1992.

[2] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[3] A. Carleson, C. Cumby, J. Rosen, and D. Roth. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department, May 1999.

[4] S. Edelman. On learning to recognize 3-d objects from examples. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(8):833–837, 1993.

[5] F. Fleuret and D. Geman. Graded learning for object detection. In *Proceedings of the IEEE workshop on Statistical and Computational Theories of Visions*, 1999.

[6] D. Haussler, M. Kearns, N. Littlestone, and M. K. Warmuth. Equivalence of models for polynomial learnability. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 42–55, Cambridge, MA, Aug. 1988.

[7] M. Kearns and M. Li. Learning in the presence of malicious error. *SIAM Journal of Computing*, 22(4), 1993.

[8] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. In *Proc. of the Annual ACM Symp. on the Theory of Computing*, 1995.

[9] E. Kushilevitz and D. Roth. On learning visual concepts and DNF formulae. *Machine Learning*, 24(1):65–85, 1996.

[10] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[11] N. Littlestone. Redundant noisy attributes, attribute errors, and linear threshold learning using winnow. In *Proceedings of the fourth Annual Workshop on Computational Learning Theory*, pages 147–156, 1991.

[12] M. Pontil and A. Verri. Support vector machines for 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6):637–646, 1998.

[13] D. Roth. Learning to resolve natural language ambiguities: A unified approach. In *Proc. of the Fifteenth National Conference on Artificial Intelligence*, pages 806–813, 1998.

[14] H. Shvaytser. Learnable and nonlearnable visual concepts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):459–466, May 1990.

[15] S. Ullman and S. Soloviev. Computation of pattern invariance in brain-like structures. *Neural Networks*, 12:1021–1036, 1999.

[16] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, Nov. 1984.

[17] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[18] M.-H. Yang, D. Roth, and N. Ahuja. A SNoW-based face detector. In *NIPS-12; The 1999 Conference on Advances in Neural Information Processing Systems*, pages 855–861. MIT Press, 2000.