From *Proceedings of the 1998 Image Understanding Workshop*. Monterrey, CA: Morgan Kaufmann.

Improving Rooftop Detection with Interactive Visual Learning*

Kamal M. Ali[◇] **Pat Langley**[◇] **Marcus A. Maloof**[†] **Stephanie Sage**[◇] **Thomas Binford**[‡]

[◇]Institute for the Study of Learning and Expertise, 2164 Staunton Court, Palo Alto, CA 94306
{ali, langley, sage}@isle.org • http://www.isle.org

[†]Department of Computer Science, Georgetown University, Washington DC 20057
maloof@cs.georgetown.edu • http://www.cs.georgetown.edu

[‡]Department of Computer Science, Stanford University, Stanford, CA 94305
binford@cs.stanford.edu • http://robotics.stanford.edu

Abstract

In this paper, we report progress on the use of machine learning to improve the process of rooftop detection in aerial images. We describe an existing system for building recognition, BUDDS, and identify its rooftop stage as a target for improvement. We then review the naive Bayesian classifier, a simple but robust approach to supervised induction, and the visual interface we developed to ease the labeling of training data. We present the results of experiments on the rooftop detection task that reveal improved recognition levels over the handcrafted BUDDS classifier, then examine the reliability and speed of the interactive labeling process itself. Finally, we consider related research and plans for future work.

1 Introduction

In the past 20 years, the computer vision community has made great strides in extending the functional coverage of image understanding systems. Researchers have developed integrated systems that operate on a variety of challenging tasks, including practical problems like the analysis of large-scale scenes in aerial imagery. But this progress has often obscured the underlying fragility of systems that have been tuned carefully to operate well on a handful of images.

The advent of programs like RADIUS, which provide a repository of common images for use in testing, has improved this situation. For specific domains, we now have image understanding systems that process many distinct images in a reasonably robust manner. Yet these programs often rely on heuristic knowledge that transfers poorly to new images, much less to new domains, and there remains considerable room for improving their behavior even on existing image libraries.

One path to such improvement invokes machine learning to refine or replace the handcrafted knowledge that currently guides an image understanding system (Bowyer et al., 1994). One can apply such techniques to any level of a complex vision system, provided data are available for training the learning algorithm. Indeed, future vision researchers might use learning methods to incorporate heuristic knowledge and to tune parameters for one stage of visual processing before addressing stages that build on it. Conveniently, the machine learning community has also developed an extensive experimental method for evaluating its algorithms (Kibler & Langley, 1988) that should carry over readily when they are used for computer vision.

In the following pages, we report our progress toward this goal. We begin by reviewing our computational framework, including a representation for visual knowledge, the vision system that uses this knowledge to process images, and the learning algorithm that aims to improve this process. We also discuss our approach to collecting training data, which relies on an interactive labeling system. Next we review some systematic experiments with visual learning, including tests of generalization to novel images. After this, we report empirical studies of the labeling process, which involves interactions between the system and a human. We close with a brief discussion of related work on visual learning and our plans for future research.

*This research was supported by the Defense Advanced Research Projects Agency under grant N00014-98-1-0543, administered by the Office of Naval Research, and by Sun Microsystems through an equipment grant.

2 An Approach to Visual Learning

To explore the potential of machine learning for computer vision, we had to select some task that requires image understanding. We decided to focus on the analysis of aerial photographs available from the DARPA RADIUS program, and in particular on the recognition and description of buildings, as this was a recognized problem of practical import on which there already existed a body of work. Another factor was the existence of BUDDS, a building recognition system developed by Lin and Nevatia (1996) at USC, which they offered to make available. Their software seemed ideal for our purposes because it was robust enough on the task to provide a good baseline, yet still seemed likely to benefit from improvement through machine learning.

In this section, we review BUDDS' representation and mechanisms, focusing on the stage that generates rooftop candidates, which we selected for our initial work. After this, we review the naive Bayesian classifier, the main learning algorithm we have used in our studies, followed by our approach to collecting data in support of the learning process. In this context, we discuss two challenges raised by this domain that are not normally addressed in machine learning: the distinct costs associated with different types of errors and the need for interactive but efficient labeling of instances.

2.1 The BUDDS System

The Buildings Detection and Description System (BUDDS) was developed by Lin and Nevatia (1996) at the University of Southern California to address the task of detecting and describing buildings in aerial photographs. Like many image understanding systems, BUDDS represents knowledge about a scene at different levels of description and operates by aggregating lower-level descriptors into higher-level ones, ultimately reaching characterizations of buildings in the scene.

Naturally, the software begins at the pixel level, describing images in terms of gray scale intensities within a rectangular grid, along with information about camera and sun angles. BUDDS invokes an edge detector to group pixels into connected segments, which it describes in terms of their starting and end points. The system then uses a line finder to group these segments into extended lines, each of which it represents using end positions, slope, and component segments.

At the next level, BUDDS constructs L junctions at the ends of each line and T junctions that occur along them. These structures it characterizes in terms of the junction vertices and the angle of their bisectors, along with their component lines. The software also groups nearby lines with similar slopes and lengths into parallel pairs, which it describes in terms of the lines involved. The ensuing level combines both junctions and parallels into three-sided structures or U contours, which again are denoted by their components.

After this, BUDDS combines pairs of nearby U's into parallelograms that constitute candidates for the rooftops of buildings in the image. The system describes these structures in terms of the component U's and the positions of their four vertices. The following stage involves elimination of some roof candidates based on global criteria (e.g., when one overlaps another). Finally, BUDDS combines the remaining rooftop candidates with evidence for walls and shadows to construct 3D wire-frame descriptions of rectilinear buildings.

At most processing stages, BUDDS' move to the next level involves two distinct operations. The system first generates candidates for the successive level, constrained mainly by locality assumptions; for example, it considers only nearby U's when constructing rooftop candidates. After this comes a filtering step that decides whether to retain or reject this candidate, typically using a handcrafted classifier that draws on computed features of the candidate. We believe that machine learning is best suited to improve decisions during this filtering step.

Discussions with R. Nevatia (personal communication, 1996) suggested that rooftop detection held the most promise for improvement through machine learning, so we should consider this stage in more detail. Again, this process combines three-sided U's to form parallelograms that constitute candidate roofs in the image. BUDDS filters these parallelograms by calculating nine continuous features and combining them into a weighted sum. For each candidate, if the result is higher than a specified threshold, the system retains it; otherwise it rejects the candidate.

Four of these features provide positive evidence that the candidate is a rooftop. For example, one such attribute specifies the extent to which edges are present in the component lines. Low contrast may obscure parts of these lines, and buildings with more complex structure will have no edges where component rectangles meet. However, the more complete the edges that make up a parallelogram, the greater the evidence for a rooftop. BUDDS uses a similar continuous attribute that favors rooftop candidates with more detected corners and ones with angles that are closer to 90 degrees. A third continuous feature reflects the extent to which opposite sides of the candidate are parallel, which can be affected indirectly by

missing edges. A fourth attribute measures evidence from shadows; taking the direction of illumination into account, it looks for expected correspondences between candidate corners and shadow corners, as well as candidate edges and shadow edges.

The remaining five features represent negative evidence that the candidate is actually a rooftop. For instance, one attribute measures the degree to which lines cross the sides of the candidate description; clearly, the presence of such intersecting lines reduces the chances that the parallelogram is a building roof. Two additional attributes indicate the extent to which sides of the candidate contain L junctions and T junctions other than at its corners; the occurrence of such junctions within a side make the candidate an implausible rooftop. A fourth negative feature measures the presence of corresponding gaps on opposite sides of the candidate, which are unlikely in buildings. Finally, BUDDS computes the displacement between the edges that make up the candidate rooftop and its inferred sides, giving higher scores for poorer matches.

Lin and Nevatia developed their classifier manually, through trial and error on a variety of images and their associated rooftop candidates. Our aim was to use some form of machine learning to produce a new classifier that improved on the handcrafted method's ability to eliminate poor rooftop candidates but still retain acceptable ones. For this we needed two items—an appropriate learning algorithm and a source of training data—to which we now turn.

2.2 The Naive Bayesian Classifier

Since BUDDS treats the filtering of rooftop candidates as a classification task, it seemed natural to formulate our learning problem in terms of supervised induction. Preliminary experiments with a number of induction methods (Maloof, Langley, Sage, & Binford, 1997) suggested that we would obtain good results with the *naive Bayesian classifier*, a simple technique for probabilistic induction that has a long history in pattern recognition (Duda & Hart, 1973) but that has gained acceptance in machine learning more recently (Clark & Niblett, 1989; Kononenko, 1990; Langley, Iba, & Thompson, 1992).

The Bayesian classifier represents its knowledge about each class in terms of a single probabilistic summary. Each such class description has an associated class probability or base rate, $p(C_k)$, which specifies the prior probability that one will observe a member of class C_k. Each description also has an associated set of conditional probabilities, specifying a probability distribution for each attribute. For a nominal attribute, one typically stores a discrete distribution, with each $p(v_j|C_k)$ term stating the probability of value v_j given an instance of class C_k. For a numeric attribute, one must represent a continuous probability distribution of that attribute given the class. This requires that one assume some general form or model, the most common choice being the normal distribution, which one can represent conveniently entirely in terms of its mean μ and its variance σ^2.

To classify a new instance I, a naive Bayesian classifier applies Bayes' theorem to determine the probability of each description given the instance,

$$p(C_i|I) = \frac{p(C_i)p(I|C_i)}{p(I)} \quad .$$

However, since I is a conjunction of j values, one can expand this expression to

$$p(C_i|\bigwedge v_j) = \frac{p(C_i)p(\bigwedge v_j|C_i)}{\sum_k p(\bigwedge v_j|C_k)p(C_k)} \quad ,$$

where the denominator sums over all classes and where $p(\bigwedge v_j|C_i)$ is the probability of the instance I given the class C_i. After calculating these probabilities for each description, the algorithm assigns I to the class with the highest overall probability.

In order to make the above expression operational, one must still specify how to compute the term $p(\bigwedge v_j|C_i)$. Naive Bayes assumes independence of attributes within each class, which lets it use the equality

$$p(\bigwedge v_j|C_i) = \prod_j p(v_j|C_i) \quad ,$$

where the values $p(v_j|C_i)$ represent the conditional probabilities stored with the class. This approach greatly simplifies the computation of class probabilities for a given observation.

The Bayesian framework also lets one specify prior probabilities for both the class and the conditional terms. A common scheme makes use of 'uninformed priors', which assign equal probabilities to each class and to the values of each attribute. However, one must also specify how much weight to give these priors relative to the training data. For example, one can use a Dirichlet distribution to initialize probabilities and give these priors the same influence as a single training instance. Clark and Niblett (1989) describe another approach (which we also use) that does not require explicit priors, but instead estimates $P(C_i)$ and $p(v_j|C_i)$ directly from their proportions in the training data. When no instances of an attribute value for a given class have been observed, it replaces the zero probability with $p(C_i)/N$, where N is the number of training cases.

Learning in the naive Bayesian classifier is an almost trivial matter. The simplest implementation increments a count each time it encounters a new instance, along with a separate count for a class each time it observes an instance of that class. Together with the prior probabilities, these counts let the classifier estimate $p(C_i)$ for each class C_i. In addition, for each instance of a class that has a given nominal value, the algorithm updates a count for that class-value pair. Together with the second count, this lets the classifier estimate $p(v_j|C_i)$. For each numeric attribute, the method retains and revises two quantities, the sum and the sum of squares, which let it compute the mean and variance for a normal curve that it uses to find $p(v_j|C_i)$. In domains that can have missing attributes, it must include a fourth count for each class-attribute pair.

In contrast to many induction methods, the naive Bayesian classifier does not carry out an extensive search through a space of possible descriptions. The basic algorithm makes no choices about how to partition the data, as with decision-tree methods, or which direction to move in a weight space, as with neural networks, and the resulting probabilistic summary is completely determined by the training data and the prior probabilities. Nor does the order of the training instances have any effect on the output; the basic process produces the same description whether it processes the data incrementally or all at once. These features make the learning algorithm both simple to understand and quite efficient.

But the naive Bayesian classifier buys these advantages at the cost of two important assumptions. First, it posits that each class can be summarized by a single probabilistic description, and that these are sufficient to distinguish the classes from one other. This idea is closely related to the assumption of linear separability in early work on neural networks and, like perceptrons, naive Bayes is typically limited to learning classes that can be separated by a single decision boundary. Second, the approach assumes that, within each class, the probability distributions for attributes are independent of each other. Clearly, this assumption is unrealistic for many real-world domains, and one can easily design nonindependent data sets on which naive Bayes does poorly. These apparent drawbacks led most machine learning researchers to ignore this promising method for many years.

However, experimental studies across a broad range of domains show that the naive Bayesian classifier often fares very well compared to much more sophisticated induction algorithms like decision-tree induction (e.g., Langley et al., 1992). In some cases, this reflects a representational bias that is more appropriate to the domain, since some class distinctions are easier to represent using global evidence combination than with decision trees. However, comparisons to methods that construct full Bayesian networks, which use the same probabilistic representation but which do not make the independence assumption, show that naive Bayes often outperforms them as well (Provan & Singh, 1996). The natural explanation is that, because naive Bayes must fit fewer parameters than more flexible techniques, it has less chance of overfitting limited training data. Recent analyses by Domingos and Pazzani (1997) reveal other factors, including the fact that naive Bayes learns the optimal classifier even in many domains that involve significant dependencies.

The naive Bayesian classifier has another advantage: it can be easily modified to take into account the cost of errors. One formulation for two-class problems, which we use below, specifies the relative cost as a parameter $-1 < c < 1$ and uses this constant to compute utilities from the posterior probabilities produced by naive Bayes. If we let $U(r|\mathbf{x})$ be the utility of classifying candidate $\mathbf{x}$ as a rooftop and let $U(n|\mathbf{x})$ be the utility of classifying $\mathbf{x}$ as a nonroof, then

$$U(r|\mathbf{x}) = P(r|\mathbf{x}) + c \cdot P(n|\mathbf{x})$$

and

$$U(n|\mathbf{x}) = P(n|\mathbf{x}) - c \cdot P(n|\mathbf{x}) \quad ,$$

where c is positive when mislabeled roofs are more expensive than mislabeled nonroofs. In other words, one calculates the utility for each class by adding a factor to the posterior probability of the more expensive class and by subtracting the same amount for the less costly class. This modified Bayesian classifier then predicts the class with the highest utility.

2.3 Interactive Labeling of Training Data

Machine learning cannot occur without some training data to drive the process. We can use BUDDS to generate candidate rooftops and their nine associated features; indeed, the system generates thousands of such candidates per image. But we also need labels that specify whether each candidate should be retained (a positive instance) or rejected (a negative instance).

Since we could identify the vertices of each actual rooftop in an image, we tried using their distance from candidate vertices to determine class labels automatically. But inspection of the results indicated that this scheme assigned labels poorly, apparently because many bad candidates had vertices near those of actual rooftops. Also, we wanted the classifier to retain candidates that *looked* like good rooftops, but were not part of buildings, for rejec-

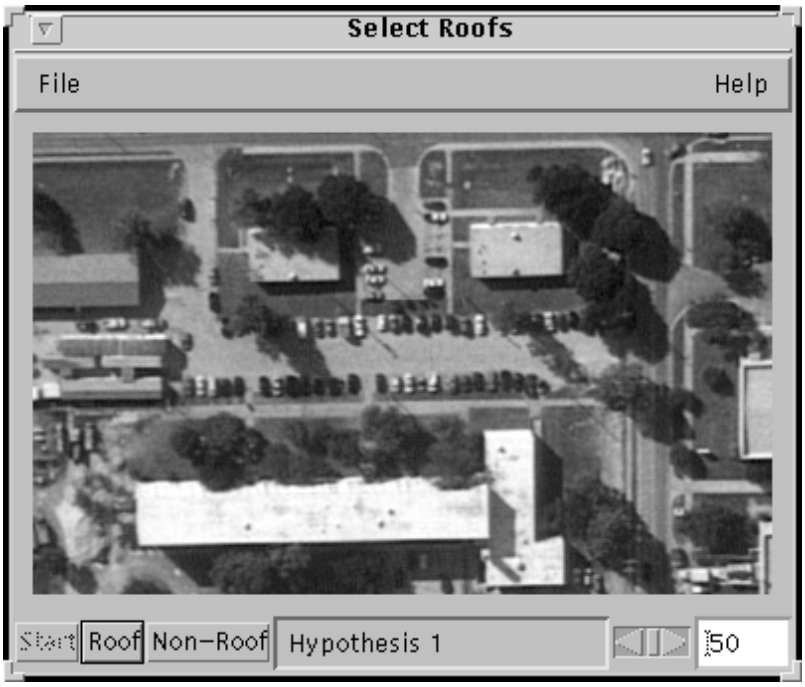

Figure 1: Visualization interface for labeling rooftop candidates. The system presents candidates to a user who labels them by clicking either the 'Roof' or 'Non-Roof' button.

tion by later stages of Budds' processing, since our USC colleagues believed that the nine features were not enough to identify rooftops on their own.

These concerns suggested that some form of manual labeling by a human was needed, but the number of candidates generated for each image made this a daunting task. In response, we implemented an interactive labeling system in Java, shown in Figure 1, that successively displays each extracted rooftop to the user. The interface draws a candidate over the portion of the image from which it was extracted, lets the user click buttons for 'Roof' or 'Non-Roof' to label it, and moves on to the next candidate.

We used this approach to generate data from two images of Fort Hood, Texas, which were collected as part of the RADIUS program (Firschein & Strat, 1997). These images, FHOV1027 and FHOV625, cover the same area but were taken from different viewpoints, one from a nadir angle and the other from an oblique angle. We subdivided each image into three subimages, focusing on locations that contained concentrations of buildings, to maximize the number of positive rooftop candidates. This gave us three pairs of images, each pair covering the same area but viewed from different aspects, as summarized in Table 1. Even though the data set contained 17,829 rooftop candidates, the interface required only about five hours to label them. This comes to under one second per candidate, which seems reasonably efficient. However, the task remains time consuming and also raises questions about the reliability of the labeling process; we will return to both of these issues in Section 4.

3 Studies of Cost-Sensitive Learning

In a previous paper (Maloof et al., 1997), we reported initial experimental studies of machine learning for rooftop classification using three different techniques: decision-tree induction, nearest neighbor, and the naive Bayesian classifier. Here we report more extensive experiments that replicate and extend our earlier results, but that focus on naive Bayes, since it fared best in those studies.

3.1 Experimental Measures and Design

Our basic hypothesis was that a learned rooftop classifier could outperform the handcrafted Budds classifier. However, before we could test this claim experimentally, we needed more than labeled examples of rooftop candidates; we also needed some clear measure of performance. The most obvious approach, usually taken in machine learning studies, was to use classification accuracy, but analysis of the domain revealed a problem with this metric. Briefly, a simple accuracy measure assumes that all

Table 1: Characteristics of the images and data sets. We began with a nadir and an oblique image of an area of Fort Hood, Texas, and we divided each into three subimages that contained concentrations of buildings. We then used BUDDS to generate rooftop candidates and interactively labeled each as either a positive or negative example of the concept "rooftop".

Image Number	Original Image	Location	Aspect	Positive Examples	Negative Examples
1	FHOV1027	1	Nadir	197	982
2	FHOV625	1	Oblique	238	1955
3	FHOV1027	2	Nadir	71	2645
4	FHOV625	2	Oblique	74	3349
5	FHOV1027	3	Nadir	87	3722
6	FHOV625	3	Oblique	114	4395

errors have the same cost, but we knew that this does not hold for the task of building detection. Given actual costs for each error type, we could calculate a weighted measure using the accuracy on each class, but we knew only that labeling buildings as nonbuildings (which later BUDDS stages cannot correct) was more expensive than the converse.

In such situations, a natural solution is to evaluate a cost-sensitive method over a *range* of cost settings and to report the results in a Receiver Operating Characteristic (ROC) curve (Swets, 1988). The basic idea is to systematically vary some aspect of the method or domain (in this case the cost ratio), then plot the true positive rate against the false negative rate for each situation. Although researchers have used such ROC curves in psychology for decades, this technique has only recently filtered into machine learning (Bradley, 1997; Provost & Fawcett, 1997).

To generate ROC curves in the studies reported below, we varied the cost parameter c and measured the resulting true positive and false positive rates for naive Bayes on a given test set of rooftop candidates. However, rather than reporting a complete ROC curve for each experimental condition, we often follow Swets' recommendation and present the area under this curve, which we approximated by summing the areas of the trapezoids defined by each pair of adjacent points.

Following the standard experimental method in machine learning, in each situation we trained the induction algorithm on labeled data, in this case rooftop candidates, that were separate from those used to test the learned classifiers. Because the BUDDS classifier was hand configured, it had no training phase, so we applied it directly to the instances in the test set, varying its threshold parameter to generate an ROC curve.

To guard against nonrepresentative samples, we report results that are averaged over ten different partitions into training and test data. Unless stated otherwise, we used 60% of the candidates for training and 40% for testing purposes. The experiments differed in whether the training and test cases came from the same or distinct images, which let us examine different forms of generalization beyond the training data.

3.2 Within-Image Learning

Our first experimental study examined how naive Bayes behaves on *within-image* learning, that is, when generalizing to test cases taken from the same image on which we trained it. Our research hypothesis was that the learned classifier would be more accurate, over a range of misclassification costs, than the handcrafted classifier. Because our measure of performance was area under the ROC curve, this translates into a prediction that the ROC curves of the learned rooftop classifiers would have larger areas than those for the BUDDS classifier.

Table 2: Approximate areas under the ROC curves, along with 95% confidence intervals, for naive Bayes and BUDDS on the within-image experiment.

	Naive Bayes	BUDDS
Image 1	0.870±0.008	0.717±0.009
Image 2	0.812±0.017	0.773±0.004
Image 3	0.962±0.013	0.899±0.015
Image 4	0.908±0.025	0.901±0.007
Image 5	0.869±0.016	0.833±0.021
Image 6	0.835±0.025	0.849±0.010

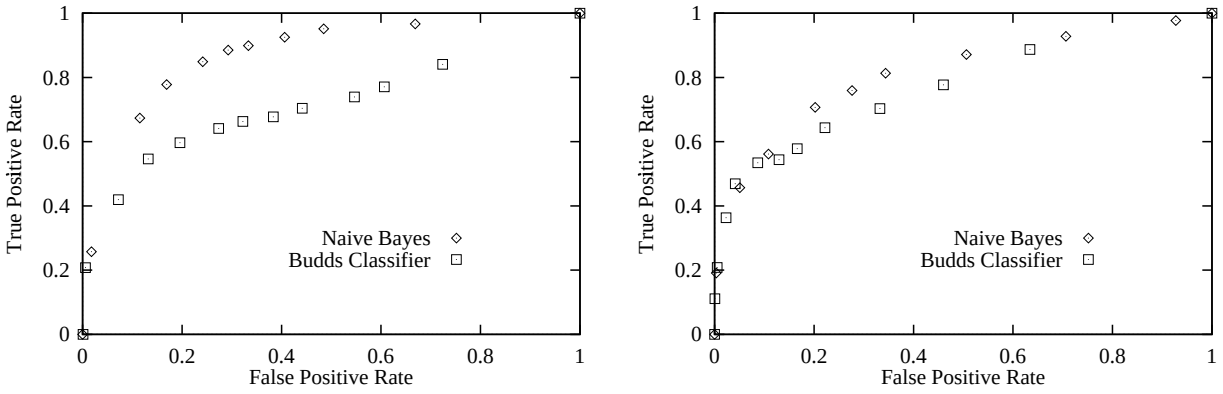

Figure 2: ROC curves for Images 1 (left) and 2 (right), which we obtained by training and testing on rooftop candidates from the same image over a range of misclassification costs and averaging over ten runs. These images cover the same location but from different aspects, with Image 1 giving a nadir view and Image 2 being oblique.

Figure 2 presents the ROC curves for Images 1 and 2, in which naive Bayes clearly fares better than the handcrafted BUDDS classifier, producing fewer false positives (nonroofs mislabeled as roofs) for a given level of true positives (correctly labeled roofs). For all except Image 6, the learned classifier produced curves with areas greater than those for BUDDS, thus generally supporting our research hypothesis.

3.3 Generalization Over Aspect

Although the results from the previous study were encouraging, we also wanted to understand better how the knowledge learned from one image generalizes to other images. Our hypothesis here was a refined version of the previous one: classifiers learned from one set of images would be more accurate on unseen images than handcrafted classifiers. However, we also expected that between-image learning would give lower accuracy than the within-image situation, since differences across images would make generalization more difficult.

Thus, our second experiment focused on how naive Bayes generalizes over aspect. Recall from Table 1 that we had images from two aspects (i.e., nadir and oblique) and from three locations. This let us train the algorithm on an image from one aspect and test the learned classifier on an image from another aspect but from the same location. As an example, for the nadir aspect, we chose Image 1 and then tested on Image 2, which is an oblique image of the same location. We ran naive Bayes in this manner using the images from each location, while varying the cost parameter and measuring its true positive and false positive rates. We then averaged these measures across the three locations and plotted the results as ROC curves, as shown in Figure 3. The areas under these curves and their 95% confidence intervals appear in Table 3.

One obvious conclusion is that the nadir images appear to pose an easier problem than the oblique images, since the curves for testing on nadir candidates are generally higher than those for testing on data from oblique images. For example, Table 3 shows that naive Bayes generates a curve with an area of 0.878 for the nadir images, but produces one with an area of 0.842 for the oblique images. In contrast, the BUDDS classifier gives an area of 0.837 for the nadir condition and 0.831 for the oblique condition.

Table 3: Approximate areas under the ROC curves, along with 95% confidence intervals, for naive Bayes and BUDDS when generalizing across aspect (A), generalizing across location (L), and generalizing within an image (W). The labels 'nadir' and 'oblique' indicate the testing condition.

		Naive Bayes	BUDDS
(A)	Nadir	0.878±0.042	0.837±0.085
	Oblique	0.842±0.063	0.831±0.068
(L)	Nadir	0.901±0.079	0.837±0.085
	Oblique	0.831±0.067	0.831±0.068
(W)	Nadir	0.900±0.012	0.837±0.085
	Oblique	0.851±0.022	0.831±0.068

3.4 Generalization Over Location

A similar study examined generalization over location. To this end, we trained naive Bayes on pairs of images from one aspect and tested on the third image from the same aspect. As an example, for the

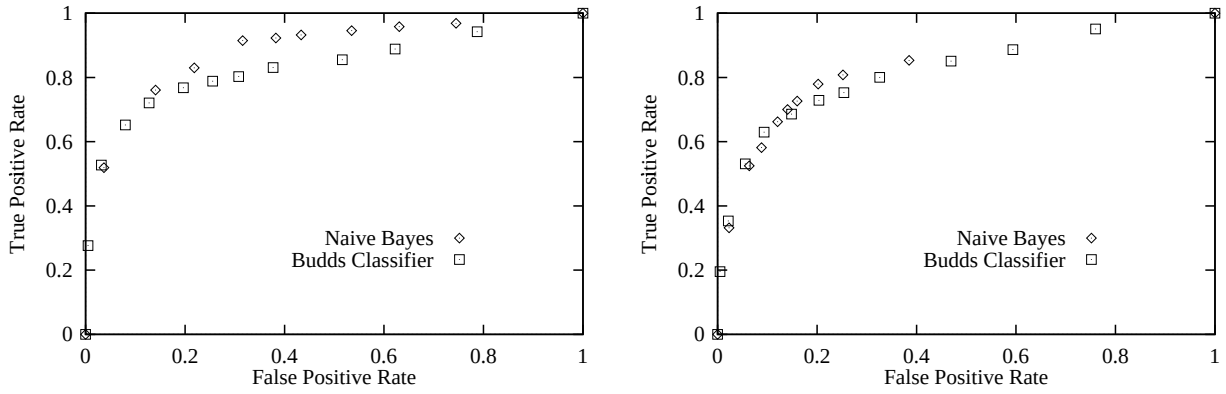

Figure 3: ROC curves for the experiment that tested generalization over aspect. For each location, we trained a classifier on the oblique image and tested it on the nadir image (left), then reversed the procedure by training on nadir images and testing on oblique ones (right).

nadir images, one of the three learning runs involved training on rooftop candidates from Images 1 and 3, then testing on candidates from Image 5. We then ran each of the algorithms across a range of costs, measuring the false positive and true positive rates. We plotted the averages of these measures across all three learning runs for one aspect in an ROC curve, as shown in Figure 4.

In this context, we again see evidence that the oblique images presented a more difficult recognition task than the nadir aspect, since the oblique areas are less than those for the nadir images. Comparing the behavior of the various methods, Table 3 shows that, for the nadir aspect, naive Bayes performs somewhat better than the BUDDS classifier, which give areas of 0.901 and 0.837, respectively. When generalizing over location on the oblique images, naive Bayes and the BUDDS classifier produced ROC curves with equal areas of 0.831.

Recall that we also anticipated generalizing across images would give lower accuracies than generalizing within images. To test this hypothesis, we must compare the results from these experiments with those from the within-image experiments in Table 2. Simple calculation shows that, for the within-image condition, naive Bayes produced an average ROC area of 0.9 for the nadir images and 0.851 for the oblique images. Thus, the results are not entirely consistent with our prediction, since naive Bayes did equally well when generalizing over location for the nadir image.

3.5 Rates of Learning

We were also interested in naive Bayes' behavior as it processed increasing amounts of training data. Both theoretical analyses (Langley et al., 1992) and experimental results in other domains (Langley & Sage, 1994) suggest that this induction method achieves high accuracy from relatively few training cases. To see if similar effects held for rooftop classification, we carried out a final study in which we systematically varied the number of training cases available to the algorithm. Here we used the rooftop candidates available from all six images, again splitting the data into training (60%) and test (40%) sets, but further dividing the training set randomly into ten subsets (10%, 20%, ..., 100%). We ran naive Bayes on each training subset and evaluated the resulting classifiers on the reserved test data, averaging our results over 25 separate training/test splits.

Figure 5 shows the resulting learning curve, each point of which corresponds to the average area under the ROC curve for a given number of training cases. The graph also includes a flat curve for the BUDDS classifier, since it involves no training and we simply applied it to the same test set for each number of training cases. Naturally, the naive Bayesian classifier improves with increasing amounts of training data. More important, its performance exceeds that for BUDDS almost from the start, after observing only 10% of the training data (less than the number of candidates from one image) and levels off after processing 30% of the cases. Not only did naive Bayes outperform BUDDS and other induction methods, but it was able to accomplish this feat using very little of the available training data.

4 Studies of Interactive Labeling

The experiments above showed that a learned Bayesian classifier typically outperforms the handcrafted BUDDS classifier, that the learning method generalizes across both aspect and location, and that it achieves good results from little training data. But these results relied on an interactive scheme in

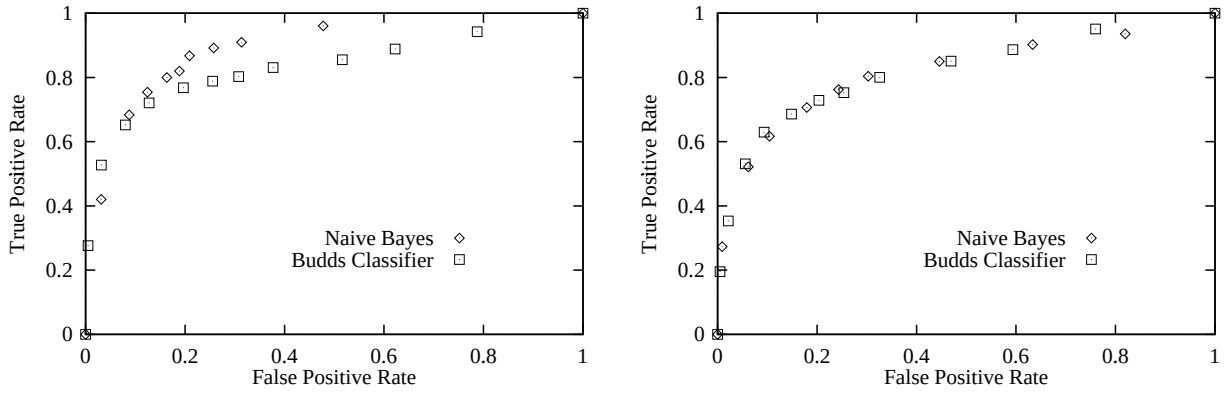

Figure 4: ROC curves for the experiment that tested generalization over location. For each pair of images with the nadir aspect, we trained on that pair and tested on the third nadir image (left), then applied the same methodology to images with the oblique aspect (right).

which a human labeled the candidates generated by BUDDS, and the studies left unanswered some central questions about the role of the developer in this process. We now present some more recent experiments that directly address the interactive aspects of our framework, in particular the reliablity of labeling decisions and the potential for reducing the number of rooftop candidates that one must label.

4.1 Consistency of Human Labeling

We developed the interactive labeling software to provide classified cases for training and testing. As described in Section 2, this interface presents the user with candidates (appearing as highlighted parallelograms in the image), each of which he must label as a roof or nonroof. This task seems straightforward, but the complex shape of some buildings, the existence of similar candidates, and the sheer length of a labeling session can all introduce subjectivity and uncertainty into user decisions. This raises questions about labeling consistency, and thus about the reliability of our results, both across different users and within a single user.

To address this issue, we examined the results of the labeling process in some detail. We combined records of labeling decisions made by one team member (Maloof) with more recent recordings from another member (Ali), who labeled the same candidates in two separate sessions. Our comparisons focused on Images 1 and 2 from Table 1, which constituted nadir and oblique views of the same Fort Hood location. Our aim was to understand how consistent these users were at labeling rooftop candidates and to understand the effects of their disagreement on the behavior of learned classifiers.

Our initial analysis examined consistency in the labeling decisions made by the two users. To this end, we generated a confusion matrix for their decisions that included the number of items they agreed were roofs or nonroofs, as well as the number one classified as a roof and the other as a nonroof. Table 4 shows the confusion matrix between Maloof's and Ali's labels on Image 1. If we treat the Maloof labels as correct, then Ali had 69% true positive candidates (which they agreed were roofs) and a 6% false negative rate (which Ali labeled as nonroofs and Maloof as roofs). Treating the Ali labels as correct gives nearly the same rates, and both points fall just above the naive Bayes curve for Image 1 in Figure 2. In other words, Maloof and Ali were little better at predicting each other's labels than was the naive Bayesian classifier, and we found qualitatively similar results for Image 2. This suggests that the learning method may be doing as well as one can reasonably expect, considering the apparent unreliability of human labelers.

Table 4: Confusion matrix for two users (Maloof and Ali) on Image 1.

	ALI ROOF	ALI NONROOF
MALOOF ROOF	134	61
MALOOF NONROOF	63	921

In our second analysis, we examined the consistency *within* a single human labeler (Ali) across two different sessions on Image 1. Table 5 presents the resulting confusion matrix. If we treat Ali_1 as correct, then Ali_2 has an 86% true positive rate and a 5% false negative rate. These numbers are much better than the between-user rates, but they are not as high as one might hope given a single labeler.

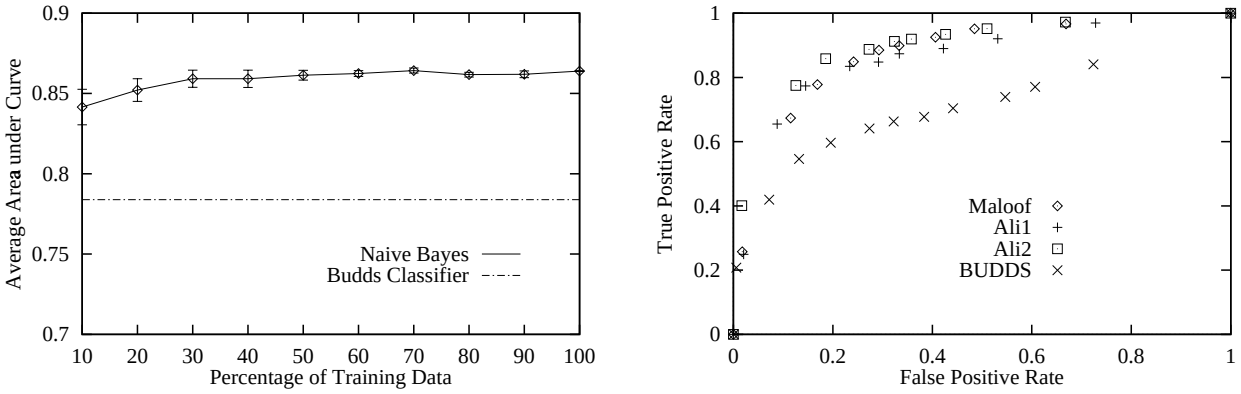

Figure 5: Learning curve for naive Bayes (left), plotting area under the ROC curve against size of the training set, averaged over 25 runs on data from all six images, and (right) ROC curves produced by training and testing naive Bayes on rooftops labeled by two users in three separate sessions, along with the curve for the BUDDS classifier.

Overall, these analyses suggest that the task of labeling rooftop candidates is a difficult one, and that the naive Bayesian classifier is doing fairly well given the inherent noise in its data.

We also wanted to study the effects of labeler differences on the behavior of learned classifiers, so we turned again to ROC analysis. Figure 5 shows four ROC curves for Image 1, one for naive Bayes trained and tested on candidates labeled by Maloof, two for naive Bayes using labels from separate sessions by Ali, and the last for the BUDDS classifier tested on Maloof labels. As before, we used 60% of the labeled candidates for training and the rest as test cases, using a variety of cost coefficients during testing. We repeated this process over ten random training/test partitions to obtain average values.

Table 5: Confusion matrix for different sessions (Ali_1 and Ali_2) of one user on Image 1.

	ALI_2 ROOF	ALI_2 NONROOF
ALI_1 ROOF	167	28
ALI_1 NONROOF	49	935

The figure's most striking aspect is the rough similarity of the three learned curves when contrasted with the curve for BUDDS. This suggests that, despite apparent variations in labeling decisions both between and within users, we can remain confident in our general results. However, there are some important differences. For instance, to achieve a 90% true positive rate, one must tolerate a false positive rate of 28% for Image 1 candidates that Ali labeled in his second session. In contrast, achieving this 90% level requires a 30% false positive rate for Maloof labels and 42% false positives for labels produced in Ali's first session. Thus, the labeled candidates from some sessions are more difficult for naive Bayes to master than those from others.

4.2 Active Learning for Reduced Labeling

Another of our concerns centered on the arduous and lengthy nature of the interactive labeling procedure. Although we were able to label all 17,829 rooftop candidates for our initial studies in around five hours, this study involved only six images. For interactive labeling to play a practical role in developing future vision systems, which would involve training on many more images, we must find some way to reduce the labeler's effort. But recall that the learning curve for naive Bayes in Section 3 showed the method reaching reasonable levels after seeing only a fraction of the training data.

This observation suggested that we modify the labeling interface to present candidate rooftops more selectively. In response, we augmented the software to update the naive Bayesian summaries after each newly labeled case, then used the revised classifier to decide whether to present the next candidate for labeling. In particular, the system showed the user only those cases about which it was uncertain, that is, on which the probability of the most likely class fell below a certain level. This approach is similar in aims and method to work on *active learning* (e.g., Cohn, Ghahramani, & Jordan, 1996), which is also motivated by domains with high labeling costs.

One design decision that arose in modifying the interface concerned the uncertainty threshold. To better understand naive Bayes' behavior, we plotted

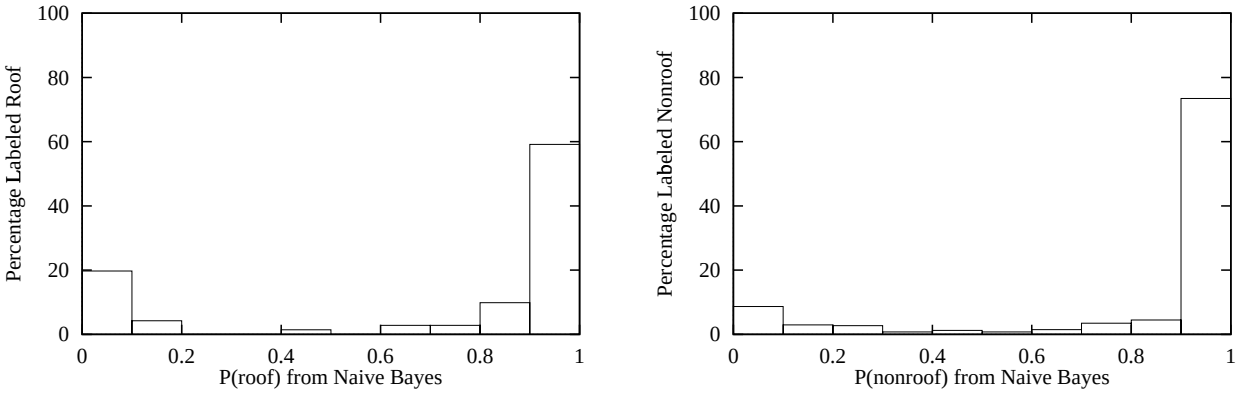

Figure 6: Distribution of actual rooftops (left) in the test set as a function of p(roof) and distribution of actual nonroofs (right) as a function of p(nonroof).

its distribution of errors on earlier runs as a function of classification confidence, or the maximum of $p(roof)$ and $p(nonroof)$. For example, the left portion of Figure 6 shows the distribution for $p(roof)$ that naive Bayes produced for test cases that were actually roofs.

This distribution is surprising on two counts. First, the classifier was much more certain about many predictions than we expected, with 60% receiving a probability above 0.9. Second, some 20% of the candidates received near-zero probabilities, even though they reflected actual rooftops, whereas very few cases fell in the middle range. The pattern repeats for the nonroof cases and the nonroof predictions, as the right part of the figure reveals. This has unfortunate consequences for any scheme that associates high confidence with lower error, as seems natural in an active learning scenario. The bimodal distribution means that $p(nonroof)$ will be near 1.0 for many actual roofs and that $p(roof)$ will approach 1.0 for many nonroof cases.

Figure 7 presents the resulting distribution of errors as a function of $p(roof)$. The graph shows that if $p(nonroof)$ is near to 1.0, then the error rate is quite low, but that errors are more common when $p(roof)$ is near 1.0. Based on this observation, we decided to implement an assymetrical form of active learning in which the interface withheld candidates from the user only when $p(nonroof)$ was high, but which presented any cases that the current classifier would label as a roof. We selected the threshold $p(nonroof) > 0.9$, since calculations showed that this level would incur only five percent errors.

Another design decision concerned the number of cases the interface should collect before active learning takes over. Since rooftops are both rare and more important, it seemed prudent to require the system to process a certain minimum number of rooftops seen before this occurred, whereas introducing such a threshold for nonrooftops seemed less important. Preliminary studies suggested 30 as a good threshold for the minimum number of rooftops, which represents 15% of the rooftops in Image 1.

Naturally, we wanted to examine the effect of embedding active learning in the interactive labeling system. We hoped that this modification would reduce labeling effort substantially but cause little reduction in the learned classifier's performance, so we used the number of labeled cases and ROC areas as our dependent measures. We were interested in comparing a passive learning scenario, in which the interface asked the user to label all training cases, with an active learning situation, in which the system presented only those cases about which it was uncertain.

As usual, we wanted to average over different runs but, for active learning, this would mean asking the user to revisit the interface many times to collect reliable results. Instead, we simulated this experimental condition by reusing labels the user generated in the passive condition. That is, the human first labeled all rooftop candidates from a given image and, for a given run with passive learning, we divided these data into separate training and test sets. Next, for each step in the corresponding run with active learning, the system selected an uncertain training case and assigned it the same label as it received in the first condition. This scheme had the added advantage that it reduced variance due to within-user inconsistency.

For each run in each condition, we used the learned Bayesian classifier to classify the same test cases, all of which were labeled by the same user. As before, we invoked these classifiers with varying costs to obtain ROC curves and their associated areas, and we averaged over ten different partitions of the data into

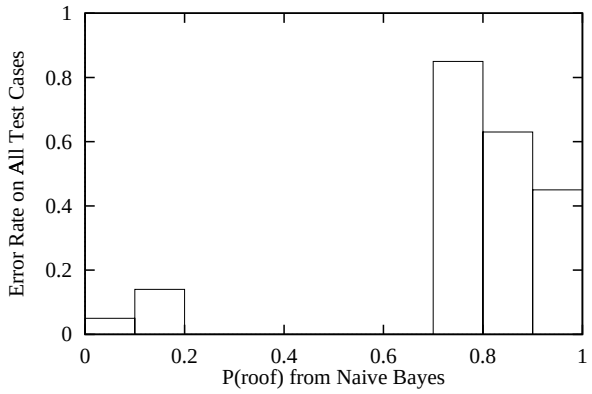

Figure 7: Test error rate for the naive Bayesian classifier as a function of $p(roof)$.

training and test sets. We carried out this experimental procedure for Images 1 and 2 from Table 1.

Figure 8 presents the ROC curves for Image 1. The key result is that we obtained very similar curves when the user (Ali) labeled all training cases and when the simulated user labeled only about 35% of the rooftop candidates. Thus, active learning reduced the user's labeling effort by two-thirds on this image with little reduction in classification performance. In fact, the area under the passive learning curve is 0.866 and that for active learning is 0.849, so the latter comes to 98% of the former. Both curves are substantially better than that for the BUDDS classifier, which had 0.717 for its area.

We can also use the ROC curves to determine the percentage of false positives (nonroofs classified as roofs) that one must tolerate to find a given percentage of the rooftops. For example, to detect 90% of the roofs, the classifier built through passive learning mislabels 42% of the nonroofs, the active-learned classifier mislabels 50%, and the BUDDS classifier some 82% of them. From this perspective, presenting only 33% of the rooftops candidates to the human labeler incurs an extra 8% false positive rate.

Figure 8 also shows corresponding results on Image 2. This oblique image is more difficult because BUDDS' uses features that favor rooftop candidates with corners close to 90 degrees. The areas under the ROC curve are less than they were for their analogs on Image 1, but again we find similar results for passive learning (0.791) and active learning (0.775). On this image, the selective interface presented only 24% of the rooftop candidates, an even greater savings than on the first one. Inspection of the ROC curves shows that active learning does slightly worse than passive learning in part of the space, but overall the two curves are difficult to distinguish, giving further evidence that active learning is effective.

5 Related and Future Research

The relationship between computer vision and machine learning has a long history. Indeed, research on pattern recognition has assumed for decades that learning can contribute to the creation of robust recognizers, and more recent work on 'appearance-based' vision often takes a similar attitude. The combination of machine learning with multi-level vision systems has been less common, but Bowyer et al. (1994) review some work along these lines, most of which addresses the acquisition of 2D or 3D models for object classes.

In contrast, our approach uses machine induction to improve the reliability of decisions within an existing image understanding system. One other effort that shares this goal is reported by Draper (1996), who also focuses on recognizing rooftops and other objects in aerial images. His formulation relies on reinforcement learning to assign credit in a multi-stage recognition system similar to BUDDS, combined with backpropagation in neural networks to learn when to select alternative operators. The most important difference between our frameworks is that Draper hopes to eliminate user interaction by automating the assignment of credit, whereas our scheme assumes continuing reliance on a human expert while trying to minimize demands on his time. However, the two approaches are not mutually exclusive, and a combined framework could well inherit benefits from both parents.

Although our experimental studies have clarified the potential for machine learning, combined with interactive labeling, to produce more robust vision systems, there remain many avenues for future research. One natural direction would involve modifying the learning algorithm to further improve the process of rooftop recognition. Previous research has shown that, when the predictive attributes are not normally distributed, one can improve the naive Bayesian classifier by using kernel estimators (John & Langley, 1995) or by discretizing their values. Automatic feature selection can also give substantial improvements in some domains (Langley & Sage, 1994). Moreover, this latter technique holds promise as a development tool that would let researchers evaluate the usefulness of new features for use in building recognition.

However, a more fundamental issue concerns extending our approach to other levels of the BUDDS software. As we noted in Section 2, the system makes earlier decisions about which junctions, parallels, and U contours to pass on to the rooftop stage, and also takes steps after this level. Each stage holds potential for improvement through machine learning, and we intend to adapt our interactive labeling

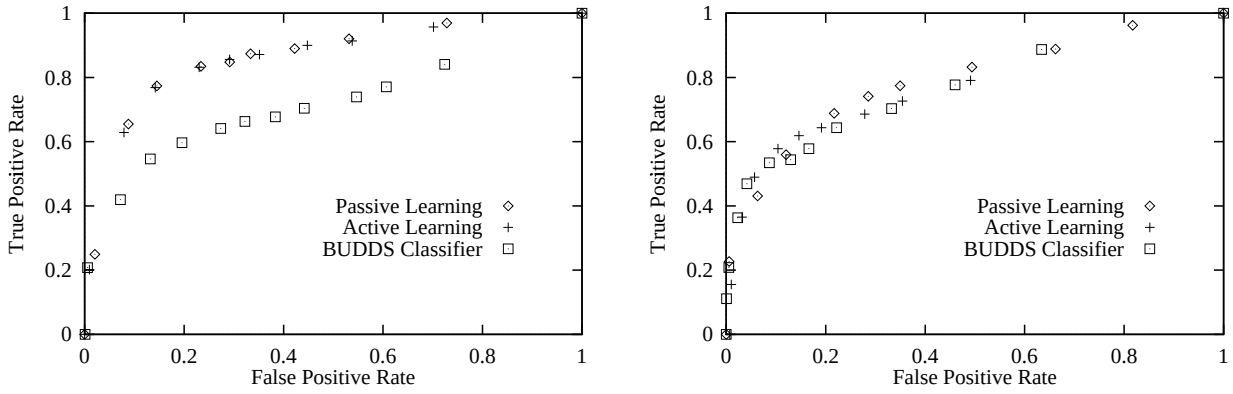

Figure 8: ROC curves for passive and active learning on the nadir Image 1 (left) and the oblique Image 2 (right). The 'passive learning' curves correspond to runs in which the user labeled the entire training set. The 'active learning' curves come from runs in which the interface selected uncertain training cases for labeling and ignored the rest. The other curves represent the BUDDS classifier.

software to operate at these other levels. One goal for such learning is to reduce BUDDS' complexity by removing poor candidates before it aggregates them into larger structures. More important, we know the system misses some rooftops because it never generates their components. We suspect that, by moving constraints on candidate generation into the filtering step, which can then be learned, we can reduce such omissions without increasing computational cost.

In the longer term, we hope to modify BUDDS to incorporate our software for learning and interactive labeling. Our intent is to support incremental improvement of the vision system as it encounters new images that cause it difficulty. When noting that the system fails to recognize a building or infers an imaginary one, the user would interactively trace its reasoning to assign blame to some particular inference step, then provide the correct labels for use in learning at that decision point. Such knowledge refinement would seem necessary to maintain any large-scale system for image understanding.

In summary, our studies have shown that machine learning can improve the behavior of an existing image understanding system, and that interactive labeling can play a important role in that process. We also showed encouraging results about the ability of learning to generalize across images and about the potential for active learning to reduce the effort of labeling training cases. Considerable work remains to be done, including the extension of interactive labeling and learning to other stages of BUDDS, but that does not minimize the progress to date. We anticipate that machine learning will come to occupy an increasingly central place in the development and maintenance of software for computer vision.

Acknowledgements

We would like to thank Ram Nevatia, Andres Huertas, and Andy Lin for providing the images and data used for experimentation and for giving valuable comments and advice. We also thank Winton Davies for useful comments on the paper. This work was conducted at the Institute for the Study of Learning and Expertise and in the Computational Learning Laboratory, Center for the Study of Language and Information, at Stanford University. The review of the naive Bayesian classifier in Section 2 repeats some material from Langley and Sage (1994).

References

Bowyer, K. W., Hall, L. O., Langley, P., Bhanu, B., & Draper, B. A. (1994). Report of the AAAI Fall Symposium on Machine Learning and Computer Vision: What, Why and How. *Proceedings of the Image Understanding Workshop* (pp. 727–731). Monterrey, CA. Morgan Kaufmann.

Bradley, A. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*, 1145–1159.

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, *3*, 261–284.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*, *4*, 129–145.

Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, *29*, 103–130.

Draper, B. (1996). Learning grouping strategies for 2D and 3D object recognition. *Proceedings of the Image Understanding Workshop* (pp. 1447–1454). Palm Springs, CA: Morgan Kaufmann.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: John Wiley & Sons.

Firschein, O., & Strat, T. (Eds.). (1997). *RADIUS: Image understanding for imagery intelligence*. San Francisco: Morgan Kaufmann.

John, G. H., & Langley, P. (1995). Estimating continuous distributions in Bayesian networks. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence* (pp. 338–345). Montreal, Quebec: Morgan Kaufmann.

Kibler, D., & Langley, P. (1988). Machine learning as an experimental science. *Proceedings of the Third European Working Session on Learning* (pp. 81–92). Glasgow: Pittman.

Kononenko, I. (1990). Comparison of inductive and naive Bayesian learning approaches to automatic knowledge acquisition. In B. Wielinga et al. (Eds.), *Current trends in knowledge acquisition*. Amsterdam: IOS Press.

Langley, P., Iba, W., & Thompson, K. (1992). An analysis of Bayesian classifiers. *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 223–228). San Jose, CA: AAAI.

Langley, P., & Sage, S. (1994). Induction of selective Bayesian classifiers. *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence* (pp. 399–406). Seattle: Morgan Kaufmann.

Lin, C., & Nevatia, R. (1996). Building detection and description from monocular aerial images. *Proceedings of the Image Understanding Workshop* (pp. 461–468). San Francisco: Morgan Kaufmann.

Maloof, M. A., Langley, P., Sage, S., & Binford, T. (1997). Learning to detect rooftops in aerial images. *Proceedings of the Image Understanding Workshop* (pp. 835–845). New Orleans: Morgan Kaufmann.

Provan, G., Langley, P., & Binford, T. O. (1996). Probabilistic learning of three-dimensional object models. *Proceedings of the Image Understanding Workshop* (pp. 1403–1413). Palm Springs, CA: Morgan Kaufmann.

Provan, G. M., & Singh, M. (1996). Data mining and model simplicity: A case study in diagnosis. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 57–62). Portland, OR: AAAI Press.

Provost, F., & Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining* (pp. 43–48). Newport Beach, CA: AAAI Press.

Swets, J. (1988). Measuring the accuracy of diagnostic systems. *Science*, *240*, 1285–1293.