Towards Automatic Discovery of Object Categories

M. Weber[†] M. Welling[‡] P. Perona[†‡§]

[†]Dept. of Computation and Neural Systems
[‡]Dept. of Electrical Engineering
California Institute of Technology
Pasadena, CA 91125, U.S.A.
{mweber,welling,perona}@vision.caltech.edu

[§]Università di Padova
Italy

Abstract

We propose a method to learn heterogeneous models of object classes for visual recognition. The training images contain a preponderance of clutter and learning is unsupervised. Our models represent objects as probabilistic constellations of rigid parts (features). The variability within a class is represented by a joint probability density function on the shape of the constellation and the appearance of the parts. Our method automatically identifies distinctive features in the training set. The set of model parameters is then learned using expectation maximization (see the companion paper [11] for details). When trained on different, unlabeled and unsegmented views of a class of objects, each component of the mixture model can adapt to represent a subset of the views. Similarly, different component models can also "specialize" on sub-classes of an object class. Experiments on images of human heads, leaves from different species of trees, and motor-cars demonstrate that the method works well over a wide variety of objects.

1 Introduction and Related Work

When we look at the first few images of Fig. 1 we see 'cars.' We may be unable to name a specific car model, but we can easily recognize a car as such. 'Cars' is an abstract object class that we have constructed through experience and which we may use for labelling previously unseen objects.

We are interested in the problem of learning object classes from unsupervised visual experience: given the images of Fig. 1, can the visual concept of 'car' be learned automatically? The main challenge we face is identifying the relevant objects amongst a preponderant amount of irrelevant clutter. Unfortunately, we do not even know in advance what to look for: the concept of 'car' will emerge from the process—for all that we know our training images could depict shoes or some type of deep-sea fish.

This problem has not yet been tackled in the computer vision literature. Traditionally, object recognition starts with a training set where the salient parts of each object are in rough geometrical correspondence. This is either obtained by segmentation and warping/alignment of each object's picture, or by direct manual identification of the main features. In either case intervention of an operator and/or controlled imaging conditions are required [9, 2, 3, 4, 6, 1].

In a companion paper we propose a method [11] for learning object classes from cluttered images without supervision. The models that we use are Gaussian densities, appropriate for reasonably homogeneous training examples. We expect that in most practical applications the data will be multi-modal; variability in the data could be due to the presence of objects from several distinct classes, a single object class being seen from different viewpoints, or visually distinct sub-classes of the same object. In this paper we extend our method to handle such diversity.

2 Overview of the Approach

We model object classes following the work of Burl et al. [3]. An object is composed of *parts* and *shape*, where 'parts' are image patches which may be detected and characterized by appropriate detectors, and 'shape' describes the geometry of the mutual position of the parts in a way that is invariant with respect to rigid and, possibly, affine transformations [10]. A joint probability density on part appearance and shape models the object class. In this paper we show that, using *mixtures* of probabilistic models of this type, it is possible to model object classes exhibiting a large degree of variability.

Object detection is performed by first running part detectors on the image, thus obtaining a set of candidate part locations. The second stage consists of forming likely object hypotheses, i.e. constellations of appropriate parts (e.g. eyes, nose, mouth, ears); both complete and partial constellations are considered, in order to allow for partial occlusion. The third stage consists of using the object's joint probability density for either calculating the likelihood that any hypothesis arises from an object (object detection), or the likelihood that one specific hypothesis arises from an object (object localization).

In order to train a model we need to decide on the key

Figure 1. Some of the images contain instances of object classes (cars and leaves), others contain only "background." How can a machine learn to recognize these objects without any further information?

parts of the object, select corresponding parts (e.g. eyes, nose etc.) on a number of training images, and lastly we need to estimate the joint probability density function on part appearance and shape. Burl et al. [2] perform the first and second act by hand, while our methods automate the first and second steps as well.

Our technique for selecting potentially informative features/regions is composed of two steps: first highly textured regions are detected in the training images by means of a standard 'interest operator' or keypoint detector. An unsupervised clustering step favoring large clusters will tend to select features that correspond to the objects of interest rather than the background. Appropriate feature detectors may be trained using these clusters. This method has been described in detail in [11] and will not be discussed here.

In order to learn an object class model from a set of training images, we need to decide on a small subset of the parts selected by the feature selection method and we then need to learn the parameters of the statistical model. We solve both problems by iteratively trying out promising subsets of parts. During each iteration, the model parameters are estimated using *expectation maximization* (EM). At the end of each iteration, the detection performance of the model is evaluated using a validation data set. Based on the performance, a part in the model might be exchanged against a more promising one.

The best performing model generated in such fashion is in the end selected as 'the model'.

Outline of the Paper

In Section 3 we review the theory underlying our probabilistic object model before we introduce the extension to mixture models. In Section 4 experimental results are provided for three datasets: cars, leaves and human heads. Explicit update rules for mixture model learning are given in the appendix.

3 Mixtures of Probabilistic Object Models

In this section, we first give a brief description of our basic object model before introducing the extension to mixture models. We then provide detailed definitions of the different model components.

3.1 Basic Generative Model

We model objects as collections of rigid parts [3]. During recognition, each part type is detected by a corresponding detector. After the part detection stage, an entire image is thus reduced to a collection of parts as well. Some of those parts might correspond to an instance of the target object class (the *foreground*), while others stem from background clutter or are simply false detections (the *background*). Throughout this paper, the only information associated with an object part is its position in the image and its identity or part *type*. We assume that there are T different types of parts. The positions of all parts extracted from one image are collected in a "matrix,"

$$X^o = \left(\begin{array}{c} x_{11}x_{12}, \ldots, x_{1N_1} \\ x_{21}x_{22}, \ldots, x_{2N_2} \\ \vdots \\ x_{T1}x_{T2}, \ldots, x_{TN_T} \end{array} \right.,$$

where the superscript 'o' indicates that these positions are *observeable* in an image, as opposed to being unobserveable or *missing*, which will be denoted by 'm.' Thus, the t^{th} row contains the locations of detections of part type t, where every entry, x_{ij}, is a two-dimensional vector. If we now assume that an object is composed of F different parts,[1] we also need to indicate which parts in X^o correspond to the foreground (the object of interest). For this we use the vector $\mathbf{h}$, a set of indices, with $h_i = j$, $j > 0$, indicating that point x_{ij} is a foreground point. If an object part is not contained in X^o, because it is occluded or otherwise undetected, the corresponding entry in $\mathbf{h}$ will be zero. When presented with an unseen image, we do not know which parts correspond to the foregound. Therefore, $\mathbf{h}$ is not observeable and we will treat it as *hidden* or *missing* data. We call $\mathbf{h}$ a *hypothesis*, since it expresses which parts of X^o are hypothesized to belong to the foreground object. It is also convenient to represent the positions of any unobserved object parts in a separate vector $\mathbf{x}^m$ which is, of course, hidden as well. We can then define a generative probabilistic model through the joint probability density

$$p(X^o, \mathbf{x}^m, \mathbf{h}). \tag{1}$$

Before we give a detailed definition of this density, we explain how we construct mixtures based on this model.

3.2 Mixture Model

We assume that a mixture model consists of Ω different components. Each component is a complete model of the type introduced in the previous section. Our learning and detection algorithms are set up such that, during recognition, these mixture components will "compete" for the explanation of an input image. Thus, an image containing an instance of an object class can only be generated or explained by a single mixture component. We follow, in this respect, the spirit of the "mixture of experts" framework introduced in [7]. However, we do not use a true mixture of experts model, in which a separate *gating network* is typically used to divide the input space into regions of responsibility for each expert.

One can also imagine an approach under which model components "collaborate." Each component could then explain only a subset of the object parts in the entire image and several experts need to be "active" for a given image. For a more detailed discussion of competitive vs. collaborative learning see [8].

[1] To simplify notation, we only consider the case where $F = T$. The extension to the general case ($F \geq T$) is straightforward.

We can write our complete model as

$$p(X^o, \mathbf{x}^m, \mathbf{h}) = \sum_{\omega=1}^{\Omega} p(X^o, \mathbf{x}^m, \mathbf{h}|\omega)p(\omega).$$

The conditional density $p(X^o, \mathbf{x}^m, \mathbf{h}|\omega)$ represents component model ω. Every component model is based on its own set of parameters, explaining how the parts in X^o can be arranged spatially in an image. However, not every component needs to make use of all part types in order to model the foreground. Hence, under certain component models, part candidates of certain types can only be accounted for as background clutter.

The probabilities $p(\omega)$ express our a priori expectation of explaining the data with component model ω.

3.2.1 Model Details

In order to provide a detailed parametrization of (1), we introduce two auxiliary variables, $\mathbf{b}$ and $\mathbf{n}$. The binary vector $\mathbf{b}$ encodes information about which features have been detected and which have been missed or occluded. Hence, $b_f = 1$ if $h_f > 0$ and $b_f = 0$ otherwise. The variable $\mathbf{n}$ is also a vector, where n_t shall denote the number of *background* candidates included in the t^{th} row of X^o. Since both variables are completely determined by $\mathbf{h}$ and the size of X^o, we have $p(X^o, \mathbf{x}^m, \mathbf{h}) = p(X^o, \mathbf{x}^m, \mathbf{h}, \mathbf{n}, \mathbf{b})$. We can now decompose in the following way

$$\begin{aligned} p(X^o, \mathbf{x}^m, \mathbf{h}, \mathbf{n}, \mathbf{b}, \omega) = p(X^o, \mathbf{x}^m|\mathbf{h}, \mathbf{n}, \omega) \times \\ p(\mathbf{h}|\mathbf{n}, \mathbf{b}, \omega)\, p(\mathbf{n})\, p(\mathbf{b}|\omega)\, p(\omega). \end{aligned} \tag{2}$$

The probability density over the number of background detections can be modeled by a Poisson distribution,

$$p(\mathbf{n}) = \prod_{t=1}^{T} \frac{1}{n_t!} (M_t)^{n_t} e^{-M_t},$$

where M_t is the average number of background detections of type t per image. Note that the background statistics are assumed independent of the model component ω.

Depending on the number of features, F, we can model the probability $p(\mathbf{b}|\omega)$ either as an explicit table (of length 2^F) of joint probabilities, or, if F is large, as F independent probabilities, governing the presence of an individual model feature.

It is important to realize that the probability $p(\mathbf{b}|\omega)$ for a vector $\mathbf{b}$ which is not consistent with model ω (e.g. it hypothesizes a feature type to be detected which is not used for the foreground in this component model) is defined to be zero.

The density $p(\mathbf{h}|\mathbf{n}, \mathbf{b}, \omega)$ is modeled by,

$$p(\mathbf{h}|\mathbf{n}, \mathbf{b}, \omega) = \begin{cases} \frac{1}{\prod_{f=1}^{F} N_f^{b_f}} & \mathbf{h} \in \mathcal{H}(\mathbf{b}, \mathbf{n}, \omega) \\ 0 & \text{other } \mathbf{h} \end{cases}$$

where $\mathcal{H}(\mathbf{b}, \mathbf{n}, \omega)$ denotes the set of all hypotheses consistent with $\mathbf{b}$, $\mathbf{n}$ and ω, and N_f denotes the total number of detections of the type of feature f. This expresses the fact that all consistent hypotheses, the number of which is $\prod_{f=1}^{F} N_f^{b_f}$, are equally likely in the absence of information on the feature locations.

Finally, we use

$$p(X^o, \mathbf{x}^m|\mathbf{h}, \mathbf{n}, \omega) = p_{\mathbf{fg}}(\mathbf{x}^o, \mathbf{x}^m|\omega)\, p_{\mathbf{bg}}(\mathbf{x}_{bg}),$$

where we defined $[\mathbf{x}^o\, \mathbf{x}^m]$ as the coordinates of all foreground detections (observed and missing) and $\mathbf{x}_{bg}$ as the coordinates of the background detections. In our experiments, $p_{\mathbf{fg}}(\mathbf{x}^o, \mathbf{x}^m|\omega)$ is modeled as a joint Gaussian with mean μ_ω and covariance Σ_ω. The positions of the background detections are modeled by a uniform density,

$$p_{\mathbf{bg}}(\mathbf{x}_{bg}) = \prod_{t=1}^{T} \frac{1}{A^{n_t}},$$

where A is the total image area.

3.3 Classification

Throughout the experiments presented here, our objective is to classify images into the classes "object present" (class $\mathcal{C}_1$) and "object absent" (class $\mathcal{C}_0$). Given the observed data, X^o, the optimal decision—minimizing the expected total classification error—is made by choosing the class with maximum a posteriori probability (MAP approach, see e.g. [5]). It is therefore convenient to consider the following ratio,

$$\frac{p(\mathcal{C}_1|X^o)}{p(\mathcal{C}_0|X^o)} \propto \frac{\sum_{\mathbf{h},\omega} p(X^o, \mathbf{h}, \omega|\mathcal{C}_1)}{\sum_{\omega} p(X^o, \mathbf{h}_0, \omega|\mathcal{C}_0)}, \tag{3}$$

where $\mathbf{h}_0$ denotes the *null hypothesis* which explains all features as background noise. The sum in the numerator includes all hypotheses, also the null hypothesis, since the object could be present but remain undetected by any feature detector. In the denominator, the only consistent hypothesis to explain "object absent" is the null hypothesis.

Although we are here concerned with classification only, our framework is by no means restricted to this problem. For instance, object localization is possible by identifying those foreground features in an image, which have the highest probability of corresponding to an occurrence of the target object.

3.4 Model Learning

A detailed description of our learning method can be found in [11] for the case of single-component models. We do not have space to reproduce the method here. However, explicit update rules of our learning algorithm for the case of mixture models have been included in the appendix.

4 Experiments

We chose to validate our method under the classification problem defined in Section 3.3, using three different data sets: images of rear and side views of cars, images of human heads, taken over a range of viewing directions from frontal to profile, and images of a sample of leaves from three different species of trees.

Since the objects to be learned were in a different, unknown location in every training image, we employed a translation invariant extension of our method (see [2] for a discussion on invariance to planar transformations). We tried, however, to assert that the objects had the same size in all images, since we have not yet implemented scale invariant learning.

4.1 Training and Test Images

Human Heads: In order to produce a large set of images with different but known head orientations, a sufficient number of subjects, as well as different, cluttered backgrounds, we resorted to a synthetic blending of head images with background scenes. Subjects were photographed in front of a blue background, facing 7 different viewing directions ($0°(\mathrm{frontal}), 15°, \ldots, 90°(\mathrm{profile})$). The background was then subtracted from the images (which were converted to grayscale) and replaced with entirely white or black regions to produce training images, as well as with random scenes to produce test images (see Fig. 2).[2] Similar scenes, without superimposed heads, were used as negative examples. We used 10 subjects for training and 10 different subjects for testing. Four pictures were taken of each subject at every viewing direction. Subjects were asked to keep a straight face and head orientation for the first two images. For the other two they were allowed to tilt their head, rotate it within the plane of view and perform facial expressions of their choice. The resolution of the training images was such that the distance from top of the head to chin spanned about 80 pixels.

The set of background scenes contained 350 pictures of landscapes, outdoor scenes of buildings, as well as indoor scenes of office and laboratory environments. This set was divided into two sets of 175 pictures each for training and testing.

Leaves: We used completely unlabeled and unsegmented images throughout the leaf (and car) experiments. We collected six leaves from each of three different species of trees and photographed each leaf 10 times in front of an arbitrary background, producing 180 images (see Fig. 1 bottom). Another 150 images from the same indoor laboratory environment were taken as negative examples. As in the other experiments, both sets were divided into an independent training and test set.

Cars: We took 300 images (150 rear views and 150 side views) on public streets and parking lots showing cars in front of a cluttered background (Fig. 1 bottom). We also took 200 images of background scenes from the same environment. We photographed vehicles of different sizes, colors and types, such as sedans, sport utility vehicles, and pick-up trucks.

[2] Although we have demonstrated that our method can learn models of human heads from entirely unlabeled cluttered scenes, we made use of this segmentation step in order to speed up the training process. Eliminating the background helped by reducing the number of features to be tried and by speeding up the convergence of the EM stage due to the absence of background detections.

Figure 2. Examples from the human head database. Segmented heads are superimposed on black and white backgrounds for training (top), and on random background scenes for testing (center). For half of the images, subjects were allowed to make facial expressions and to tilt or rotate their head around the camera axis. On the bottom we show random background scenes used as negative test examples.

The car and leaf images were high-pass filtered in order to promote invariance with respect to different colors of cars and different lighting conditions. All images were taken with a digital camera; they were converted to a grayscale representation and downsampled to a resolution of 240×160 pixels. No images were discarded by hand prior to the experiments.

The EM model learning was carried out on the positive training images. In order to optimize for the model configuration, the classification performance of the model was evaluated after each pass of EM, using the positive and negative training images. No individual car/head/leave was present in the test *and* training set at the same time. The final model was tested on the previously unseen target and background images.

4.2 Automatically Selected Parts

Parts were automatically selected according to the procedure described in [11]. The Förstner interest operator was applied to the positive training images. We then extracted patches of size 11×11 around the points of interest and clustered those using vector quantization. A different set of patterns was produced for each object class (Fig. 3).

Figure 3. A sample of the patterns obtained from our feature selection method (based on k-means clustering) is shown for cars (left), heads (center) and leaves (right). The leaf and car images were high-pass filtered before feature selection. The total number of patterns selected was 175 for heads (across a 90° range of viewing directions), 89 for cars (across two orthogonal viewing directions) and 81 for leaves.

4.3 Results

Classification Performance

To assess classification performance, we learned models with two mixture components of three features each.

We found the EM algorithm to converge in about 100 iterations. One iteration took about $200\mathrm{ms}$ using a Matlab implementation with subroutines written in 'C'. The entire training process took several hours on a PC with $450\mathrm{MHz}$ Pentium II processor.

Rather than classifying every image by applying a fixed decision threshold according to Equation 3, we computed receiver operating characteristics (ROCs) based on the ratio of posterior probabilities. In order to reduce sensitivity to noise due to the limited number of training images, we used the area under the ROC curve as a measure of the classification performance driving the optimization of the model configuration. In Figure 5, we report total classification errors which are computed by choosing a decision threshold such that the same error rate on foreground and background images is obtained. The differences between training and test errors in the Cars and Leaves experiments suggest that we have experienced a certain degree of *overfitting*. The asymptotical classification error for an infinitely large training set lies somewhere between our observed training and test error.

One might also be surprised by the limited improvement of the mixture models over the single component models. The reason for this is the remarkable performance of single component models which, e.g., in the case of heads are often able to work realiably over a large range of viewing directions (see also Fig. 4).

Examples of misclassifications are shown in Fig. 7.

Tuning of Mixture Components

For this experiment, we learned models with two (cars), three (leaves), and four (heads) components of three features each, exhausting the entire set of available data. We then investigated the tuning properties of the individual component models by assigning every training image to that particular component, ω_k, for which the "responsibility," $p(\omega_k|X^o)$, was maximal. If no component had a responsibility of more than 80%, we assigned the corresponding image to an "undecided" category. We were able to observe clear tuning characterstics for all three data sets. In the case

	Cars		Heads			Leaves		
# comp.	1	2	1	2	$4^\star$	1	2	$3^\star$
train	17%	12%	14%	13%	5%	8%	8%	6%
test	18%	16%	14%	13%	n/a	16%	16%	n/a
ZFA-DR	46%	54%	57%	61%	n/a	59%	60%	n/a

Figure 5. Test and training performance for the detection experiments. We show the total misclassification error, obtained at the decision treshold where an equal number of positive and negative images is misclassified. We also report the detection rate at zero false alarms (ZFA-DR). This performance is important for applications such as digital library searches. *Models with three (leaves) and four (heads) components were trained on *all* available data for the tuning experiments.

Figure 4. Analysis of a four-component mixture model with three features per component. This model was trained on the *entire* set of available sata. We show the three parts selected for each component (left) by our training algorithm. Ellipses indicating a three-standard deviation distance from the mean part positions according to the Gaussian pdf governing the foreground part positions are shown (center). They have been superimposed on an arbitrary image for which the corresponding component had a high "responsibility." Since our models are translation invariant, the models have been aligned with the image by hand for illustrative purposes. On the right we show a second arbitrarily selected image, also with a high responsibility of the corresponding component. Detected candidate part locations are shown *only* for the three part types of the respective component (○ = 'A', □ = 'B', ◇ = 'C'). Note that the model components are able to represent the heads over a fairly large range of viewing directions—about $45°$ in this case—mainly due to the stability of the chosen features. This effect was even more pronounced when we trained models with only two components over a $90°$ viewing range. In this case, single components were often able to detect a head over the entire range. Hence, the dataset could be considered not difficult enough for the mixture models.

of leaves, only a model with three components showed tuning characteristics, but not with two. In the head experiment we only observed tuning for models with at least four components. Figure 6 shows histograms illustrating these findings. In both cases, each model component is between two and four times as likely to detect a particular view or sub-class than any other.

Separately Trained Components

In another experiment, we trained single component models separately on labeled side and back views of cars. We repeated this experiments 10 times, to avoid local extrema and obtained an average performance of 82% correct for back views and 86% correct for side views. We then merged those models into two-component models, choosing $p(\omega) = 0.5$ for both components. The performance of the resulting models was only 76% correct on average and, therefore, significantly worse than those of two-component models trained on unlabeled data with our algorithm for mixture model learning. Hence, it appears that training components *simultaneously* is vital to exploit the benefits of our mixture models.

5 Discussion and Future Work

We have presented ideas for learning mixture models of objects from inhomogeneous training images in an unsupervised setting. A set of unsegmented and unlabeled images (in the case of leaves and cars) containing examples of objects amongst clutter is supplied; our algorithm automatically selects distinctive features of the object class, and learns the joint probability density function encoding the object's appearance. This allows the automatic construction of an efficient object detector which is robust to clutter and occlusion.

We have demonstrated that our model learning algorithm works successfully on three different data sets: human heads viewed over a $90°$ range of viewing directions, cars seen from the rear and the side, and images of three different species of leaves. In the case of heads, discrimination of images containing the desired object vs. background images approaches 90% correct with simple models composed of 2 components with 3 features each. Performance on cars is 84% correct and on leaves we obtained 85% correctly classified test images. The detection rate at zero false alarms was consistently above 50%–and often significantly higher–which is promising for content-based searching of

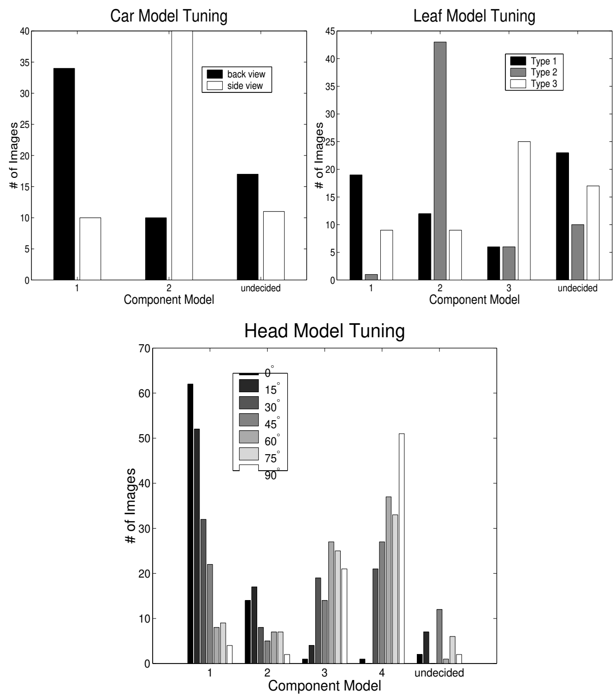

Figure 6. Histograms showing tuning properties of the mixture models models. For every component model the number of images for which it is "responsible" is plotted. The right-most group of bars show "undecided" images, where no model reached a value of more than 80% probability. The number of components was two (cars), three (leaves) and four (heads). Head models were found to be partially tuned to head orientation but also to small remaining differences in head sizes in our data sets.

large image databases. We demonstrated that our method is capable of identifying different sub-classes amongst the unlabeled training images. These sub-classes are due to different viewpoints (cars, heads) or genuinely different subcategories (leaves).

While training is computationally expensive, requiring several hours of computer time on training sets composed of approximately 100 images, detection is efficient, requiring less than half a second in our hybrid 'C'-Matlab implementation. This suggests that training should be seen as an off-line process, while detection may be implemented in real-time.

Many aspects of our implementation are suboptimal and susceptible of improvement. To list a few: we implemented the part detectors using normalized correlation. More sophisticated detection algorithms, involving multiscale image processing, multiorientation-multiresolution filters, neural network etc. should be considered and tested. Moreover, in our current implementation only part of the information supplied by the detectors, i.e. the candidate feature's location, is used; the scale and orientation of the image patch, parameters describing the appearance of the patch, as well as its likelihood, should be incorporated. Our interest operator as well as the unsupervised clustering of the features have not been optimized in any respect; the choice of the algorithms deserves further scrutiny as well. An important aspect where our implementation falls short of generality is invariance: the models we learned and tested are translation invariant, but not rotation, scale or affine invariant. There is no conceptual limit to this generalization, although the computational cost grows significantly. We also need to explore a principled way of deciding on the number of mixture components. This could be done by monitoring the generalization error as a function of the number of components. Finally, the possibility to trade-off the number of components against the number of parts in every component should be investigated.

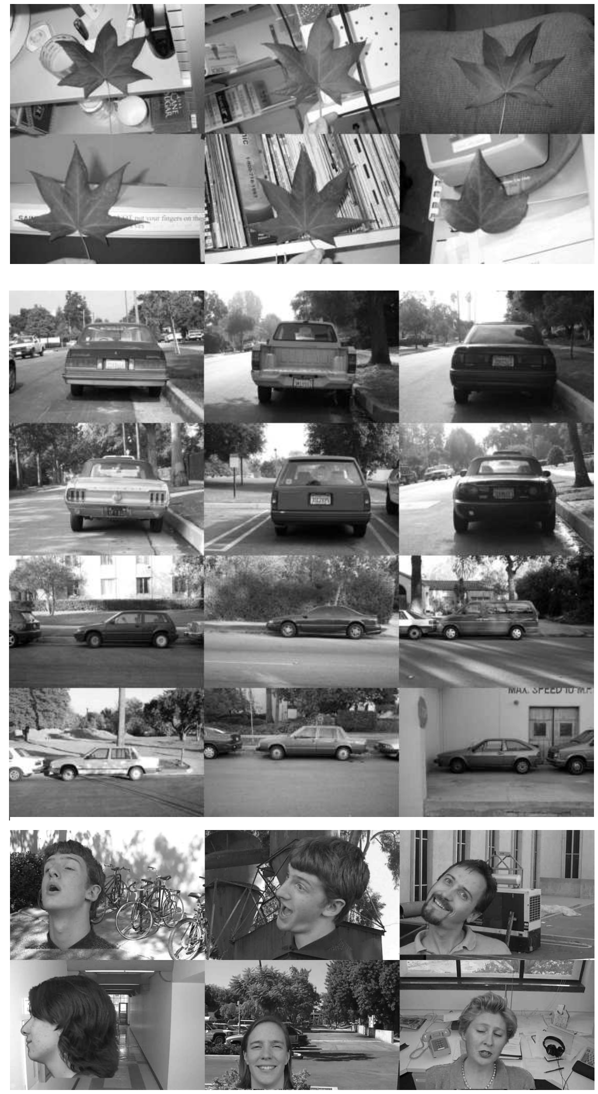

Figure 7. Examples of misclassified images from the three test sets.

Acknowledgements

Funded by the NSF Engineering Research Center for Neuromorphic Systems Engineering (CNSE) at Caltech (NSF9402726), and an NSF National Young Investigator Award to P.P. (NSF9457618). M. Welling was supported by the Sloan Foundation.

We are also very grateful to Rob Fergus, Catharine Stebbins and Justin Smith for helping with collecting the databases. We are grateful to Thomas Leung, Mike Burl, Jitendra Malik and David Forsyth for many helpful comments.

A M-step and E-step

For details on the derivation of our basic learning algorithm the reader is refered to [11], where we derive the update rules for a single component model.

We employ the EM algorithm to optimize the likelihood of the observed data,

$$L(X^o|\theta) = \sum_{i=1}^{I} \log \sum_{\mathbf{h}_i,\omega_i} \int p(X_i^o, \mathbf{x}_i^m, \mathbf{h}_i, \omega_i|\theta)\, d\mathbf{x}_i^m,$$

with respect to the model parameters, $\theta = \{\mu_\omega,\ \Sigma_\omega,\ p(\mathbf{b}|\omega),\ \mathbf{M},\ p(\omega)\}$. Since this is difficult to achieve analytically, EM iteratively maximizes a sequence of cost functions,

$$Q(\tilde{\theta}|\theta) = \sum_{i=1}^{I} E[\log p(X_i^o, \mathbf{x}_i^m, \mathbf{h}_i, \omega_i|\tilde{\theta})].$$

Taking the derivative of $Q(\tilde{\theta}|\theta)$ with respect to the parameters and equating to zero produces the following update rules,

$$\tilde{\mu}_\omega = \frac{\sum_{i=1}^{I} p(\omega|X_i^o)\, E_\omega[\mathbf{z}_i]}{\sum_{i=1}^{I} p(\omega|X_i^o)},$$

$$\tilde{\Sigma}_\omega = \frac{\sum_{i=1}^{I} p(\omega|X_i^o)\, E_\omega[\mathbf{z}_i\mathbf{z}_i^T]}{\sum_{i=1}^{I} p(\omega|X_i^o)} - \tilde{\mu}_\omega\tilde{\mu}_\omega^T,$$

$$\tilde{p}(\bar{\mathbf{b}}|\omega) = \frac{\sum_{i=1}^{I} p(\omega|X_i^o)\, E_\omega[\delta_{\mathbf{b}_i,\bar{\mathbf{b}}}]}{\sum_{i=1}^{I} p(\omega|X_i^o)},$$

$$\tilde{\mathbf{M}} = \frac{1}{N}\sum_{i=1}^{I}\sum_{\omega} p(\omega|X_i^o)\, E_\omega[\mathbf{n}_i],$$

$$\tilde{p}(\omega) = \frac{1}{N}\sum_{i=1}^{I} p(\omega|X_i^o),$$

where $\mathbf{z}^T = [\mathbf{x}^o\ \ \mathbf{x}^m]$ and $E_\omega[.]$ denotes taking the expectation with respect to the posterior density $p(\mathbf{h}_i, \mathbf{x}_i^m|X_i^o, \omega, \theta)$. These update rules constitute the M-step.

The E-step amounts to the calulation of the sufficient statistics $E_\omega[\mathbf{z}]$, $E_\omega[\mathbf{zz}]$, $E_\omega[\delta_{\mathbf{b},\bar{\mathbf{b}}}]$, $E_\omega[\mathbf{n}]$ and the posterior density

$$p(\omega|X_i^o) = \frac{p(X_i^o|\omega)\, p(\omega)}{\sum_\omega p(X_i^o|\omega)\, p(\omega)},$$

with

$$p(X_i^o|\omega) = \sum_{\mathbf{h}_i} \int p(\mathbf{h}_i, \mathbf{x}_i^m, X_i^o|\omega, \theta)\ d\mathbf{x}_i^m.$$

Explicitely, $p(X_i^o|\omega)$ is calculated as follows. Choose a hypothesis consistent with the observed data. Integrate out the missing data by eliminating the appropriate dimensions from the Gaussian foreground pdf. Calculate $\mathbf{b}(\mathbf{h})$ and $\mathbf{n}(\mathbf{h})$ and insert them into the joint density. Finally, sum over all possible hypotheses. The expectations of the statistics are calculated in a similar fashion. $E_\omega[\delta_{\mathbf{b},\bar{\mathbf{b}}}]$ is calculated by summing only over those hypotheses consistent with $\bar{\mathbf{b}}$ in the numerator and dividing by $p(X_i^o|\omega)$. Similarly, $E_\omega[\mathbf{n}]$ is calculated by averaging $\mathbf{n}(\mathbf{h})$ over all hypotheses. For $E_\omega[\mathbf{z}] = (\mathbf{x}^o\ E_\omega[\mathbf{x}^m])$ we need

$$\int \mathbf{x}^m\ G(\mathbf{z}|\mu_\omega, \Sigma_\omega)\ d\mathbf{x}^m = \mu_\omega^m + \Sigma_\omega^{mo}\Sigma_\omega^{oo-1}(\mathbf{x}^o - \mu_\omega^o),$$

where we defined $\mu_\omega = (\mu_\omega^o\ \mu_\omega^m)$ and a similar decomposition for Σ_ω. For the calculation of $E_\omega[\mathbf{z}\mathbf{z}^T]$ we need the following result

$$E[\mathbf{x}^m\mathbf{x}^{mT}] = \Sigma_\omega^{mm} - \Sigma_\omega^{mo}\Sigma_\omega^{oo-1}\Sigma_\omega^{moT} + E_\omega[\mathbf{x}^m]E_\omega[\mathbf{x}^m]^T.$$

References

[1] Y. Amit. A neural network architecture for visual selection. Available at http://galton.uchicago.edu/˜amit, November 1998.

[2] M. Burl, T. Leung, and P. Perona. “Recognition of Planar Object Classes”. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, 1996.

[3] M. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *Proc. 5th Europ. Conf. Comput. Vision, H. Burkhardt and B. Neumann (Eds.), LNCS-Series Vol. 1406–1407, Springer-Verlag*, pages 628–641, 1998.

[4] T. Cootes and C. Taylor. “Locating Objects of Varying Shape Using Statistical Feature Detectors”. In *European Conf. on Computer Vision*, pages 465–474, 1996.

[5] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, Inc., 1973.

[6] G. Edwards, T.F.Cootes, and C.J.Taylor. Face recognition using active appearance models. In *Proc. 5th Europ. Conf. Comput. Vision, H. Burkhardt and B. Neumann (Eds.), LNCS-Series Vol. 1406–1407, Springer-Verlag*, pages 581–595, 1998.

[7] J. Hampshire and A. Waibel. The meta-pi network: Building distributed knowledge representations for robust pattern recognition. Technical Report CMU-CA-89-166, Carnegie Mellon University, Pittsburgh, PA, 1989.

[8] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

[9] M. Lades, J. Vorbruggen, J. Buhmann, J. Lange, C. v.d. Malsburg, R. Wurtz, and W. Konen. “Distortion Invariant Object Recognition in the Dynamic Link Architecture”. *IEEE Trans. Comput.*, 42(3):300–311, Mar 1993.

[10] T. Leung, M. Burl, and P. Perona. Probabilistic affine invariants for recognition. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 678–684, 1998.

[11] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *Proc. 6th Europ. Conf. Comput. Vision*, Dublin, Ireland, to appear June 2000.