Unsupervised Learning of Models for Visual Object Class Recognition

M. Weber† M. Welling‡ P. Perona†‡

†Dept. of Computation and Neural Systems
‡Dept. of Electrical Engineering
California Institute of Technology
Pasadena, CA 91125

Abstract

We present a method to learn object class models for the purpose of object recognition. We focus on a particular type of model where objects are represented as constellations of rigid features (parts). The variability within a class is represented by a joint probability density function (pdf) on the shape of the constellation and the output of feature detectors. The pdf may be estimated from training data once a model structure (type and number of features) has been specified. The method automatically identifies distinctive features in the training set and learns the statistical shape model. It is assumed that a set of generic feature detectors is available for the learning algorithm to choose from. The entire set of model parameters is learned using expectation maximization.

1 Introduction and Related Work

We are interested in the problem of recognizing members of object classes, where we define an *object class* as a collection of objects which share characteristic parts or *features* that are visually similar and occur in similar spatial configurations.

Several solutions to this problem have recently been proposed. In particular, Amit and Geman have developed a method for visual selection which learns a hierarchical model with a simple type of feature detector (edge elements) at the lowest level [1]. The method assumes that training images are registered with respect to a reference grid. After an exhaustive search through all possbile local feature detectors, a global model is built, under which shape variability is encoded in the form of small regions in which local features can move freely.

We have proposed a statistical model in which shape variability is modeled in a probabilistic setting using Dryden-Mardia shape space densities [2, 3].

One should also mention the active appearance models of Taylor et al. [4, 6] which model global deformations using Eigenspace methods, as well as the Dynamic Link Architecture of v. der Malsburg and colleagues, who consider deformation energy of a grid that links landmark points on the surface of objects [7]. Also Yuille has proposed a recognition method based on gradient descent on a deformation energy function in [9].

In this paper, we are concerned with several shortcomings of the above methods, related, in particular, to the training of such models. Learning models like the ones described above, typically requires a supervised stage in which the following questions need to be answered:

- Which objects are to be recognized?
- Which object features are distinctive and stable?
- What are the parameters of the global geometry or *shape* model that best describe the training data?

By the first question we mean that target objects must be identified in training images, be it by selection of points or regions on the surface of the objects or by segmenting the objects from the background. The question of which object features are distinctive as well as stable across different objects of the same class is often left unanswered or approached by heuristics or procedures of hand-selection by a trained human operator. However, it is generally not established, that features that appear distinctive to the human observer will also lend themselves to successful detection by a machine. A notable exception represents the work by Walker et al. who address this problem in [8], albeit outside the realm of statistical shape models. The third problem is commonly solved through some optimization technique.

We propose in this paper a method which addresses all three problems at once. A compelling reason to treat these problems jointly, is the existing trade-off between localizability and distinctiveness. A very distinctive feature can be a strong cue, even if it appears in an arbitrary location on the surface of an object—think e.g. of a manufacturer's logo on a car. On the other hand, a less distinct feature can only contribute information if it occurs in a stable *spatial relationship* trelative to other features.

2 Approach

We assume that instances of an object class are described through a characteristic set of features (parts), which can occur at variable spatial locations. The parts are modeled as rigid, photometric patterns and the positional variability is represented using a probability density function over the point locations of the object features. Although our previously introduced object model allows to treat part constellations in *shape space*, and thus achieves pose invariant detection, the learning method presented here incorporates only invariance with respect to translation, not rotation and scale. We assume that the part positions, after translation has been eliminated by describing all feature positions relative to one reference feature, are well represented by a Gaussian pdf.

Formally, the detection problem amounts to deciding whether an instance of the class (*the foreground*) is present in the image, or whether the image contains only clutter (*background*). The problem of detecting the exact position of the object in the image can be treated under the same framework.

The two stages of our recognition method are an independent detection of object features, using different types of feature detectors, followed by a hypothesis evaluation stage. During the latter, small sets of candidate locations in the image, which have been labeled by the feature detectors, are evaluated as to their likelihood of actually corresponding to an instance of the object class.

We assume that the feature detector responses evoked by the background (the *false detections*) are distributed uniformly across the image and are independent of each other and of the foreground. This assumption is key to our method, which, in a nutshell, can be seen as fitting a mixture density model to the data, which consists of a joint Gaussian density over all foreground detector reponses and a uniform density over background responses.

The problem we are addressing in this paper is three-fold, although we propose a single algorithm to solve it.

Firstly, we would like to avoid segmentation or labeling of training images by hand. Ideally, the training algorithm should determine which parts of the image contain the objects of interest. This may sound contradictory, as detecting the objects was precisely the problem we set out to solve in the first place, but our experiments show that it is possible under certain conditions. Secondly, we want to select from a larger set of feature detectors the ones which are able to consistently identify a shared feature of the object class. Finally, also the global shape representation should be learned autonomously.

We assume that we have at our disposal T training images, identified by subscripts τ, and we suppose that a large number of simple generic detectors for features such as corners, dots or other points of high texture content is available.

The problem then becomes to select a subset of feature detectors, i.e. to choose a *model configuration*, and to learn the parameters of the global geometry model, with the final goal of maximizing recognition performace.

To solve this problem, we first apply all feature detectors to the training images and retain only the positions at which a given detector has maximal response (locally) on a given image. The only training data extracted from the images are these *candidate locations.* In order to achieve a high recognition performance, we then optimize for the *maximum likelihood* fit of our object model to the training data, using the EM algorithm.

2.1 Notation

For the remainder of this paper, we assume that a number, F, of feature detectors has been selected to be part of the model. Although an object could, in principle, exhibit several features of the same type, we assume for now, that every detector is included in the model at most once, to avoid further complication of the following presentation. The extension to multiple features of the same type is straightforward. As a further simplification, we derive the learning algorithm for a Gaussian density of part positions in the image. The necessary changes to obtain the translation ivariant version used in the experiments are minor. This is due to the fact that switching to a representation where feature positions are described relative to a reference feature, involves only a linear transformation and there is thus no need to depart from the class of Gaussian pdf's.

All information extracted from a training image, I_τ, is represented in the following matrix of feature candidate positions,

$$X^o = \left(\begin{array}{c} x_{11} x_{12}, \ldots, x_{1N_1} \\ x_{21} x_{22}, \ldots, x_{2N_2} \\ \vdots \\ x_{F1} x_{F2}, \ldots, x_{FN_F} \end{array} \right.\quad .$$

Every row contains the (two dimensional) locations of detections of feature type f.

We will use the following random variables, which represent either explicit or unobserved information,

$$\mathcal{D} = \{X^o_\tau, \mathbf{x}^m_\tau, \mathbf{n}_\tau, \mathbf{h}_\tau, \mathbf{b}_\tau\}.$$

Here, $\mathbf{h}$ denotes a set of indices, also called *a hypothesis*, for reasons to become evident later, indicating which points in X^o are from the foreground distribution (i.e. on the surface of the object), so that $h_i = j$, $j > 0$, means that point x_{ij} is a foreground point, while $j = 0$ indicates that the corresponding feature has not been included in X^o because it is has not been detected. We denote by $\mathbf{b}$ a binary vector which has entry $b_f = 1$ if $h_f > 0$ and zero otherwise. The positions of the occluded or missed foreground features are collected in a separate vector $\mathbf{x}^m$. The size of $\mathbf{x}^m$ varies between 0 and F depending on the number of unobserved features. Finally $\mathbf{n}_\tau$ denotes the number of background detections. All variables, except X^o are considered hidden.

2.2 The Model

For a given training image, I_τ, we can write the probability density function modeling the data as:

$$p(X^o_\tau, \mathbf{x}^m_\tau, \mathbf{h}_\tau, \mathbf{n}_\tau, \mathbf{b}_\tau) = \\ p(\mathbf{n}_\tau)\, p(\mathbf{b}_\tau)\, p(\mathbf{h}_\tau|\mathbf{n}_\tau, \mathbf{b}_\tau) \times \\ p(X^o_\tau, \mathbf{x}^m_\tau|\mathbf{h}_\tau, \mathbf{n}_\tau, \mathbf{b}_\tau).$$

The probability density over the number of background detections is modeled by a Poisson[1] distribution,

$$p(\mathbf{n}) = \prod_{f=1}^{F} \frac{1}{n_f!} (M_f)^{n_f} e^{-M_f},$$

where M_f is the average number of background detections per image. Admitting a different M_f for every feature allows us to model different detector statistics and, ultimately, to distinguish between more or less reliable detectors.

The probability $p(\mathbf{b})$ is modeled explicitely by a table of size 2^F which equals the number of possible binary vectors of length F. If F is large, the explicit probability mass-table of length 2^F might become unreasonably long. In that case we can assume independence between the feature detectors and model $p(\mathbf{b})$ by a product of independent densities,

$$p(\mathbf{b}) = \prod_{f=1}^{F} p(b_f).$$

The number of parameters reduces in that case from 2^F to F.

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

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

where $\mathcal{H}_\mathbf{b}$ denotes the set of all hypotheses consistent with $\mathbf{b}$ and $\mathbf{n}$, and N_f denotes the total number of detections of feature f.

[1] Given that we are dealing with a discrete set of pixel locations, a binomial distribution might seem more natural. However, since the Gaussian foreground density is defined over a continuum of part positions, the Poisson distribution is the natural counterpart for the background process.

Finally, we use

$$p(X^o, \mathbf{x}^m|\mathbf{h}, \mathbf{n}) = G(\mathbf{z}|\mu, \Sigma)\, U(\mathbf{x}_{bg}),$$

where we defined $\mathbf{z}^T = (\mathbf{x}^o\, \mathbf{x}^m)$ as the coordinates of the *hypothesized* foreground detections (observed and missing) and $\mathbf{x}_{bg}$ as the coordinates of the background detections. $G(\mathbf{z}|\mu, \Sigma)$ denotes a Gaussian with mean μ and covariance Σ. The positions of the background detections are modeled by a uniform density,

$$U(\mathbf{x}_{bg}) = \prod_{f=1}^{F} \frac{1}{A^{n_f}},$$

where A is the area covered by the image.

3 The EM algorithm

Since our detection method relies on the *maximum a posteriori probability* (MAP) principle, we will obtain maximum detection performance for those parameters which optimize the joint data likelihood. As mentioned above, we treat as missing data the $\mathbf{h}$, the $\mathbf{b}$ and the $\mathbf{n}$. The positions of unobserved foreground features, i.e. those corresponding to zero entries in $\mathbf{h}$, are collected in $\mathbf{x}^m$ and are, quite naturally, considered missing as well.

In standard *EM* fashion, we attempt to maximize the log-likelihood of the observed data, which is given as

$$L(X^o|\tilde{\theta}) = \\ \sum_{\tau=1}^{T} \log \sum_{\mathbf{h}_\tau} \sum_{\mathbf{b}_\tau} \sum_{\mathbf{n}_\tau} \int p(X^o_\tau, \mathbf{x}^m_\tau, \mathbf{h}_\tau, \mathbf{n}_\tau, \mathbf{b}_\tau|\theta)\, d\mathbf{x}^m_\tau,$$

where θ represents the set of all parameters of the model. Since maximizing sums and integrals of a logarithm is difficult in practice, we choose to compute

$$Q(\tilde{\theta}|\theta) = \sum_{\tau=1}^{T} E[\log p(X^o_\tau, \mathbf{x}^m_\tau, \mathbf{h}_\tau, \mathbf{n}_\tau, \mathbf{b}_\tau|\tilde{\theta})].$$

where $E[.]$ denotes taking the expectation with respect to $p(\mathbf{h}_\tau, \mathbf{x}^m_\tau, \mathbf{n}_\tau, \mathbf{b}_\tau|X^o_\tau, \theta)$. Throughout this section, a tilde denotes parameters we are optimizing for, while no tilde implies that the values from the previous iteration are substituted. EM theory [5] guarantees that the maximum of $Q(\tilde{\theta}|\theta)$ is at the maximum of the log-likelihood. The EM algorithm is a recipe to find this maximum (or at least a local maximum) iteratively.

Let us now derive update rules that will be used in the M-step of the EM algorithm. The parameters we need to consider are those of the Gaussian governing the distribution of the foreground features, i.e. μ and Σ, the parameters of the density $p(\mathbf{b})$ which we will model explicitly as a table of probability masses and the parameters governing the background densities, $\mathbf{M}$. It will be helpful to decompose Q into four parts

$$
\begin{aligned}
Q(\tilde{\theta}|\theta) &= Q_1(\tilde{\theta}|\theta) + Q_2(\tilde{\theta}|\theta) + Q_3(\tilde{\theta}|\theta) + Q_4(\theta) \\
&= \sum_{\tau=1}^{T} E[\log p(\mathbf{n}_\tau|\theta)] + \sum_{\tau=1}^{T} E[\log p(\mathbf{b}_\tau|\theta)] \\
&+ \sum_{\tau=1}^{T} E[\log p(X_\tau^o, \mathbf{x}_\tau^m|\mathbf{h}_\tau, \mathbf{n}_\tau, \theta)] \\
&+ \sum_{\tau=1}^{T} E[\log p(\mathbf{h}_\tau|\mathbf{n}_\tau, \mathbf{b}_\tau)]
\end{aligned}
$$

The first three terms contain parameters that will be updated in the EM, while the last term contains no new parameters.

Update rule for μ

Since only Q_3 depends on $\tilde{\mu}$, taking the derivative of the expected likelihood yields

$$\frac{\partial}{\partial \tilde{\mu}} Q_3(\tilde{\theta}|\theta) = \sum_{\tau=1}^{T} E\left[\tilde{\Sigma}^{-1}(\mathbf{z}_\tau - \tilde{\mu})\right],$$

where $\mathbf{z}^T = (\mathbf{x}^o\,\mathbf{x}^m)$ according to our definition above. Setting the derivative to zero yields the following update rule

$$\tilde{\mu} = \frac{1}{T}\sum_{\tau=1}^{T} E[\mathbf{z}_\tau].$$

Update rule for Σ

Similarly, we obtain for the derivative with respect to the inverse covariance matrix

$$\frac{\partial}{\partial \tilde{\Sigma}^{-1}} Q_3(\tilde{\theta}|\theta) = \sum_{\tau=1}^{T} E\left[\frac{1}{2}\tilde{\Sigma} - \frac{1}{2}(\mathbf{z}_\tau - \tilde{\mu})(\mathbf{z}_\tau - \tilde{\mu})^T\right].$$

Equating with zero leads to

$$\tilde{\Sigma} = \frac{1}{T}\sum_{\tau=1}^{T} E[(\mathbf{z}_\tau - \tilde{\mu})(\mathbf{z}_\tau - \tilde{\mu})^T] = \frac{1}{T}\sum_{\tau=1}^{T} E[\mathbf{z}_\tau \mathbf{z}_\tau^T] - \tilde{\mu}\tilde{\mu}^T.$$

Update rule for $p(\mathbf{b})$

To find the update rule for the 2^F probability masses of $p(\mathbf{b})$, we need to consider $Q_2(\tilde{\theta}|\theta)$ because this is the only term that depends on these parameters. Taking the derivative with respect to $\tilde{p}(\bar{\mathbf{b}})$ we find,

$$\frac{\partial}{\partial \tilde{p}(\bar{\mathbf{b}})} Q_2(\tilde{\theta}|\theta) = \sum_{\tau=1}^{T} \frac{E[\delta_{\mathbf{b},\bar{\mathbf{b}}}]}{\tilde{p}(\bar{\mathbf{b}})}$$

Imposing the constraint $\sum_{\bar{\mathbf{b}}\in\mathcal{B}} \tilde{p}(\bar{\mathbf{b}}) = 1$, for instance by adding a Lagrange multiplier term), we find the following update rule for $\tilde{p}(\bar{\mathbf{b}})$,

$$\tilde{p}(\bar{\mathbf{b}}) = \frac{1}{T}\sum_{\tau=1}^{T} E[\delta_{\mathbf{b},\bar{\mathbf{b}}}].$$

Update rule for $\mathbf{M}$

Finally, we notice that $Q_3(\tilde{\theta}|\theta)$ is the only term containing information about the mean number of background points per feature M_f. Differentiating $Q_3(\tilde{\theta}|\theta)$ with respect to $\tilde{\mathbf{M}}$ we find,

$$\frac{\partial}{\partial \tilde{\mathbf{M}}} Q_3(\tilde{\theta}|\theta) = \sum_{\tau=1}^{T} \frac{E[\mathbf{n}_\tau]}{\tilde{\mathbf{M}}} - I.$$

Equating to zero gives the intuitive result,

$$\tilde{\mathbf{M}} = \frac{1}{T}\sum_{\tau=1}^{T} E[\mathbf{n}_\tau].$$

Computing the Sufficient Statistics

In the previous sections we calculated the expressions needed for the M-step in the EM algorithm. All updates rules are expressed in terms of the so called 'sufficient statistics' $E[\mathbf{z}]$, $E[\mathbf{z}\mathbf{z}^T]$, $E[\delta_{\mathbf{b},\bar{\mathbf{b}}}]$ and $E[\mathbf{n}]$. We will now calculate these sufficient statistics in what is formally called the E-step of the EM algorithm. In order to do this we need to consider the *posterior* density. It is given by,

$$p(\mathbf{h}_\tau, \mathbf{x}_\tau^m, \mathbf{n}_\tau, \mathbf{b}_\tau | X_\tau^o, \theta) = \frac{p(\mathbf{h}_\tau, \mathbf{x}_\tau^m, \mathbf{n}_\tau, \mathbf{b}_\tau, X_\tau^o|\theta)}{\sum_{\mathbf{h}_\tau\in\mathcal{H}_\mathbf{b}} \sum_{\mathbf{b}_\tau\in\mathcal{B}} \sum_{\mathbf{n}_\tau=0}^{\infty} \int p(\mathbf{h}_\tau, \mathbf{x}_\tau^m, \mathbf{n}_\tau, \mathbf{b}_\tau, X_\tau^o|\theta)\, d\mathbf{x}_\tau^m}$$

We will first simplify the 'zoo' of variables a bit by observing that if we perform summations in the following order,

$$\sum_{\mathbf{h}_\tau\in\mathcal{H}_\mathbf{b}} \sum_{\mathbf{b}_\tau\in\mathcal{B}} \sum_{\mathbf{n}_\tau=0}^{\infty}$$

we may replace $\mathbf{n} = \mathbf{N} - \mathbf{b}$, where $\mathbf{N}$ is the total number of detections per feature type, and forget the summation over $\mathbf{n}$. Furthermore, we may replace $\mathbf{b} = sign(\mathbf{h})$ (with $sign(0) = 0$) and forget the summation over $\mathbf{b}$. In the following we will assume this simplification and treat $\mathbf{n}$ and $\mathbf{b}$ as functions of $\mathbf{h}$. The denominator in the expression above, $p(X_\tau^o)$, is calculated as follows. Choose a hypothesis consistent with the observed data. Integrate out the missing data in that hyposthesis[2]. Calculate $\mathbf{b}(\mathbf{h})$ and $\mathbf{n}(\mathbf{h})$ and insert them into the joint density. Finally, sum over all possible hypothesis. The expectations of the statistics are calculated in a similar fashion. $E[\delta_{\mathbf{b},\bar{\mathbf{b}}}]$ is calculated by

[2]Integrating out dimensions of a Gaussian is simply done by deleting the means and covariances of those dimensions

summing only over those hypotheses consistent with $\bar{\mathbf{b}}$ in the numerator and dividing by $p(X_\tau^o)$. Similarly, $E[\mathbf{n}_\tau]$ is calculated by averaging $\mathbf{n}(\mathbf{h})$ over all hypotheses. The case $E[\mathbf{z}]$ is slightly more complicated. For every hypothesis we make the split $\mathbf{z}^T = (\mathbf{x}^o \ \mathbf{x}^m)$. $E[\mathbf{x}^o] = \mathbf{x}^o$ because there is no dependence on hidden information. For $E[\mathbf{x}^m]$ one needs to calculate,

$$\int \mathbf{x}^m \ G(\mathbf{z}|\mu, \sigma) \ d\mathbf{x}^m = \mu^m + \Sigma^{mo}\Sigma^{oo-1}(\mathbf{x}^o - \mu^o),$$

where we defined,

$$\mu = \begin{pmatrix} \mu^o \\ \mu^m \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \Sigma^{oo} & \Sigma^{om} \\ \Sigma^{mo} & \Sigma^{mm} \end{pmatrix}$$

Doing this for every consistent hypothesis, summing and dividing by $p(X_\tau^o)$ establishes the result. Finally we need to calculate

$$E[\mathbf{z}\mathbf{z}^T] = \begin{pmatrix} \mathbf{x}^o\mathbf{x}^{oT} & \mathbf{x}^o E[\mathbf{x}^m]^T \\ E[\mathbf{x}^m]\mathbf{x}^{oT} & E[\mathbf{x}^m\mathbf{x}^{mT}] \end{pmatrix}$$

Only the part $E[\mathbf{x}^m\mathbf{x}^{mT}]$ has not been considered. Again, we will make the split $\mathbf{z}^T = (\mathbf{x}^o \ \mathbf{x}^m)$ for evey consistent hypothesis. Integrating out the missing dimensions, $\mathbf{x}^m$, now involves,

$$\int \mathbf{x}^m\mathbf{x}^{mT} \ G(\mathbf{z}|\mu, \sigma) \ d\mathbf{x}^m =$$
$$\Sigma^{mm} - \Sigma^{mo}\Sigma^{oo-1}\Sigma^{moT} + E[\mathbf{x}^m]E[\mathbf{x}^m]^T.$$

Looping through all possible hypotheses and dividing by $p(X_\tau^o)$ again provides the desired result. This concludes the E-step of the EM algorithm.

As described so far, our method assumes that a *model configuration* (the number and types of feature detectors) has been chosen prior to the EM phase. Without any further elaborate strategy, we would have to fit a model for every possible model configuration, hoping to find a model with satisfactory recognition performance. We will present, in the next section, a method to avoid this exhaustive search through configuration space.

4 Experiments

For the experiments described in this section, we used a translation invariant extension of the above derivations. The algorithm could have been used as described above, if the training images had been prepared such that the target object is in the same location in every image. This is inconvenient, especially since our goal is to eliminate the need for user intervention.

4.1 Choice of Detectors

For a general recognition task, the set of feature detectors should contain a large number of simple generic detectors of basic features such as corners, T-junctions, line endings, line crossings etc. However, it is of course possible to add more specific detectors, in the case where prior knowledge about the possible target objects is available.

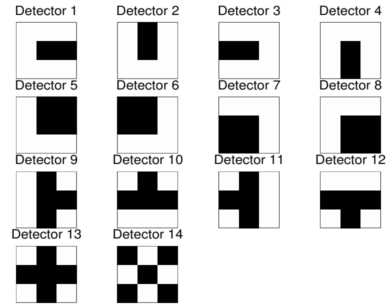

Figure 1: The above 14 templates were used as correlation masks for the letter recognition experiment. The templates are normalized such that the mean is equal to zero.

For the following demonstration of our method, we used a set of 14 simple correlation templates, which are shown in Figure 1.

4.2 Learning Letters

We performed a recognition experiment on a set of 30 “Dilbert” comic strips. The cartoons had a size of 500 $\times$ 200pixels. Since the original resolution was fairly low, we applied the templates only at the highest available resolution. To illustrate the feasability of our method we attempted to learn several letters (‘E’, ‘T’, ‘H’ and ‘L’) from the text segments of the comic strips. This choice was in part motivated by the fact that the letters are hand-written, which ensures a certain degree of variability across each class. We used 6 strips for training, which provided us with 40–60 training samples per letter. We presented each letter in a window of approximately 12 times the area of a single letter. Every letter was thus seen in its “natural surrounding” (Fig. 2). The main limitation for the amount of background area in the training images is the number of feature candidates encountered in the background and the resulting computational cost due to the large number of hypothesis that need to be considered during model learning.

Greedy Configuration Search

After applying all feature detectors to the training samples, we chose the following greedy strategy to explore different model configurations.

In a first step, we explore configurations with a few (e.g. three) different features. The configuration which yields the smallest training error (measured as the probability of misclassification), is chosen and augmented by one feature, trying all possible types. The best of these augmented models is then retained for subsequent augmentation. This process should be

Figure 2: A text segment of a cartoon is shown on the top. One strip typically contains up to six such segments. The frame in the upper image illustrates the size of a training image. On the bottom we show a collection of 9 training "patches" for each letter. Note that adjacent letters are included together with the target letter, which is appearing in a different location in every training image.

stopped as soon as a criterion for model complexity (such as an MDL measure) or desired detection performance is met. In our case we simply chose a total number of five features in the model as a complexity threshold. It is possible, that no further improvement in detection performance is obtained even *before* the maximum number of features is reached. This was the case for the 'L' and 'T' models presented below.

We found the EM algorithm to converge rapidly in 10 - 100 iterations. One iteration took about a second using a Matlab implementation of the method.

Two of the learned models are presented in Figure 3.

Performance Evaluation

We measured detection performance using 20 comic strips that were not included in the training set. Here,

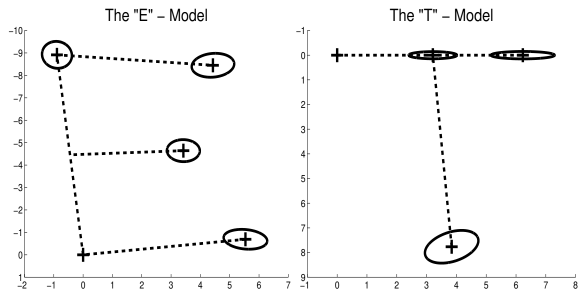

Figure 3: We show the models learned for the letters 'E' and 'T'. Ellipses are shown at a one standard deviation distance from the mean feature positions. The feature position are measured relative to a reference feature. No variance is shown for this feature, as the attached variability would correspond to overall translation of the object. The features chosen for the 'E' are of types 3, 5 and 8, those for the 'T' are of types 1 to 3 and 12.

we used the entire strips, including not only the text regions but also the line drawings which comprised some textured regions.

Rather than classifying every location in the image by applying a fixed threshold, we computed receiver operating characteristics (ROCs), which are shown in Fig. 4.

The overall detection performance was good. This is, in part, due to the fact that line drawings do not suffer from the typical changes in appearance due to different lighting conditions or changes of pose in three dimensions, so that a reliable detection of the features was obtained even with the simple detectors. The performance on individual letters seems to be governed by the performance of the feature detectors. In particular, objects tend to be missed when individual features are not detected. This occured mostly when letters were too close to adjacent ones.

As can be seen in the figure, the performance of the 'L'-detector was considerably worse than that of the other three. This is due to the fact that the model of the 'L' contains only three features. The result are many false negatives, since the letter is missed easily as soon as one feature is missed.

We present examples of image patches with a high probability of being misclassified (see Figs. 5 and 6). Without settling on a particular decision threshold, we can identify those by picking the foreground samples with the lowest likelihood and the background locations with the highest likelihood.

A complete strip with detections of the letter 'E' is shown in Fig. 7.

An interesting observation made during these experiments is that the model sometimes included features from neighboring letters. If we suppose that the background patterns in the training set resemble those

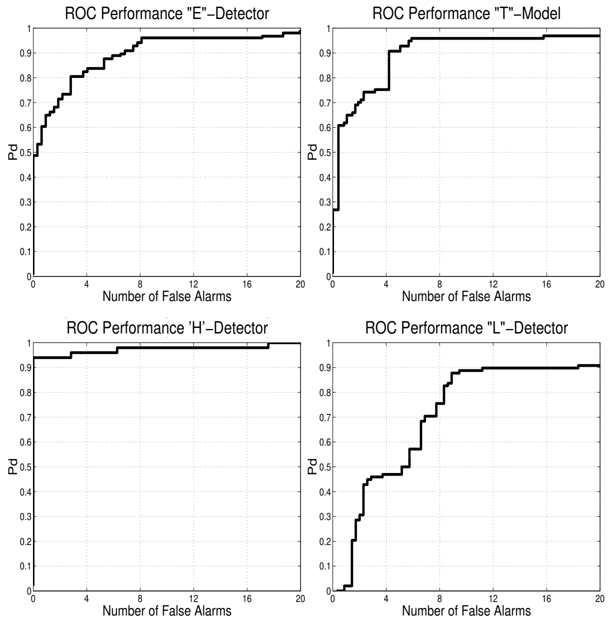

Figure 4: ROC curves show the detection performance of all four letter detectors. The probability of false alarm is rescaled to reflect the expected performance on an image of the size of the cartoon strips. The probabilities of error at an operating point where the probilities of false positives and false negatives are equal, are 3.9% for ‘E’, 4.2% for ‘T’, 2.0% for ‘H’ and 9.2% for ‘L’.

encountered when the model is finally put to work, then knowledge about the neighbors of a letter to be detected can and should be used to improve the performance.

5 Conclusion and Future Work

We have presented a method which can learn an optimal object class model—in a maximum likelihood sense—with respect to the set of employed feature detectors, as well as all other model parameters. The method significantly facilitates model acquisition, since human supervision is reduced to a minimum.

Limitations are the remaining need for normalization for rotation and scale. The greedy strategy used to search possible model configurations also proved to be relatively slow, which is why we are currently investigating a method which dynamically introduces or suppresses Gaussian clusters during the maximization process. We are also investigating the possiblity to directly learn a true *shape* model instead of a pdf over feature position in the *image plane*.

References

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

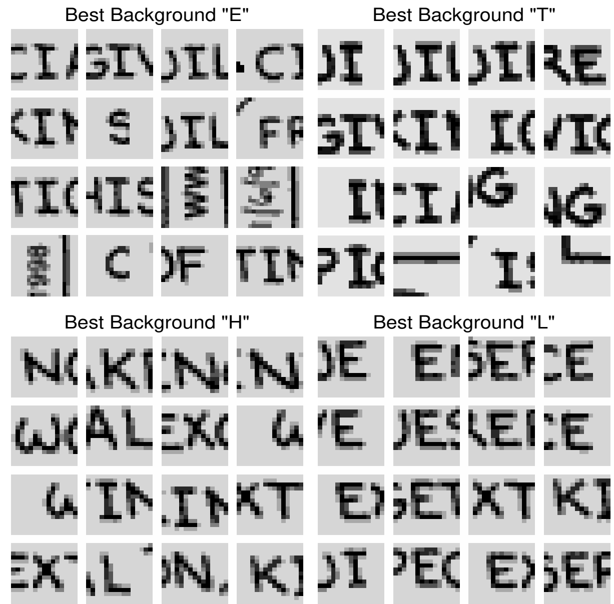

Figure 5: Shown are, for every letter, the 16 locations on the background of test images which yielded the highest likelihood.

Figure 6: For every letter, we show the 16 lowest scoring correct locations.

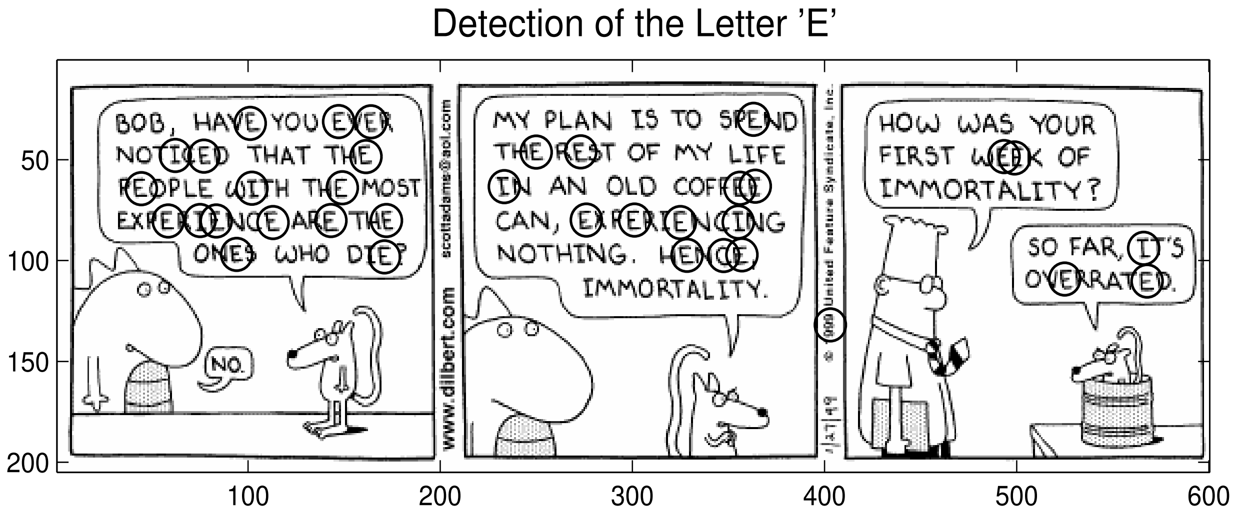

Figure 7: An entire comic strip with detections of the letter ‘E’. A few ‘E’s were missed and a few ‘I’s as well as other background noise were detected. The threshold for this experiment was set such that at least 90% of all true occurences of the letter were detected throughout the 20 test strips.

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

[3] M.C. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *proc. ECCV’98*, pages 628–641, 1998.

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

[5] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society* **B**, 39:1–38, 1976.

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

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

[8] K. N. Walker, T. F. Cootes, and C. J. Taylor. Locating salient facial features. In *Int. Conf. on Automatic Face and Gesture Recognition*, Nara, Japan, 1998.

[9] A.L. Yuille. “Deformable Templates for Face Recognition”. *J. of Cognitive Neurosci.*, 3(1):59–70, 1991.