# *TextonBoost*: Joint Appearance, Shape and Context Modeling for Multi-class Object Recognition and Segmentation

Jamie Shotton[2], John Winn[1], Carsten Rother[1], and Antonio Criminisi[1]

[1] Microsoft Research Ltd., Cambridge, UK
{jwinn, carrot, antcrim}@microsoft.com
[2] Department of Engineering,
University of Cambridge
jdjs2@cam.ac.uk

**Abstract.** This paper proposes a new approach to learning a discriminative model of object classes, incorporating appearance, shape and context information efficiently. The learned model is used for automatic visual recognition and semantic segmentation of photographs. Our discriminative model exploits novel features, based on textons, which jointly model shape and texture. Unary classification and feature selection is achieved using shared boosting to give an efficient classifier which can be applied to a large number of classes. Accurate image segmentation is achieved by incorporating these classifiers in a conditional random field. Efficient training of the model on very large datasets is achieved by exploiting both random feature selection and piecewise training methods.

High classification and segmentation accuracy are demonstrated on three different databases: i) our own 21-object class database of photographs of real objects viewed under general lighting conditions, poses and viewpoints, ii) the 7-class Corel subset and iii) the 7-class Sowerby database used in [1]. The proposed algorithm gives competitive results both for highly textured (e.g. grass, trees), highly structured (e.g. cars, faces, bikes, aeroplanes) and articulated objects (e.g. body, cow).

## 1 Introduction

This paper investigates the problem of achieving automatic detection, recognition and segmentation of object classes in photographs. Precisely, given an image, the system should automatically partition it into semantically meaningful areas each labeled with a specific object class. The challenge is to handle a large number of both structured and unstructured object classes, while modeling their variabilities. Our focus is not only the accuracy of segmentation and recognition, but also the efficiency of the algorithm, which becomes particularly important when dealing with large image collections.

At a local level, the *appearance* of an image patch leads to ambiguities in its class label. For example, a window can be part of a car, a building or an aeroplane. To overcome these ambiguities, it is necessary to incorporate longer

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part I, LNCS 3951, pp. 1–15, 2006.

range information such as the spatial configuration of the patches on an object (the object *shape*) and also *contextual* information from the surrounding image. To achieve this we construct a discriminative model for labeling images which exploits all three types of information: appearance, shape and context.

**Related work.** Whilst the fields of object recognition and segmentation have been extremely active in recent years, many authors have considered these two tasks separately. For example, recognition of particular object classes has been achieved using the constellation models of Fergus et al. [2], the deformable shape models of Berg et al. [3] and the texture models of Winn et al. [4]. None of these methods leads to a pixel-wise segmentation of the image. Conversely, other authors have considered only the segmentation task, e.g. [5, 6].

Joint detection and segmentation of a *single* object class has been achieved by several authors [7, 8, 9]. Typically, these approaches exploit a global shape model and are therefore unable to cope with arbitrary viewpoints or severe occlusion. Additionally, only highly structured object classes are addressed.

A similar task as addressed in this paper was considered in [10] where a classifier was used to label regions found by automatic segmentation. However such segmentations often do not correlate with semantic objects. Our solution to this problem is to perform segmentation and recognition in the same unified framework rather than in two separate steps. Such a unified approach has been presented in [11] where only text and faces are recognized and at a high computational cost. Konishi and Yuille [12] label images using a unary classifier and hence do not achieve spatially coherent segmentations.

The most similar work to ours is that of He et al. [1] which incorporate region and global label features to model shape and context in a Conditional Random Field. Their work uses Gibbs sampling for both the parameter learning and label inference and is therefore limited in the size of dataset and number of classes which can be handled efficiently. Our focus on the speed of training and inference allows us to use larger datasets with many more object classes. We currently handle 21 classes (compared to the seven classes of [1]) and it would be tractable to train our model on even larger datasets than presented here.

Our contributions in this paper are threefold. First, we present a discriminative model which is capable of fusing shape, appearance and context information to recognize efficiently the object classes present in an image, whilst exploiting edge information to provide an accurate segmentation. Second, we propose features, based on textons, which are capable of modeling object shape, appearance and context. Finally, we demonstrate how to train the model efficiently on a very large dataset by exploiting both boosting and piecewise training methods.

The paper is structured as follows. In the next section we describe the image database used in our experiments. Section 3 introduces the high-level model, a Conditional Random Field, while section 4 presents our novel low-level image features and their use in constructing a boosted classifier. Experiments, performance evaluation and conclusions are given in the final two sections.

## 2 Image Databases

Our object class models are learned from a set of labeled training images. In this paper we consider three different labeled image databases. Our own database[1] is composed of 591 photographs of the following 21 object classes: building, grass, tree, cow, sheep, sky, aeroplane, water, face, car, bike, flower, sign, bird, book, chair, road, cat, dog, body, boat (fig. 1). The training images were hand-labeled with


**Fig. 1. The labeled image database.** A selection of images in our 21-class database and their corresponding ground-truth annotations. Colors map uniquely to object class labels. All images are approximately $320 \times 240$ pixels.

the assigned colors acting as indices into the list of object classes. Note that we consider completely general lighting conditions, camera viewpoint, scene geometry, object pose and articulation. Our database is split randomly into roughly 45% training, 10% validation and 45% test sets, while ensuring approximately proportional contributions from each class.

Note that the ground-truth labeling of the 21-class database contains pixels labeled as 'void'. These were included both to cope with pixels that do not belong to a database class, and to allow for a rough and quick hand-segmentation which does not align exactly with the object boundaries. Void pixels are ignored for both training and testing.

For comparison with previous work we have also used the 7-class Corel database subset (where images are $180 \times 120$ pixels) and the 7-class Sowerby database ($96 \times 64$ pixels) used in [1]. For those two databases the numbers of images in the training and test sets are exactly as for [1].

## 3 A Conditional Random Field Model of Object Classes

We use a Conditional Random Field (CRF) model [13] to learn the conditional distribution over the class labeling given an image. The use of a Conditional Random Field allows us to incorporate shape, texture, color, location and edge cues in a single unified model. We define the conditional probability of the class labels $\mathbf{c}$ given an image $\mathbf{x}$ as

$$\log P(\mathbf{c}|\mathbf{x},\boldsymbol{\theta}) = \sum_i \overbrace{\psi_i(c_i,\mathbf{x};\boldsymbol{\theta}_\psi)}^{\text{shape-texture}} + \overbrace{\pi(c_i,\mathbf{x}_i;\boldsymbol{\theta}_\pi)}^{\text{color}} + \overbrace{\lambda(c_i,i;\boldsymbol{\theta}_\lambda)}^{\text{location}} + \sum_{(i,j)\in\mathcal{E}} \overbrace{\phi(c_i,c_j,\mathbf{g}_{ij}(\mathbf{x});\boldsymbol{\theta}_\phi)}^{\text{edge}} - \log Z(\boldsymbol{\theta},\mathbf{x}) \quad (1)$$

[1] Publicly available at http://research.microsoft.com/vision/cambridge/recognition/

where $\mathcal{E}$ is the set of edges in the 4-connected grid, $Z(\boldsymbol{\theta}, \mathbf{x})$ is the partition function, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_\psi, \boldsymbol{\theta}_\pi, \boldsymbol{\theta}_\lambda, \boldsymbol{\theta}_\phi\}$ are the model parameters, and $i$ and $j$ index nodes in the grid (corresponding to positions in the image).

**Shape-texture potentials.** The shape-texture potentials $\psi$ use features selected by boosting to represent the shape, texture and appearance context of the object classes. These features and the boosting procedure used to perform feature selection while training a multi-class logistic classifier are described in section 4. We use this classifier directly as a potential in the CRF, so that

$$\psi_i(c_i, \mathbf{x}; \boldsymbol{\theta}_\psi) = \log \widetilde{P}_i(c_i|\mathbf{x}) \tag{2}$$

where $\widetilde{P}_i(c_i|\mathbf{x})$ is the normalized distribution given by the classifier using learned parameters $\boldsymbol{\theta}_\psi$.

**Edge potentials.** The pairwise edge potentials $\phi$ have the form of a contrast sensitive Potts model [14],

$$\phi(c_i, c_j, \mathbf{g}_{ij}(\mathbf{x}); \boldsymbol{\theta}_\phi) = -\boldsymbol{\theta}_\phi^T \mathbf{g}_{ij}(\mathbf{x}) \delta(c_i \neq c_j). \tag{3}$$

In this work, we set the edge feature $\mathbf{g}_{ij}$ to measure the difference in color between the neighboring pixels, as suggested by [15], $\mathbf{g}_{ij} = [\exp(-\beta \|x_i - x_j\|^2), 1]^T$ where $x_i$ and $x_j$ are three-dimensional vectors representing the color of the $i$th and $j$th pixels. Including the unit element allows a bias to be learned, to remove small, isolated regions. The quantity $\beta$ is set (separately for each image) to $(2\langle \|x_i - x_j\|^2 \rangle)^{-1}$, where $\langle \cdot \rangle$ averages over the image.

**Color potentials.** Capture the color distribution of the instances of a class in a *particular image*. This choice is motivated by the fact that, whilst the distribution of color across an entire class of objects is broad, the color distribution across one or a few instances of the class is typically compact. Hence the parameters $\boldsymbol{\theta}_\pi$ are learned separately for each image (and so this learning step needs to be carried out at test time). This aspect of the model captures the more precise image-specific appearance that a solely class-specific recognition system cannot.

Color models are represented as mixtures of Gaussians (GMM) in color space where the mixture coefficients depend on the class label. The conditional probability of the color of a pixel $x$ is given by

$$P(x|c) = \sum_k P(k|c) \mathcal{N}(x \,|\, \bar{x}_k, \Sigma_k) \tag{4}$$

where $k$ is a random variable representing the component the pixel is assigned to, and $\bar{x}_k$ and $\Sigma_k$ are the mixture mean and variance respectively. Notice that the mixture components are shared between different classes and only the coefficients depend on the class label, making the model much more efficient to learn than a separate GMM for each class. For a particular pixel $x_i$ we compute a fixed

soft assignment to the mixture components $P(k|x_i)$.[2] Given this assignment, we choose our color potential to have the form

$$\pi(c_i, x_i; \boldsymbol{\theta}_\pi) = \log \sum_k \boldsymbol{\theta}_\pi(c_i, k) P(k|x_i) \tag{5}$$

where parameters $\boldsymbol{\theta}_\pi$ act as a probability lookup-table; see (8).

**Location potentials.** capture the weak dependence of the class label on the absolute location of the pixel in the image. The potential takes the form of a look-up table with an entry for each class and pixel location,

$$\lambda_i(c_i, i; \boldsymbol{\theta}_\lambda) = \log \boldsymbol{\theta}_\lambda(c_i, \hat{i}). \tag{6}$$

The index $\hat{i}$ is the normalized version of the pixel index $i$, where the normalization allows for images of different sizes; e.g. if the image is mapped onto a canonical square then $\hat{i}$ indicates the pixel position within this canonical square.

### 3.1 Learning the CRF Parameters

Ideally, we would learn the model parameters by maximizing the conditional likelihood of the true class labels given the training data. This can be achieved using gradient ascent, and computing the gradient of the likelihood with respect to each parameter, requiring the evaluation of marginals over the class labels for each training image. Exact computation of these marginals is intractable due to the complexity of the partition function $Z(\mathbf{x}, \boldsymbol{\theta})$ in (1). Instead, we approximated the label marginals by the mode, i.e. the most probable labeling, computed as discussed later in this section. This choice of approximation was made because the size of our datasets limited the time available to estimate marginals. Using this approximation, conjugate gradient ascent did converge but unfortunately the learned parameters gave poor results (almost no improvement on unary classification alone).

Given these problems with directly maximizing the conditional likelihood, we decided to use a method based on *piecewise training* [16] instead. Piecewise training involves dividing the CRF model into pieces, each of which is trained independently. As discussed in [16], this training method minimizes an upper bound on the log partition function. However, this bound is generally an extremely loose one and performing parameter training in this way leads to problems with overcounting during inference in the combined model. Modifying piecewise training to incorporate fixed powers can compensate for overcounting. It can be shown that this leads to an approximate partition function of similar form of that used in [16], except that it is no longer an upper bound on the true partition function. Optimal selection of those powers is an area of active research. In this work, we added power parameters for the location and color potentials and optimized them discriminatively.

[2] A soft assignment was seen to give a marginal improvement over a hard assignment, at negligible extra cost.

Each of the potential types is therefore trained separately to produce a normalized model. For the shape-texture potentials, we simply use the parameters learned during boosting. For the location potentials, we train the parameters by maximizing the likelihood of the normalized model containing just that potential and raising the result to a fixed power $w_\lambda$ (specified in section 5) to compensate for overcounting. Hence, the location parameters are learned using

$$\boldsymbol{\theta}_\lambda(c_i, \hat{i}) = \left( \frac{N_{c,\hat{i}} + \alpha_\lambda}{N_{\hat{i}} + \alpha_\lambda} \right)^{w_\lambda} \tag{7}$$

where $N_{c,\hat{i}}$ is the number of pixels of class $c$ at normalized location $\hat{i}$ in the training set, $N_{\hat{i}}$ is the total number of pixels at location $\hat{i}$ and $\alpha_\lambda$ is a small integer (we use $\alpha_\lambda = 1$) corresponding to a weak Dirichlet prior on $\boldsymbol{\theta}_\lambda$.

At test time the color parameters are learned for each image in a piecewise fashion using Iterative Conditional Modes, similar to [15]. First a class labeling $\mathbf{c}^\star$ is inferred and then the color parameters are updated using

$$\boldsymbol{\theta}_\pi(c_i, k) = \left( \frac{\sum_i \delta(c_i = c_i^\star) P(k|x_i) + \alpha_\pi}{\sum_i P(k|x_i) + \alpha_\pi} \right)^{w_\pi}. \tag{8}$$

Given this new parameter setting, a new class labeling is inferred and this procedure is iterated [15]. The Dirichlet prior parameter $\alpha_\pi$ was set to 0.1, and the power parameter is $w_\pi$. In practice, $w_\pi = 3$, fifteen color components and two iterations of this procedure gave good results. Because we are training in pieces, the color parameters do not need to be learned for the training set.

Learning the edge potential parameters $\boldsymbol{\theta}_\phi$ by maximum likelihood was also attempted. Unfortunately, the lack of alignment between object edges and label boundaries in the roughly labeled training set forced the learned parameters to tend towards zero. Instead, the values of the only two contrast-related parameters were manually selected to minimize the error on the validation set.

### 3.2 Inference in the CRF Model

Given a set of parameters learned for the CRF model, we wish to find the most probable labeling $\mathbf{c}^\star$; i.e. the labeling that maximizes the conditional probability (1). The optimal labeling is found by applying the alpha-expansion graph-cut algorithm of [14] (note that our energy is *regular*). In our case the initial configuration is given by the mode of the unary potentials, though the MAP solution was not in practice sensitive to this initialization.

## 4 Boosted Learning of Shape, Texture and Context

The most important part of the CRF energy is the unary potential, which is based on a novel set of features which we call *shape filters*. These features are capable of capturing shape, texture and appearance context jointly. We describe shape filters next, together with the process for automatic feature selection.

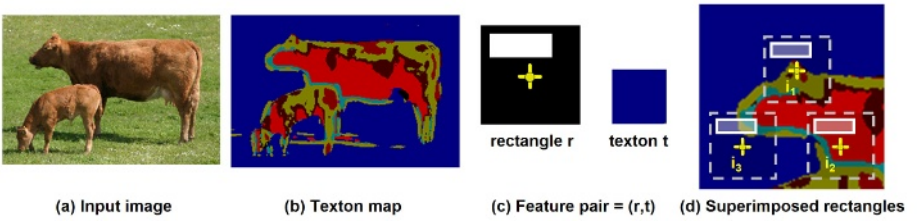


**Fig. 2. Shape filter responses and appearance context.** (**a, b**) An image and its corresponding texton map (colors map uniquely to texton indices). (**c**) A rectangle mask $r$ (white) is offset from the center (yellow cross), and paired with a texton index $t$ which here maps to the blue color. (**d**) As an example, the feature response $v(i, r, t)$ is calculated at three positions in the texton map (zoomed). If $A$ is the area of $r$, then in this example $v(i_1, r, t) \approx A$, $v(i_2, r, t) \approx 0$, and $v(i_3, r, t) \approx A/2$. For this feature where $t$ is a 'grass' texton, our algorithm learns that points $i$ (such as $i_1$) belonging to 'cow' regions tend to produce large counts $v(i, r, t)$, and hence exploits the contextual information that 'cow' pixels tend to be surrounded by 'grass' pixels.

**Textons.** Efficiency demands compact representations for the range of different appearances of an object. For this we utilize *textons* [17] which have been proven effective in categorizing materials [18] as well as generic object classes [4]. A dictionary of textons is learned by convolving a 17-dimensional filter bank[3] with all the training images and running $K$-means clustering (using Mahalanobis distance) on the filter responses. Finally, each pixel in each image is assigned to the nearest cluster center, thus providing the *texton map* (see fig. 2(a,b)).

**Shape filters.** Consist of a set of $N_R$ rectangular regions whose four corners are chosen at random within a fixed bounding box covering about half the image area. For a particular texton $t$, the feature response at location $i$ is the count of instances of that texton under the offset rectangle mask (see fig. 2(c,d)). These filter responses can be efficiently computed over a whole image with integral images [19] ($K$ for each image, where $K$ is the number of textons).

Shape filters with their pairing of rectangular masks and textons can be seen as an extension of the features used in [19]. Our features are sufficiently general to allow us to *learn* automatically shape and context information, in contrast to techniques such as Shape Context [20] which utilize a hand-picked shape descriptor. Figure 2 illustrates how shape filters are able to model appearance-based context. Modeling shape is demonstrated for a toy example in fig. 3.

**Joint Boosting for unary classification.** A multi-class classifier is learned using an adapted version of the Joint Boosting algorithm of [21]. The algorithm iteratively builds a strong classifier as a sum of 'weak classifiers', simultaneously

[3] The filter bank used here is identical to that in [4], consisting of scaled Gaussians, $x$ and $y$ derivatives of Gaussians, and Laplacians of Gaussians. The Gaussians are applied to all three color channels, while the remaining filters only to the luminance. The perceptually uniform CIELab color space is used.

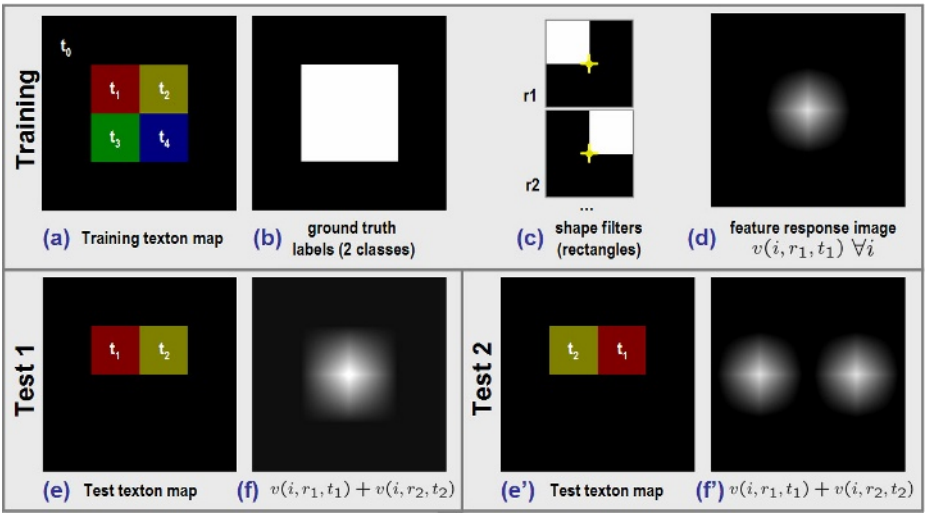


**Fig. 3. Capturing local shape information.** This toy example illustrates how our *shape filters* capture relative positions of textons. (**a**) Input texton map. (**b**) Input binary ground-truth label map (e.g. foreground=white, background=black). (**c**) Example rectangle masks ($r_1$ and $r_2$). (**d**) The feature response image $v(i, r_1, t_1)$ shows a positive response within the foreground region and zero in the background. An identical response image is computed for feature $(r_2, t_2)$. Boosting would pick both these features as discriminative. (**e**) A test input with textons $t_1$ and $t_2$ in the *same* relative position as that of training. (**f**) Illustration that the two feature responses *reinforce* each other. (**e'**) A second test with $t_1$ and $t_2$ swapped. (**f'**) The summed feature responses do not reinforce, giving a weaker signal for classification. Note (**f**) and (**f'**) are illustrative only since boosting actually combines thresholded feature responses.

selecting discriminative features. Each weak classifier is a decision stump based on a thresholded feature response, and is *shared* between a set of classes, allowing a single feature to help classify several classes at once. The sharing of features between classes allows for classification with cost sub-linear in the number of classes, and also leads to improved generalization.

The learned 'strong' classifier is an additive model of the form $H(c_i) = \sum_{m=1}^{M} h_m(c_i)$, summing the classification confidence of $M$ weak classifiers. This confidence value can be reinterpreted as a probability distribution over $c_i$ using the softmax transformation $\widetilde{P}_i(c_i|\mathbf{x}) = \frac{\exp(H(c_i))}{\sum_{c_i'} \exp(H(c_i'))}$ [22].

Each weak-learner is a decision stump of the form

$$h(c_i) = \begin{cases} a\delta(v(i, r, t) > \theta) + b & \text{if } c_i \in N \\ k_{c_i} & \text{otherwise} \end{cases} \tag{9}$$

with parameters $(a, b, \{k_c\}_{c \notin N}, \theta, N, r, t)$ and where $\delta(\cdot)$ is a 0-1 indicator function. The $r$ and $t$ indices together specify the shape filter feature (rectangle mask

and texton respectively), with $v(i, r, t)$ representing the corresponding feature response at position $i$. For those classes that share this feature ($c_i \in N$), the weak learner gives $h(c_i) \in \{a + b, b\}$ depending on the comparison of $v(i, r, t)$ to a threshold $\theta$. For each class not sharing the feature ($c_i \notin N$) there is a constant $k_{c_i}$ that ensures asymmetrical sets of positive and negative training examples do not adversely affect the learning procedure.

The boosting algorithm iteratively minimizes an error function which unfortunately requires an expensive brute-force search over the sharing set $N$, the features ($r$ and $t$), and the thresholds $\theta$. Given these parameters, a closed form solution exists for $a$, $b$ and $\{k_c\}_{c \notin N}$. The set of all possible sharing sets is exponentially large, and so we employ the quadratic-cost greedy approximation of [21]. To speed up the minimization over features we employ the random feature selection procedure described below. Optimization over $\theta \in \Theta$ for a discrete set $\Theta$ can be made efficient by careful use of histograms of feature responses.

**Sub-sampling and random feature selection for training efficiency.** The considerable memory and processing requirements make training on a per-pixel basis impractical. Computational expense is reduced by calculating filter responses on a $\Delta \times \Delta$ grid (either $3 \times 3$ for the smaller databases or $5 \times 5$ for the largest database). The shape filter responses themselves are still calculated at full resolution to enable per-pixel accurate classification at test time.

One consequence of this sub-sampling is that a small degree of shift-invariance is learned. On its own, this would lead to inaccurate segmentation at object boundaries. However, when applied in the context of the CRF, the edge and color potentials come into effect to locate the object boundary accurately.

Even with sub-sampling, exhaustive searching over all features (pairs of rectangle and texton) at each round of boosting is prohibitive. However, our algorithm examines only a fraction $\tau \ll 1$ of features, randomly chosen at each round

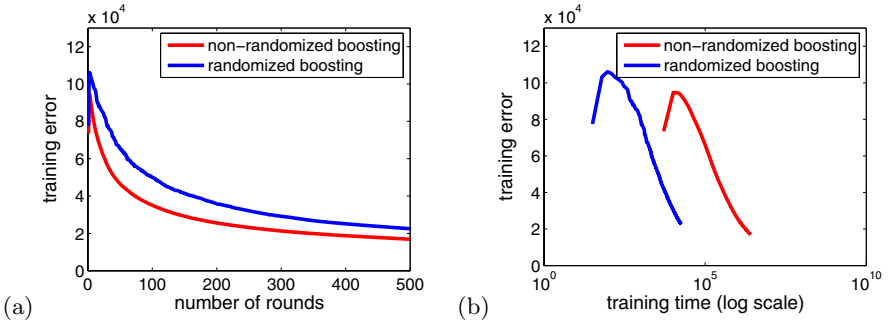


**Fig. 4. Effect of random feature selection on a toy example.** (**a**) Training error as a function of the number of rounds (axis scales are unimportant). (**b**) Training error as function of time. Randomization makes learning two orders of magnitude faster here, with very little increase in training error for the same number of rounds. The peak in error in the first few rounds is due to an artefact of the learning algorithm.

(see [23]). All our results use $\tau = 0.003$ so that, over several thousand rounds, there is high probability of testing all features at least once.

To analyze the effect of random feature selection, we compared the results of boosting on a toy data set of ten images with ten rectangle masks, 400 textons, and $\tau = 0.003$. The results in fig. 4 show that using random feature selection improves the training time by several orders of magnitude whilst having only a small impact on the training error.

## 5 Results and Comparisons

**Boosting accuracy.** Fig. 5(a) illustrates the effect of training the boosted classifier in isolation, i.e. separately from the CRF. As expected, the error decreases (non-linearly) as the number of weak classifiers increases. Furthermore, fig. 5(b) shows the accuracy of classification with respect to the validation set, which after about 5000 rounds flattens out to a value of approximately 73%.

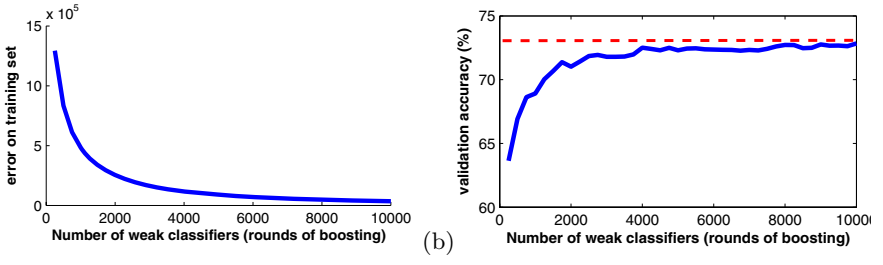


**Fig. 5. Error plots.** Training error (**a**) and accuracy on the validation set (**b**) as function of the number of weak classifiers. While the training error decreases almost to zero, the validation set accuracy rises to a maximum of about 73%.

The boosting procedure takes 42 hours for 5000 rounds on the 21-class training set of 276 images on a 2.1 Ghz machine with 2GB memory. Without random feature selection, the training time would be around 14000 hours. Note that due to memory constraints, the training integral images had to be computed on-the-fly which slowed the learning down by at least a factor two.

**Object class recognition and segmentation.** This section presents results for the full CRF model on our 21-class database. Our unoptimized implementation takes approximately three minutes to segment each test image. The majority of this time is spent evaluating all the $\widetilde{P}_i(c_i|\mathbf{x})$ involving a few thousand weak-classifier evaluations. Evaluating those potentials on a $\Delta \times \Delta$ grid (with $\Delta = 5$) produces almost as good results in about twenty-five seconds per test image.

Example results of simultaneous recognition and segmentation are shown in fig. 6. The figure shows both the original photographs and the color-coded output labeling. Note for instance that despite large occlusions, bicycles are recognized and segmented correctly, and large variations in the appearance of grass and road

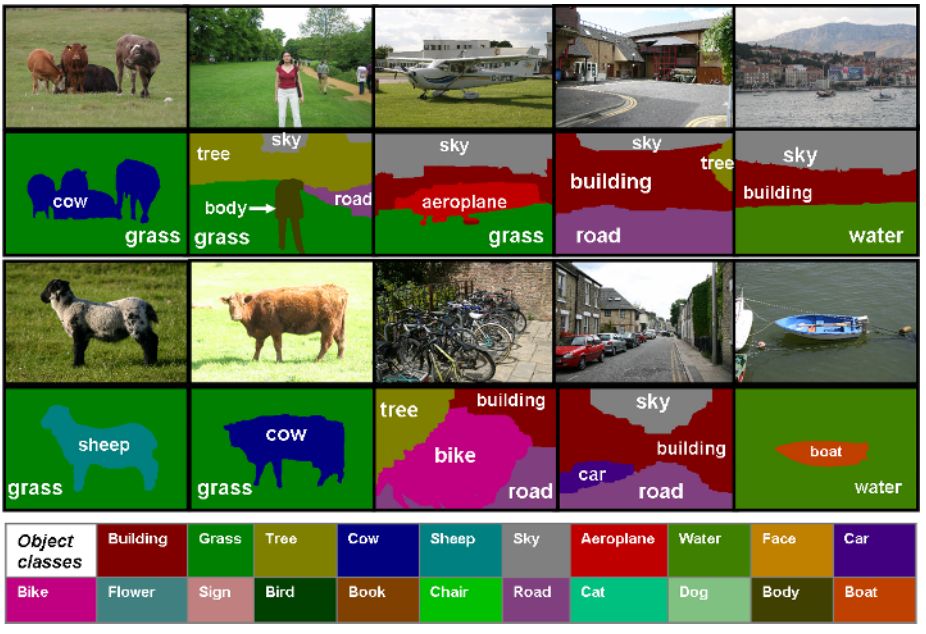


**Fig. 6. Some example results.** Above, original images with corresponding color-coded output object-class maps. Below, color-coding legend for the 21 object classes. For clarity, textual labels have also been superimposed on the result object maps.

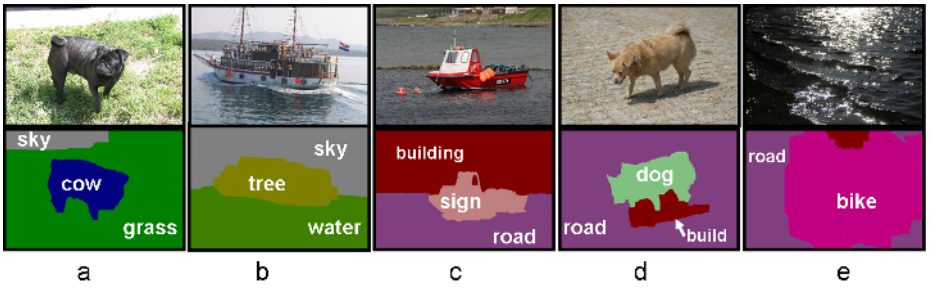


**Fig. 7. Some examples where recognition works less well.** Input test images with corresponding color-coded output object-class maps. Note that even when recognition fails segmentation may still be quite accurate.

are correctly modeled. In order to better understand the behavior of our algorithm we also present some examples which work less well, in fig. 7. In fig. 7(a,d) despite the recognition of the central figure being incorrect, the segmentation is still accurate. For cases like these, the algorithm of [24] could be used to refine the class labeling. In fig. 7(e) the entire image is incorrectly recognized due to lack of similar examples of water in the training data, a typical drawback of discriminative learning.

| True class \ Inferred class | building | grass | tree | cow | sheep | sky | aeroplane | water | face | car | bike | flower | sign | bird | book | chair | road | cat | dog | body | boat |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| building | **61.6** | 4.7 | 9.7 | 0.3 | | 2.5 | 0.6 | 1.3 | 2.0 | 2.6 | 2.1 | | 0.6 | 0.2 | 4.8 | | 6.3 | 0.4 | | 0.5 | |
| grass | 0.3 | **97.6** | 0.5 | | | | | | | | 0.1 | | | | | | | | | 1.3 | |
| tree | 1.2 | 4.4 | **86.3** | 0.5 | | 2.9 | 1.4 | 1.9 | 0.8 | 0.1 | | | | | | | 0.1 | | 0.2 | 0.1 | |
| cow | | 30.9 | 0.7 | **58.3** | | | | 0.9 | 0.4 | | | 0.4 | | | 4.2 | | | | | 4.1 | |
| sheep | 16.5 | 25.5 | 4.8 | 1.9 | **50.4** | | | | | | | | | 0.6 | | | 0.2 | | | | |
| sky | 3.4 | 0.2 | 1.1 | | | **82.6** | | 7.5 | | | | | | | | | 5.2 | | | | |
| aeroplane | 21.5 | 7.2 | | | | 3.0 | **59.6** | 8.5 | | | | | | | | | | | | | |
| water | 8.7 | 7.5 | 1.5 | 0.2 | | 4.5 | | **52.9** | | 0.7 | 4.9 | | | 0.2 | 4.2 | | 14.1 | 0.4 | | | |
| face | 4.1 | | 1.1 | | | | | | **73.5** | 7.1 | | | | | 8.4 | | | 0.4 | 0.2 | 5.2 | |
| car | 10.1 | | 1.7 | | | | | | | **62.5** | 3.8 | | 5.9 | 0.2 | | | 15.7 | | | | |
| bike | 9.3 | | 1.3 | | | | | | | 1.0 | **74.5** | | 2.5 | | | 3.9 | 5.9 | | 1.6 | | |
| flower | | 6.6 | 19.3 | 3.0 | | | | | | | | **62.8** | | | 7.3 | | 1.0 | | | | |
| sign | 31.5 | 0.2 | 11.5 | 2.1 | | 0.5 | | 6.0 | | 1.5 | | 2.5 | **35.1** | | 3.6 | 2.7 | 0.8 | 0.3 | | 1.8 | |
| bird | 16.9 | 18.4 | 9.8 | 6.3 | 8.9 | 1.8 | | 9.4 | | | | | | **19.4** | | | 4.6 | 4.5 | | | |
| book | 2.6 | | 0.6 | | | | | | 0.4 | | | 2.0 | | | **91.9** | | | | | 2.4 | |
| chair | 20.6 | 24.8 | 9.6 | 18.2 | | 0.2 | | | | | 3.7 | | | | 1.9 | **15.4** | 4.5 | | 1.1 | | |
| road | 5.0 | 1.1 | 0.7 | | | | | 3.4 | 0.3 | 0.7 | 0.6 | | 0.1 | 0.1 | 1.1 | | **86.0** | | | 0.7 | |
| cat | 5.0 | | 1.1 | 8.9 | | | | 0.2 | | 2.0 | | | | | 0.6 | | 28.4 | **53.6** | 0.2 | | |
| dog | 29.0 | 2.2 | 12.9 | 7.1 | | | | 9.7 | | | | | | | 8.1 | | 11.7 | | **19.2** | | |
| body | 4.6 | 2.8 | 2.0 | 2.1 | 1.3 | 0.2 | | | 6.0 | 1.1 | | | | | 9.9 | | 1.7 | 4.0 | 2.1 | **62.1** | |
| boat | 25.1 | | 11.5 | | | 3.8 | | 30.6 | | 2.0 | 8.6 | | 6.4 | 5.1 | | | 0.3 | | | | **6.6** |

**Fig. 8. Accuracy of segmentation for the 21-class database.** Confusion matrix with percentages row-normalized. Overall pixel-wise accuracy 72.2%.

*Quantitative evaluation.* Figure 8 shows the confusion matrix obtained by applying our algorithm to the test image set. Accuracy values in the table are computed as percentage of image pixels assigned to the correct class label, ignoring pixels labeled as void in the ground-truth. The overall classification accuracy is 72.2%; random chance would give $1/21 = 4.76\%$, and thus our results are about 15 times better than chance. For comparison, the boosted classifier alone gives an overall accuracy of 69.6% and so the color, edge and location potentials increase the accuracy by 2.6%. This seemingly small numerical improvement corresponds to a large perceptual improvement (cf. fig. 10). The parameter settings, learned against the validation set, were $M = 5000$ rounds, $N_t = 400$ textons, edge potential parameters $\boldsymbol{\theta}_\phi = [45, 10]^T$, and location potential power $w_\lambda = 0.1$.

The greatest accuracies are for classes which have low visual variability and many training examples (e.g. grass, book, tree, road, sky and bicycle) whilst the lowest accuracies are for classes with high visual variability and fewer training examples (e.g. boat, chair, bird, dog). We expect more training data to boost considerably the recognition accuracy for those difficult classes. Additionally, using features with better lighting invariance properties would help considerably.

Let us now focus on some of the largest mistakes in the confusion matrix to gather some intuition on how the algorithm may be improved. Structured objects such as aeroplanes, chairs, signs, boats are sometimes incorrectly classified as buildings. Perhaps this kind of problem may be fixed by a part-based modeling approach. For example, detecting windows and roofs should resolve many such ambiguities. Furthermore, objects such as cows, sheep and chairs (benches) which in training are always seen sitting on grass do get confused with grass.

**Table 1.** Comparison of segmentation/recognition accuracy and efficiency

| | Accuracy | | Speed (Train/Test) | |
|---|---|---|---|---|
| | Sowerby | Corel | Sowerby | Corel |
| This paper – Full CRF model | 88.6% | 74.6% | 5h/10s | 12h/30s |
| This paper – Unary classifier only | 85.6% | 68.4% | | |
| He et al. – mCRF model [1] | 89.5% | 80.0% | Gibbs | Gibbs |
| He et al. – unary classifier only | 82.4% | 66.9% | | |

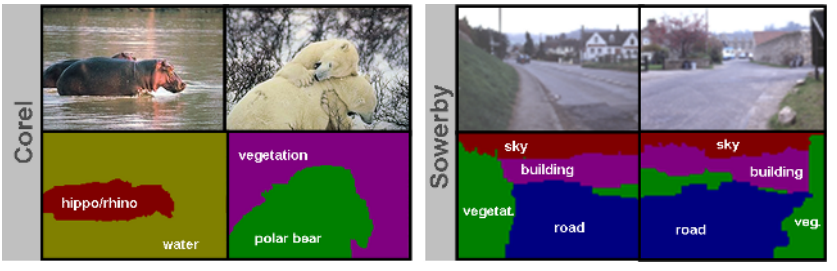


**Fig. 9. Example results on the Corel and Sowerby databases.** A different set of object class labels and thus different color-coding is used here. Textual labels are superimposed for clarity.

This latter effect is probably due to inaccuracies in the manual ground-truth labeling where pixels belonging to such classes are often labeled as grass near the boundary.

**Comparison with existing methods.** To assess how much the shape and context modeling help with recognition we have compared the accuracy of our system against the framework of [4], i.e. given a (manually) selected region, assign one single class label to it and then measure classification accuracy. On the 21-class database, our algorithm achieves 70.5% region-based recognition accuracy beating our implementation of [4] which achieves 67.6% using 5000 textons and their Gaussian class models. Moreover, the significant advantages of our proposed algorithm are that: i) no regions need to be specified manually, ii) a pixel-wise labeling (segmentation) of the image is obtained.

We have also compared our results with those of He et al [1] on their Corel and Sowerby databases, as shown in table 1 and fig. 9. For both models we show the results of the unary classifier alone as well as results for the full model. For the Sowerby database the parameters were set as $M = 6500$, $K = 250$, $\boldsymbol{\theta}_\phi = [10, 2]^T$, and $w_\lambda = 2$. For the Corel database, all images were first automatically color and intensity normalized and the training set was augmented by applying random affine intensity changes to give the classifier improved invariance to illumination. The parameters were set as $M = 5000$, $K = 400$, $\boldsymbol{\theta}_\phi = [20, 2]^T$, and $w_\lambda = 4$.

Our method gives comparable or better (with unary classifier alone) results than [1]. However, the careful choice of efficient features and learning techniques,

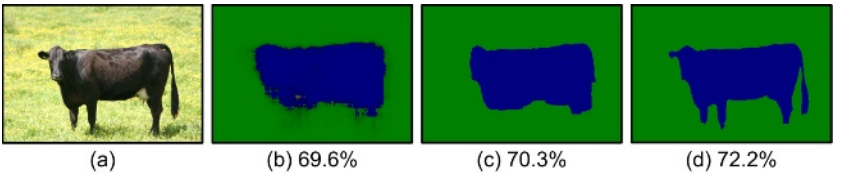


**Fig. 10. Effect of different model potentials.** The original input image (**a**) and the result from the boosted classifier alone (**b**), with no explicit spatial coherency; brighter pixels correspond to lower entropy of the unary potentials. (**c**) Results for the CRF model without color modeling, i.e. omitting term $\pi$ in (1), and (**d**) for the full CRF model. Segmentation accuracy figures are given over the whole dataset. Observe the marked improvement in perceived segmentation accuracy of the full model over the boosted classifier alone, despite a seemingly small numerical improvement.

and the avoidance of inefficient Gibbs sampling enables our algorithm to scale much better with the number of training images *and* object classes. Incorporating *semantic* context information as [1] is likely to improve our performance.

**The effect of different model potentials.** Figure 10 shows results for variations of our model with different potentials included. It is evident that imposing spatial coherency (c) as well as an image dependent color model (d) improves the results considerably. The percentage accuracies in fig. 10 show that each term in our model captures essential information from the training set. Note that the improvement given by the full model over just the unary classifiers, while numerically small, corresponds to a significant increase in perceived accuracy (compare fig. 10b with 10d) since the object contour is accurately delineated.

## 6 Conclusions

This paper has presented a new discriminative model for efficient recognition and simultaneous semantic segmentation of objects in images. We have: i) introduced new features which capture simultaneous appearance, shape and context information, ii) trained our model efficiently by exploiting both boosting and piecewise training techniques, iii) achieved efficient labeling by a combination of integral image processing and feature sharing. The result is an accurate algorithm which recognizes and locates a large number of object classes in photographs.

In the future we hope to integrate explicit *semantic* context information such as in [1] to improve further the classification accuracy. We are also interested in learning object parts (for structured objects) and their spatial arrangement. While we currently capture shape and thereby some implicit notion of objects 'parts', an explicit treatment of these would better model structured objects.

**Acknowledgements.** The authors would like to thank Florian Schroff, Roberto Cipolla, Andrew Blake and Andrew Zisserman for their invaluable help.

## References


1. He, X., Zemel, R.S., Carreira-Perpiñán, M.A.: Multiscale conditional random fields for image labeling. Proc. of IEEE CVPR (2004)
2. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: CVPR'03. Volume II. (2003) 264–271
3. Berg, A.C., Berg, T.L., Malik, J.: Shape matching and object recognition using low distortion correspondences. In: CVPR. (2005)
4. Winn, J., Criminisi, A., Minka, T.: Categorization by learned universal visual dictionary. Int. Conf. of Computer Vision (2005)
5. Kumar, S., Herbert, M.: Discriminative fields for modeling spatial dependencies in natural images. In: NIPS. (2004)
6. Borenstein, E., Sharon, E., Ullman, S.: Combining top-down and bottom-up segmentation. In: Proceedings IEEE workshop on Perceptual Organization in Computer Vision, CVPR 2004. (2004)
7. Winn, J., Jojic, N.: LOCUS: Learning Object Classes with Unsupervised Segmentation. Proc. of IEEE ICCV. (2005)
8. Kumar, P., Torr, P., Zisserman, A.: Obj cut. Proc. of IEEE CVPR. (2005)
9. Leibe, B., Schiele, B.: Interleaved object categorization and segmentation. In: BMVC'03. Volume II. (2003) 264–271
10. Duygulu, P., Barnard, K., de Freitas, N., Forsyth, D.: Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. ECCV (2002)
11. Tu, Z., Chen, X., Yuille, A.L., Zhu, S.: Image parsing: Unifying segmentation, detection, and recognition. In: CVPR. (2003)
12. Konishi, S., Yuille, A.L.: Statistical cues for domain specific image segmentation with performance analysis. In: CVPR. (2000)
13. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML. (2001)
14. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. Proc. of IEEE ICCV. (2001)
15. Rother, C., Kolmogorov, V., Blake, A.: Interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics (SIGGRAPH'04). (2004)
16. Sutton, C., McCallum, A.: Piecewise training of undirected models. In: 21st Conference on Uncertainty in Artificial Intelligence. (2005)
17. Leung, T., Malik, J.: Representing and recognizing the visual appearance of materials using three-dimensional textons. IJCV **43** (2001) 29–44
18. Varma, M., Zisserman, A.: A statistical approach to texture classification from single images. International Journal of Computer Vision: Special Issue on Texture Analysis and Synthesis **62** (2005) 61–81
19. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR01. (2001) I:511–518
20. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. PAMI **24** (2002) 509–522
21. Torralba, A., Murphy, K., Freeman, W.: Sharing features: efficient boosting procedures for multiclass object detection. Proc. of IEEE CVPR (2004) 762–769
22. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Technical report, Dept. of Statistics, Stanford University. (1998)
23. Baluja, S., Rowley, H.A.: Boosting sex identification performance. In: AAAI. (2005) 1508–1513
24. Kumar, S., Hebert, M.: A hierarchical field framework for unified context-based classification. In: ICCV05. (2005) II: 1284–1291