Object Class Segmentation using Random Forests

F. Schroff[1], A. Criminisi[2], A. Zisserman[1]
[1]Dept. of Engineering Science, University of Oxford
{schroff,az}@robots.ox.ac.uk
[2]Microsoft Research Ltd., antcrim@microsoft.com

Abstract

This work investigates the use of Random Forests for class based pixel-wise segmentation of images.

The contribution of this paper is three-fold. First, we show that apparently quite dissimilar classifiers (such as nearest neighbour matching to texton class histograms) can be mapped onto a Random Forest architecture. Second, based on this insight, we show that the performance of such classifiers can be improved by incorporating the spatial context and discriminative learning that arises naturally in the Random Forest framework. Finally, we show that the ability of Random Forests to combine multiple features leads to a further increase in performance when textons, colour, filterbanks, and HOG features are used simultaneously.

The benefit of the multi-feature classifier is demonstrated with extensive experimentation on existing labelled image datasets. The method equals or exceeds the state of the art on these datasets.

1 Introduction

This paper addresses the problem of simultaneously segmenting an image into its constituent semantic regions and automatically labelling each region as belonging to a class, out of a predetermined set of classes. The core discriminative classifier employed here is a modification of Random Forests.

Random Forests (often also referred to as Randomized Trees) were introduced in the machine learning community by [1, 4]. Their popularity in the computer vision community arose mainly from the work of Lepetit *et al.* in [12, 15], and a large number of papers have applied them to various supervised classification tasks [2, 5, 8, 13, 14, 18, 24, 25].

Appealing features of Random Forests include: (i) their computational efficiency in both training and classification, (ii) their probabilistic output, (iii) the seamless handling of a large variety of visual features (e.g. colour, texture, shape, depth etc.), and (iv) the inherent feature sharing of a multi-class classifier (see also [20] for feature sharing in boosting).

Previous work on class based image segmentation has often used quite simple local features for node tests (or weak classifiers), such as pixel differences, or differences between sums of filter responses accumulated over local rectangular regions [18, 19, 22, 24].

[2]This work was supported in part by Microsoft Research and the Royal Academy of Engineering.

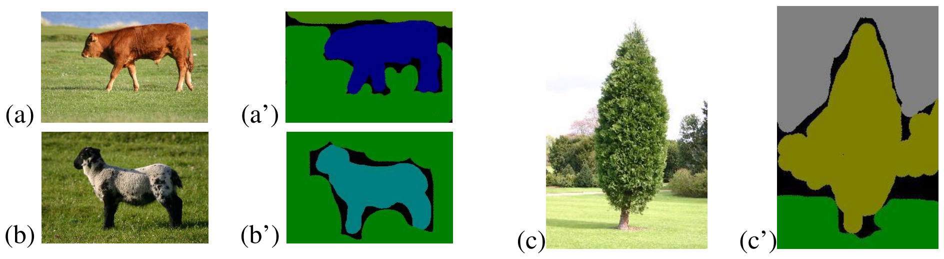

Figure 1: **The MSRC labelled image dataset**. (a–c) Example images from the MSRC dataset and (a'–c') their ground-truth class segmentation maps. The colour indicates the object class, with different kinds of grass, for example, associated with a common "grass" label. Note the presence of unlabelled (black) pixels.

In contrast, Schroff *et al.* [17] achieved pixel-wise classification by comparing histograms estimated over sliding windows to pre-learned, global "single-histogram class models"' (SHCM).

This paper connects Random Forest classifiers using local features to classifiers using nearest neighbour matching class models, such as Schroff *et al.* [17]. We show that a Random Forest classifier can improve over the classification performance of pre-learned class models, since the local features can incorporate spatial context, and the forest allows multi-class discriminative learning. Experiments carried out on standard labelled datasets demonstrate that our results are state of the art or better.

Datasets. The validity of our approach is demonstrated by extensive quantitative experimentation on two datasets. The initial experiments and the comparison with [17, 21] are performed on the 9-class MSRC dataset ([6]). This is a subset of the 21-class version [6], which is used for comparisons with [19]. Both these datasets consist of natural images and their corresponding ground truth object class maps (see fig. 1). We also report the performance on the challenging VOC2007 [9] data.

Paper outline. Section 2 describes the core Random Forest classifier. Section 3 presents the main contribution, describing the mapping of global class models into Random Forests, and section 4 shows how relaxing constraints on the node functions yields improved classification. Section 5 extends the set of visual features further. Finally, section 6 imposes spatial smoothing via a Conditional Random Field (CRF) model. Next we introduce the Random Forests classifier.

2 Random Forests

Decision tree classifiers have been known for a long time [16] but they have shown problems related to over-fitting and lack of generalization. The main idea behind Random Forest is to try and mitigate such problems by: (i) injecting randomness into the training of the trees, and (ii) combining the output of multiple randomized trees into a single classifier. Random Forests have been demonstrated to produce lower *test* errors than conventional decision trees [25] and performance comparable to SVMs in multi-class problems [2], while maintaining high computational efficiency.

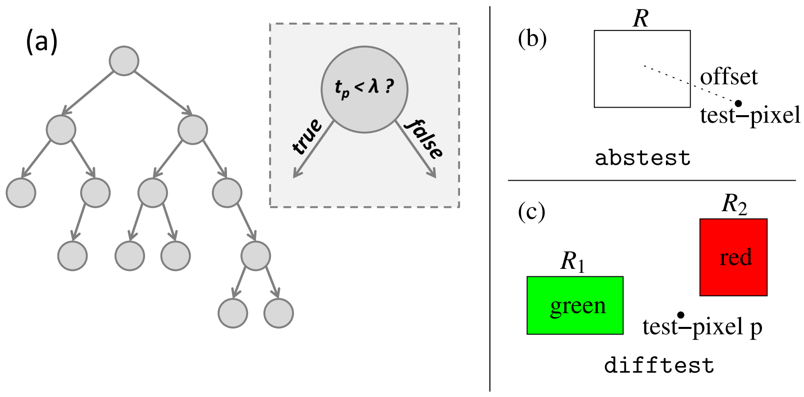

Figure 2: **Decision trees and node functions**: **(a)** Binary decision tree with its node functions t. **(b)** A node function can be based on filter responses accumulated over a single rectangular region as in [19]. **(c)** Alternatively, responses accumulated over two rectangles can also be used [25]. The pixel differences in [12] and Haar-like responses in [22] are special cases.

2.1 Decision trees

In this work we consider binary trees (fig. 2a), with their structure and decision nodes learned discriminatively as follows. Starting from the root, given the labelled training data, the function t and threshold λ which maximize the information gain are found. Then training proceeds down to the children nodes.

Note that the underlying features can be of general nature, *e.g.*: RGB, HOG [7], the output of a filter bank *etc*. Also the function t is of a very general nature: it could be a sum of feature responses in a defined spatial extent (fig. 2b), a linear classifier, the output of another tree, the output of AdaBoost *etc*.

For a pixel in position p, the node function t_p may be described as:

$$t_p = \sum_{r \in \mathscr{S}} \mathbf{w}_r \cdot \mathbf{f}_r \qquad (1)$$

where r indexes one or two rectangles as described in fig. 2(b,c) (*i.e.* $\mathscr{S} = \{1\}$ or $\{1,2\}$), $\mathbf{w}_r$ describes a filter selecting the pixels in the rectangle $\mathtt{R}_r$ and a weighting for each dimension of the feature vector $\mathbf{f}_r$ (a concatenation of all (used) feature channels and pixels in $\mathtt{R}_r$, *e.g.* $\mathbf{f}_1 = [G_1\ G_2 \cdots G_n]$, if $\mathtt{R}_1$ accumulates over the green channel G, and n is the number of pixels in $\mathtt{R}_1$).

In this work we use two types of node functions (see fig. 2b,c): (i) `abstest` [19] consists of the accumulated response in one feature channel in a single rectangle ($\mathscr{S} = \{1\}$). This is described in (1) where $\mathbf{w}_r$ is 1 for all pixels; (ii) `difftest` [24] is defined by two rectangles ($\mathscr{S} = \{1,2\}$), positioned relative to the test-pixel p. Each of them defines an area for which the cumulative response of a specific channel is computed. The node function t is the difference of the aggregated values from the two rectangles. This is described in (1) where the weight vector is $+1$ for all pixels in $\mathtt{R}_1$ and -1 for all pixels in $\mathtt{R}_2$. $\mathbf{f}_r$ is a concatenation of the values of one specific feature channel for all pixels in $\mathtt{R}_r$.

2.2 Training and testing Random Forests

Random Forests are composed of multiple independently learnt random decision trees.

During training of the tree each node has available only a *randomly* chosen subset of the entire pool $\mathscr{P}$ of possible node functions (*i.e.* variously shaped non-centred rectangles accumulating different channels). Training is achieved by finding the node function and threshold λ which yields maximum information gain within such restricted, randomized search space [2, 24]. Note, [12, 14] do not optimize λ. The training data is subsampled to increase independence of the trees [4] and reduce training time. During training the empirical class posterior distributions are stored in the leaf nodes.

Testing proceeds by applying all trees to each input pixel in the test image. The output per-pixel class posterior is achieved as the average of the posteriors across all trees.

3 Fusing global and local cues into Random Forests

The previous section has given a general model for the kinds of node functions typically used in Random Forests. This section extends the node function to incorporate *global* knowledge about the object classes. First we review the single-histogram class models (SHCM) introduced in [17] and then show how those can effectively be mapped within an efficient Random Forests classifier.

3.1 Global class modeling via SHCM

In [17] it is shown that given an image patch, the MAP estimate of its class is equivalent to a nearest neighbour classifier to single class histograms, where similarity is measured using KL divergence. In [17] a texton representation is used, and each class SHCM is computed as the average of texton distributions across all training exemplar images. In the implementation, the textons are computed on dense 5×5 color patches using k-means clustering, resulting in a 8000 texton dictionary.

The resulting SHCMs are used to segment a test image into classes by a sliding window classifier. In detail, a sliding window s is used to assign the object class $\hat{i}$ to the centre pixel p of s. With $\mathbf{h}$ being the distribution over textons in window s and $\mathbf{q}^i$ being the SHCMs the classification result for p is $\hat{i}$:

$$\mathbf{q}^{\hat{i}} = \arg\min_{\mathbf{q}^i} \left(KL(\mathbf{h}||\mathbf{q}^i) \right) \qquad (2)$$

Note that the NN classifier in [17] is *not* learnt discriminatively and does *not* learn spatial information about the object classes.

3.2 Casting SHCM classification as a node function

We first note that finding the NN class histogram using (2) can be accomplished by a series of hierarchical tests that compare the test window histogram $\mathbf{h}$ to a pair of SHCMs and select the more likely one. The goal is to combine this pairwise comparison into a single efficient node function that can be used in a decision tree. This can be achieved by combining two SHCMs $\mathbf{q}^i$ and $\mathbf{q}^j$ into $\mathbf{w}^{i,j} = \log\left(\frac{\mathbf{q}^i}{\mathbf{q}^j}\right)$. Now we define the node function which compares the KL-Divergence of a *single* test-histogram $\mathbf{h}_r$ (computed from one rectangle $\mathtt{R}_r$, similar to `abstest` in fig. 2) to two class models $\mathbf{q}^i$ and $\mathbf{q}^j$ in a single test (3):

$$KL(\mathbf{h}_r||\mathbf{q}^i) < KL(\mathbf{h}_r||\mathbf{q}^j) \quad \Leftrightarrow \quad t_p < 0 \text{ with } t = \mathbf{w}^{i,j} \cdot \mathbf{h}_r \qquad (3)$$

Note that unlike in (1) here $\mathbf{w}^{i,j}$ is not just 1 or -1, but each feature channel (here texton) is weighted by the globally induced weights. The node function t_p defined in (3) is smaller than 0, if and only if class model i is more likely to explain the rectangle $\mathtt{R}_r$ than class model j. This equivalence is verified experimentally in sec. 4.

If there are n object classes, *i.e.* n SHCMs $\mathbf{q}^i$ with $i \in \{1,\dots,n\}$, a tree of depth $n-1$ can compute $\arg\min_{\mathbf{q}^i}\left(KL(\mathbf{h}||\mathbf{q}^i)\right)$, by means of a series of the tests defined in (3). This scheme is called *fixed-tree* in the following.

Consider the case where the class distributions $\mathbf{q}^i$ and $\mathbf{q}^j$ have identical counts (*i.e.* similar occurrence frequencies) in all but one bin/channel. Then $\mathbf{w}^{i,j} = \log\left(\frac{\mathbf{q}^i}{\mathbf{q}^j}\right)$ will be zero for all but one weight. This is the limiting case of a single discriminative texton between the two classes. It shows that class histograms can underpin the features used successfully in two weak detectors/node functions: (i) TextonBoost [18, 19], where textons that are discriminative for a pair of classes are learnt (in that case using boosting) *i.e.* all weights but one are zero in (3); and (ii) the linear classifier (with a set of non-zero $\mathbf{w}^{i,j}$s) used in the node function of [2].

Based on this connection we now use class histograms and the patch classifier of [17] as a starting point for a Random Forest classifier. By casting such classifiers into the Random Forest framework we can introduce discriminative learning, and spatial information can be learnt by the use of less restricted patches and weightings.

4 From SHCM to local features

One of the main contributions of this work is that of combining the efficiency and discriminative power of decision trees with the descriptive power of global class models.

Here we show that introducing flexibility to the *fixed-tree* (see sec. 3.2) node functions which were inspired by global class models improve the performance. This is due to the fact that the Random Forests are able to learn multiple classes discriminatively (unlike binary SVMs for example). Additionally, a discriminative selection of the extent of the spatial support and context awareness result in a performance increase. The following experiments progressively introduce more freedom into the feature selection process: (i) *Fixed-tree*: learn the posterior from the data, (ii) *Learnt tree*: select the basic test *i.e.* the pair (i,j), (iii) *Flexible rectangles*: explore various aspects of the rectangles position (*shape* of the window w and an *offset*, *i.e.* the test-pixel not restricted to the centre of the rectangle).

4.1 Experimental evaluation

All the experiments in this section (except the *fixed-tree*) use a Random Forest of size 10 and a maximum depth for each tree of 15. During training the one, out of 100 randomly sampled node functions as in (3), that maximises the information gain is picked for each node. In this case λ is set to zero following (3). All results are on the 9-class MSRC dataset.

Fixed-tree. We use the *single* fixed-tree and just learn the class posteriors, by applying the tree to each training point. Using this tree gives virtually the same performance 75.2% as reported in [17] with the same features (25×25 pixel wide centred windows).

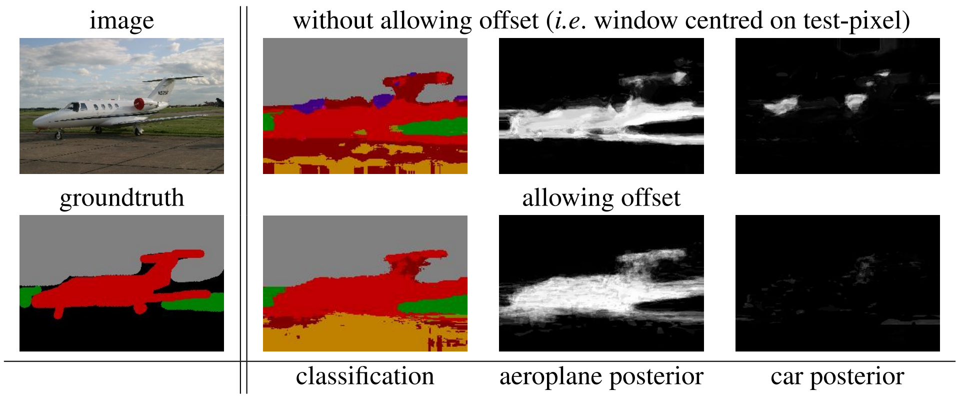

Figure 3: **Spatial Context**: This figure shows how an offset incorporates spatial context. The classification with a maximum offset of 30 is in the bottom row, the one with the test-pixel centred in the rectangle in the top row. Next to the classification are shown the posteriors for the object class (aeroplane) and the car class (marked "purple" in classification) that is confused with aeroplane, if no context is used.

window size\offset	0	10	20	30
25:25	76.2%	78.8%	81.0%	82.2%
10:35	77.7%	79.3%	80.8%	82.0%

Table 1: **Spatial influence**: The window size [smallest extent : largest extent] describes the range of the width and height of the rectangles used in the node function. The offset describes the maximum distance the centre of the rectangle can have to the test-pixel. Allowing for additional "freedom" especially a large offset clearly improves the performance.

This is a nice result indicating that the class histogram $\mathbf{q}^i$ minimizing $\left(KL(\mathbf{h}||\mathbf{q}^i)\right)$ indeed corresponds to the MAP of the class.

Learnt tree (fixed rectangles). Instead of using the fixed tree of depth 8 (9 classes -1), we learn a Random Forest keeping the rectangle size fixed to 25×25 and centred on the test pixel. Incorporating the flexibility to select the pair (i, j) in each node increases the performance marginally to 76.2% (see tab. 1).

Flexible rectangles. Allowing for different rectangle *shapes/sizes* as well as *off centre test-pixels* (see fig. 2a) and thereby taking full advantage of the discriminative and spatial learning of Random Forests results in a dramatic performance boost. The node functions itself still follow (3).

Tab. 1 shows that allowing different *window sizes* (smallest edge of rectangle can be 10, largest 35 pixels wide) improves the performance, as these rectangles can adapt to the objects and encode fine/large grained structures. The rectangles are around 23 ± 7 pixels each dimension and tend to be slightly smaller towards the bottom of the tree. The average ratio of small side to long side is 0.69 ± 0.19, *i.e.* to capture the full shape of the objects "real" elongated rectangles are advantageous.

However, a significantly larger improvement is achieved by allowing an *offset* of the centre of the rectangles to the test-pixel, in which case the context of the test-pixel is taken into account. The offset is around 15 ± 8 pixels in the x and y direction. Fig. 3 illustrates that the Random Forest learns that car is unlikely to occur next to aeroplane

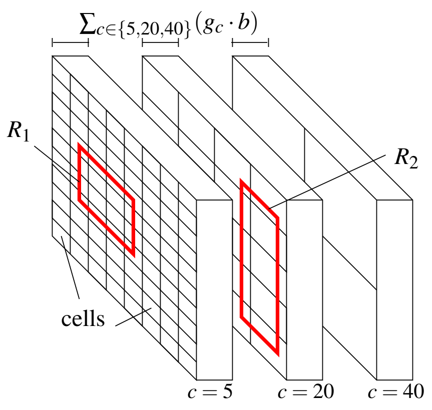

Figure 4: **HOG features**: We use stacked HOG features computed with different cell-sizes $c = \{5,20,40\}$ and gradient binnings $g = \{6,6,12\}$. One block always contains 4 cells ($b = 4$). Each rectangle response is computed by adding the responses for one of the HOG dimensions of a pixel (also see fig. 4). Cells partially included in the rectangle area contribute proportionally to the included area.

and sky. The important observation is that confusions like aeroplane with car are dramatically suppressed. This is indicated by the posterior response for the car class, where the two high response areas in the top right image. It is worth noting that the classification obtained by the context features (bottom row) has a less crisp object boundary, which can be explained by the fact that pixels occurring near the boundary have a higher likelihood for both classes when context is incorporated (*e.g.* aeroplane/grass confusion).

Conclusion. Using *a priori* knowledge (SHCMs) to compute a weighted sum of textons occurring in a rectangular region near the test-pixel provides strong initial node functions. The functions are improved by introducing flexibility and this increases performance from 76.2% to 82.0% (see tab. 1), exceeding the performance of the original features in [17] dramatically.

5 Introducing more visual cues

In the previous section we laid out how "reusing" features suitable for nearest neighbour classification can result in powerful features for Random Forests and by exploiting properties of these, namely discriminative selection of parameters (window size, offset) performance can be improved.

Now we relax the restrictions on the features even further and introduce additional cues. We focus on the node functions described in fig. 2b and apply them first to RGB and then to HOG [7]. Finally all features will be combined in the decision trees of a Random Forest, and additionally filterbanks will be used. The optimal most suitable features are selected by means of the discriminative learning of the decision trees.

RGB features. The node functions are simple differences of rectangles computed over one of the three channels (R, G, or B) per rectangle (see sec. 2.1). Our baseline classifier uses single-histogram class models based on pixel-wise colour histograms of size 8000 (20 bins for each colour) and gives a pixel-wise classification performance of 67.3%. This coincides with the findings of [21], who report 67.1% for simple appearance based features. Tab. 2 shows the performance of RGB only, and again the flexibility in selecting the shape and offset of the rectangles results in increased performance (72.2%).

HOG features. As RGB based Random Forests are not strong enough to capture the spatial variation of classes, we also introduce features based on the HOG descriptor [7]. We compute the HOG for the whole image using various cellsizes c in pixels, blocksizes b in cells, and number of gradient bins g. This leads to a $g \cdot b$ dimensional feature vector for each cell (see fig. 4). The stacked HOG consists of $c = \{5,20,40\}$ and $g = \{6,6,12\}$

	RGB	HOG	HOG,RGB	HOG,RGB,T	CRF	Others
9-class	72.2%	75.9%	84.5%	84.0%	**87.2%**	84.9% [21]
21-class	54.0%	56.3%	69.9%	69.7%	**71.7%**	72.7% [19]

Table 2: **Feature combinations:** Pixelwise classification performance on the 9 and 21-class MSRC datasets using RGB, HOG and texton (T) features. For the 9-class dataset the forest consists of 20 trees, each of a maximum depth 20; each node test is picked as the best of 500 node functions per feature using information gain. For each node function the threshold λ which maximizes the information gain is selected. For the 21-class dataset a forest of size 30 was used.

oriented gradient bins for each of the c values (with $b = 4$ cells in each block). This results in $6 \cdot 4 + 6 \cdot 4 + 12 \cdot 4 = 96$ channels for each pixel p. It was important to have cellsizes c ranging from small 5 to large 40. The performance of HOG alone is shown in tab. 2.

F17 filterbank. The filterbank described in [23] is used as an additional cue in the same manner as the RGB.

5.1 Combining features

We combine these various feature types, whereby each feature (RGB, stacked HOG, textons T, F17) is treated as a separate pool and the feature maximising the information gain is selected for each node.

Tab. 2 show the results of various feature combinations. Using all features combined by exploiting the feature selection properties of Random Forests gives state of the art results 84.5% on the MSRC 9-class, and 69.9% on the 21-class ([19] achieved 69.6% with a boosted classifier). It can clearly be seen that the combination of RGB and HOG gives a *very* strong boost in performance. The addition of texton features (although very strong alone) is not able to increase performance (probably due to redundancy with HOG and RGB), but the discriminative learning assures that the right features are selected and performance does not decrease, thus the classifier is able to improve over state of the art.

VOC2007 [9]. On the very challenging VOC data our method using RGB,HOG, and F17 achieves state state of the art performance ([18] reaches an average class performance of 20%), and in combination with the CRF is slightly better (see following section).

6 CRF stage

After having learnt the class posteriors using the methods described in section 3, a CRF [11] stage is applied to the pixel grid, where the state space of each pixel is the number of classes, *e.g.* 9 in the case of the 9-class MSRC dataset.

The class posteriors from the Random Forest are used as unary potentials ψ and constitute the most important part of the CRF-model. Additional terms are the contrast sensitive Potts model ϕ [3] and a colour term π, based on a Gaussian mixture model estimated from labels inferred using (4) with $\pi = 0$, on a *per* image basis. The colour model is included to use colours peculiar to that image, rather than simply the colour node tests learnt over all training images and included already in the posteriors/unary terms.

Given the image I, $\{ij \in I\}$ describes the set of edges (4 neighbourhood in our case) and $\mathbf{g}_{ij} \in \mathbb{R}^d$ describes the d dimensional colour difference vector, between pixel i and j.

$$\log P(\mathbf{c}|\mathbf{I}) = \sum_i (\psi_i(c_i, \mathbf{I}) + \pi(c_i, \mathbf{I})) + \sum_{ij \in I} \phi(c_i, c_j, \mathbf{g}_{ij}) \tag{4}$$

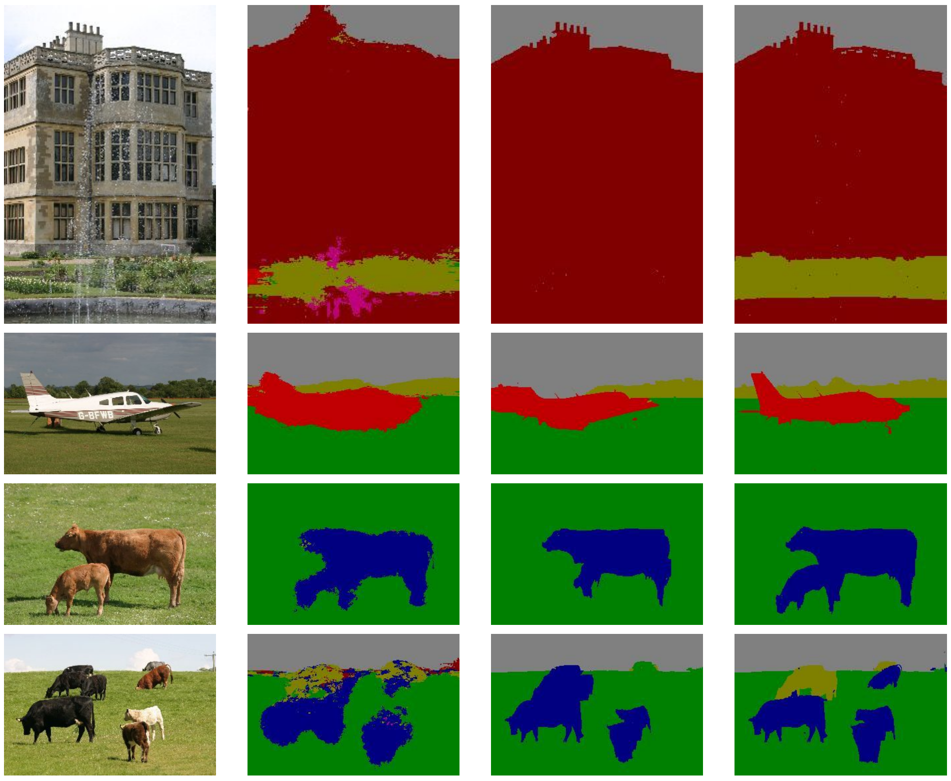

image	MAP posterior	CRF w/o colour	CRF with colour
acc. 9-class	84.4%	**87.2**%	84.8%

Figure 5: **The CRF** helps to retrieve crisper object boundaries. Using an additional per image colour model improves 'good' classifications further, but can hurt certain cases, *e.g.* last row. The accuracy for the whole dataset in all three cases. The features used here were stacked HOG and RGB.

describes the CRF energy with c_i being the label for pixel I_i. We use TRW [10] to minimize (4). The results in fig. 5, show that the CRF improves the performance slightly, but more importantly improves the visual appearance of the segmentation. Using this additional colour model improves 'good' segmentations further see fig. 5 (aeroplane, building), but can deteriorate 'bad' classifications (black cow in the last row). In general the results are very sensitive to the CRF parameters.

The performance on the 9-classes (87.2% without colour model; 84.8% with colour model) is shown in fig. 5. For the 21-classes we achieve 71.7% without colour model and 70.8% with the colour model. On the VOC2007 data we achieve 22% average class performance. These results exceed state of the art performance or perform equally well ([18, 19, 21]).

7 Conclusion & future work

This paper has shown how to incorporate globally learnt class models into a Random Forest classifier. It has also investigated the use of various global, local and context-rich feature types in Random Forests. The features used in other work [2, 12, 18, 19] are interpreted as special cases of the general family of features introduced here. Furthermore, we have demonstrated how relaxing different constraints on such features leads to state of the art classification accuracy on standard image datasets.

Interesting areas for future research include incorporating image level class 'priors' (as in [18]) to further boost performance, and moving from direct pixel classification to an intermediate object detection stage before pixel classification.

References

[1] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588, 1997.

[2] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Proc. ICCV*, 2007.

[3] Y. Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *Proc. ICCV*, volume 2, pages 105–112, 2001.

[4] L. Breiman. Random forests. *ML Journal*, 45(1):5–32, 2001.

[5] H. A. Chipman, E. L. George, and R. E. McCulloch. Bayesian Ensemble Learning. In *NIPS*, 2006.

[6] A. Criminisi. Microsoft research cambridge object recognition image dataset. version 1.0, 2004.

[7] N. Dalal and B Triggs. Histogram of Oriented Gradients for Human Detection. In *Proc. CVPR*, volume 2, pages 886–893, 2005.

[8] T. Deselaers, A. Criminisi, J. Winn, and A. Agarwal. Incorporating On-demand Stereo for Real-Time Recognition. In *Proc. CVPR*, 2007.

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[10] V. Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization, 2005.

[11] J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proc. ICML*, 2001.

[12] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE PAMI*, 28(9):1465–1479, 2006.

[13] R. Marée, P. Geurts, J. Piater, and L Wehenkel. Random Subwindows for Robust Image Classification. In *Proc. CVPR*, 2005.

[14] F. Moosman, B. Triggs, and F. Jurie. Fast Discriminative Visual Codebook using Randomized Clustering Forests. In *NIPS*, 2006.

[15] M. Ozuysal, P. Fua, and V. Lepetit. Fast Keypoint Recognition in Ten Lines of Code. In *Proc. CVPR*, 2007.

[16] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[17] F. Schroff, A. Criminisi, and A. Zisserman. Single-Histogram Class Models for Image Segmentation. In *Proc. ICVGIP*, 2006.

[18] J. Shotton, M. Johnson, and R. Cipolla. Semantic Texton Forests for Image Categorization and Segmentation. In *Proc. CVPR*, 2008.

[19] J. Shotton, J. Winn, C. Rother, and A. Criminisi. *TextonBoost*: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proc. ECCV*, pages 1–15, 2006.

[20] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multi-class object detection. In *Proc. CVPR*, pages 762–769, 2004.

[21] J. Verbeek and Triggs B. Scene Segmentation with CRFs Learned from Partially Labeled Images. In *NIPS*, 2008.

[22] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. CVPR*, pages 511–518, 2001.

[23] J. Winn, Criminisi, A., and T. Minka. Object Categorization by Learned Universal Visual Dictionary. *Proc. ICCV*, 2005.

[24] J. Winn and A. Criminisi. Object Class Recognition at a Glance. In *Proc. CVPR*, 2006.

[25] P. Yin, A. Criminisi, J. Winn, and I. Essa. Tree-based Classifiers for Bilayer Video Segmentation. In *Proc. CVPR*, 2007.