Oriented Edge Forests for Boundary Detection

Sam Hallman, Charless C. Fowlkes
Department of Computer Science, University of California, Irvine

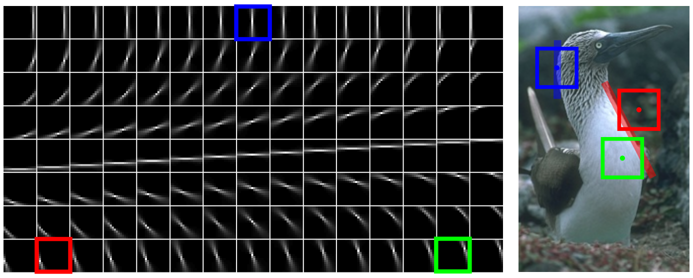

Figure 1: Our boundary detector consists of a decision forest that analyzes local patches and outputs probability distributions over the space of oriented edges passing through the patch. This space is indexed by orientation and signed distance to the edge (d, θ). These local predictions are calibrated and fused over an image pyramid to yield a final oriented boundary map.

In recent years, work in boundary detection has seen a strong push toward data-driven approaches. For example, [6] uses sparse coding to improve the hand-designed gradient features from [1], while [4, 5] predict the probability of a boundary at an image location using a cascade or randomized decision forest built over simple image features. The highly successful structured forest detector [3] evolved from [5] by moving from a fixed set of classes to an unrestricted output space, and it was posited that this was key to its performance advantage.

In this work we show that, in fact, with the right set of fixed clusters the resulting model outperforms published results on the challenging BSDS500 boundary detection benchmark. Specifically, our approach applies the robust machinery of random decision forests to the simple task of accurately detecting straight-line boundaries at different candidate orientations and positions within a small image patch (Fig. 1). Unlike SCG [6] or, [5], which also trains a forest from a fixed set of cluster labels, we are not restricted to predicting edges that pass through the center of the detection window.

This space of edge patterns has a simple parameterization. From a ground truth boundary image, we categorize patches either as non-boundary or as belonging to one of a fixed number of edge categories. Boundary patches are labeled according to the distance d and orientation θ of the edge pixel closest to the patch center. Thus, patches with $d = 0$ have an edge running through the center. This label space is illustrated in Fig. 1. We bin the space of distances and angles and assign every boundary patch one of K discrete labels. This allows for easy application of off-the-shelf decision forest training code for k-way classification.

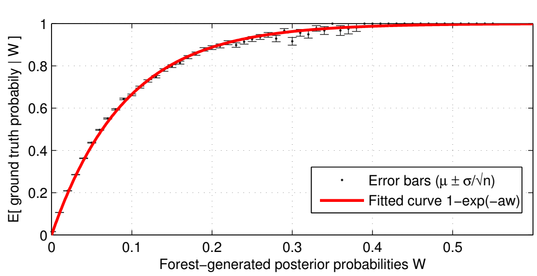

Figure 2: We provide a simple procedure for calibrating forest-generated posterior probabilities. This reliability plot shows the empirical probability of a ground-truth edge label as a function of the score output by the forest. The red curve shows a simple functional fit $1 - \exp(-\beta w)$ which appears to match the empirical distribution well. Per-scale calibration prior to combining predictions across scales improves performance.

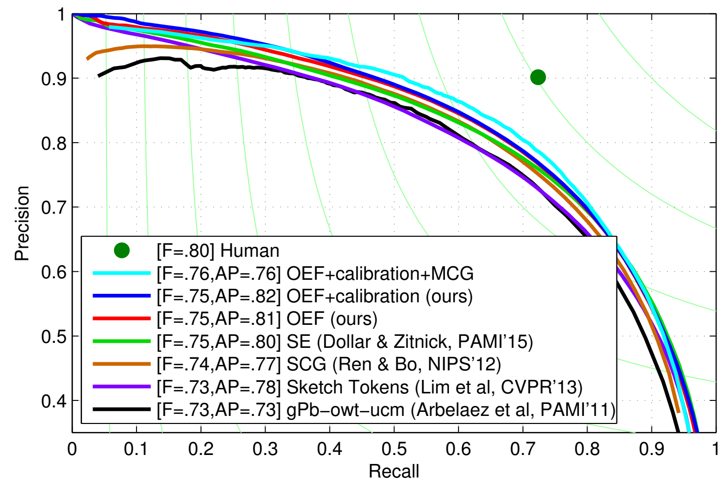

Figure 3: Despite its simplicity, our decision forest model achieves top performance on BSDS (red curve). An additional step of calibrating the posterior probabilities emitted by the forest further increases performance (blue curve). The full system runs in 2 seconds per image and, if combined with MCG [2], achieves an F-measure of 0.76 (cyan curve).

Once a forest is trained to recognize these oriented edge patterns, we apply it over the input image in a scanning-window fashion, generating a posterior distribution over the $K+1$ classes at each location. Because these distributions express the likelihood of both centered ($d = 0$) as well as distant, off-center ($d \neq 0$) edges, the probability of boundary at a given location is necessarily determined by the tree predictions over an entire neighborhood around that location. We show that it helps to *calibrate* these predicted probabilities to match the true posterior distribution over edge types for that patch. Our calibration approach is illustrated in Fig. 2. We aggregate calibrated predictions across the whole image by compositing the boundary map predictions into the image, each weighted by its posterior probability.

The final system achieves state-of-the-art performance. Fig. 3 shows the performance of our model on the BSDS500 test set over the full range of operating thresholds. Our system outperforms all existing methods in the high precision regime, and is virtually identical to SE [3] at high recall. As a result, AP is increased a full *two points* over competing methods. A conclusion of our work is that simply detecting oriented lines is as effective as any of the more complex strategies that attempt to model corners, junctions, and other complex edge patterns.

[1] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *PAMI*, 2011.

[2] Pablo Arbeláez, Jordi Pont-Tuset, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping. CVPR, 2014.

[3] Piotr Dollár and C. Lawrence Zitnick. Fast edge detection using structured forests. *PAMI*, 2015.

[4] Piotr Dollar, Zhuowen Tu, and Serge Belongie. Supervised learning of edges and object boundaries. In *CVPR*, pages 1964–1971. IEEE, 2006.

[5] Joseph J Lim, C Lawrence Zitnick, and Piotr Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In *CVPR*, pages 3158–3165. IEEE, 2013.

[6] Ren Xiaofeng and Liefeng Bo. Discriminatively trained sparse code gradients for contour detection. In *NIPS*, pages 584–592, 2012.

This is an extended abstract. The full paper is available at the Computer Vision Foundation webpage.