Supervised Learning of Edges and Object Boundaries

Piotr Dollár *
Computer Science & Engineering
University of California, San Diego
pdollar@cs.ucsd.edu

Zhuowen Tu *
Lab of Neuro Imaging
University of California, Los Angeles
zhuowen.tu@loni.ucla.edu

Serge Belongie
Computer Science & Engineering
University of California, San Diego
sjb@cs.ucsd.edu

Abstract

Edge detection is one of the most studied problems in computer vision, yet it remains a very challenging task. It is difficult since often the decision for an edge cannot be made purely based on low level cues such as gradient, instead we need to engage all levels of information, low, middle, and high, in order to decide where to put edges. In this paper we propose a novel supervised learning algorithm for edge and object boundary detection which we refer to as Boosted Edge Learning or `BEL` *for short. A decision of an edge point is made independently at each location in the image; a very large aperture is used providing significant context for each decision. In the learning stage, the algorithm selects and combines a large number of features across different scales in order to learn a discriminative model using an extended version of the Probabilistic Boosting Tree classification algorithm. The learning based framework is highly adaptive and there are no parameters to tune. We show applications for edge detection in a number of specific image domains as well as on natural images. We test on various datasets including the Berkeley dataset and the results obtained are very good.*

1. Introduction

Edge detection is one of the most studied problems in computer vision. It finds application in tasks such as object detection/recognition, structure from motion, segmentation and tracking, *e.g*. [17, 18, 1, 21]. Edges reduce the dimensionality of the original data while retaining rich information about the contents of the image. They can also serve as a basis for other forms of image representation, such as the primal sketch [10, 8]. Nevertheless, high quality general edge detection remains elusive. Methods that rely on local features, such as the Canny [2] detector, do not take into account the context (*e.g*. if the surrounding area is textured), mid-level information (*e.g*. the Gestalt laws [12]), or high level information (*e.g*. object knowledge [19]). Canny also cannot take into account local information at multiple scales. Such information is important: sometimes we hallucinate a boundary where there is weak or even no local evidence (*e.g*. certain parts of an object may have the same intensity pattern as the background), other times we do not see a boundary even if there are strong local cues that would imply its existence (*e.g*. in the presence of shadows). Complex generative models, such as presented in [19, 15], have the potential to integrate both low-level and high-level information but present significant computational challenges.

Even as intensive research into general edge detection continues, there is general consensus in the community that edge detection is somewhat ill defined in that it is not quite clear what defines a correct output [11]. From an application driven point of view, a general edge detection algorithm is possibly inappropriate since relevant boundaries in a scene depend on the components of interest, which in turn depend on the task being performed. For example, if the goal is to detect object boundaries for object detection then all other detected edges are noise, but if the task changes the object boundaries may no longer be relevant. One of the motivations for this work is that while designing individual edge detectors for particular tasks in particular domains is not feasible, often times it is simple to obtain training images with labeled boundaries, and depending on the task there need not be any ambiguity as to what constitutes a correct output.

In this paper we propose a novel supervised learning algorithm for edge and object boundary detection which we refer to as Boosted Edge Learning or `BEL` for short. A decision of an edge point is made independently at each location in the image; a very large aperture is used providing significant context for each decision. In the learning stage, the algorithm selects and combines a set of features out of a pool with tens of thousands of generic, efficient Haar wavelets in order to learn a discriminative model. The scope of the machine learning problem is formidable: we have tens of millions of training points with tens of thousands of features each. We present an extension of the probabilistic

*This work was performed while PD and ZT were at Siemens Corporate Research.

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

boosting tree algorithm that copes with this data much better than did either boosting or cascade approaches [22]. Our method outputs true probabilities, whereas other edge detection methods either output a binary value or a soft value based on edge strength (which is not a true probability). We show how the method implicitly combines low-level, mid-level, and context information across different scales when making a decision. The learning based framework is highly adaptive and there are no parameters to tune. We show applications for edge detection in a number of specific image domains as well as on natural images. We test on various datasets including the Berkeley dataset and the results obtained are very good.

1.1. Related work

There have been many methods proposed for edge detection, we have already mentioned a few. A representative set is given by [2, 14, 19, 15], we review each in some detail below. The Canny edge detector [2] is perhaps the most widely used edge detector; it is based only on local gradient and has a scale parameter to tune. Ramesh [14] systematically analyze the performance of a number of edge detectors w.r.t. their parameter setting. The method in [19] uses edges obtained from bottom-up (discriminative) processes as proposals to guide the top-down (generative) search. Recently, Ren *et al*. [15] build graphs to reinforce some mid-level cues to give more complete edges. In general, however, it is difficult to encode all the rules needed for edge detection, for example how to exploit all sorts of mid-level Gestalt laws such as junction, parallelism, symmetry, and closure, how to deal with texture and color, how to resolve disagreements between cues that give conflicting local information, and so on, even if mid-level and high-level information can be modeled (for example in a generative framework), a search for optimal solutions can be daunting.

Two other relevant algorithms are `Pb`, proposed by [11], and the method presented in [13]. Both have data driven and learning components, although aside from this obvious similarity they bear little resemblance to our approach. `Pb` uses learning to perform cue combination on 9 carefully designed local features (texture gradient, brightness gradient and color gradient at 3 scales each), learning improves performance over setting the weights by hand. The learning component in both [11] and [13] improves the overall results of the algorithms on general edge detection; our method, however, relies entirely on learning. This makes our method very versatile, and as we show in the experiments section we were able to apply it to a very broad range of domains.

2. Problem formulation

A number of tasks in computer vision can be formulated as finding a likely interpretation W for an observed image

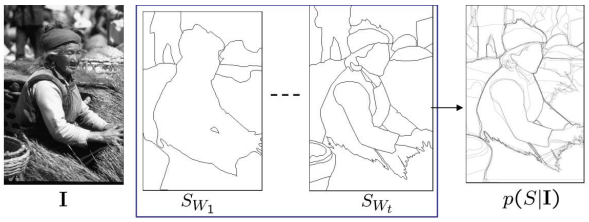


Figure 1. Manual segmentations of an example image and the corresponding edge maps. For illustration purpose, we show the negative of the probability so the darker the pixel, the higher the probability of an edge.

$\mathbf{I}$, where W includes information about the spatial location and extent of objects, regions, object boundaries, curves and so on. Let S_W be a function associated with a scene interpretation W that encodes the spatial location and extent of a component of interest, where $S_W(i,j)$ is 1 for each image location (i,j) that belongs to the component and 0 elsewhere. Given an image, obtaining an optimal or even likely scene interpretation W, or associated S_W, can be difficult. Instead, we can ask what is the probability a given location in a given image belongs to the component of interest:

$$p(S(i,j)|\mathbf{I}) = \sum_{W_t} S_{W_t}(i,j)p(W_t|\mathbf{I}). \quad (1)$$

See Figure (1). Calculating $p(S(i,j)|\mathbf{I})$ using the above is difficult. Instead we seek to learn this distribution directly from image data. To further reduce the complexity, we seek to learn a discriminative model $p(S(i,j)|\mathbf{I}_{N(i,j)})$ where $\mathbf{I}_{N(i,j)}$ is an image patch centered at (i,j). If we throw away the absolute coordinates, then the major focus of this paper is to learn:

$$p(S(c)|\mathbf{I}_{N(c)}), \quad (2)$$

where c is the center of an image patch. So the decision for a single point is made based on an image patch centered at it. A large enough patch contains low-level features and also some mid-level and context information. We want to piece this information together and learn a discriminative model.

Edge detection is easily placed in the above framework if we let the scene interpretation W of an image be its segmentation, and define $S_W(i,j) = 1$ if any region boundary in W passes through location (i,j), and otherwise $S_W(i,j) = 0$. Then $p(S(i,j)|\mathbf{I})$ gives the edge probability at each location in the image. To obtain ground truth for our discriminative model, we use the manually labeled segmentations from [11]. Given an image and a number of different segmentations W_t, we can obtain an approximation, $\hat{p}(S|\mathbf{I})$, by considering each of the human segmentations to have an equal probability $p(W_t|\mathbf{I})$. We can then sample positive and negative example patches from $\mathbf{I}$ according to $\hat{p}(S|\mathbf{I})$. Figure (1) shows an example where human subjects drew different segmentations. The edge probability map summing

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

up all manual segmentations is shown on the right. Figure (2) shows some sample patches.

We describe other applications of this framework, as well as other methods for generating the ground truth, in the experiments Section 4. However, for much of this paper we refer to $p(S(c)|\mathbf{I}_{N(c)})$ as the edge probability at c.

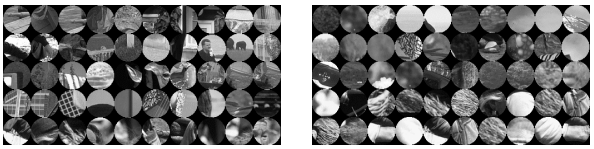
3. Learning edge probability

Our goal is to train a discriminative model $p(S(c)|\mathbf{I}_{N(c)})$ that predicts the probability of a location being an edge point based on an image patch centered at it.

3.1. Features

Informative features greatly facilitate the training stage of a classification algorithm. Their design should compromise among generality, speed and effectiveness. As mentioned, `Pb` performs cue combination on 9 features. These features are a culmination of years of effort, and they are in fact very effective for edge detection in natural images. Instead, our approach is to use tens of thousands of very simple features, calculated over a much larger image region. The primary advantage of such an approach is that the human effort is minimized, instead the work is shifted to the classification algorithm. We can measure the time it took to design and implement our features in terms of days. Also, such features tend to be much more general so applying the algorithm to a different domain is straightforward. Including features far from the center of the patch implicitly provides mid-level and context information to the discriminative algorithm. The primary disadvantage is that the classification stage is far more challenging and the choice of a discriminative algorithm more sensitive.

We used a large number of generic features at multiple locations, orientations, scales, aspect ratios and so on, calculated over a large image patch (*e.g.* of size 50×50). Features included gradients at multiple scales and locations, differences between histograms computed over filter responses (difference of Gaussian (DoG) and difference of offset Gaussian (DooG)) again at multiple scales and locations, and also Haar wavelets[22]. We experimented with using the output of the Canny edge detector at various scales as input to our method, although these Canny features were not very informative and for speed reasons were not included in the final version of our classifier. The classifier has to handle edges at different scales implicitly, and also different types of edges, so it is important to have a large pool of informative features. Integral images, filter responses, and so on were computed once for each input image, not once per patch, increasing efficiency when adjacent patches must be evaluated. For color images we additionally calculated the above features (gradients, histograms of filter responses, Haar wavelets) over each color channel.



(A) Positives (B) Negatives

Figure 2. (A) Some positive (center pixel is an edge point) and (B) negative (center pixel is none edge) image patches.

We use approximately 50000 features. The same features were used in all applications reported. We emphasize that little effort went into optimizing our feature set.

3.2. Classification framework

Given a training image, along with an estimate of the probability that each location is an edge point, $\hat{p}(S|\mathbf{I})$, we can sample positive and negative example patches. The number of samples from a single image is equal to the number of locations in the image (although typically the majority of locations contain negative samples), and even more if we consider the image at multiple orientations, scales and so on. Adjacent sample are highly similar, nevertheless, we often have very large training sets ($\mathcal{O}(10^8)$ samples). Learning an accurate decision boundary for this data is difficult, and we would like to use as much data as possible.

Advances in machine learning have allowed us to take advantage of the size of this high dimensional dataset. Boosting [4, 5] deals well with high dimensional data. Cascades of boosted classifiers [22] allow for efficient evaluation, and combined with bootstrapping allow for training on very large datasets. In an early attempt we trained a cascade of AdaBoost classifier, unfortunately the cascade did not give us sufficiently good results. Even with a large number bootstrapping stages the error remained fairly high (a comparison is given in the experiments section, specifically see Figure 7).

Instead we chose to use an extension of the probabilistic boosting tree (`PBT`) proposed in [20]. `PBT` can be seen as a combination of a decision tree with boosting (a cascade is then just a special case of a tree). In this paper, we give an extended `PBT` that combines the bootstrapping procedure directly into the tree formation while properly maintaining priors. We give a description of the algorithm below, our presentation of the extended `PBT` algorithm uses a different notation from [20].

3.3. Probabilistic Boosting Tree - training

Training `PBT` is similar to training a decision tree, except at each node a boosted classifier is used to split the data. The tree is trained recursively: at each node the empirical distribution $\hat{q}(y)$ of the data is calculated, and if the node is not pure ($0 < \hat{q}(y) < 1$), a strong classifier is trained on the data at the node. Each sample is then passed to the left and right subtrees, weighted by $q(-1|x_i)$ and $q(+1|x_i)$

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

respectively, where $q(+1|x_i)$ is the probability that x_i is a positive sample according to the strong classifier. Thus, the strong classifier at each node is used not to return the class of the sample but rather to assign the sample to the left or right subtree. Training proceeds recursively. Details for the extended version of the algorithm are given below, see [20] for information about the original algorithm. To train:

1. Given a set of images with edges annotated, retrieve a training set $S = \{(x_1, y_1, w_1), ..., (x_m, y_m, w_m);\ \ x_i \in \chi,\ \ y_i \in \{-1, +1\}, \sum_i w_i = 1$.
2. If the number (or weight) of either positive or negative samples in S is too small, perform bootstrapping to augment S (see below).
3. Compute the empirical distribution of S, $\hat{q}(y) = \sum_i w_i \delta(y_i = y)$. Continue if the depth of the node does not exceed some maximum value and $\theta \leq \hat{q}(+1) \leq (1 - \theta)$, *e.g.* $\theta = 0.99$, else stop.
4. On training set S, train a strong boosted classifier (with a limited number of weak learners).
5. Split the data into two sets S_L and S_R using the decision boundary of the learned classifier and a tolerance ϵ. For each sample (x_i, y_i, w_i) compute $q(+1|x_i)$ and $q(-1|x_i)$, then:
   $(x_i, y_i, w_i * q(+1|x_i)) \rightarrow S_R$
   $(x_i, y_i, w_i * q(-1|x_i)) \rightarrow S_L$.
   Finally normalize all the weights in S_L and also S_R.
6. Train the left and right children recursively using S_L and S_R respectively (go to step 2).

The bootstrapping step (2) for a given node is similar to bootstrapping when training a cascade, except that both positive and negative examples are bootstrapped. Let $l_1, ...l_k$ denote the path to the current node at depth $k + 1$. The weight of a sample (x, y, w) when it reaches the node in is given by $w' = w \prod q(x|l_i)$. By resampling the original data after reweighing according to the above (and renormalizing), one can augment the data S at the given node. This bootstrapping procedure allows us to deal with very large data sets while properly maintaining priors.

Finally, to make training more efficient, step (5) can be altered so a given sample is passed to both S_L and S_R only if it is near the decision boundary (the bootstrapping procedure must be altered accordingly):

If $q(+1|x_i) - \frac{1}{2} > \varepsilon$:
  $(x_i, y_i, w_i) \rightarrow S_R$
else if $q(-1|x_i) - \frac{1}{2} > \varepsilon$:
  $(x_i, y_i, w_i) \rightarrow S_L$
else:
  $(x_i, y_i, w_i * q(+1|x_i)) \rightarrow S_R$
  $(x_i, y_i, w_i * q(-1|x_i)) \rightarrow S_L$.

3.4. Computing probability

Given a trained tree, the posterior $\tilde{p}(y|x)$ is computed recursively. If the tree has no children, the posterior is simply the learned empirical distribution at the node $\tilde{p}(y|x) = \hat{q}(y)$. Otherwise the posterior is defined recursively:

$$\tilde{p}(y|x) = q(+1|x)\tilde{p}_R(y|x) + q(-1|x)\tilde{p}_L(y|x)$$

Here $q(y|x)$ is the posterior of the classifier, and $\tilde{p}_L(y|x)$ and $\tilde{p}_R(y|x)$ are the posteriors of the left and right trees.

In other words, to compute the posterior at a non-terminal node of the tree, the posterior of the left and right subtrees are calculated and the resulting posterior is a weighted combination according to the output of the strong classifier associated with the given node. Just as in training, we avoid traversing the entire tree by recursing to both subtrees only if the sample is near the decision boundary:

If $q(+1|x) - \frac{1}{2} > \varepsilon$:
  $\tilde{p}(y|x) = q(+1|x)\tilde{p}_R(y|x) + q(-1|x)\hat{q}_L(y)$
else if $q(-1|x) - \frac{1}{2} > \varepsilon$:
  $\tilde{p}(y|x) = q(+1|x)\hat{q}_R(y) + q(-1|x)\tilde{p}_L(y|x)$
else:
  $\tilde{p}(y|x) = q(+1|x)\tilde{p}_R(y|x) + q(-1|x)\tilde{p}_L(y|x)$

Typically, using this approximation, only a few paths in the tree are traversed; thus the amount of computation to calculate $\tilde{p}(y|x)$ is roughly linear in the depth of the tree.

4. Experiments

We show the application of our framework to 4 different domains: (1) illustrations of the Gestalt laws, (2) detection of object boundaries, (3) road detection and (4) edge detection in natural images. We use nearly identical parameters in all domains, except we change the depth of the tree depending on the amount of data available to avoid overfitting.

4.1. Illustrations of Gestalt laws

The Gestalt laws of perceptual organization, including symmetry, closure, parallelism and so on, are rules of how component parts are organized into overall patterns [12]. The Gestalt laws play an important role in determining object grouping and region boundaries. Explicit studies of mid-level structures include their representations, spatial relationships, and explicit probability distributions. Though there has been substantial work done in this vein [3, 7, 8, 15, 23], individual studies tend to focus on particular Gestalt laws, and it is not clear how to combine them into a unified framework.

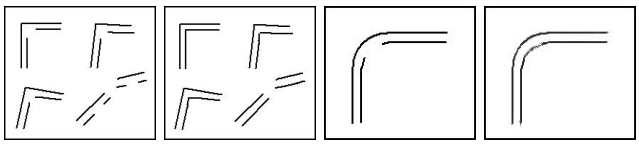
Typically, applying the Gestalt laws is seen as a separate 'mid-level' processing stage that comes into play after low-level features have been calculated. Instead, we show how in our framework the Gestalt laws can be exploited implicitly in a discriminative model that uses very simple image features. The advantages to this type of approach are as follows. First, providing training data is simple – we just need to annotate the edges where the Gestalt laws apply. The same training process can then be used. We do not have to define the individual laws, or how they interact, instead the learning algorithm combines a set of features/weak classifiers to generalize based on the training samples.

In the remainder of this section we provide a few of examples in the form of analogies: we provide a training image and the annotated ground truth, train and apply the discriminative model to a novel test image. That is we ask "A is to B as C is to ?", akin to the work of [9]. The input

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

images in these examples are binary. The results for these examples were easy to obtain – they required no special parameter settings or tweaks to the algorithm. These are meant to be illustrative, we argue that on real data similar principles are implicitly exploited to achieve good results.

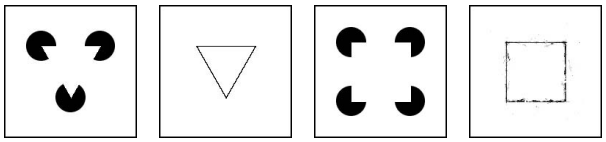
Parallelism



(A) (B) (C) (D)

Training is done on the image in (A) with the ground truth given in (B), and the result when the classifier is applied to (C) is given in (D). To correctly classify points as edge points the local gradient is insufficient. Instead, some edges must be filled in using 'parallelism'. Note that on the test image the learned classifier outputs a high edge probability not only in locations where local gradient was present but also in the gap. The learned classifier was able to generalize to two parallel curves with a smooth turn from training data that contained only straight parallel lines.

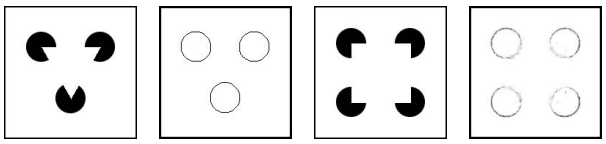
Modal Completion



(A) (B) (C) (D)

Modal completion occurs when portions of an object are occluded by another object that happens to have the same color as nearby regions [16]. Shown the so called Kanizsa triangle in figure (A), observers typically report a white triangle occluding three black disks. In cases such as this, there is a perception of a contrast border even though there is no local contrast. Our algorithm is easily able to generalize to the example of the square. This is significant because it shows that the method can hallucinate edges by fusing features on a relatively large image neighborhood. Note that edge hallucination is not possible with the algorithm of Martin *et al*. [11], where only local gradients are available.

Alternate interpretation of the same data



(A) (B) (C) (D)

We are agnostic with respect to preconceived notions about what defines an edge. The algorithm learns based on the training data it is given.

4.2. Detecting object boundaries

We can train the algorithm to detect edges specifically between an object and the rest of the image. By training on object boundaries the discriminative model learns to suppress other edges in the image, regardless of local gradients.

If we let the scene interpretation W of an image include the location and extent of an object of interest, and define $S_W(i,j) = 1$ if the object boundary passes through location (i,j), then $p(S(i,j)|\mathbf{I})$, defined as before, gives the probability that each location in the image is on the boundary of the object of interest. To obtain ground truth we roughly outlined the object in each image using a paint program. Since the labeling was error prone we smoothed the binary edge map using a Gaussian kernel, which assumes a Gaussian model of the localization error. The result was an approximation of the probability that each location contains an edge, $\hat{p}(S(i,j)|\mathbf{I})$, as before.

Results are shown in Figure (4). Results on the testing data match the human labeled images well, although in some parts the mouse edges were not detected (especially around the ears and tail). Very few non-object edges were detected. The results are far superior to both Canny and `Pb` [11] which are not tuned for this specific domain. Although `Pb` has a data driven component, code for training is not available online, furthermore even if retrained we would not expect `Pb` to suppress non-object edges since only local gradients are available as features.

In Figure (3) we show a closeup of the final testing image, and two cropped patches, that demonstrate the importance of using significant context information.

Successful detection of object boundaries can facilitate object detection or tracking. Given ten images containing a given object provides only ten instance of the object, yet tens of thousands of points on the boundaries of the object (although these points may be highly correlated). Detecting the general location of the mouse in the images in Figure (4)

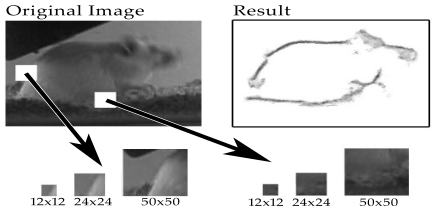


Figure 3. Zoomed in view of final testing image for mouse boundary detection and result of classifier. The classifier fails to detect edges near the moving head due to considerable motion blur, an artifact not observed in the training data. Two positive patches of size 24x24 are shown, and corresponding patches of about half and double size. The decision for the left patch can be made based on local information (the small patch is sufficient), but for the right patch context information is crucial (the larger patch is necessary). A patch size of 50x50 is used throughout this work to make context information available to the discriminative method.

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

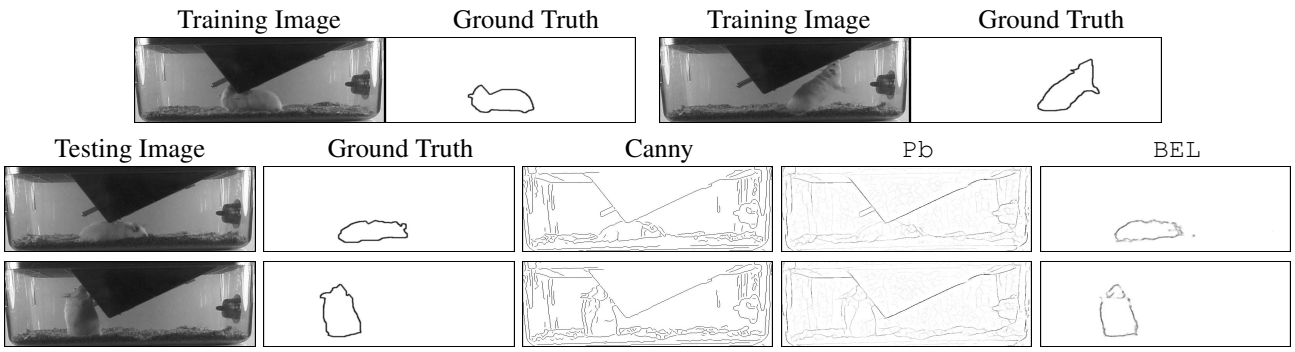


Figure 4. Illustration of object boundary detection. Two of fourteen training images are shown in the first row. The next rows show two of the seven testing images – BEL is able to generalize to the unseen images. Not only are boundaries of the mouse found, but also other edges are suppressed – something that traditional edge detection algorithms cannot do. Both Canny and Pb perform poorly, the 'F' score for BEL was .79 while for Canny it is .10 and for Pb it is .13 (for more on evaluation methodology see Section 4.4). Not only do Canny and Pb detect non-object edges but detection of the mouse edges is poor.

is trivial given the output of our algorithm.

One popular technique used in tracking is to perform background subtraction. However, this requires both the camera and the background to be fixed - our framework provides an alternate solution. The authors of [1] use the output of Pb as input to their mouse tracker on very similar images, we would expect that our method which is more accurate and faster would prove more useful.

4.3. Road detection

Our framework can also be applied to detect roads in satellite images. We obtained the satellite images and corresponding road maps using Google Maps. Each satellite image is aligned with its road map; to generate the ground truth we converted the road map to a binary mask (the process could easily be automated). Road detection is a well known problem, see for example [6]; here we show that our algorithm is directly applicable.

The goal is to classify each pixel as belonging to a road or not. That is we define $S_W(i,j) = 1$ if a location in an image is part of a road according to scene interpretation W, and otherwise 0. In this case, there is only one interpretation W for each image, derived from the corresponding road map, so $\hat{p}(S(i,j)|\mathbf{I}) = S_W(i,j)$. We smooth S_W to allow for some positional uncertainty (alignment of the road to the map is not perfect).

Note that roads are multiple pixels thick. Road detection is not edge detection, rather, the task is pixel assignment. Some results are shown in Figure (5).

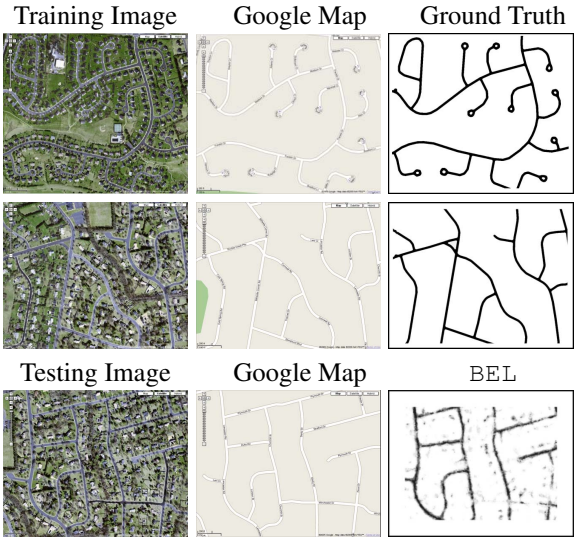


Figure 5. Some results on road detection. Three satellite images were obtained from Google Maps, along with their corresponding road maps. The first two images were used for training, their corresponding road maps were converted to probability distributions of pixel membership. Results on the third image are shown. Close inspection reveals that 'Winchester Dr.' was not detected, it appears that it is much darker than any road in the training data.

4.4. Edge detection in natural images

One of the strengths of our algorithm is that it can learn an edge detector tuned for a specific domain. If labeled data is available, and the regions or boundaries of interest are of a specific form (for example the boundaries on a mouse or roads in satellite images), then our method outperforms other edge detection algorithms which are designed to respond to all edges, not just specific edges, and cannot take advantage of the particular properties of the edges of interest. However, it is interesting to ask if our algorithm were trained to respond to edges in natural images, how would its performance as a general edge detection algorithm rank?

To train our algorithm to respond to edges in natural images, and to test its performance, we used the Berkeley Segmentation Dataset and Benchmark [11]. We trained on a subset of the 200 training images ($\frac{1}{6}$ of each of the first 100 images) and applied the algorithm to 100 test images. We repeated the experiment in both gray scale and color (in our framework, adding features based on color simply involves computing the same features we used for the gray scale im-

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

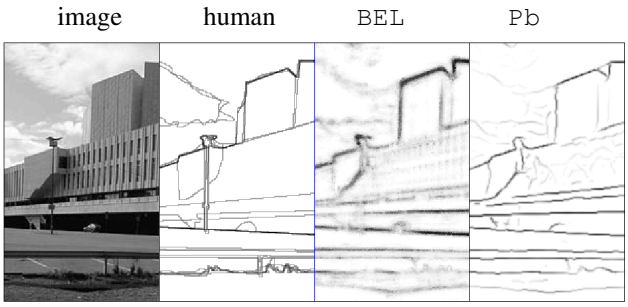


Figure 6. Zoomed in view of a test image form the natural image dataset. Qualitative properties of our output can be seen: (1) BEL gives a true probability, not just a measure of edge strength. The strongest responses correlate well with regions where the largest number of people marked edges. (2) The curvature of BEL edges reflects the curvature of human edges, that is straight edges do not become wavy and corners remain sharp (contrast this with the output of Pb). (3) Sharp, clear boundaries (like the building boundaries) give rise to tightly peaked strong responses, boundaries that are harder to localize (near the clouds and grass) give rise to more diffuse weaker responses. This is consistent with human edges near the clouds and grass which also had significant positional uncertainty. The 'F' score for this image is .79 for Pb and .71 for BEL, i.e. according to the benchmark Pb performs significantly better. When computing the overall precision and recall of the output it must be thresholded and thinned, we suspect this adversely affected our measured performance.

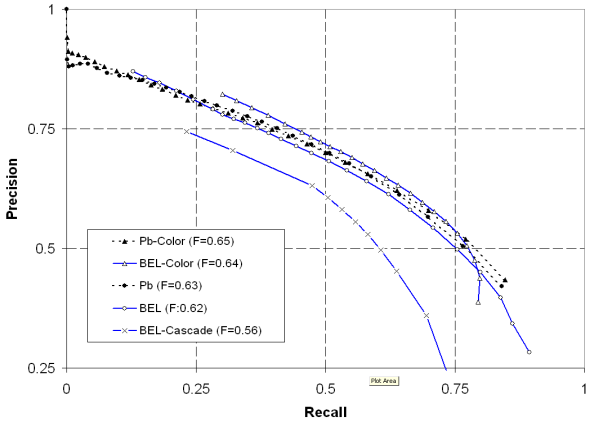


Figure 7. Precision recall curves of BEL and Pb on gray scale and color versions of the Berkeley test set. The overall performance on gray scale images of BEL is very close to Pb, whose performance is the highest reported in the literature. Similarly for color images, BEL-Color outperforms Pb-Color for all but a few precision/recall values (the 'F' score is an overall measure of performance based on all precision/recall values). Training a cascade classifier BEL-Cascade for gray scale images failed to give as good of results, showing the importance of the classification algorithm in our framework.

age over each of the color channels).

Some results on gray scale images are shown in Figure (8). Our output gives a true probability whereas the output of Pb, like of most edge detection algorithms, gives a strength or confidence of an edge being present, and not a true probability. See Figure (6) for a qualitative comparison of the output.

We report quantitative results of our algorithm, calculated using the Berkeley benchmark, in Figure (7). More information on how the precision and recall are computed and the significance of the curve can be found in [11]. We compare our performance to multiple variants of Pb. The color and gray scale versions of Pb have the highest reported performance in the literature. The overall performance of our methods on gray scale and color images is very similar to the corresponding versions of Pb; the precision is higher for some values of recall and lower for others. The performance of all the algorithms shown in Figure (7) is significantly higher than the performance of methods based only on brightness gradients, such as the Canny detector. For a full comparison see [11].

We believe that if changed how we do edge thinning, increased the amount of training data we use, trained a deeper tree, or tweaked any number of other factors we could further improve the overall performance, however, this is not central to our agenda, since as mentioned we believe that the true strength of our method lies in its adaptability.

5. Discussion

In this paper, we have presented a learning based algorithm for edge detection which implicitly combines low-level, mid-level and context information across different scales to learn and compute discriminative models for complex patterns. We treat edge detection as a machine learning problem with a very challenging dataset. We use generic, fast features and make no assumptions about what defines an edge, thus avoiding the need to specifically define ad-hoc rules. We use almost identical parameters for training and there are no parameter to specify once the model is learned.

The resulting algorithm is highly adaptive and scalable. We apply it to a number of different datasets, achieving good results. There are a number of applications, such as object detection and tracking, where edge detection could be used but typically is not because edge detectors tend to be inaccurate and give strong responses in uninteresting regions and weak responses in relevant regions. For many of these areas of application sample labeled data can be made available, and we hope that by making the algorithm adaptive it will find use in these applications.

We conclude by acknowledging that there is a limit to how far a discriminative model such as our can go. The method tends to do very well when adapted to a specific domain, but still has problems on the more general problem of edge detection in natural images, as demonstrated by the difficult images in the Berkeley dataset. We have presented our method in view of an underlying generative model, and argued its merit on the basis of its efficiency and good overall performance. Eventually, however, models that explic-

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

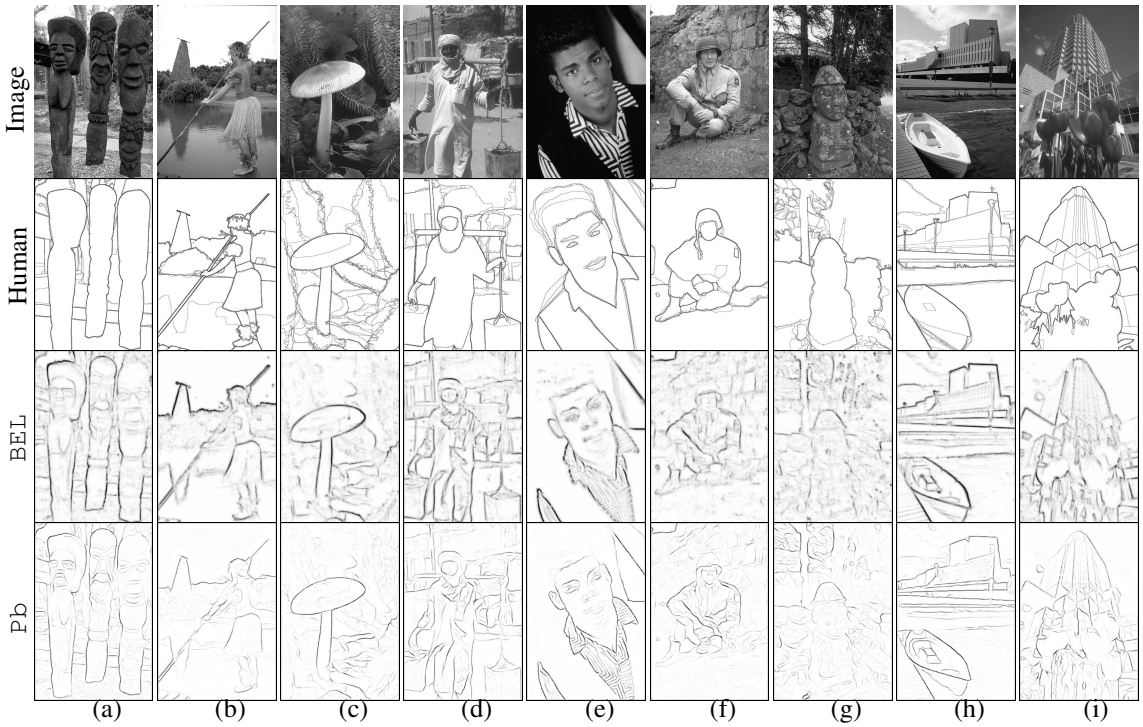


Figure 8. The first row contains gray scale images from the Berkeley dataset (http://www.cs.berkeley.edu/projects/vision/grouping/segbench), and the second the overlayed manual segmentations. For each image we give our results and the result of the Berkeley gray scale edge detector for comparison. The images chosen are the same ones as in Figure 15 in [11].

itly represent and make use of high-level knowledge must be engaged.

Acknowledgements

We would like to thank Visvanathan Ramesh and Dorin Comaniciu for valuable discussion and feedback. This work was done primarily while PD and ZT were at Siemens Corporate Research with support from the Office of Naval Research Award No. N000014-05-1-0543, CFDA No. 12.300. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research. PD is supported by NSF IGERT Grant DGE-0333451. SB is supported by NSF-CAREER Award #0448615 and the Alfred P. Sloan Research Fellowship.

References

[1] K. Branson and S. Belongie, "Tracking Multiple Mouse Contours (without Too Many Samples)", *CVPR*, 2005.
[2] J. F. Canny, "A Computational Approach to Edge Detection", *PAMI*, Nov. 1986.
[3] B. Dubuc and S.W. Zucker, "Complexity, Confusion, and Perceptual Grouping", *IJCV*, 2001.
[4] Y. Freund and R. Schapire, "A Decision-theoretic Generalization of On-line Learning And an Application to Boosting", *ICML*, 1996.
[5] J. Friedman, T. Hastie and R. Tibshirani, "Additive logistic regression: a statistical view of boosting", Stanford Tech Report. 1998.
[6] D. Geman and B. Jedynak, "An Active Testing Model for Tracking Roads in Satellite Images", *PAMI*, 1996.
[7] B. Geisler, "Perceptual Grouping and the Bayesian Statistics of Natural Scenes," *Bayes*, San Francisco, 2001.
[8] C.E. Guo, S.C. Zhu, and Y.N. Wu, "A mathematical theory of primal sketch and sketchability", *ICCV*, 2003.
[9] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, D. Salesin. "Image Analogies", *SIGGRAPH*, 2001
[10] D. Marr, "Vision", 1982.
[11] D. Martin, C. Fowlkes, & J. Malik, "Learning to Detect Natural Image Boundaries Using Local Brightness, Color and Texture Cues", *PAMI04*.
[12] K. Koffka, "Principles of Gestalt Psychology", New York: Harcourt, Brace and company 1935.
[13] S. Konishi, A. Yuille, J. Coughlan and S.C. Zhu, "Statistical Edge Detection: Learning and Evaluating Edge Cues", *PAMI*, Jan, 2003.
[14] V. Ramesh, "Performance Characterization of Image Understanding Algorithms", Ph.D. thesis, University of Washington.
[15] X. Ren, C. Fowlkes, and J. Malik, "Scale-Invariant Contour Completion using Conditional Random Fields", *ICCV*, 2005.
[16] M. Singh. "Modal and Amodal Completion Generate Different Shapes", *Psychological Science*, 2004.
[17] J. Sullivan and S. Carlsson "Recognizing and Tracking Human Action", *ECCV*, 2002.
[18] C. Taylor and D. Kriegman, "Structure and Motion from Line Segments in Multiple Images", *PAMI*, 1995.
[19] Z. Tu, X. Chen, A. Yuille and S.C. Zhu, "Image Parsing: Unifying Segmentation, Detection, and Object Recognition", *IJCV*, 2005.
[20] Z. Tu, "Probabilistic Boosting-Tree: Learning Discriminative Models for Classification, Recognition, and Clustering", *ICCV05*.
[21] S. Ullman R. Basri "Recognition by Linear Combinations of Models", *PAMI*, May, 1991.
[22] P. Viola and M. Jones, "Robust Real Time Object Detection", *SCTV*, 2001.
[23] L.R. Williams and D.W. Jacobs, "Stochastic Completion Fields: A Neural Model of Illusory Contour Shape and Salience", *Neural Computation*, 1997.