Layered Object Models for Image Segmentation

Yi Yang, *Member, IEEE,* Sam Hallman, *Member, IEEE,* Deva Ramanan, *Member, IEEE,* and Charless C. Fowlkes, *Member, IEEE*

Abstract—We formulate a layered model for object detection and image segmentation. We describe a generative probabilistic model that composites the output of a bank of object detectors in order to define shape masks and explain the appearance, depth ordering, and labels of all pixels in an image. Notably, our system estimates both class labels and object instance labels. Building on previous benchmark criteria for object detection and image segmentation, we define a novel score that evaluates both class and instance segmentation. We evaluate our system on the PASCAL 2009 and 2010 segmentation challenge datasets and show good test results with state of the art performance in several categories including segmenting humans.

1 Introduction

Object detection and semantic image segmentation are both fundamental tasks in computer vision. However, these two problems have typically been tackled using substantially different techniques and evaluated using very different criteria, as evidenced by the separate detection and segmentation challenges in the popular PASCAL Visual Object Recognition Challenge (VOC) [1]. Candidate bounding boxes are often generated using a scanning window approach and scored using a classifier trained on positive and negative examples [2], [3], [4]. In contrast, semantic segmentation models have largely been built on top of Markov Random Field (MRF) models which enforce smoothness across pixel labels [5], [6], [7], [8], [9].

We posit that these two problems should be addressed jointly. Per-pixel labels in semantic segmentation should benefit from highly discriminative template-based object detectors. Similarly, object detections should be consistent with some underlying segmentation of the image. Our approach works by processing a set of object detections, represented as a collection of object and part shape masks. We describe an algorithm for compositing these shape masks into a layered model that produces a consistent labeling of each pixel. The algorithm works by integrating top-down shape information from the part masks with bottom-up cues such as object color and boundary information. When compared to previous approaches, our primary contributions are two-fold:

Layered models: We describe a simple probabilistic model that captures the shape, appearance and depth ordering of a collection of detections within an image. It explicitly represents the shapes of a collection of detected object in terms of a layered, per-pixel segmentation. This shape estimate is driven by a novel deformable spatial prior for object shape that adapts to particular instances based on the response of deformable part-based detectors. Given an ordering of layers, these object detections are *composited* to yield a complete generative explanation of pixel colors, their semantic class labels, and object instance labels. Explicitly representing detections with a layered model not only captures depth ordering but can also be advantageous in guiding more precise segmentation. Layering allows for one to link disjoint object segments separated by an occluder (Figure 1) based on estimating the layer appearance (e.g. color and texture).

Benchmark evaluation: We introduce novel scoring criteria for evaluating the accuracy with which individual object instances are segmented. Previous criteria for scoring object detection or segmentation are limited in some respects. Object detectors are typically scored using a ranked list of bounding box detections, which are clearly poor approximations of objects with complex shapes. Furthermore, ranked lists are not internally consistent as boxes of different classes may overlap the same pixel regions. Semantic image segmentations are typically evaluated using pixel-level class labels, which address both limitations. However, such labels ignore the fundamental notion of object instances, necessary for such basic analysis as counting the number of objects in a scene. We propose a novel and simple instance-based segmentation score that address both these shortcomings. We provide extensive experimental evaluation of our model on benchmark datasets for semantic image segmentation, achieving or surpassing state-of-the-art results.

After a brief discussion of related work, we describe our layered representation in detail in Section 3, discuss how to perform inference in Section 4 and how

• *Y. Yang, S. Hallman, D. Ramanan, and C. Fowlkes are with the Department of Computer Science, University of California at Irvine, Irvine, CA 92697. E-mail: {yyang8,shallman,dramanan,fowlkes}@ics.uci.edu*

Fig. 1. Multi-class object detections algorithms typically predict bounding box locations and class labels (**left**) but this only provides coarse information about object localization. We propose to use object detector outputs to guide multi-class segmentation algorithms that provide class labels for every pixel (**center**). To do so, we introduce layered representations (**right**) that reason about relative depth orderings of detected objects and can link disconnected object segments separated by an occluder (e.g., the feet and head of the center person occluded by the horse).

parameters are learned from training data in Section 5. We then show experimental results on the PASCAL Segmentation challenge in Section 6, demonstrating state-of-the art results across many categories using both standard class segmentation scoring criteria and our novel instance-based criteria.

2 Related work

The reconciliation of recognition and segmentation has been an active area of research. Early approaches bias a segmentation engine using the output of object models [10], [11], while others attempt to directly fuse bottom-up and top-down cues during detection [12], [13], [14], [15], [16]. In terms of prior art, our approach is most similar to the ObjCut framework [10] which uses a part-based model to bias a bottom-up grouping process. However, our work differs from previous efforts in that we focus on segmenting images containing multiple instances drawn from multiple object categories. Our approach is also similar to the recent works of [17] and [18], [19]. The former biases a hierarchical CRF model using object detection windows across multiple categories, while the latter generates segmentations using part-specific shape models. Our work combines the multi-category model of the former with the multi-part model of the latter, using a layered representation to construct a globally consistent pixel-level model of the image.

Our work is also inspired by image representations that reason about occlusion through the use of "2.1D" or layered models. Such approaches are typically applied in the video domain and include examples such as layered motion models [20], video sprites [21], and layered pictorial structures [22]. Such layered approaches are less commonly applied to static images but have been explored in e.g., [23], [24], [25].

As the name suggests, 2.1D models live on a continuum between 2D and full 3D representations of scene geometry. At one extreme, minimal ordinal information about depth is described by the figure-ground assignment along each occlusion boundary. This boundary labeling problem has been tackled using various computational frameworks (see e.g., [26], [27], [28], [29]). At the other extreme, one can attempt to estimate the three-dimensional geometry of all visible surfaces from a single images as in [30], [31]. The layered 2.1D model we explore here is an intermediate between these two which specifies a total depth order on object segmentations in an image but stops short of representing metric depth.

A preliminary version of this work appeared in [32]. The system described here differs in a number of ways. We now use a significantly richer order model. We also introduce novel instance-based segmentation scores, and use them to evaluate our model for both class and object segmentation. We also provide additional experimental results on new datasets as well as additional diagnostic analysis of our system components.

3 Layered Model

We now describe our layered generative model for object segmentation.

Detections: For a particular image, let d_n encode the class, score, and bounding box coordinates of the n^{th} detection, where $1 \leq n \leq N$. We assume that the detectors have been calibrated on training data so that detections across classes have comparable scores and thresholding scores at 0 yields an appropriate number of detections on average (we describe details of this calibration in the experimental results section).

Importantly, we model each detection in 2.1D and order them from back to front with some permutation π so that $d_{\pi(N)}$ is the front-most detection, $d_{\pi(N-1)}$ is the second, etc. We define $d_{\pi(0)}$ to be a default background detection associated with a background layer that is included for all images. Let θ_n be the parameters of the appearance model associated with the $\pi(n)^{th}$ detection. We will model appearance with a color histogram.

Pixel Labels: Let x_i be the feature value associated with the i^{th} pixel. Because there is a one-to-one correspondence between a detection and a layer, we write $z_i \in \{0 \dots N\}$ for a label that simultaneously specify both the layer and detection associated with pixel i. Each layer also has its own binary segmentation mask denoted by $b_{in} \in \{0, 1\}$, where we define $b_{i0} = 1$. Note that a pixel i may belong to multiple segmentation masks but can only have one final object label (e.g., both b_{in} and b_{im} are 1 but due to occlusion, either $z_i = n$ or $z_i = m$)

Joint Model: By convention, we use the lack of subscript to denote the set obtained by including all instances of the omitted subscript - e.g., $b_i = \{b_{i0} \dots b_{iN}\}$. Our first assumption is that, given the set of ordered detections d and appearance models θ,

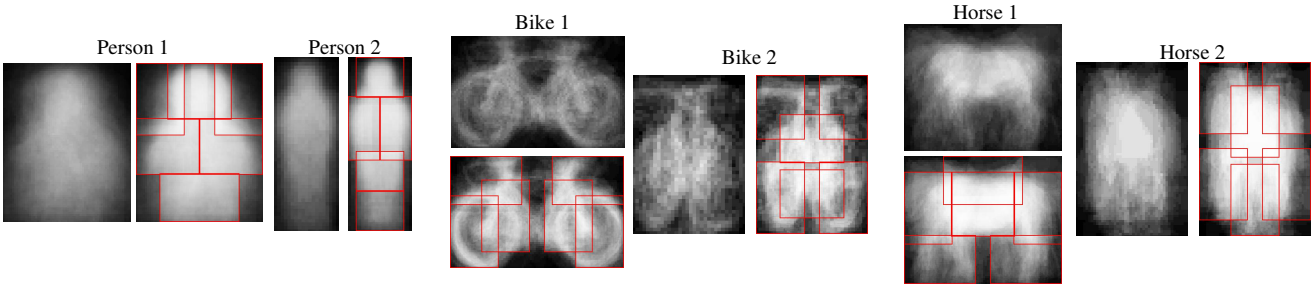


Fig. 2. Examples of class-specific shape priors α represented as soft segmentation masks. We show priors derived from a mixture model of deformable parts including both "root" and "part" templates. Note that the mixture models capture shapes corresponding to different aspects, and part shape models tend to be more tightly localized than the root shape. For example, the horse's legs are blurred out in the first mixture component, but are visible in the composited part model. This is because part models are learned from deformable part annotations, while root shape models are learned from rigid bounding boxes.

the joint probability of pixel features x and labels z factors into a product over pixels:

$$P(z, x|\theta, d_\pi) = \prod_i P(z_i, x_i|\theta, d_\pi), \quad (1)$$

The model for each pixel can be further factored:

$$P(z_i = n, x_i|\theta, d_\pi) = P(z_i = n|d_\pi)P(x_i|\theta_n), \quad (2)$$

where $n \in \{0 \dots N\}$ is a constant that indicates what layer we are considering. The second term on the right-hand side is a standard "likelihood" model that scores pixel x_i under the appearance model for this particular layer which has parameters θ_n. The first term is a distribution over labels induced by the ordered detections.

3.1 Layered Label Distributions

We obtain the distribution over labels by integrating over all layered binary segmentations:

$$P(z_i = m|d_\pi) = \sum_{b_i} P(z_i = m|b_i)p(b_i|d_\pi) \quad (3)$$

$$\text{where} \quad P(z_i = m|b_i) = b_{im} \prod_{n=m+1}^{N} (1 - b_{in}), \quad (4)$$

$$\text{and} \quad P(b_i|d_\pi) = \prod_{n=0}^{N} P(b_{in}|d_{\pi(n)}). \quad (5)$$

We define $b_{i0} = 1$ since the background segment spans the whole image. Thus all pixels are labeled as background by default unless they are explicitly covered by a detection. We combine the previous three equations into a single expression:

$$P(z_i = m|d_\pi) = \sum_{b_i} b_{im} \prod_{n=m+1}^{N} (1 - b_{in}) \prod_{n=0}^{N} P(b_{in}|d_{\pi(n)}) \quad (6)$$

$$= \Big(\sum_{b_{im}} b_{im} P(b_{im}|d_{\pi(m)}) \Big) \prod_{n=m+1}^{N} \sum_{b_{in}} (1 - b_{in}) P(b_{in}|d_{\pi(n)}) \quad (7)$$

One can derive (7) from (6) by distributing the summation over b_i inside the remaining terms. Notably, the summation over b_{in} for layers $n < m$ evaluates to 1 and so disappears from the expression. Intuitively, we only need integrate (3) over binary segmentations in layers in front of m. The above can be simplified by recalling that b_{in} are binary random variables parameterized by a scalar value $\beta_{in} = P(b_{in} = 1|d_{\pi(n)})$:

$$P(z_i = m|d_\pi) = \beta_{im} \prod_{n=m+1}^{N} (1 - \beta_{in}) \quad (8)$$

3.2 Shape model

In this section, we consider different models for deriving the parameter β_{in}, which captures the probability that a given pixel belongs to a detected object. Arguably the simplest model is to associate a shape prior with each class that specifies a "soft" mask or alpha-matte that records the probability of a pixel at some location relative to the center of the detection belonging to the object.

Let c_n as the class label of the n^{th} layer and $i' = \mathcal{T}_n(i)$ be the index of a pixel i which has been mapped by some transformation $\mathcal{T}_n$ into the coordinate system of the corresponding detection. In our experiments, this transformation is a translation and scaling corresponding to the location and scale at which a detector fired. We can then specify a per-detection shape distribution by:

$$\beta_{in} = \alpha_{i',c_n} \quad (9)$$

We visualize examples of such shape priors α in Figure 2.

Object Pose: Local detectors based on mixture models return a mixture component label l_n for each detection. This label often captures the pose of an object - e.g., side versus frontal cars. It is natural to define a shape model for each discrete pose as:

$$\beta_{in} = \alpha_{i',c_n,l_n} \quad (10)$$

Part Pose: Finally, part-based detectors also return a vector of part locations $\{p_1 \dots p_T\}$ for each detection.

We assume each part carries with it a localized alpha matte (as described above) which captures the probability that a nearby pixels belongs to the part. Because parts may overlap, we assume that they are layered in depth and derive a model similar to Section 3.1 that composites their contributions. Part t will contribute the labeling of a pixel so long as the $T-t$ parts in front do not account for that pixel:

$$\beta_{in} = \sum_{t=1}^{T} \alpha_{i',c_n,p_{tn}} \prod_{s=t+1}^{T} (1 - \alpha_{i',c_n,p_{sn}}) \qquad (11)$$

where $i' = \mathcal{T}_{tn}(i)$, the location of pixel i in the coordinate system of the t^{th} part of the detection at layer n. One can also define a shape prior from a mixture of part models by adding an additional mixture index l_n to Equation (11).

3.3 Order distribution

The previous model is conditioned on the ordering π. To examine different orderings, it will be useful to model π as a random variable by writing:

$$P(x, z, \pi | d, \theta) = P(x, z | \pi, d, \theta) P(\pi | d) \qquad (12)$$
$$= P(x, z | d_\pi, \theta) P(\pi | d) \qquad (13)$$

The first term is on the right equivalent to Equation (1). The second term is a distribution over orderings of detections.

One choice for $P(\pi|d)$ would be an uninformative prior that doesn't favor one depth ordering over another. However, there are a variety of cues that may be useful to improve estimates of ordering. First, it is reasonable to assume that most local object models produce higher scores on unoccluded instances compared to occluded instances. This assumption suggests that one should favor depth orderings that place high scoring objects in front of lower scoring objects. A second feature which is useful in ordering detections is that when multiple objects rest on a ground-plane, the object whose bottom edge is lower in the image is typically closer to the camera. A third feature is that objects with smaller image projections tend to be further from the camera. This size cue naturally depends on the size of the object in question. If an airplane and person detection are equal in image area, then the person should be closer to the camera.

Let us write $f_n = (s_n, y_n, h_n, \bar{h}_n)$ for the feature vector containing the score, lower-y coordinate, height and relative height of a given detection n. Assuming that objects of class c are a fixed height H_c, when viewed in perspective the relative height $\bar{h}_n = \frac{h_n}{H_{c_n}}$ gives an additional cue to depth. To integrate these local cues into a global model of ordering, we define a conditional Markov Random Field (MRF) distribution on permutations by:

$$P(\pi|d) = \frac{1}{Z(d)} \prod_{m<n} e^{-w^T (f_{\pi(m)} - f_{\pi(n)})} \qquad (14)$$

Fig. 4. An example superpixel grouping from [33] tuned to return roughly 200 superpixels. We use this bottom-up information in our probabilistic model by constraining all pixels within a superpixel to share the same label.

where w is a vector of model parameters and $Z(d)$ is a normalizing constant. We use our model to produce a relative ordering rather than an absolute ordering As such, it is natural to use difference features to construct a probability distribution over orderings. By symmetry, if we swap the features of the objects, the probabilities of the corresponding orderings will also be swapped.

3.4 Exploiting Bottom-up Grouping

The model as described makes no use direct use of bottom-up grouping constraints such as the presence of contours separating object boundaries. A simple way to incorporate such information is to utilize a segmentation engine which generates superpixels (we use [33]) and assign superpixels to layers instead of pixels (see Figure 4). In this case, we can use the same notation but let i index a superpixel instead of a pixel. For example, z_i will indicate the label of superpixel i and x_i a feature vector (e.g. color distribution) extracted from i. Since superpixels are image dependent, we still maintain a per-pixel alpha-matte which we use to define a distribution over z_i:

$$P(z_i = m | d_\pi) \propto \prod_{j \in \mathcal{S}_i} \beta_{jm} \prod_{n=m+1}^{N} (1 - \beta_{jn}) \qquad (15)$$

The superpixel-constrained label distribution is equivalent to the label distribution from Section 3.1 conditioned on the fact that groups of pixels in the same superpixel must share the same label. This conditioning requires the use of a proportionality sign in (15) to ensure that the left-hand side is a proper probability distribution.

4 Inference

Given an image and a set of detections, we would like to infer the class labels for each pixel z_i. Ideally, one would like to estimate the labels z by marginalizing out over the color models θ. Marginalization is difficult because color models are typically continuous (e.g., the probabilities associated with each bin of a color histogram) and the induced joint potential between θ and z is non-Gaussian. Furthermore,

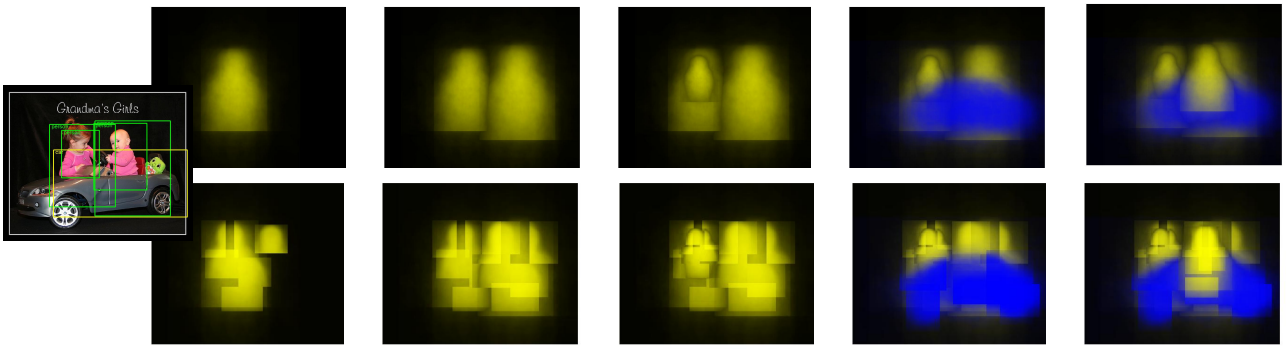


Fig. 3. Examples of an order-dependent layered label distribution $P(z|d_\pi)$. The original image with overlaid candidate detections is shown on the **left**. In the **top** row, we show the composited layered distribution iteratively built from detections ordered back to front. For visualization purposes, we color distributions according to the object type although each detection maintains its own instance layer. In the **bottom** row, we show the distribution built from part-based detections which deform to better match the shape of the detected instance. In general the part composites are more accurate. For example, the front car wheel is better modeled with parts. One notable exception is the mislocalized head in the first instanced person.

marginalizing over θ must be done jointly where multiple detections overlap. Instead, it is natural to approximate the distribution over θ by its maximum likelihood value using coordinate descent or the Expectation-Maximization (EM) algorithm. We first consider inference for the simpler case of a fixed ordering of detections.

4.1 Coordinate ascent

We outline here a coordinate ascent algorithm for maximizing Equation (1) by iterating between updates for z and θ:

$$\begin{aligned} &1. \quad z^t = \arg\max_z P(x, z|\theta^{t-1}, d_\pi) \\ &2. \quad \theta^t = \arg\max_\theta P(x, z^t|\theta, d_\pi) \end{aligned} \tag{16}$$

Step 1 requires computing $P(z_i = n, x_i|\theta^{t-1}, d_\pi)$ for each pixel i and possible label n. Step 2 corresponds to standard maximum likelihood estimation (MLE) which can be solved for each color model independently by computing

$$\arg\max_{\theta_n} \prod_{i:z_i^t=n} P(x_i|\theta_n)$$

In our case, we use color histogram models so θ_n can be estimated using empirical counts.

4.2 EM

As an alternative to coordinate ascent, one could also define an EM algorithm that learns histograms using weighted MLE, where the weight of pixel i for histogram θ_n is given by $P(z_i = n|x, \theta, d_\pi)$. Let us write the expected complete log-likelihood, treating θ as the model parameter to be maximized and z and the hidden variables to the marginalized:

$$L(q, \theta) = E_{q(z)}[\log P(x, z|\theta, d_\pi)] \tag{17}$$

The E and M steps, which perform coordinate ascent on a lower-bounding auxiliary function, are:

$$\begin{aligned} &1. \text{ E step} \quad q^t(z_i) = P(z_i|x, \theta^{t-1}, d_\pi) \quad \forall i \\ &2. \text{ M step} \quad \theta^t = \operatorname*{argmax}_\theta E_{q^t(z)}[\log P(x, z|\theta, d_\pi)] \end{aligned} \tag{18, 19}$$

Step 1 is performed by computing $P(z_i = m, x_i|\theta^{t-1}, d_\pi)$ for each pixel i and label m as before and then normalizing to yield a probability distribution over z_i. Step 2 corresponds to weighted MLE. In the case of histograms this amounts to using weighted frequency counts where the contribution of pixel i to θ_m is given by $q(z_i = m)$.

We have not implemented the above algorithm, but include it for completeness as it gives a probabilistic motivation for our coordinate descent algorithm.

4.3 Orderings

The previous sections assumed that the ordering was fixed. We would now also like to optimize over the ordering as well:

$$\begin{aligned} &\max_{z,\theta,\pi} P(x, z|\theta, d, \pi) P(\pi|d) \\ &\quad = \max_\pi P(\pi|d) \max_{z,\theta} P(x, z|\theta, d, \pi) \end{aligned} \tag{20}$$

For each ordering π, we can compute the inner maximization by coordinate ascent as previously described (or alternately replace the inner maximization with a maximization of the expected complete log-likelihood using EM).

Because the number of detections in an image is usually small, it is often practical to perform a brute force search over orderings. The search space for the maximization over π can be further restricted by noting it is only necessary to enumerate those orderings that generate distinct label priors $P(z|d)$. If

an image only contains two detections that do not overlap, then either order generates the same label prior. A simple method for exploiting this observation is to construct a $N \times N$ adjacency graph of overlapping detections, and ignore the relative ordering between different connected components.

It is also worth noting that if the ordering distribution is sharply peaked around a single ordering, it can dominate the color-shape likelihood term in the maximization in Equation (20). In this case, one can avoid the search over ordering and simply use the most probable ordering under the distribution term. This has the advantage of substantially speeding up the model inference.

5 Learning

We now describe a MLE procedure for learning the shape priors α (defined in Section 3.2) from labeled training data. For simplicity, we include equations for estimating $\alpha_{i'k}$ from a single training image but the extension to multiple images and pose/part-based shape priors are straightforward.

Bernoulli models: First consider the fully observed case in which layered segmentation masks b_{in} are given. Let c_n denote the class of the n^{th} layer. Learning corresponds to standard Bernoulli MLE:

$$\alpha_{i',k} = \underset{\gamma}{\operatorname{argmax}} \prod_{n:c_n=k} P(b_{in}|\gamma) \quad \text{where} \quad i = T_n^{-1}(i')$$
$$= \underset{\gamma}{\operatorname{argmax}} \sum_{n:c_n=k} b_{in} \log\gamma + (1-b_{in}) \log(1-\gamma)$$

where we write $i = T_n^{-1}(i')$ for the inverse transformation that warps a pixel from shape mask coordinates i' to image coordinates i. In our case, this transformation is a translation and scaling corresponding to the location and scale of the n^{th} training instance. The above equations indicate that $\alpha_{i',k}$ is set to the fraction of times the i^{th} pixel for class k is 'on'.

Layered Bernoulli models: In practice, it is easier to label z rather than b because one does not need to estimate the spatial extent of occluded objects. Fortunately, one can still compute MLE estimate of α by marginalizing out labels for occluded regions:

$$\alpha_{i',k} = \underset{\gamma}{\operatorname{argmax}} \prod_{n:c_n=k} P(z_i|\gamma) \quad \text{where} \quad i = T_n^{-1}(i')$$
$$= \underset{\gamma}{\operatorname{argmax}} \sum_{n:c_n=k} \mathbf{1}_{[z_i=n]} \log\gamma + \mathbf{1}_{[z_i<n]} \log(1-\gamma)$$

The above formulation is very similar to standard Bernoulli MLE except that occluded pixels are ignored.

Order model: Learning the parameters of our order-model (14) is more difficult since one requires iterative algorithms or approximations for learning MRFs [34]. Since we have a small number of weights, we experimented with both manually tuning them and learning them with a local logistic regression model (trained to predict the order of pairs of detections given the relative pairwise features $f_n = (s_n, y_n, h_n, \bar{h}_n)$ from (14)).

Learned ordering: We use weights learned using logistic regression in our final experiments, using a subset of features selected through cross validation. When learning the regression model using detection windows produced by our detectors, we find that simply using the detection score s_n produces the best result. For further diagnostic evaluation, we also evaluate our model on ground-truth object windows, as described in Sec.6.5. When trained using such ground-truth data, we find that *all* features are useful, with the lower-y coordinate y_n being the most informative by far. Interestingly, the regression model learns a *negative* weight for the height and relative height $h_n, \bar{h}_n$ features. We initially hypothesized that smaller objects should be placed further from the camera, but doing so may produce poor segmentations because small detections can be fully occluded by larger detections.

6 Experimental Results

In this section, we present results on the PASCAL VOC segmentation competition [35], [1]. PASCAL is widely-acknowledged as a difficult available testbed for both object detection and multi-class segmentation. The competition contains 2000 training and validation images along with ground-truth labelings which give per-pixel labelings for 4200 instances of 20 object categories. Test annotations for the dataset are not released. Instead, benchmarking algorithm performance is done on a held-back test set through a web interface.

6.1 Implementation

Given an image x and a set of calibrated detections d, our final algorithm is as follows:

For each ordering π, iterate until convergence:

1. $z_{S_i} := \underset{m}{\operatorname{argmax}} \prod_{j \in S_i} P(z_j = m|d_\pi) P(x_j|\theta_m) P(\pi|d)$
2. $\theta_m(j) := \frac{\sum \mathbf{1}_{[x_i=j, z_i=m]}}{\sum \mathbf{1}_{[z_i=m]}}$

Output superpixel labels z_{S_i} with most probable ordering π

Our algorithm repeats the above for every ordering π, and returns the ordering and segmentation with the highest probability. In Step 1, we label each superpixel S_i with a label m that maximizes the joint model. In Step 2, we re-estimate a color histogram model for the m^{th} detection by counting the fraction of pixels with a bin value of j.

To generate detections, we used the part-based detector of [4] which was trained using the PASCAL VOC training dataset. We show results for both

version-3 and version-4 of the publicly available detection code [36]. We used version-3 as the input for the 2009 data, and version-4 as the input for the 2010 data (corresponding to the timetable at which both releases were available). Version-3 models objects using two pose-specific mixture components, while version-4 uses six pose-specific mixture components. We demonstrate that our segmentation system can greatly benefit from this refinement because our shape models are now more accurate.

Computation: The bottleneck of our system is the initial object detection and segmentation. Segmentation using the GPU implementation of gPb [37] takes 5s, running the 20 object detectors from [36] takes 30s. Given the superpixels and detections, our inference algorithm takes 3-5s to estimate the segmentation conditioned on the ordering. There is a wide variation in the number of distinct orderings per image, with the median number being 2. This means that the median running time is close to a minute per image.

6.2 Calibration

We use detectors which are trained independently for each object class using a support vector machine (SVM), producing scores which are not directly comparable. In order to calibrate the detectors with respect to segmentation, we estimated an optimal threshold for each detector by evaluating the segmentation benchmark at different threshold settings. We independently select a threshold for each class by scoring the following simple segmentation model:

1) Select all detections of a given class above the threshold
2) Label all pixels inside those detection windows as belonging to the given class.

Figure 5 shows the resulting average segmentation benchmark score as a function of threshold for the 20 classes on a validation set.

It is clear from the figure that the optimal threshold varies widely across different classes (as does the maximal detector performance). The inability of the SVM to learn consistent bias terms for each detector presumably relates to the disconnect between the segmentation benchmark and the detection benchmark. We utilized the per class threshold by simply subtracting the optimal threshold from the detector score and only utilizing detections which scored greater than 0. The offset detector scores were also used in the layer order model.

6.3 Benchmark Results

Figure 6 shows the quantitative performance of our system on the 2009 and 2010 PASCAL segmentation challenge. We compare our results to other top results reported at the both workshops [35], [1], ignoring our own previous entry that was a preliminary version

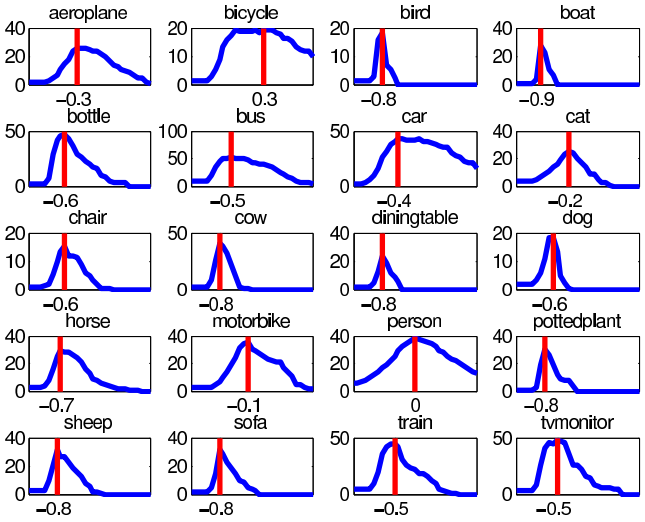


Fig. 5. In order to calibrate the detectors with respect to segmentation, we found a threshold for each detector that optimized independent segmentation performance. The segmentation performance was quantified using the overlap between the set of above-threshold detector bounding boxes and the ground-truth segments for the given class on validation images not used in training the detectors. The graphs show this bounding-box segmentation accuracy as a function of the detector threshold for each of the 20 classes. We observe the optimal threshold varies widely across different classes.

of the system described here. Our system performs quite well compared to the average performance across entries into the competition. Specifically, in 2009, our system ranks first over all other entries in the "person", "bicycle", and "car" categories. Because people are the overwhelmingly common object in the PASCAL dataset, our system tends to produce quite reasonable segmentations for many images. We present example image segmentation results in Figure 12.

Overall, we rank 7 among all entries for both years. We also see a noticeable improvement in average performance of 23% in 2009 to 30% in 2010. This improvement is likely due to two factors. Firstly, the local detectors themselves are more accurate due to the additional mixture components. Secondly, our system also learns more accurate shape masks since each root and part mask is tuned to capture a smaller range of poses.

It is useful to compare our approach with"' other methods that performed better. Many are based on conditional random field (CRF) models, where class-specific appearance models (typically based on HOG and color) are smoothed using models of label consistency. Often the local potentials are biased to respect the output of object detectors (Brookes, CVC-Barcelona, and Stanford entries). This suggests that our approach may also be improved by incorporating additional label consistently constraints. Most simi-

PASCAL Segmentation Challenge

	2009				2010			
	Mean	**Max**	**Us**	**Our rank**	**Mean**	**Max**	**Us**	**Our rank**
background	41.2	83.5	78.0	8	38.6	84.2	81.5	3
aeroplane	18.8	56.3	32.8	7	27.4	58.3	45.6	6
bicycle	10.4	26.6	29.4	1	14.5	27.4	23.9	3
bird	11.0	40.6	3.2	17	13.4	39	10.4	11
boat	11.5	36.1	5.0	16	14.6	37.8	22.1	7
bottle	18.2	46.1	33.1	3	24.6	47.4	35.2	6
bus	25.5	50.5	43.4	3	46.2	63.2	54	6
car	20.6	42.3	43.8	1	33.4	62.4	53.5	3
cat	12.6	35.3	8.3	12	21.2	42.6	14.3	15
chair	4.2	9.1	5.1	9	5.2	9.6	9.6	1
cow	11.7	33.1	11.9	9	18.8	36.8	19.8	9
diningtable	9.1	27.0	8.2	11	11.5	25.2	6.6	13
dog	9.1	24.5	5.6	14	14.6	34.1	9.6	15
horse	17.5	42.7	21.0	7	20.4	37.5	30.5	6
motorbike	23.4	56.4	24.4	9	31.0	60.6	32.8	9
person	20.9	37.5	38.6	1	22.6	44.9	42.4	3
pottedplant	9.7	37.1	14.6	6	12.1	36.8	23.6	4
sheep	19.7	43.6	14.8	13	24.3	50.3	23	9
sofa	8.5	21.9	3.5	17	11.8	21.9	16.1	6
train	19.2	41.0	27.5	7	24.2	45.6	34.5	5
tvmonitor	22.3	47.8	45.7	2	26.9	48.5	41.1	2
average	16.4	36.2	23.7	7	21.8	40.1	30	7

Fig. 6. A performance evaluation of our system using the held-out test set of the 2009 PASCAL Segmentation Challenge [35] and the 2010 PASCAL Segmentation Challenge [1]. Our 2009 entry uses version 3 of local detectors of [36], [4], while our 2010 entry uses version 4 (which has additional mixture components). We compare to all the original systems entered in the competition, omitting our own entry in 2009 that was a preliminary version of the system described here. We perform quite well compared to the average performance across all entries. For "people", "bicycles", and "cars" we obtain the best performance on the 2009 dataset. Since "people" are common in the PASCAL dataset, our system tends to produce quite reasonable segmentations for many images. Overall, we rank 7 among all entries for both years, and see a noticeable improvement in average performance from 2009 to 2010. This improvement is partly due to the more-accurate shape model we learn from the additional mixture models in the version-4 detectors of [36]. We show examples in Figure 12.

lar to us is the UofCTTI entry, which also pastes down root shape masks on detection windows (but without parts, color-models, or bottom-up ground). We compare to such versions of our system in our diagnostic experiments below, and believe their strong performance is due to an improved object detector. Notably, the top-performing method of Bonn [38] does not use object detection windows, but rather ranks putative segments based on a combination of appearance and shape features.

6.4 Diagnostic experiments

Figure 7 documents experiments where we analyzed the contribution of different model components to the overall performance. These performance results were computed on the set of 2010 "validation" segmentation images (rather than using the online test protocol). To avoid testing on data used to train the local detectors, we tested only on those validation images that were not in the segmentation or detection training sets.

Instance-specific appearance model: To turn-off instance-specific appearance modeling, we ignore the color term $P(x_i|\theta_n)$ from (2). This is equivalent to pasting down a shape mask without any coordinate descent optimization. Our instance-specific appearance model turns out to be a strong cue, increasing average performance from 34.7% to 38.4%. We see large improvements for classes such as people, whose instance appearance varies greatly due to clothing. By estimating an instance-specific color model, our system is able to use clothing-specific cues to help segment out the person. Because only a single model is estimated, our system oftentimes will segment out regions associated with one dominant color. This suggests a useful extension is learning a part-specific color model that can capture the difference in appearance between the torso and legs, for example.

Mixture-of-deformable-parts shape prior: To turn off our deformable part prior, we simply ignore part masks when computing the shape model from Sec.3.2. The mixture-of-deformable part spatial prior also tends to help, increasing average performance from 36.5% to 38.4%. We see particularly large improvements for classes such as bicycle and bottle. We hypothesize that the part models are able to better capture anisotropic scalings of the object models not present in the discrete mixtures (e.g., deforming parts can better model tall and short bottles).

PASCAL 2010 Validation set

	baseline	**¬part**	**¬color**	**¬superpixel**	**¬order**	**Worst order**	**Best order**	**Our model**
background	70.6	79.3	78.8	78.4	79.7	79.6	79.7	79.6
aeroplane	21.8	40.9	38.4	37.4	42.6	40.8	44.0	43.8
bicycle	15.3	17.8	10.8	21.9	23.9	20.4	25.9	25.5
bird	10.2	14.8	13.1	13.2	14.1	13.0	15.5	15.3
boat	16.3	16.8	17.1	17.1	18.3	18.1	18.3	18.3
bottle	32.9	37.8	37.3	35.5	40.2	36.1	41.9	39.5
bus	46.4	48.2	50.8	47.0	45.6	42.4	51.4	50.2
car	40.6	44.3	48.2	44.3	47.1	45.9	48.2	47.6
cat	16.9	15.5	14.9	15.2	13.7	11.6	15.2	15.3
chair	10.3	8.8	8.3	9.5	8.0	6.2	10.4	10.3
cow	17.9	16.0	11.4	16.7	12.8	10.1	18.9	15.7
diningtable	4.3	6.7	5.9	6.0	7.1	6.3	7.2	6.4
dog	7.9	8.7	8.5	7.4	8.7	8.1	9.4	8.9
horse	16.4	19.8	18.7	19.4	18.8	11.2	21.6	20.6
motorbike	16.0	18.0	16.1	16.9	17.9	17.2	20.4	17.5
person	33.5	34.9	30.1	35.0	34.8	32.2	37.1	36.4
pottedplant	19.4	15.1	12.9	15.8	14.1	13.0	15.3	14.4
sheep	15.6	17.3	14.5	16.1	15.2	13.5	18.4	17.7
sofa	11.0	9.2	8.8	10.6	9.7	8.8	9.9	9.7
train	31.2	33.2	34.1	33.2	33.7	30.6	35.6	35.0
tvmonitor	34.8	34.3	36.3	34.2	34.2	33.6	35.2	35.3
average	23.3	25.6	24.5	25.3	25.7	23.7	27.6	26.8

Fig. 7. The contribution of different components of our system to performance on the 2010 segmentation validation data. The rightmost column is our full system. The left-most column represents our implementation of the **baseline** approach used by the benchmark organizers to generate segmentations from a detection algorithm; box-shaped segmentation masks are composited in order of detector score (after calibration). The next four columns represent our full system minus particular components, such as instance-specific color estimation, bottom-up grouping, and part-based priors, and layering. In all cases, our system outperforms the baseline. To evaluate **¬order**, we construct a non-ordered shape prior by marginalizing over all possible orderings. We also compare the performance of the ordering selected by our model with the best-possible and worst-possible orderings. Overall, each component plays an important role in our final model, with the instance-specific color model and order-reasoning having the largest impact. Notably, in searching over orderings, our model chooses an ordering that gives segmentation results quite close to those given by the best possible ordering. See Section 6.4 for further detailed analysis and discussion.

Bottom-up grouping: To turn off bottom-up grouping, we treat each pixel independently, removing the "constraint" from Sec.3.4 that all pixels from a superpixel must take on the same label. Overall, the bottom-up grouping constraints provide a noticeable improvement, increasing average performance from 36.3% to 38.4%. We see the largest improvements for classes whose objects tend to have strong, smooth boundaries. We observe this phenomena for many rigid objects such as airplanes, cars, and buses, which often produce strong boundaries due to characteristic backgrounds of sky or road.

Layering: To examine the effect of our layered representation, we would like to consider a version of our system without layering. This is difficult to construct since our probabilistic framework requires a consistent shape model for pixels that overlap two or more detection windows. We created an non-layered, per-pixel shape model by marginalizing the ordered model over all orderings $P(z_i|d) \propto \sum_\pi P(z_i|d_\pi)$.

We see a small, but noticeable improvement in moving from the non-layered model to our full model that explicitly reasons about a globally-consistent ordering of detections. We hypothesize this lack of impact on the benchmark score holds for two reasons. Firstly, because images are sparsely labeled with 20 object categories, it is relatively rare for two objects of different classes to overlap. Only 40% of images had overlaps in the ground-truth bounding boxes, of which half only had a single overlap. Secondly, our local detectors often fail to detect partially occluded objects. Both these facts suggest there are relatively few interesting cases where occlusion reasoning could help, thus limiting the effect of layering on the benchmark score. We further examine this phenomena in the next section.

Order model: To examine the effectiveness of our layer order model (Equation (14)), we show the performance of our system using the best-possible and worst-possible ordering per image, as determined by the average performance across all classes for each image. The performance of our final model (38.4%) appears close to the upper-bound given by the optimal ordering (39.4%). However, we observed that the best-ordering often placed false positive detections behind true positive detections, hence using ordering to inadvertently remove false positives. Since we would like to examine the influence of ordering in reasoning about relative depth, we also considered

an additional experiment (described in Sec.6.5) using only true positive detections.

Detector accuracy: To examine the behavior of our system with different quality detectors, we performed another experiment using ground-truth detection windows (e.g., an "ideal detector" with zero false positives and zero missed detections). When evaluated on the entire PASCAL 2010 Validation set, our final segmentation accuracy doubles to 60.9% (due to space restrictions, we don't include the associated column in Fig.7. This suggests that the performance of our system is closely tied to the accuracy of the initial detectors.

6.5 Evaluation on images with overlap

To further examine the role of ordering detections, we conducted an experiment using a set of true positive detections culled from the subset of images where two or more detections overlapped. True-positive detections were those detection outputs above threshold whose overlap with some ground-truth bounding box was greater than 50%. We show results on this subset of the data in Figure 8. We note that the overall performance of our system (59%) and relative impact of each component is greater for this test-set, since we utilize known-good detections and test on images where overlapping detections are much more common. For example, on this test set, removing global order-reasoning from the model reduces average performance from 59% down to 57%.

We also examine the effect of different depth orders on this set of images. The worst ordering scored 52.7%, the best possible scored 60.0%, and our system scored 59.0%. Because we are examining only true positive detections, the best-ordering score is no longer inflated by the ability to remove false positives by layering them behind other detections. This suggests that our model does captures much of the gain to be had by reasoning about depth order. Furthermore, given accurate detectors (no false-positives), our system nearly doubles in performance from 38% to 59%.

6.6 Instance-based segmentation benchmark

The PASCAL segmentation challenge scores the task of semantic segmentation, where each pixel must be labeled with 1 of 20 object labels or background. This makes sense for classes which consist of "stuff" such as grass, sky, mud, etc. In contrast, for "things", i.e. semantic classes denoting objects defined primarily by shape, it seems far more natural to score object-instance labels. Consider the image of three dogs in Figure 9; the instance-level segmentation naturally produced by our system is clearly more detailed than the class segmentation scored by PASCAL. Furthermore, due to our layered representation, we are able to reason about the instance labels of occluded pixels as well (though scoring such output is difficult).

In this section, we evaluate the performance of our system using two novel instance-based segmentation benchmarks. The first is a natural extension of the existing class-based benchmark, where the notion of a correctly-labelled pixel is refined to require both class and instance labels to agree. We also propose an alternate score that decomposes into a sum of per-instance scores. The latter can also be viewed as a novel detection benchmark, unifying the traditionally disparate evaluation criteria for detection and segmentation.

PASCAL class benchmark: First, we introduce notation for describing the PASCAL segmentation benchmark. Let $\mathcal{G}_k$ and $\mathcal{M}_k$ denote the set of ground-truth and machine-generated segments respectively for the k^{th} object class. Let $G_i \in \mathcal{G}_k$ denote the set of pixels corresponding to a particular segment. The PASCAL class segmentation accuracy is given by:

$$\mathrm{acc}_{\mathrm{class}}(k) = \frac{\sum_{G_i \in \mathcal{G}_k} \sum_{M_j \in \mathcal{M}_k} |G_i \cap M_j|}{|\bigcup_{G_i \in \mathcal{G}_k} \bigcup_{M_j \in \mathcal{M}_k} G_i \cup M_j|} \quad (21)$$

This score ranges between 0 and 1 and measures the area of overlap of ground-truth and machine marked pixels relative to the union of their areas.

Instance benchmark: To extend this performance metric to object instance segmentations, we need to establish a 1-to-1 correspondence between machine and ground-truth segments. We describe such a correspondence by a function p so that G_i is matched to a particular instance $M_{p(i)}$. Let $\mathcal{P}$ be the set of all such matchings.

We define the instance benchmark accuracy, $\mathrm{acc}_{\mathrm{inst}}(k)$, to be a straightforward extension of (21) by changing the numerator so that pixels are counted as true positives only if class labels and instance assignments agree:

$$\mathrm{acc}_{\mathrm{inst}}(k) = \max_{p \in \mathcal{P}} \frac{\sum_{G_i \in \mathcal{G}_k} |G_i \cap M_{p(i)}|}{|\bigcup_{G_i \in \mathcal{G}_k} \bigcup_{M_j \in \mathcal{M}_k} G_i \cup M_j|} \quad (22)$$

The optimal correspondence p can be found by solving a maximum-weight bipartite matching problem containing edges that connect ground-truth and candidate segments of the same class within the same image. Edge weights are given by pixel overlap counts $|G_i \cap M_j|$. Leftover segments are matched to "dummy" nodes with zero overlap.

The instance benchmark is clearly stricter than the class benchmark since the summation in the instance benchmark numerator contains a subset of the terms in the class benchmark numerator. By enforcing a matching, the instance benchmark can appropriately penalize undersegmentations which fail to split apart objects of the same class.

Segment-Instance Precision and Recall: Both the benchmarks defined thus far count the number of correctly classified pixels, so larger objects are more important to the total accuracy than smaller objects. To

Subset of PASCAL 2010 Validation set with verified, overlapping detections

	¬part	¬color	¬superpixel	¬order	Worst order	Best order	Our model
background	81.9	79.5	81.6	82.8	82.5	83.0	82.8
aeroplane	68.8	73.9	61.8	75.9	75.9	75.9	75.9
bicycle	22.4	17.5	25.6	28.3	20.8	34.3	33.9
bird	73.3	58.3	68.1	74.8	74.9	74.9	74.9
boat	47.6	44.9	44.9	42.3	41.6	42.7	42.2
bottle	68.1	64.2	72.0	73.1	58.7	77.0	76.4
bus	74.0	78.3	75.3	79.2	78.8	79.5	79.5
car	64.6	62.6	61.6	68.2	63.8	68.5	67.4
cat	51.1	49.7	53.9	47.6	33.6	60.7	60.4
chair	31.4	26.7	34.3	34.5	26.7	41.1	39.4
cow	38.8	35.6	49.9	46.5	41.1	47.7	45.5
diningtable	52.3	45.5	46.6	48.9	45.5	52.0	50.4
dog	62.7	60.9	61.9	60.2	50.3	65.5	64.7
horse	52.8	44.4	51.8	57.6	54.2	58.1	55.3
motorbike	61.5	58.1	54.2	63.1	58.2	63.9	62.0
person	52.9	43.1	52.8	53.3	48.7	57.9	56.1
pottedplant	52.7	45.6	48.1	49.4	44.0	51.4	50.8
sheep	46.2	38.2	47.8	48.5	48.2	52.7	51.4
sofa	39.0	32.8	50.7	38.7	35.0	44.0	43.7
train	58.7	58.2	58.4	59.1	59.0	59.5	59.5
tvmonitor	66.2	65.6	72.4	68.6	65.1	69.5	67.0
average	55.6	51.6	55.9	57.2	52.7	60.0	59.0

Fig. 8. We analyze our system using verified (true-positive) detections on the subset of PASCAL 2010 validation images where two or more such detections overlapped. In general, we see a large impact for each component of our system. For example, a no-order version of system drops in performance from 59% to 57%. Our overall performance of 59% almost doubles our performance on the full validation set. This suggests our model could produce quite accurate segmentations given ideal detectors run on images where multiple overlapping objects were common.

give each instance equal importance, we can instead compute the intersection-over-union overlap score on a per-instance basis rather than for the whole pool of segments. In this case, an object instance G_i matched to a segment M_j contributes a value

$$O_{ij} = \frac{|G_i \cap M_j|}{|G_i \cup M_j|} \tag{23}$$

to the final score. This per-object score is between 0 and 1 regardless of object size. The final score for category k is the average value of O_{ij} across instances and can be normalized with respect to either the number of ground-truth segments (recall) or the number of machine-produced segments (precision):

$$\mathrm{acc}_{\mathrm{rec}}(k) = \max_{p \in \mathcal{P}} \frac{1}{|\mathcal{G}_k|} \sum_{G_i \in \mathcal{G}_k} \frac{|G_i \cap M_{p(i)}|}{|G_i \cup M_{p(i)}|} \tag{24}$$

$$\mathrm{acc}_{\mathrm{prec}}(k) = \max_{p \in \mathcal{P}} \frac{1}{|\mathcal{M}_k|} \sum_{M_j \in \mathcal{M}_k} \frac{|G_{p^{-1}(j)} \cap M_j|}{|G_{p^{-1}(j)} \cup M_j|} \tag{25}$$

In this case the optimal correspondence p for both scores is identical and given by the solution of a maximum-weight bipartite matching problem with weights O_{ij}.

If we replace O_{ij} in the equations above with a thresholded indicator function $\mathbf{1}_{[O_{ij}>.5]}$, the resulting averages are equivalent to precision-recall values computed in the PASCAL detection benchmark with two differences: we compute overlaps with segmentation masks rather than bounding-boxes, and we compute an optimal correspondence p rather than a approximate one [1]. This makes it directly possible to directly compare the performance of object detectors that return bounding boxes and instance-based segmentation algorithms using a unified scoring criteria.

Results: In Figure 10, we evaluate performance using $\mathrm{acc}_{\mathrm{inst}}$. Because this is a strictly harder criteria than $\mathrm{acc}_{\mathrm{class}}$ (the set of true positive pixels must now be smaller), all the performance numbers decrease. We evaluate our system on the entire validation set (as in Fig.7) and the subset of images with verified, overlapping detections (as in Fig.8), though we present the full diagnostic analysis for the latter because it is more interpretable. On either set, the overall decrease in segmentation accuracy is relatively small; from 26.8% to 25.9% for the former and 59.0% to 57.7% for the latter. This indicates that our system can be used to segment individual object instances as well as generating class segmentations.

In Figure 11, we evaluate performance using $\mathrm{acc}_{\mathrm{prec}}$ and $\mathrm{acc}_{\mathrm{rec}}$. Our segmentations tend to operate at the high-precision low-recall regime, much like the base detectors we use. Note that the precision-recall benchmark differs from $\mathrm{acc}_{\mathrm{inst}}$ in that all instances are weighted equally (rather than by size in the image). This is visible in the results. For example, we segment people more accurately than tables (56% to 50%) according to $\mathrm{acc}_{\mathrm{inst}}$ but both have a similar F1-score in the precision-recall benchmark. One explanation is that people may occur at larger scales than dining

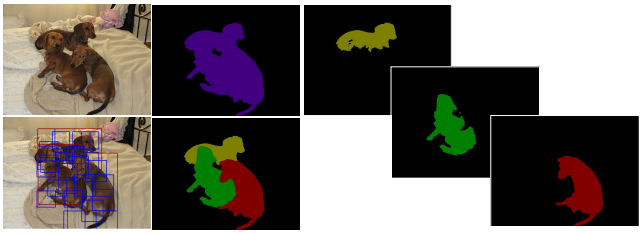


Fig. 9. We show an example output of our system on the image from the **top-left** using the true positive detections on the **bottom-left**. Our system produces class labels for each pixel, show on the **top-right**. This is the output scored by the PASCAL segmentation benchmark. Our system can also return multi-class *instance* labels z, as shown in the **bottom-right** image. Moreover, due to our layered representation, we can explicitly reason about the spatial layout of occluded regions of objects. We show the binary segmentation labels b on the three **right** images, where images are ordered from back to front. Note that our system correctly estimates the depth order of the three dogs as well as inferring spatial extent of occluded parts of the dogs.

tables in PASCAL.

Detection benchmark: One can also evaluate our instance segmentations using the standard PASCAL detection benchmark. This requires generating bounding-boxes from the segmentation masks output by our system. We found that this was not straightforward, as isolated pixels could produced skewed bounding boxes, which in turn produced a poor detection benchmark score. One interpretation of this result is that when scoring detection accuracy, it is more natural to use pixel overlap (as $\mathrm{acc}_{\mathrm{prec}}$ and $\mathrm{acc}_{\mathrm{rec}}$ do) rather than bounding box overlap. Fig.11 suggests that our system does produce better detections when measured with the former.

7 Conclusion

We have proposed a simple model which performs pixel labeling based on the output of scanning window classifiers. It does so by combining top-down deformable shape priors with bottom-up grouping constraints and instance-specific color models. It reconciles all these cues in a globally consistent, 2.1D interpretation of the image obtained by layering objects in depth. Based on extensive diagnostic analysis, we have verified that each component of our system is important in producing high-quality segmentations. We also demonstrate that our algorithm extracts much of the possible information about depth order inferable from given current detection systems.

In terms of performance, our system achieves or surpasses the performance of current state-of-the-art approaches for multiclass segmentation. Our system produces much richer outputs than current systems, in that it estimates the spatial layout of individual

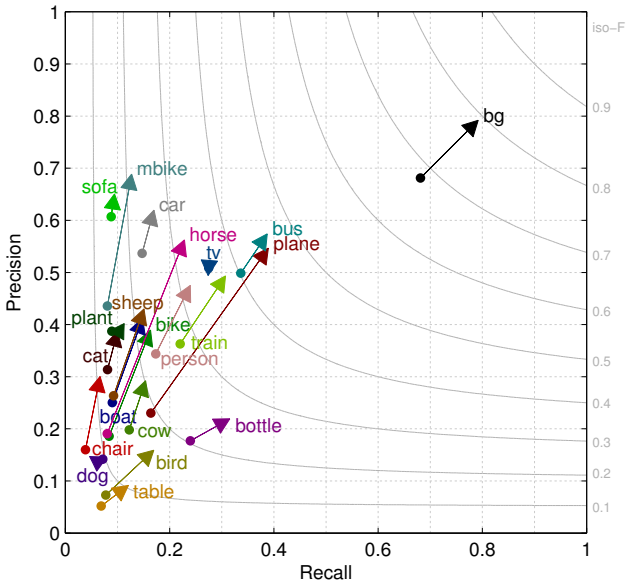


Fig. 11. We analyze our system on the full PASCAL VOC 2010 Validation set, but using our new precision-recall, instance-based, scoring criteria ($\mathrm{acc}_{\mathrm{prec}}$ and $\mathrm{acc}_{\mathrm{rec}}$). We compare the result of our system (the head of the arrow) to the baseline approach from Fig.7 (the tail of each arrow). We plot isocontours of constant F1-score, the harmonic mean of precision and recall. Our system produces better F1-scores for all classes. Our system generally operates at a high-precision low-recall regime, most likely due to the properties of our calibrated detectors (which operate at a similar regime).

objects, including both visible and occluded regions. To score the accuracy of instance-level segmentation, we introduce two new benchmarks that reconcile the traditionally disparate evaluation criteria for object detection and segmentation. Evaluating our new criteria on benchmark data, we demonstrate that our system can fairly reliably segment individual object instances.

Acknowledgments

Funding for this research was provided by a UC Labs research program grant, NSF Grant IIS-0954083, ONR-MURI Grant N00014-10-1-0933, and support from Microsoft, Google, and Intel.

References

[1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results."
[2] P. A. Viola and M. J. Jones, "Robust real-time face detection," *IJCV*, vol. 57, no. 2, pp. 137–154, 2004.
[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005, pp. I: 886–893.
[4] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE PAMI*, 2009.
[5] X. He, R. Zemel, and M. Carreira-Perpinan, "Multiscale conditional random fields for image labeling," in *CVPR*, vol. 2, 2004.

Instance benchmark on PASCAL 2010 Validation with verified, overlapping detections

	¬part	¬color	¬superpixel	¬order	Worst order	Best order	Our model
background	81.9	79.5	81.6	82.8	82.5	83.0	82.8
aeroplane	68.8	73.9	61.8	75.9	75.9	75.9	75.9
bicycle	21.9	17.4	24.8	27.8	20.8	33.1	33.2
bird	72.7	57.3	67.2	73.6	73.7	73.9	73.9
boat	38.7	37.0	38.0	33.7	32.9	34.6	34.4
bottle	67.8	64.0	71.2	72.7	58.9	76.6	76.1
bus	73.6	77.6	74.3	78.8	78.2	79.2	78.9
car	64.2	62.3	61.1	67.6	63.1	68.2	66.6
cat	51.1	49.7	53.9	47.6	33.6	60.7	60.4
chair	28.8	24.2	31.1	31.7	24.8	37.9	36.2
cow	38.0	35.1	47.3	45.2	39.9	46.5	44.2
diningtable	52.3	45.5	46.4	48.9	45.6	51.9	50.4
dog	60.4	59.2	59.7	57.9	47.0	62.9	62.3
horse	50.5	42.9	50.3	54.1	50.5	56.5	53.5
motorbike	61.4	58.0	53.9	63.0	59.2	62.7	61.9
person	51.4	41.7	50.9	51.7	46.7	56.3	54.3
pottedplant	49.8	43.4	45.1	47.0	41.8	49.3	48.1
sheep	44.2	36.7	43.7	46.2	45.5	50.3	49.1
sofa	39.0	32.7	50.6	38.7	35.0	43.9	43.7
train	58.3	57.8	58.0	58.6	58.3	59.1	58.8
tvmonitor	65.4	64.8	71.2	67.8	64.6	68.4	66.5
average	54.3	50.5	54.4	55.8	51.4	58.6	57.7

Fig. 10. We analyze our system on the same dataset as Figure 8, but using our newly proposed instance-based segmentation benchmark $\mathrm{acc}_{\mathrm{inst}}$. Because labeling pixels with instance labels is harder than assigning class labels, these performance numbers are lower than those reported in Figure 8. The small overall drop in performance (from 59% to 58%) suggests our system is able to quite accurately label object instances as well as class labels. For reference, we also evaluated our instance benchmark on the full validation set, as in Figure 7. We similarly see a small drop in average segmentation accuracy, from 26.8% to 25.9%

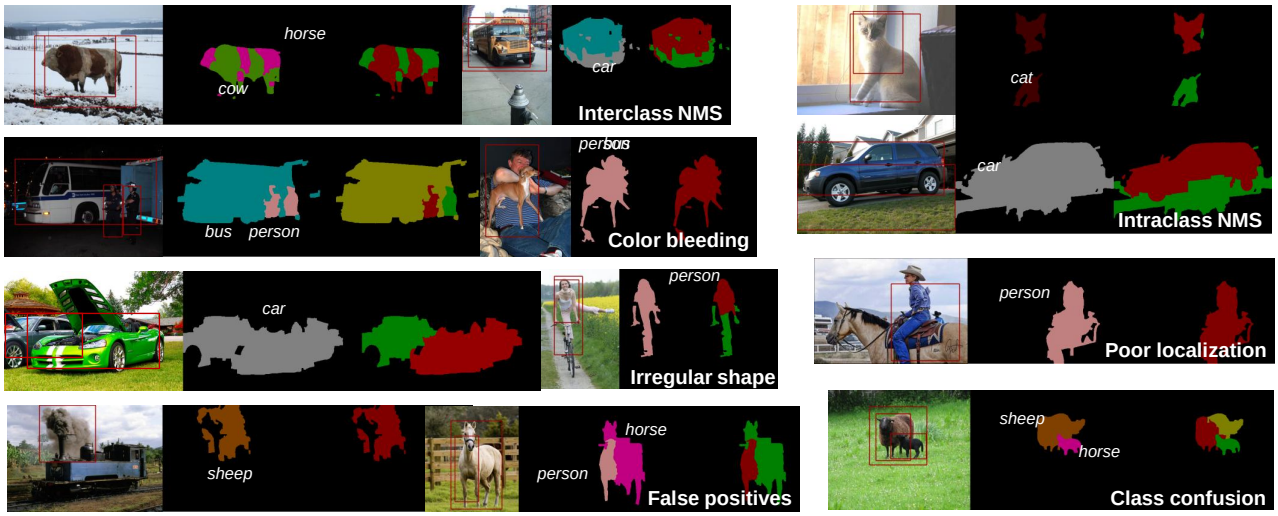
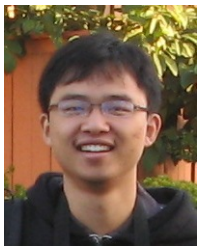


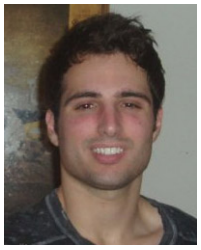
Fig. 12. Example failure modes of our system on the 2010 PASCAL dataset. We show triplets corresponding to the original image, a class segmentation following the color scheme from PASCAL, and an instance segmentation using an arbitrary color scheme. Some failures such as false positives, class confusion and poor localization are attributable to shortcomings of the detector and are quantified by standard detector benchmarks. However, there are also several failure modes that involve interactions of both components and non-maximum suppression (NMS) which are only diagnosed by the segmentation or instance benchmark. For example, multiple animal detectors often fire on the same image region, making independent pixel label assignment difficult. Failures such as color bleeding and irregular shapes could be eliminated by improved segmentation models.

[6] A. Torralba, K. Murphy, and W. Freeman, "Contextual models for object detection using boosted random fields," *NIPS*, 2004.
[7] S. Kumar and M. Hebert, "A hierarchical field framework for unified context-based classification," in *ICCV*, vol. 2, 2005.
[8] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," *ECCV*, vol. 3951, p. 1, 2006.
[9] Z. Tu, "Auto-context and its application to high-level vision tasks," in *IEEE CVPR*, 2008.
[10] M. Kumar, P. Torr, and A. Zisserman, "Obj cut," in *CVPR*, vol. 1, 2005.

[11] D. Ramanan, "Using segmentation to verify object hypotheses," *CVPR*, 2006.
[12] S. Yu, R. Gross, and J. Shi, "Concurrent object recognition and segmentation by graph partitioning," *NIPS*, pp. 1407–1414, 2003.
[13] B. Leibe, A. Leonardis, and B. Schiele, "Combined object categorization and segmentation with an implicit shape model," in *ECCV 04 workshop on statistical learning in computer vision*, 2004, pp. 17–32.
[14] A. Levin and Y. Weiss, "Learning to combine bottom-up and top-down segmentation," *International Journal of Computer Vision*, vol. 81, no. 1, pp. 105–118, 2009.
[15] Z. Tu, X. Chen, A. Yuille, and S. Zhu, "Image parsing: Unifying segmentation, detection, and recognition," *IJCV*, vol. 63, no. 2, pp. 113–140, 2005.
[16] X. Ren, C. Fowlkes, and J. Malik, "Cue integration for figure/ground labeling," in *NIPS*, 2005.
[17] L. Ladicky, P. Sturgess, K. Alahari, C. Russell, and P. H. S. Torr, "What, where and how many? combining object detectors and crfs," in *European Conference on Computer Vision*, 2010, pp. 424–437.
[18] T. Brox, L. Bourdev, S. Maji, and J. Malik, "Object segmentation by alignment of poselet activations to image contours," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. [Online]. Available: http://www.eecs.berkeley.edu/ lbourdev/poselets
[19] L. Bourdev, S. Maji, T. Brox, and J. Malik, "Detecting people using mutually consistent poselet activations," in *European Conference on Computer Vision (ECCV)*, 2010. [Online]. Available: http://www.eecs.berkeley.edu/ lbourdev/poselets
[20] J. Wang and E. Adelson, "Representing moving images with layers," *IEEE Trans on Image Processing*, vol. 3, no. 5, pp. 625–638, 1994.
[21] N. Jojic and B. Frey, "Learning flexible sprites in video layers," in *CVPR*, vol. 1, 2001.
[22] M. Kumar, P. Torr, and A. Zisserman, "Learning layered pictorial structures from video," in *Indian Conf on Comp Vis, Graphics and Image Proc*, 2004, pp. 158–163.
[23] M. Nitzberg, D. Mumford, and T. Shiota, *Filtering, Segmentation and Depth*. Springer-Verlag, 1993.
[24] R. Gao, T. Wu, S. Zhu, and N. Sang, "Bayesian inference for layer representation with mixed markov random field," in *Energy Minimization Methods in CVPR*, pp. 213–224.
[25] I. Liechter and M. Lindenbaum, "Boundary ownership by lifting to 2.1d," in *ICCV*, 2009.
[26] J. Winn and J. Shotton, "The layout consistent random field for recognizing and segmenting partially occluded objects," in *CVPR*, 2006.
[27] X. Ren, C. Fowlkes, and J. Malik, "Figure/ground assignment in natural images," in *ECCV*, 2006.
[28] D. Hoiem, C. Rother, and J. Winn, "3d layout crf for multi-view object class recognition and segmentation," in *CVPR*, 2007.
[29] M. Maire, "Simultaneous segmentation and figure/ground organization using anuglar embedding," in *ECCV*, 2010.
[30] D. Hoiem, A. Stein, A. Efros, and M. Hebert, "Recovering occlusion boundaries from a single image," in *ICCV*, 2007.
[31] A. Saxena, M. Sun, and A. Ng, "Make3D: Learning 3D Scene Structure from a Single Still Image," *IEEE TPAMI*, pp. 824–840, 2009.
[32] Y. Yang, S. Hallman, D. Ramanan, and C. Fowlkes, "Layered object detection for multi-class segmentation," *CVPR*, 2010.
[33] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in *CVPR*, 2009.
[34] S. Li, "Markov random field models in computer vision," *Computer VisionECCV'94*, pp. 361–370, 1994.
[35] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results."
[36] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Discriminatively trained deformable part models, release 4," http://people.cs.uchicago.edu/ pff/latent-release4/.
[37] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.
[38] F. Li, J. Carreira, and C. Sminchisescu, "Object Recognition as Ranking Holistic Figure-Ground Hypotheses," in *IEEE International Conference on Computer Vision and Pattern Recognition*, June 2010, description of our winning PASCAL VOC 2009 and 2010 recognition entry.



Yi Yang Yi Yang received a BS with honors from Tsinghua University in 2006 and received a Master of Philosophy degree in Industrial Engineering in Hong Kong University of Science and Technology in 2008. He is currently a PhD student in the Department of Computer Science at the University of California, Irvine. His research interests are in artificial intelligence, machine learning and computer vision.



Sam Hallman Sam Hallman received a BS degree in computer science from the University of California, Irvine in 2009. He returned as a PhD student in 2010 and has since been studying computer vision. His main interests are in image segmentation and object recognition.

Deva Ramanan is an assistant professor of Computer Science at the University of California at Irvine. He received his Ph.D. in Electrical Engineering and Computer Science from UC Berkeley in 2005. His research interests span computer vision, machine learning, and computer graphics, with a focus on the application of understanding people through images and video.

Charless C. Fowlkes received a BS with honors from Caltech in 2000 and a PhD in Computer Science from UC Berkeley in 2005, where his research was supported by a US National Science Foundation Graduate Research Fellowship. He is currently an Assistant Professor in the Department of Computer Science at the University of California, Irvine. His research interests are in computer and human vision, and in applications to biological image analysis.