Finding Things: Image Parsing with Regions and Per-Exemplar Detectors

Joseph Tighe
University of North Carolina at Chapel Hill
jtighe@cs.unc.edu

Svetlana Lazebnik
University of Illinois at Urbana-Champaign
slazebni@illinois.edu

Abstract

This paper presents a system for image parsing, or labeling each pixel in an image with its semantic category, aimed at achieving broad coverage across hundreds of object categories, many of them sparsely sampled. The system combines region-level features with per-exemplar sliding window detectors. Per-exemplar detectors are better suited for our parsing task than traditional bounding box detectors: they perform well on classes with little training data and high intra-class variation, and they allow object masks to be transferred into the test image for pixel-level segmentation. The proposed system achieves state-of-the-art accuracy on three challenging datasets, the largest of which contains 45,676 images and 232 labels.

1. Introduction

This paper addresses the problem of image parsing, or labeling each pixel in an image with its semantic category. Our goal is achieving *broad coverage* – the ability to recognize hundreds or thousands of object classes that commonly occur in everyday street scenes and indoor environments. A major challenge in doing this is posed by the non-uniform statistics of these classes in realistic scene images. A small number of classes – mainly ones associated with large regions or "stuff," such as road, sky, trees, buildings, etc. – constitute the majority of all image pixels and object instances in the dataset. But a much larger number of "thing" classes – people, cars, dogs, mailboxes, vases, stop signs – occupy a small percentage of image pixels and have relatively few instances each.

"Stuff" categories have no consistent shape but fairly consistent texture, so they can be adequately handled by image parsing systems based on pixel- or region-level features [5, 7, 8, 18, 21, 22, 25, 26, 27, 29]. However, these systems have difficulty with "thing" categories, which are better characterized by overall shape than local appearance. In order to improve performance on "things," a few recent image parsing approaches [1, 10, 12, 14, 16] have attempted to incorporate sliding window detectors. Many of these approaches rely on detectors like HOG templates [6] and deformable part-based models (DPMs) [9], which produce only bounding box hypotheses. However, attempting to infer a pixel-level segmentation from a bounding box is a complex and error-prone process. More sophisticated detection frameworks like implicit shape models [17] and poselets [3] provide a better way to do per-pixel reasoning, but they tend to require a lot of extensively annotated positive training examples. None of these schemes are well suited for handling large numbers of sparsely-sampled classes with high intra-class variation.

In this paper, we propose an image parsing system that integrates region-based cues with the promising novel framework of *per-exemplar detectors* or *exemplar-SVMs* [19]. Per-exemplar detectors are more appropriate than traditional sliding window detectors for classes with few training samples and wide variability. They also meet our need for pixel-level localization: when a per-exemplar detector fires on a test image, we can take the segmentation mask from the corresponding training exemplar and transfer it into the test image to form a segmentation hypothesis.

The idea of transferring object segmentation masks from training to test images – either whole or in "fragments" – has been explored before in the literature – see, e.g., [2, 17, 20]. However, most existing work uses local feature matches to transfer mask hypotheses, and focuses on one class at a time. To our knowledge, our approach is the first to transfer masks using per-exemplar detectors (Malisiewicz et al. [19] suggest this idea, but do not evaluate it quantitatively) and to output a dense many-category labeling, as opposed to a segmentation of a single class.

Our proposed method is outlined in Figure 1. It combines the region-based parser from our earlier work [27] with a novel parser based on per-exemplar detectors. Each parser produces a score or *data term* for each possible label at each pixel location, and the data terms are combined using a support vector machine (SVM) to generate the final labeling. This scheme produces state-of-the-art results on three challenging datasets: SIFT Flow [18], LM+SUN [27], and CamVid [5]. In particular, the LM+SUN dataset, with 45,676 images and 232 labels, has the broadest coverage of any image parsing benchmark to date.

Complete code and results for our system can be found at `http://www.cs.unc.edu/SuperParsing`.

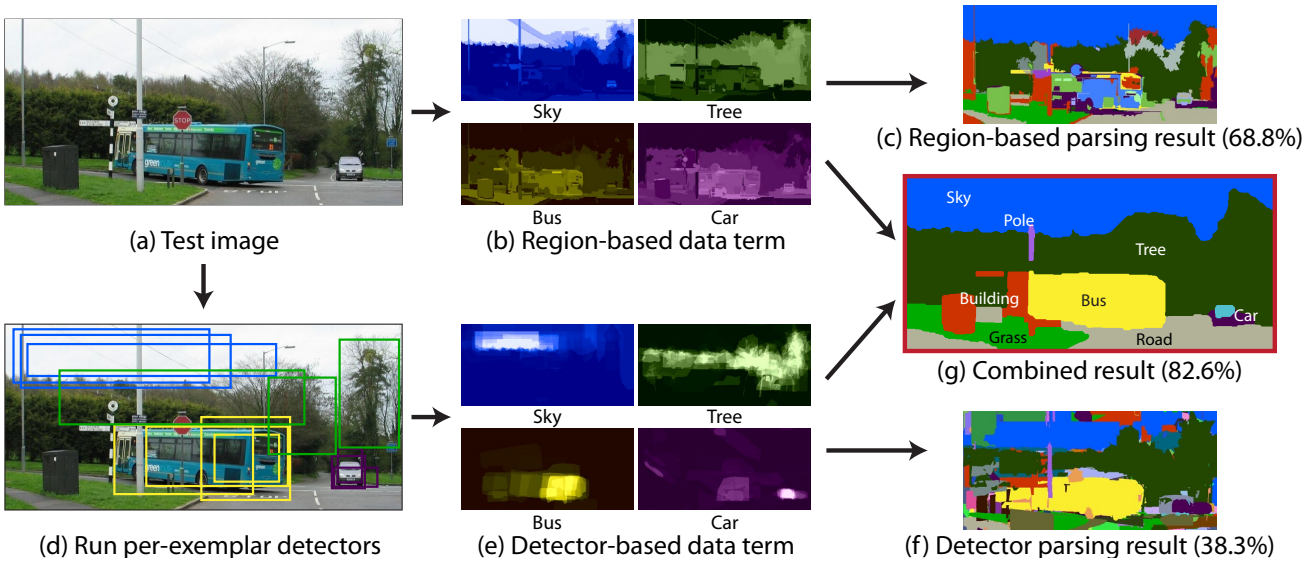

Figure 1. Overview and sample result of our approach. The test image (a) contains a bus – a relatively rare "thing" class. Our region-based parsing system [27] computes class likelihoods (b) based on superpixel features, and it correctly identifies "stuff" regions like sky, road, and trees, but is not able to get the bus (c). To find "things" like bus and car, we run per-exemplar detectors [19] on the test image (d) and transfer masks corresponding to detected training exemplars (e). Since the detectors are not well suited for "stuff," the result of detector-based parsing (f) is poor. However, combining region-based and detection-based data terms (g) gives the highest accuracy of all and correctly labels most of the bus and part of the car.

2. Method

This section presents our hybrid image parsing method as illustrated in Figure 1. Sections 2.1 and 2.2 describe its region- and detector-based components, and Section 2.3 details our proposed combination method.

2.1. Region-Based Parsing

For region-based parsing, we use the scalable nonparametric system we have developed earlier [27]. Given a query image, this system first uses global image descriptors to identify a *retrieval set* of training images similar to the query. Then the query is segmented into regions or superpixels; each region is matched to regions in the retrieval set based on 20 features representing position, shape, color, and texture and these matches are used to produce a log-likelihood ratio score $L(s_i, c)$ for label c at region s_i:

$$L(s_i, c) = \log \frac{P(s_i|c)}{P(s_i|\bar{c})} = \sum_k \log \frac{P(f_i^k|c)}{P(f_i^k|\bar{c})}\,, \qquad (1)$$

where $P(f_i^k|c)$ (resp. $P(f_i^k|\bar{c})$) is the likelihood of feature type k for region i given class c (resp. all classes but c).

For large-scale datasets with many labels (SIFT Flow and LM+SUN in our experiments), we obtain the log-likelihood ratio score based on nonparametric nearest-neighbor estimates (see [27] for details). For smaller-scale datasets with few classes (CamVid), we obtain it from the output of a boosted decision tree classifier. Either way, we use this score to define our *region-based data term* E_{R} for each pixel p and class c:

$$E_{\mathrm{R}}(p, c) = L(s_p, c)\,, \qquad (2)$$

where s_p is the region containing p. Figure 1(b) shows the detector-based data terms for the test image in Figure 1(a).

2.2. Detector-Based Parsing

Following the per-exemplar framework of [19], we train a per-exemplar detector for each labeled object instance in our dataset. While it may seem intuitive to only train detectors for "thing" categories, we train them for *all* categories, including ones seemingly inappropriate for a sliding window approach, such as "sky." As our experiments will demonstrate, this actually yields the best results for the combined region- and detector-based system. We follow the detector training procedure of [19], with negative mining done on all training images that do not contain an object of the same class. For our largest LM+SUN dataset we only do negative mining on 1,000 training images most similar to the positive exemplar's image (we have found that using more does not increase the detection accuracy).

At test time, given an image that needs to be parsed, we first obtain a retrieval set of globally similar training images as in Section 2.1. Then we run the detectors associated with the first k instances of each class in that retrieval set (the instances are ranked in decreasing order of the similarity of their image to the test image, and different instances in the same image are ranked arbitrarily). We restrict k purely to reduce computation; all our experiments use $k = 100$. Next, we take all detections that are above a given threshold t_d (we use the negative margin or $t_d = -1$ as suggested in [19]). For each detection we project the associated object mask into the detected bounding box (Figure 2). To compute the *detector-based data term* E_{D} for a class c and pixel

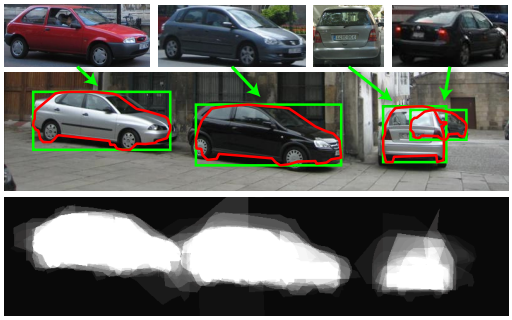

Figure 2. Computation of the detector-based data term. For each positive detection (green bounding box) in the test image (middle row) we transfer the mask (red polygon) from the associated exemplar (top) into the test image. The data term for "car" (bottom) is obtained by summing all the masks weighted by their detector responses.

p, we simply take the sum of all detection masks from that class weighted by their detection scores:

$$E_{\mathrm{D}}(p,c) = \sum_{d \in D_{p,c}} (w_d - t_d)\,, \tag{3}$$

where $D_{p,c}$ is the set of all detections for class c whose transferred mask overlaps pixel p and w_d is the detection score of d. Figure 1(e) shows some detector-based data terms for the test image of Figure 1(a).

Note that the full training framework of [19] includes computationally intensive calibration and contextual pooling procedures that are meant to make scores of different per-exemplar detectors more consistent. In our implementation, we have found these steps to be unnecessary, as they are at least partially superceded by the combined region- and detector-based inference scheme described next.

2.3. SVM Combination and MRF Smoothing

Once we run the parsing systems of Sections 2.1 and 2.2 on a test image, for each pixel p and each class c we end up with two data terms, $E_{\mathrm{R}}(p,c)$ and $E_{\mathrm{D}}(p,c)$, as defined by eqs. (2) and (3). For a dataset with C classes, concatenating these values gives us a $2C$-dimensional feature vector at each pixel. Next, we train C one-vs-all SVMs, each of which takes as input the $2C$-dimensional feature vectors and returns final per-pixel scores for a given class c.

Training data for each SVM is generated by running region- and detector-based parsing on the entire training set using a leave-one-out method: for each training image a retrieval set of similar training images is obtained, regions are matched to generate $E_{\mathrm{R}}(p,c)$, and the per-exemplar detectors from the retrieval set are run to generate $E_{\mathrm{D}}(p,c)$. Unfortunately, the resulting amount of data is huge: our largest dataset has over nine billion pixels, which would require 30 terabytes of storage. To make SVM training feasible, we must subsample the data – a tricky task given the unbalanced class frequencies in our many-category datasets.

We could subsample the data uniformly (i.e., reduce the number of points by a fixed factor without regard to class labels). This preserves the relative class frequencies, but in practice it heavily biases the classifier towards the more common classes. Conversely, subsampling the data so that each class has a roughly equal number of points produces a bias towards the rare classes. We have found that combining these two schemes in a 2:1 ratio achieves a good trade-off on all our datasets. That is, we obtain 67% of the training data by uniform sampling and 33% by even per-class sampling. We train all SVMs on 250,000 points – using more did not significantly improve performance for any of our setups.

For training one-vs-all SVMs, we normalize each feature dimension by its standard deviation and use fivefold cross-validation to find the regularization constant. Another important aspect of the implementation is the choice of the SVM kernel. As will be shown in Section 3, the linear kernel already does quite well, but we can obtain further improvements with the RBF kernel. Since it is infeasible to train a nonlinear SVM with the RBF kernel on our largest dataset, we approximate it by training a linear SVM on top of the random Fourier feature embedding [23]. We set the dimensionality of the embedding to 4,000 and find the kernel bandwidth using fivefold cross-validation. Experiments on our two smaller datasets confirm the quality of the approximation (Table 2).

The resulting SVMs produce C responses at each pixel. Let $E_{\mathrm{SVM}}(p_i,c_i)$ denote the response of the SVM for class c_i at pixel p_i. To obtain the final labeling, we can simply take the highest-scoring label at each pixel, but this produces noisy results. We smooth the labels with an MRF energy function similar to [18, 25] defined over the field of pixel labels $\mathbf{c}$:

$$\begin{aligned} J(\mathbf{c}) = &\sum_{p_i \in I} \max[0, M - E_{\mathrm{SVM}}(p_i,c_i)] \\ &+\lambda \sum_{(p_i,p_j)\in\epsilon} E_{\mathrm{smooth}}(c_i,c_j)\,, \end{aligned} \tag{4}$$

where I is the set of all pixels in the image, ϵ is the set of adjacent pixels, M is the highest expected value of the SVM response (about 10 on our data), λ is a smoothing constant (we set $\lambda = 16$), and $E_{\mathrm{smooth}}(c_i,c_j)$ imposes a penalty when two adjacent pixels (p_i,p_j) are similar but are assigned different labels (c_i,c_j) (see eq. (8) in [18]). We perform MRF inference using α-expansion [4, 15].

3. Evaluation

3.1. Datasets

The first dataset in our experiments, **SIFT Flow** [18], consists of outdoor scenes. It has 2,488 training images, 200 test images, and 33 labels. For this dataset, we use retrieval set size of 200. Our second dataset, **LM+SUN** [27], was collected from the SUN dataset [28] and LabelMe [24]. It contains 45,676 images (21,182 indoor and 24,494 outdoor)

	SIFT Flow		LM+Sun		CamVid	
	Per-Pixel	Per-Class	Per-Pixel	Per-Class	Per-Pixel	Per-Class
Detector ML	65.1	25.8	33.0	14.1	61.2	45.5
Detector SVM	62.5	25.4	46.1	12.0	61.4	47.0
Detector SVM MRF	71.1	26.7	52.5	11.3	63.8	47.3
Region ML	74.1	30.2	51.5	7.5	82.7	51.2
Region SVM	75.0	35.9	56.3	6.7	81.4	55.7
Region SVM MRF	77.7	32.8	58.3	5.9	83.5	55.7
Region + Thing SVM	74.4	36.9	58.5	14.1	82.4	60.0
Region + Thing SVM MRF	77.5	35.7	60.0	12.9	84.2	59.5
Combined SVM	75.6	**41.1**	59.6	**15.5**	82.3	62.1
Combined SVM MRF	**78.6**	39.2	**61.4**	15.2	**84.0**	**62.2**

Table 1. Comparison of different data terms. All SVMs use the approximate RBF embedding. Detector ML and Region ML directly assign the highest-scoring label at each pixel based on the respective data terms, while Detector SVM and Region SVM use SVM outputs trained on the individual data terms. Region + Thing uses the SVM trained on the full region data term and the subset of the detector data term corresponding to "thing" classes.

	SIFT Flow		LM+Sun		CamVid	
	Per-Pixel	Per-Class	Per-Pixel	Per-Class	Per-Pixel	Per-Class
Linear	75.4	40.0	57.2	16.6	82.4	60.7
Linear MRF	77.5	40.2	59.5	15.9	83.8	60.7
Approx. RBF	75.6	41.1	59.6	15.5	82.3	62.1
Approx. RBF MRF	78.6	39.2	61.4	15.2	83.9	62.5
Exact RBF	75.4	41.6	N/A	N/A	82.3	61.9
Exact RBF MRF	77.6	42.0	N/A	N/A	84.0	62.2

Table 2. Comparison of different SVM kernels. The RBF kernel has a slight edge over the linear kernel, and the approximate RBF embedding of [23] has comparable performance to the exact nonlinear RBF. Note that training the exact RBF on the largest LM+SUN dataset was computationally infeasible.

and 232 labels. We use the split of [27], which consists of 45,176 training and 500 test images. Since this is a much larger dataset, we set the retrieval set size to 800.

Our third dataset, **CamVid** [5], consists of video sequences taken from a moving vehicle and densely labeled at 1Hz with 11 class labels. It is much smaller – a total of 701 labeled frames split into 468 training and 233 testing. While CamVid is not our target dataset type, we use it for comparison with a number of recent approaches [5, 10, 16, 26, 29]. Unlike SIFT Flow and LM+SUN, CamVid does not have object polygons, only per-pixel labels. For training detectors, we fit a bounding box and a segmentation mask to each connected component of the same label type. Thus, if multiple object instances (e.g., cars) overlap, they are treated as one exemplar. Following our earlier work [27], we segment the video using the method of Grundmann et al. [11], and use boosted decision tree classifiers instead of nonparametric likelihood estimates. To obtain training data for the SVM, we compute the responses of the boosted decision tree classifiers on the same images on which they were trained (we have found this to work better than cross-validation on this dataset). We do not enforce any spatio-temporal consistency in the final labeling (in [27], enforcing consistency produced more visually pleasing results but had little effect on the numbers).

On all datasets, we report the overall per-pixel rate (percent of test set pixels correctly labeled), which is dominated by the most common classes, as well as the average of per-class rates, which is dominated by the rarer classes.

3.2. Experiments

First, we analyze the contributions of individual components of our system. In particular, one may wonder whether the power of our approach truly lies in combining detector- and region-based cues, or whether most of our performance gain comes from the extra layers of SVM and MRF inference. Table 1 shows the performance of various combinations of region- and detector-based data terms with and without SVM training, with and without MRF smoothing. The region-based data term obtains higher per-pixel accuracy than the detector-based one on all three datasets, and higher per-class accuracy on SIFT Flow and CamVid. On the LM+SUN dataset, which has the largest number of rare "thing" classes, the detector-based data term actually obtains higher per-class accuracy than the region-based one. While the SVM can sometimes improve performance when applied to the individual data terms, applying it to their combination gives by far the biggest and most consistent improvements. As observed in [27], MRF inference further raises the per-pixel rate, but often lowers the per-class rate by smoothing away some of the smaller objects.

Because part of our motivation for incorporating detectors is to improve performance on the "thing" classes, we want to know what will happen if we train detectors only on "things" – if detectors are completely inappropriate for "stuff," then not using them on "stuff" may improve accuracy, not to mention speed up the system considerably. The "Region + Thing" section of Table 1 shows the performance of the SVM trained on the full region-based data term and the subset of the detector-based data term corresponding

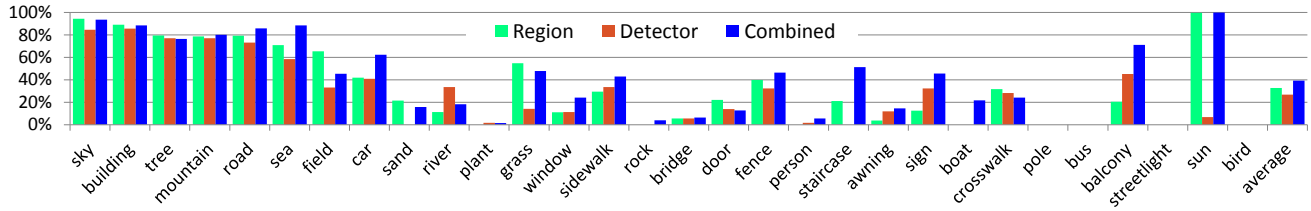

Figure 3. Classification rates of individual classes (ordered from most to least frequent) on the SIFT Flow dataset for region-based, detector-based, and combined parsing. All results include SVM and MRF smoothing.

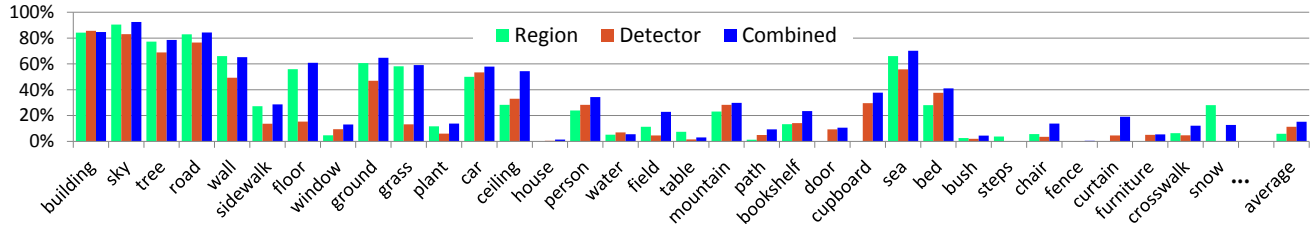

Figure 4. Classification rates of the most common individual classes (ordered from most to least frequent) on the LM+SUN dataset for region-based, detector-based, and combined parsing. All results include SVM and MRF smoothing.

SIFT Flow	Per-Pixel	Per-Class
Ours: Combined MRF	78.6	39.2
Tighe and Lazebnik [27]	77.0	30.1
Liu et al. [18]	76.7	N/A
Farabet et al. [8]	78.5	29.6
Farabet et al. [8] balanced	74.2	46.0
Eigen and Fergus [7]	77.1	32.5
Myeong et al. [22]	77.1	32.3

Table 3. Comparison to state-of-the-art on the SIFT Flow dataset.

LM+SUN	Per-Pixel	Per-Class
Ours: Combined MRF	61.4	15.2
Outdoor Images	65.5	15.3
Indoor Images	46.3	12.2
Tighe and Lazebnik [27]	54.9	7.1
Outdoor Images	60.8	7.7
Indoor Images	32.1	4.8

Table 4. Comparison to [27] on the LM+SUN dataset with results broken down by outdoor and indoor test images.

only to "thing" classes (specified manually). Interestingly, the results for this setup are weaker than those of the full combined system using both "thing" and "stuff" detectors.

Next, Table 2 compares SVMs with the linear kernel, approximate RBF embedding [23], and exact nonlinear RBF. The linear kernel may be a better choice if speed is a concern, but the approximate and exact RBF are able to boost performance by 1-2%. All subsequent figures and tables will report only the SVM results with the approximate RBF.

Figures 3 and 4 show the per-class rates of our system on the most common classes in the SIFT Flow and LM+SUN datasets, respectively. As expected, adding detectors significantly improves many "thing" classes (including car, sign, and balcony) but also some "stuff" classes (road, sea, sidewalk, fence). Figure 5 gives a close-up look at our performance on many small object categories, and Figure 6 shows several parsing examples on the LM+SUN dataset.

Table 3 compares our combined system to a number of state-of-the-art approaches on the SIFT Flow dataset. We outperform them, in many cases beating the average per-class rate by up to 10% while maintaining or exceeding the per-pixel rates. The one exception is the system of Farabet et al. [8] when tuned for balanced per-class rates, but their per-pixel rate is much lower than ours in this case. When their system is tuned to a per-pixel rate similar to ours, their average per-class rate drops significantly below ours.

On LM+SUN, which has an order of magnitude more images and labels than SIFT Flow, the only previously reported results are from our earlier region-based system [27]. As Table 4 shows, by augmenting the region-based term with a novel detector-based data term and SVM inference, we are able to raise the per-pixel rate from 54.9% to 61.4% and the per-class rate from 7.1% to 15.2%.

Table 5 compares our per-class rates on the CamVid dataset to a number of recent methods. When compared to our region-based system [27], we improve performance for every class except for building and sky, towards which the region-based parser seems to be overly biased. Furthermore, we match the state-of-the-art method of Ladicky et al. [16], which incorporates DPM detectors, bounding box segmentation, and a complex CRF model. Figure 7 shows the output of our system on example CamVid images.

3.3. Running Time

Finally, we examine the computational requirements of our system on our largest dataset, LM+SUN, by timing our MATLAB implementation (feature extraction and file I/O excluded) on a six-core 3.4 GHz Intel Xeon processor with 48 GB of RAM. There are a total of 354,592 objects in the training set, and we train a per-exemplar detector for each of them. The average training time per detector is 472.3 seconds; training all of them would require 1,938 days on a single CPU, but we do it on a 512-node cluster in approxi-

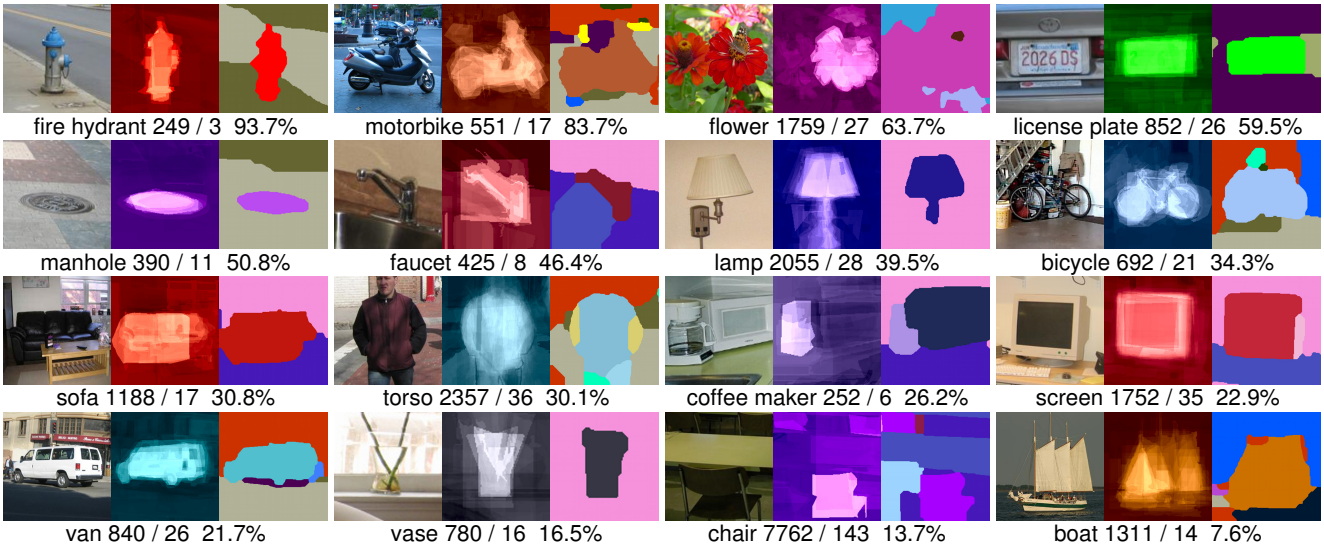

Figure 5. Examples of "thing" classes on LM+SUN. For each class we show a crop of an image, the SVM combined output, and the smoothed final result. The caption for each class shows: (# of training instances of that class) / (# of test instances) (per-pixel rate on the test set)%. Best viewed in color.

	Building	Tree	Sky	Car	Sign	Road	Pedestrian	Fence	Pole	Sidewalk	Bicyclist	Per-class	Per-pixel
Ours: Combined MRF	83.1	73.5	94.6	78.1	**48**	96	**58.6**	32.8	5.3	71.2	**45.9**	**62.5**	**83.9**
Tighe and Lazebnik [27]	**87.0**	67.1	96.9	62.7	30.1	95.9	14.7	17.9	1.7	70.0	19.4	51.2	83.3
Brostow et al. [5]	46.2	61.9	89.7	68.6	42.9	89.5	53.6	46.6	0.7	60.5	22.5	53.0	69.1
Sturgess et al. [26]	84.5	72.6	**97.5**	72.7	34.1	95.3	34.2	45.7	8.1	77.6	28.5	59.2	83.8
Zhang et al. [29]	85.3	57.3	95.4	69.2	46.5	**98.5**	23.8	44.3	**22.0**	38.1	28.7	55.4	82.1
Floros et al. [10]	80.4	76.1	96.1	86.7	20.4	95.1	47.1	47.3	8.3	79.1	19.5	59.6	83.2
Ladicky et al. [16]	81.5	**76.6**	96.2	**78.7**	40.2	93.9	43.0	**47.6**	14.3	**81.5**	33.9	**62.5**	83.8

Table 5. Comparison to state of the art on the CamVid dataset.

mately four days. Leave-one-out parsing of the training set (see below for average region- and detector-based parsing times per image) takes 939 hours on a single CPU, or about two hours on the cluster. Next, training a set of 232 one-vs-all SVMs takes a total of one hour on a single machine for the linear SVM and ten hours for the approximate RBF. Note that the respective feature dimensionalities are 464 and 4,000; this nearly tenfold dimensionality increase accounts for the tenfold increase in running time. Tuning the SVM parameters by fivefold cross-validation on the cluster only increases the training time by a factor of two.

At test time, the region-based parsing takes an average of 27.5 seconds per image. The detector-based parser runs an average of 4,842 detectors per image in 47.4 seconds total. The final combination (SVM testing) step takes an average of 8.9 seconds for the linear kernel and 124 seconds for the approximate RBF (once again, the tenfold increase in feature dimensionality and the overhead of computing the embedding account for the increase in running time). MRF inference takes only 6.8 seconds per image.

4. Discussion

Our current system achieves very promising results, but at a considerable computational cost. Reducing this cost is an important future research direction. To speed up training of per-exemplar detectors, we plan to try the whitened HOG method of Hariharan et al. [13]. At test time, we would like to to reduce the number of detectors that need to be run per image. As shown in Table 1, doing this naively, e.g., by running only "thing" detectors, lowers the accuracy. Instead, we want to develop methods for dynamically selecting detectors for each test image based on context. Also, SVM testing with the approximate RBF embedding imposes a heavy overhead in our current implementation. However, this is a nuisance that can be resolved with better choice of embedding dimensionality and more optimized code.

Ultimately, we want our system to function on *open universe* datasets, such as LabelMe [24], that are constantly evolving and do not have a pre-defined list of classes of interest. The region-based component of our system [27] already has this property. In principle, per-exemplar detectors are also compatible with the open-universe setting, since they can be trained independently as new exemplars come in. Our SVM combination step is the only one that relies on batch offline training (including leave-one-out parsing of the entire training set). In the future, we plan to investigate online methods for this step.

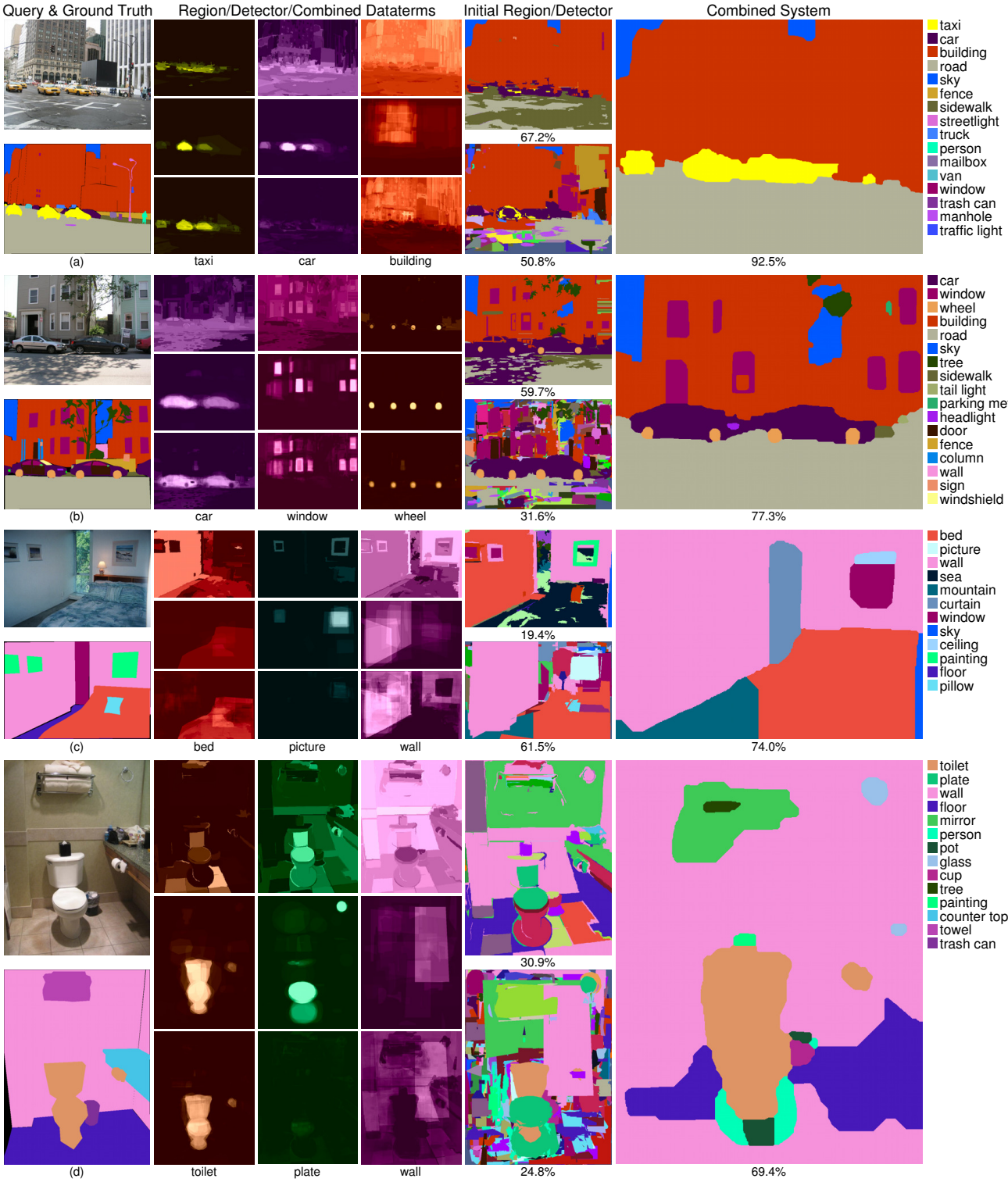

Figure 6. Example results on the LM+SUN dataset (best viewed in color). First column: query image (top) and ground truth (bottom). Second through fourth columns: region-based data term (top), detector-based data term (middle), and SVM combination (bottom) for three selected class labels. Fifth column: region-based parsing results (top) and detector-based parsing results (bottom) without SVM or MRF smoothing. Right-most column: smoothed combined output. The example in (a) has strong detector responses for both "car" and "taxi," and the SVM suppresses the former in favor of the latter. In (b), the system correctly identifies the wheels of the cars and the headlight of the left car. In (c), the detectors correctly identify the wall and most of the bed. Note that the region-based parser alone mislabels most of the bed as "sea"; the detector-based parser does much better but still mislabels part of the bed as "mountain." In this example, the detector-based parser also finds two pictures and a lamp that do not survive in the final output. In (d), our system successfully finds the toilet. Note that both the region- and detector-based data terms assign very high likelihood of "plate" to the toilet bowl, but the SVM suppresses "plate" in favor of "toilet."

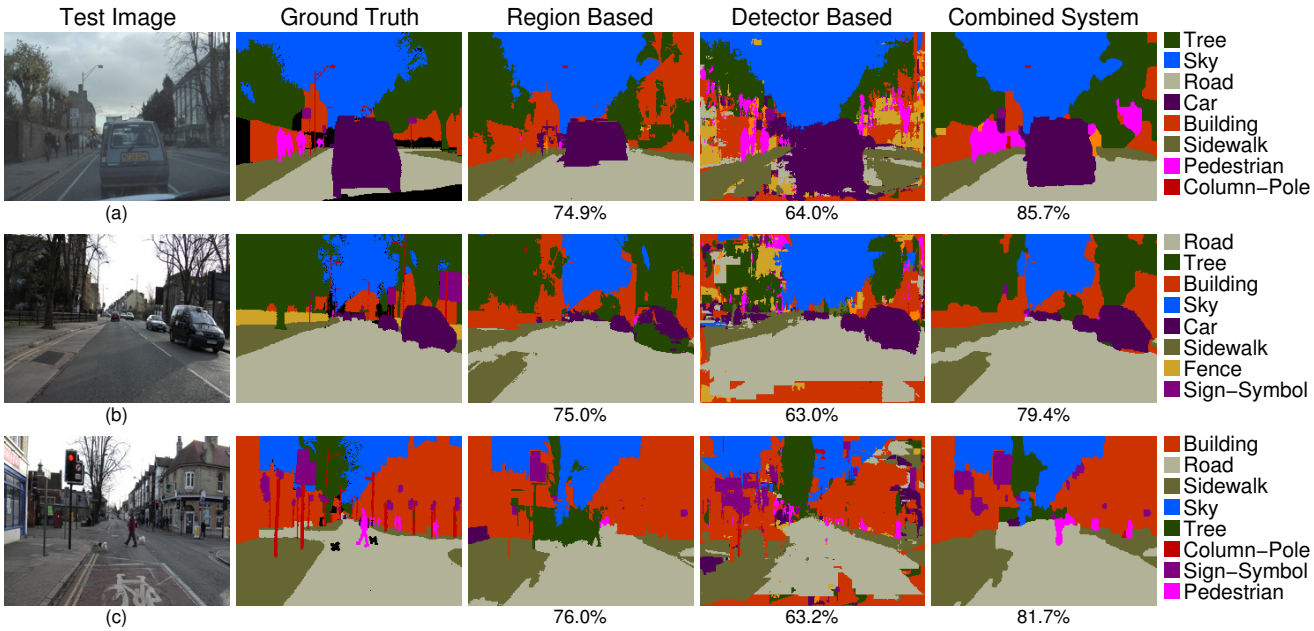

Figure 7. Example results on the CamVid dataset. The region- and detector-based results are shown without SVM or MRF smoothing; the final output has both. Notice how the detectors are able to complete the car in (a) and (b). In (c) we are able to correctly parse a number of pedestrians and signs.

Acknowledgments. This research was supported in part by NSF grant IIS 1228082, DARPA Computer Science Study Group (D12AP00305), Microsoft Research Faculty Fellowship, and Xerox.

References

[1] P. Arbelaez, B. Hariharan, C. Gu, S. Gupta, and L. Bourdev. Semantic segmentation using regions and parts. In *CVPR*, 2012. 1

[2] E. Borenstein and S. Ullman. Class-specific, top-down segmentation. In *ECCV*, 2002. 1

[3] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009. 1

[4] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124–37, Sept. 2004. 3

[5] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *ECCV*, 2008. 1, 4, 6

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1

[7] D. Eigen and R. Fergus. Nonparametric image parsing using adaptive neighbor sets. In *CVPR*, 2012. 1, 5

[8] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Scene parsing with multiscale feature learning, purity trees, and optimal covers. *Arxiv preprint arXiv:1202.2160 [cs.CV]*, 2012. 1, 5

[9] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008. 1

[10] G. Floros, K. Rematas, and B. Leibe. Multi-class image labeling with top-down segmentation and generalized robust P^N potentials. In *BMVC*, 2011. 1, 4, 6

[11] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, 2010. 4

[12] R. Guo and D. Hoiem. Beyond the line of sight: labeling the underlying surfaces. In *ECCV*, 2012. 1

[13] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *ECCV*, 2012. 6

[14] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, pages 30–43, 2008. 1

[15] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2):147–59, Feb. 2004. 3

[16] L. Ladický, P. Sturgess, K. Alahari, C. Russell, and P. H. S. Torr. What, where and how many? Combining object detectors and CRFs. In *ECCV*, 2010. 1, 4, 5, 6

[17] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *IJCV*, 77(13):259289, 2008. 1

[18] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing via label transfer. *PAMI*, 33(12):2368–2382, June 2011. 1, 3, 5

[19] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-SVMs for object detection and beyond. In *ICCV*, 2011. 1, 2, 3

[20] M. Marszałek and C. Schmid. Accurate object recognition with shape masks. *IJCV*, 97(2):191–209, 2012. 1

[21] D. Munoz, J. A. Bagnell, and M. Hebert. Stacked hierarchical labeling. In *ECCV*, pages 57–70, 2010. 1

[22] H. J. Myeong, Y. Chang, and K. M. Lee. Learning object relationships via graph-based context model. *CVPR*, June 2012. 1, 5

[23] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007. 3, 4, 5

[24] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008. 3, 6

[25] J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006. 1, 3

[26] P. Sturgess, K. Alahari, L. Ladický, and P. H. S. Torr. Combining appearance and structure from motion features for road scene understanding. *BMVC*, 2009. 1, 4, 6

[27] J. Tighe and S. Lazebnik. SuperParsing: Scalable nonparametric image parsing with superpixels. *IJCV*, 101(2):329–349, Jan 2013. 1, 2, 3, 4, 5, 6

[28] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, June 2010. 3

[29] C. Zhang, L. Wang, and R. Yang. Semantic segmentation of urban scenes using dense depth maps. In *ECCV*, 2010. 1, 4, 6