RGB-(D) Scene Labeling: Features and Algorithms

Xiaofeng Ren
ISTC-Pervasive Computing
Intel Labs
xiaofeng.ren@intel.com

Liefeng Bo and Dieter Fox
Computer Science and Engineering
University of Washington
{lfb,fox}@cs.washington.edu

Abstract

Scene labeling research has mostly focused on outdoor scenes, leaving the harder case of indoor scenes poorly understood. Microsoft Kinect dramatically changed the landscape, showing great potentials for RGB-D perception (color+depth). Our main objective is to empirically understand the promises and challenges of scene labeling with RGB-D. We use the NYU Depth Dataset as collected and analyzed by Silberman and Fergus [30]. For RGB-D features, we adapt the framework of kernel descriptors *that converts local similarities (kernels) to patch descriptors. For contextual modeling, we combine two lines of approaches, one using a superpixel MRF, and the other using a segmentation tree. We find that (1) kernel descriptors are very effective in capturing appearance (RGB) and shape (D) similarities; (2) both superpixel MRF and segmentation tree are useful in modeling context; and (3) the key to labeling accuracy is the ability to efficiently train and test with large-scale data. We improve labeling accuracy on the NYU Dataset from* 56.6% *to* $\mathbf{76.1}\%$*. We also apply our approach to image-only scene labeling and improve the accuracy on the Stanford Background Dataset from* 79.4% *to* $\mathbf{82.9}\%$*.*

1. Introduction

Scene labeling, aiming to densely label everything in a scene, is a fundamental problem and extensively studied. Most scene labeling research focused on outdoor scenes [29, 13, 8]. Perhaps with the exception of Manhattan world layout [20, 11], indoor scene labeling has been largely ignored, even though people spend most time indoors. This is partly because indoor scenes are the harder case [25]: challenges include large variations of scene types, lack of distinctive features, and poor illumination.

The release of Microsoft Kinect [22], and the wide availability of affordable RGB-D sensors (color+depth), changed the landscape of indoor scene analysis. Using active sensing, these RGB-D cameras provide synchronized color and depth information. They not only provide direct 3D information that's lost in typical camera projection, but also provide a channel independent of ambient illumination. For a wide range of problems, depth and color+depth dramatically increased accuracy and robustness, such as in body pose estimation [28], 3D mapping [12], object recognition [19], and 3D modeling and interaction [14].

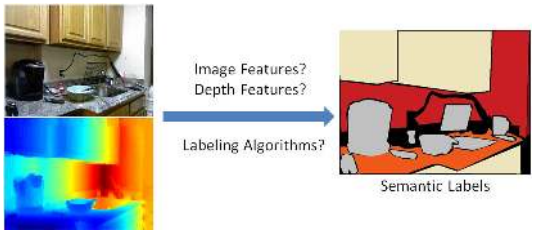


Figure 1. We jointly use color and depth from a Kinect-style RGB-D sensor to label indoor scenes.

How much does the RGB-D revolution change indoor scene labeling? Silberman and Fergus [30] early adopted Kinect for RGB-D scene labeling. The NYU work achieved 56.6% accuracy on 13 semantic categories over 7 scene types with encouraging results on SIFT features, relative depth, and MRFs. A related work from Cornell also showed promising results labeling 3D point clouds from merged Kinect frames [15].

In this paper, we build on the NYU work and develop and evaluate a scene labeling approach that combines rich RGB-D features and contextual models using MRFs and hierarchical segmentation. We carry out extensive studies of features and models on the NYU dataset. The class-average accuracy of our best model reaches $\mathbf{76.1}\%$ accuracy, a large step forward on the state of the art in RGB-D scene labeling.

We achieve high labeling accuracy by studying both RGB-D features and labeling algorithms. Motivated by progress in object recognition, we use *kernel descriptors* [2, 3] to capture a variety of RGB-D features such as gradient, color, and surface normal. We show that linear SVMs work well to utilize large-scale data to classify superpixels [26, 33] and paths in segmentation trees [21, 23], and, interestingly, combine well with superpixel MRFs.

While our main focus is on RGB-D scene labeling, our approach also applies to image-only scene labeling. We val-

idate using the Stanford Background Dataset [8] with 8 semantic categories. Again, we find large improvements using our approach: while previous works reported pixel accuracy between 76% and 79%, we achieve **82.9**% accuracy using kernel descriptors along with segmentation tree plus MRF.

2. Related Works

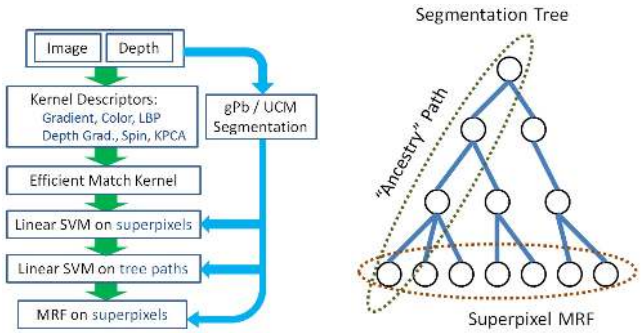
Scene labeling has been studied extensively. A lot of work has been put into modeling *context*, through the use of Markov random fields (MRF) [8, 18, 16, 32] or conditional random fields (CRF) [17, 10, 13, 29, 9]. He et al. proposed multi-scale CRFs for learning label patterns [10]. Gould et al. encoded relative positions between labels [9] and developed inference techniques for MRFs with pairwise potentials [8]. Ladicky et al. used hierarchical MRFs combining pixels and regions [18]. Tighe and Lazebnik combined scene recognition and MRF-based superpixel matching on large datasets [32]. Socher et al. used recursive neural networks for scene parsing [31].

While superpixels have been widely used both for efficiency [13] and for aggregating local cues [33], it is well known that segmentation is far from perfect and its imprecision hurts labeling accuracy. This motivated approaches using multiple segmentations [27, 16, 6] or hierarchical segmentations [21, 23]. Kumar and Koller searched through multiple segmentations [16]. Lim et al. used segmentation "ancestry" or paths to do exemplar-based distance learning [21]. Munoz et al. used stacked classification to classify nodes in a segmentation tree from top down [23].

The release of Kinect [22] and other cheap depth sensors [24] has been transforming the landscape of vision research. The Kinect work used a large-data approach to solve the body pose problem using depth only [28]. At the junction of vision and robotics, there have been a series of works on RGB-D perception, combining color and depth channels for 3D mapping and modeling [12, 14], object recognition [19] and point cloud labeling [15].

For indoor scene labeling, Silberman and Fergus [30] presented a large-scale RGB-D scene dataset, and carried out extensive studies using SIFT and MRFs. Our work is motivated and directly built on top of theirs, demonstrating the need for rich features and large-scale data. Related but different are the works on indoor scene recognition [25] and Manhattan world box labeling [20, 11].

What features should be used in scene labeling? The TextonBoost features from Shotton et al [29] have been popular in scene labeling [8, 18, 16], which use boosting to transform the outputs from a multi-scale texton filterbank. Recently, SIFT and HOG features start to see more use in scene labeling [30], bringing it closer to the mainstream of object recognition. We will use the newly developed *kernel descriptors*, which capture different aspects of similarity in a unified framework [2, 4, 3].



(a) Flowchart (b) Tree path + MRF

Figure 2. We use kernel descriptors (KDES) [2] to capture both image and depth cues. Transformed through efficient match kernels (EMK) [4], we use linear SVM classification both on superpixels and on paths in segmentation trees. We compare and combine tree path classification with pairwise MRF.

3. Overview

Indoor scene labeling is a challenging and poorly understood problem. Kinect-style RGB-D cameras, active sensors that provide synchronized depth and color, provide high hopes but do not automatically solve the problem. The pioneering work of Silberman and Fergus [30] showed that RGB-D significantly improves scene labeling, but still the accuracy is near 50%, much lower than outdoor scenes [8].

We seek robust solutions to RGB-D scene labeling that can achieve a much higher accuracy, studying both features and labeling algorithms. An outline of our approach is shown in Figure 2. For RGB-D features, we follow a proven strategy in object recognition by extracting rich features at low-level and encoding them to be used in an efficient classifier, linear SVM in our case. We use *kernel descriptors* (KDES) [2, 3], a unified framework that uses different aspects of similarity (kernel) to derive patch descriptors. Kernel descriptors are aggregated over superpixels and transformed using *efficient match kernels* (EMK) [4].

For contextual modeling, we combine and validate two strategies, one using segmentation trees, and the other using superpixel MRFs. We use gPb [1] (modified for RGB-D) to construct a fixed-height segmentation tree, and classify features accumulated over paths from leaf to root. For the MRF we use linear SVM scores and gPb transitions.

3.1. Scene Labeling Datasets

The main focus of our work is on RGB-D scene labeling, validated using the recently collected and released NYU Depth Dataset [30] from Silberman and Fergus. The data covers 7 scene types in 2,284 Kinect frames (480x640) annotated through Mechanical Turk. Following the setup in [30], we use WordNet to reduce the labels to 12 common categories (see Fig. 8), plus one meta-category "background" for all other objects. We use class-average accu-

racy (mean of the diagonal of the confusion matrix) as the main evaluation criterion, excluding unlabeled regions.

We also show evaluations on the Stanford Background Dataset [8], commonly used in scene labeling research. The Stanford dataset contains 715 images of size 240x320, with 8 semantic categories as well as 3 geometric/surface categories. The evaluation criterion is the pixel-wise accuracy.

3.2. Generating Segmentation Trees

To generate segmentation trees, we use the widely used gPb/UCM hierarchical segmentation from Arbelaez et al. [1]. gPb combines a number of local and global contrast cues into a probability-of-boundary map on pixels. It is an interesting question, and beyond the scope of our work, what is the best way of adapting gPb to RGB-D frames. We use a simple strategy to combine color and depth images: run the gPb algorithm on the color image to obtain (soft and oriented) gPb_rgb, run the same algorithm on the depth map (in meters) to obtain gPb_d, and linearly combine them (before non-maximum suppression)

$$\text{gPb\_rgbd} = (1-\alpha)\cdot \text{gPb\_rgb} + \alpha \cdot \text{gPb\_d} \qquad (1)$$

While this linear combination is crude, we empirically find that it significantly improves gPb performance. We use $\alpha = 0.25$, and the F-measure values (from precision-recall evaluation as in [1]) are listed in Table 1.

Boundary Maps	Image	Depth	Image+Depth
F-measure	0.465	0.421	0.481

Table 1. F-measure evaluation for RGB-D boundary detection.

We threshold the UCM boundary map from gPb_rgbd at multiple levels. Each threshold creates a cut through the segmentation hierarchy, and the result is a tree of fixed height. The thresholds are chosen such that the number of segments are roughly half-octave apart.

4. Image and Depth Features

For image and depth features, we learn from object recognition research and employ features more sophisticated than typically used in scene labeling. In particular, we extensively use *kernel descriptors*, a flexible framework that has proven to be useful for RGB-D object recognition [3].

4.1. RGB-D kernel descriptors

Kernel descriptors (KDES) [2] is a unified framework for local descriptors: for any pixel-level similarity function, KDES transforms it into a descriptor over a patch. We use and evaluate six kernel descriptors (as in [3]): gradient (G), color (C), local binary pattern (L), depth gradient (GD), spin/surface normal (S), and KPCA/self-similarity (K).

As an example, we briefly describe the gradient kernel descriptor over depth patches. We treat depth images as

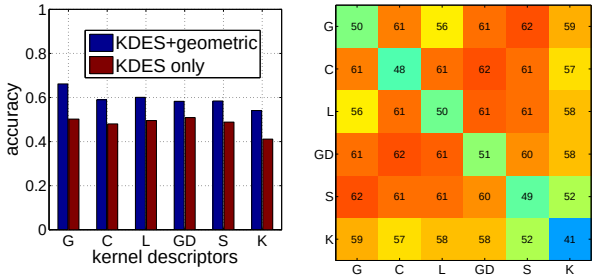


Figure 3. Evaluating six types of kernel descriptors. (left) Labeling accuracy using a single KDES (with and without geometric features), and (right) using a pair (without geometric features).

grayscale images and compute gradients at pixels. The gradient kernel descriptor F_{grad} is constructed from the pixel gradient similarity function k_o

$$F^t_{\text{grad}}(Z) = \sum_{i=1}^{d_o}\sum_{j=1}^{d_s} \alpha^t_{ij} \left\{ \sum_{z\in Z} \tilde{m}_z k_o(\tilde{\theta}_z, p_i) k_s(z, q_j) \right\} \qquad (2)$$

where Z is a depth patch, and $z \in Z$ are the 2D relative position of a pixel in a depth patch (normalized to $[0,1]$). $\tilde{\theta}_z$ and $\tilde{m}_z$ are the normalized orientation and magnitude of the depth gradient at a pixel z. The orientation kernel $k_o(\tilde{\theta}_z, \tilde{\theta}_x) = \exp(-\gamma_o \|\tilde{\theta}_z - \tilde{\theta}_x\|^2)$ computes the similarity of gradient orientations. The position Gaussian kernel $k_s(z,x) = \exp(-\gamma_s \|z-x\|^2)$ measures how close two pixels are spatially. $\{p_i\}_{i=1}^{d_o}$ and $\{q_j\}_{j=1}^{d_s}$ are uniformly sampled from their support region, d_o and d_s are the numbers of sampled basis vectors for the orientation and position kernels. α^t_{ij} are projection coefficients computed using kernel principal component analysis. Other kernel descriptors are constructed in a similar fashion from pixel-level similarity functions (see [2] and [3] for details).

In addition to the appearance features provided by KDES, we add a standard set of geometric/prior features [8]: position (up to 2^{nd} order), area, perimeter, moments. For RGB-D, we add relative depth (as in [30], up to 2^{nd} order) as well as the percentage of missing depth.

Figure 3 shows labeling accuracy using individual KDES and their combinations. For single KDES, the left panel shows results with and without geometric features. We see that they all perform reasonably well on the task, with the SIFT-like gradient KDES performing best on both image and depth, and the KPCA descriptor being the weakest. As can be seen in the right panel, a good feature pair typically mixes an image descriptor (No. 1-3) with a depth descriptor (No. 4-6). We observe the correlation (and redundancy) between gradient and local binary pattern descriptors, which is expected as they capture local variations in similar ways. While spin images were not used in [30], we find extensive uses for our spin KDES, a spin-image like descriptor encoding normals without orientation invariance. The best

combinations are: image gradient + spin/normal, and color + depth gradient. We use all four in our final experiments.

4.2. Classifying superpixels

We extract kernel descriptors over a dense grid, and use efficient match kernels (EMK) [4] to transform and aggregate descriptors in a set $\mathcal{S}$ (grid locations in the interior of a superpixel s). EMK combines the strengths of both bag-of-words and set kernels that maps kernel descriptors to a low dimensional feature space (see [4, 2] for details). We average the EMK features over the spatial support to obtain fixed-length features on superpixels.

Let $\Phi(s)$ be the combined features (KDES+geometric) over a superpixel s. For each layer (height) $t \in \{1, \cdots, T\}$ in the segmentation tree, we *separately* train a 1-vs-All linear SVM classifier: for each semantic class $c \in \{1, \cdots, C\}$ at height t, we have a linear scoring function

$$f_{t,c}(s) = w_{t,c}^{\top} \Phi(s) + b_{t,c} \tag{3}$$

One interesting question is how to weigh the data instances, as superpixels have very different sizes. Let c be the groundtruth class of s, A_s the area of s, and Q_c the set of all the superpixels q in the class c. We weigh $f(s)$ by

$$A_s / (\sum_{q \in Q_c} A_q)^p \tag{4}$$

where $0 \leq p \leq 1$ defines a tradeoff between class balance and pixel accuracy. We use $p = 1$ for NYU Depth, and $p = 0$ for Stanford Background. We will discuss the balancing issue in the experiments. A weighted version of *liblinear* is used for efficient training [7]. We set the groundtruth label of a superpixel as the majority class of the pixels within.

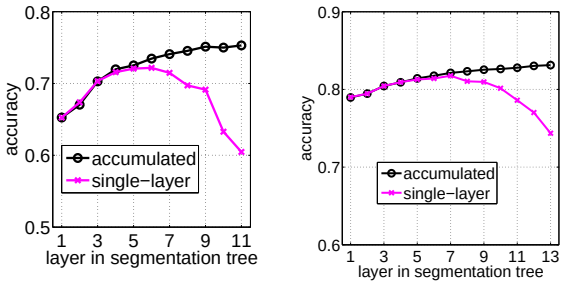
5. Contextual Models

For contextual modeling, we use both superpixel MRFs and paths in segmentation trees. Our superpixel MRF is defined over linear SVM outputs and gPb boundaries. Our tree path classification directly uses a linear SVM on concatenated features. We show that both models help and complement each other: combining them leads to a large boost.

5.1. Classifying paths in segmentation tree

As discussed, we construct a single segmentation tree of fixed height for each scene. For each leaf node, there is a unique path to the root. Comparing to earlier works that used the paths for exemplar distance learning [21] or stacked classification [23], we choose a direct approach by concatenating the *outputs* from (separately trained) linear SVMs computed on features at each layer, generating a tree feature for each superpixel s:

$$\text{Tree}(s) = \{f_{t,c}(s_t)\}, \ \forall t, c \tag{5}$$



(a) NYU Depth (b) Stanford

Figure 4. Change in accuracy when going through layers in segmentation trees: accumulating features or current layer only.

where s is a superpixel at the bottom layer, $\{s_t\}, t \in \{1, \cdots, T\}$ the ancestors of s, and $\{c\}$ all the classes. Classifiers on $\text{Tree}(s)$ are trained with groundtruth labels at the bottom layer. We again find linear SVMs efficient and performing well for classifying $\text{Tree}(s)$, better or no worse than other choices such as kernel SVM or sparse coding.

In Figure 4 we show superpixel labeling accuracies for each layer (height) in the segmentation tree, as well as the accuracies when we accumulate features along segmentation paths up to a certain layer. We see that single-layer classification has a "sweet spot", with superpixels being not too small, not too large. If we accumulate features over paths, the accuracy continues to increase to the top level, which has only a handful of segments. On the other hand, the initial part of the curves overlap, suggesting there is little benefit going to superpixels at too fine scales (about 200 per image suffice), consistent with previous studies.

5.2. Superpixel MRF with gPb

Our second contextual model is a standard MRF formulation. We use Graph Cut [5] to find the labeling that minimizes the energy of a pairwise MRF:

$$E(y_1, \cdots, y_{|S|}) = \sum_{s \in S} D_s(y_s) + \sum_{\{s,r\} \in N} V_{s,r}(y_s, y_r) \tag{6}$$

where y_s is the label of superpixel s, N is a set of all pairs of neighbors. For the data term D_s, we use $-f_{t,c}$, the output from the per-layer SVM, weighted by area. For the pairwise term $V_{s,r}$, we use

$$V_{s,r} = \beta \exp\left(-\gamma \cdot \text{gPb\_rgbd}(s, r)\right) \tag{7}$$

weighted by the length of the boundary between s and r. As the RGB-D gPb captures various grouping cues into a single value, the MRF only has two parameters, easy to set with cross-validation. We find the superpixel MRF useful both by itself and when combined with treepath classification.

6. Experimental Evaluations

We show our experimental analysis of the kernel descriptors and the labeling algorithms on both the NYU

SIFT+MRF [30]	$56.6 \pm 2.9\%$
KDES Superpixel (RGB)	$66.2 \pm 0.3\%$
KDES Superpixel (Depth)	$63.4 \pm 0.6\%$
KDES Superpixel (RGB-D)	$71.4 \pm 0.6\%$
KDES Treepath	$74.6 \pm 0.7\%$
KDES Superpixel MRF	$74.6 \pm 0.5\%$
KDES Treepath+Superpixel MRF	$\mathbf{76.1} \pm 0.9\%$

Table 2. Class-average accuracy on the NYU Depth dataset. Superpixel results (w/ and w/o MRF) are reported using the *best* layer in the segmentation tree (5^{th}).

Depth [30] and the Stanford Background Dataset [8]. We follow standard practices: for NYU Depth, we use 60% data for training and 40% for testing; for Stanford Background, 572 images for training and 143 images for testing. Unless otherwise mentioned, the accuracy on NYU Depth is the average over the diagonal of the confusion matrix, and the accuracy on Stanford Background the pixelwise accuracy. The final results in Table 2, 3, and 4 and Figure 5 are averaged over 5 random runs.

Extracting kernel descriptors. Kernel descriptors (KDES) are computed over a regular grid (with a stride of 2 pixels). They are transformed using efficient match kernels (EMK), to "soft" visual words, then averaged over superpixels. For gradient, color, local binary pattern, and depth gradient descriptors, we use a patch size of 16x16. For spin and kpca descriptors, we use a larger patch size of 40x40. Comparing to [3], we use larger scales for normal computation and spin descriptors, adapting the descriptors from targeting small objects to large scene elements.

Classifying superpixels and paths. For NYU Depth, we use 4 kernel descriptors: gradient, color, depth gradient, and spin kernel descriptors, with 200 words in the EMK transform. With 15 geometric features, the length of the feature vector is 815. We use segmentation trees of 11 levels. For Stanford Background, we use 3 kernel descriptors: gradient, color, and local binary pattern, and 400 words in EMK, making the feature length 1211. We use segmentation trees of 13 levels. For single-layer results, we always use the best performing layer (not the bottom layer).

Final Results on NYU Depth. Results are shown in Table 2. Comparing to [30], we achieve much higher accuracies in all cases. Single-layer superpixel classification gives us 71%. The two contextual models, classifying paths in segmentation tree and pairwise MRF using gPb boundaries, both provide a boost of over 3%. We empirically find that our path classification approach compares favorably (about 1% higher) to a stacked approach where classification scores are passed from layer to layer. Our MRF model is also effective compared to that in [30], partly because gPb captures all low-level boundary cues in one value, making our MRF easy to configure.

One interesting finding, which has not been explored before, is the combination of contextual modeling from segmentation paths and superpixel MRF. We find that they complement each other well, leading to the final accuracy of $\mathbf{76.1}\%$. Retrospectively, this makes a lot of sense, considering that a single segmentation tree may capture the "mode" of groupings and generate large-scale (and imprecise) segments, while a superpixel MRF captures soft local connectivities. This opens up future possibilities for exploring more sophisticated interactions of these two types of models. Examples are shown in Fig. 8 and 9.

	Pixelwise	Average
Region-based energy [8]	$76.4 \pm 1.2\%$	65.5%
Selecting regions [16]	$79.4 \pm 1.4\%$	-
Stacked Labeling [23]	76.9%	66.2%
Superpixel MRF [32]	77.5%	-
Recursive NN [31]	78.1%	-
This Work	$\mathbf{82.9} \pm 0.9\%$	$\mathbf{74.5}\%$

Table 3. Pixelwise and class-average accuracies on the Stanford Background dataset (8 semantic classes).

Region-based energy [8]	$91.0 \pm 0.6\%$
Superpixel MRF [32]	90.6%
This Work	$\mathbf{92.2} \pm 0.5\%$

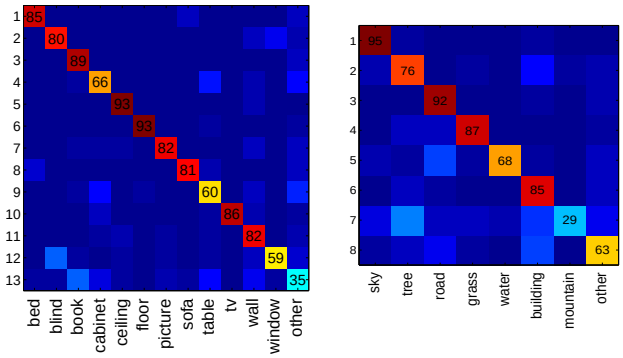
Table 4. Pixelwise accuracy on the Stanford Background dataset (3 geometric classes).

The 13-class confusion matrix of our final results is shown in Figure 5(a). We can see that the easy classes are: bookshelf 89%, ceiling 93%, and floor 93%. The hard classes are: cabinet 66%, table 60%, window 59%, and background 35%. The background class is an interesting case: as it covers all other objects in a scene, we find it difficult to model its appearance. We also see confusions between Blind and Window, and Cabinet and Table, partly because there are inconsistent labels in the groundtruth.

Final results on Stanford Background. Table 3 shows our results on the Stanford Background Dataset, comparing to existing works. We achieve pixelwise accuracy of $\mathbf{82.9}\%$, a large improvement from the previous state of the art 79.4%. We improve the class-average accuracy from 66% to $\mathbf{74.5}\%$. Examples are shown in Fig. 10.

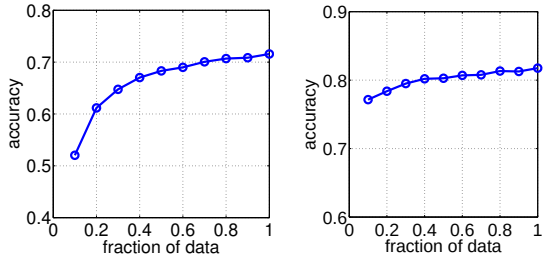
The 8-class confusion matrix of our final results is shown in Figure 5(b). Consistent with other studies, the hard classes are: water 68%, mountain 29%, and foreground 63%. It is interesting to compare these to the per-class accuracies reported in [23, 8]. We do much better on the hardest category *mountain*, achieving 29% vs 14% and 5%.

One naturally wonders why we achieve high accuracies; is it because of kernel descriptors or SVM classification on trees? We believe the answer is both. We conduct a comparative experiment, where we feed kernel descriptor features into the boosting+CRF approach of [8] using the authors' code. We increase the rounds of boosting to 2500 (from 500). The resulting accuracy is $79.9 \pm 1.1\%$, higher than



(a) NYU Depth (b) Stanford

Figure 5. The confusion matrices of our final results on the NYU (RGB-D, 13 classes) and Stanford (RGB only, 8 classes) datasets.



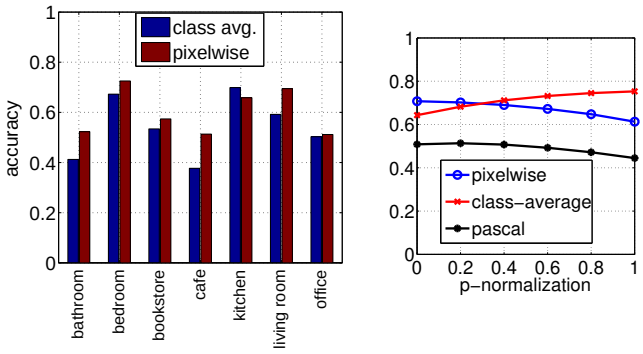
(a) NYU Depth (b) Stanford

Figure 6. How labeling accuracy varies with different amounts of training data (single layer). Accuracy increases fast and is not saturated, suggesting the need for all the data (and more).

using TextonBoost features [8] but still 3% lower than our complete pipeline. Using Texton features in linear SVM, on the other hand, yields lower scores. The kernel descriptor features go well hand-in-hand with linear SVM.

Amount of training data. One major advantage of our KDES+linear SVM approach is that we are capable of using a large amount of training data. In our approach, each superpixel is a training instance, and we routinely deal with $200K$ to $800K$ data points. To verify the need of data, we run classification experiments (on a single layer) using a varying amount of training data, from 10% to 100%. The results are shown in Figure 6. On the NYU dataset, we see a large drop in accuracy when subsampling (3% lower for using 50% data), and there is no saturation around full data. This suggests that although the NYU dataset contains over 2000 frames, there are a lot of variations in the indoor scenes, and the training data is far from enough. The effect also exists for the Stanford dataset in a weaker form. This finding is consistent with that from the recognition community, that large-scale data is a key to high accuracy.

Scene types. The NYU Depth dataset has 7 scene types, from bedroom, kitchen to office. In Figure 7(a), we show both class-average and pixelwise accuracies for each scene type. The distribution of semantic labels across scene types are very different (e.g. no Bed in Kitchen), making the class-average accuracies lower than the aggregate case. Nonetheless, the patterns are consistent, with bathroom (close view), cafe (small amount of training data) and office (large layout variations) being the most difficult scene types. Future work will need to address these scene types with either stronger models or more data.



(a) Scene Type (b) Balancing

Figure 7. (a) Accuracies for individual scene types in NYU Depth. (b) Imbalanced classes with vastly different pixel counts: pixelwise, class-average, and PASCAL labeling score when using different amounts of balancing (with $0 \leq p \leq 1$).

Balancing classes. As noted in [30], the (im)balance between semantic classes is a delicate issue for the NYU Depth dataset. The "large" classes, such as Wall, occupy over 20% of all pixels. The "small" classes, such as Television, are less than 1%. This not only creates a problem for learning, but also reveals a conceptual issue: should we "weigh" Wall vs Television equally, or by their occurrences? There is no definitive answer.

We use a value $0 \leq p \leq 1$ to normalize weights across classes (Eq. 4), from unbalanced (p=0) to completely balanced (p=1). Figure 7(b) shows how labeling performance, under different evaluation criteria, changes with p. Pixelwise accuracy prefers the unbalanced case, class accuracy prefers balanced, and the PASCAL score $TP/(TP+FP+FN)$ seems more robust to balancing. This suggests the PASCAL score being a better criterion or the need to show such curves as accuracies vary with balancing.

7. Discussion

In this paper we presented our study on scene labeling both with RGB-D and with images. We improved RGB-D labeling accuracy on the NYU Depth dataset from 56.6% to **76.1**%, and that on the Stanford Background dataset from 79.4% to **82.9**%. We achieved high labeling accuracy by a combination of color (and depth) features using *kernel descriptors*, and by combining MRF with segmentation tree. One key finding of our work, not a surprise, is the need to have and to use a large amount of data. We showed that labeling accuracy increased with the size of training data. Linear SVM, efficiently utilizing possibly high dimensional

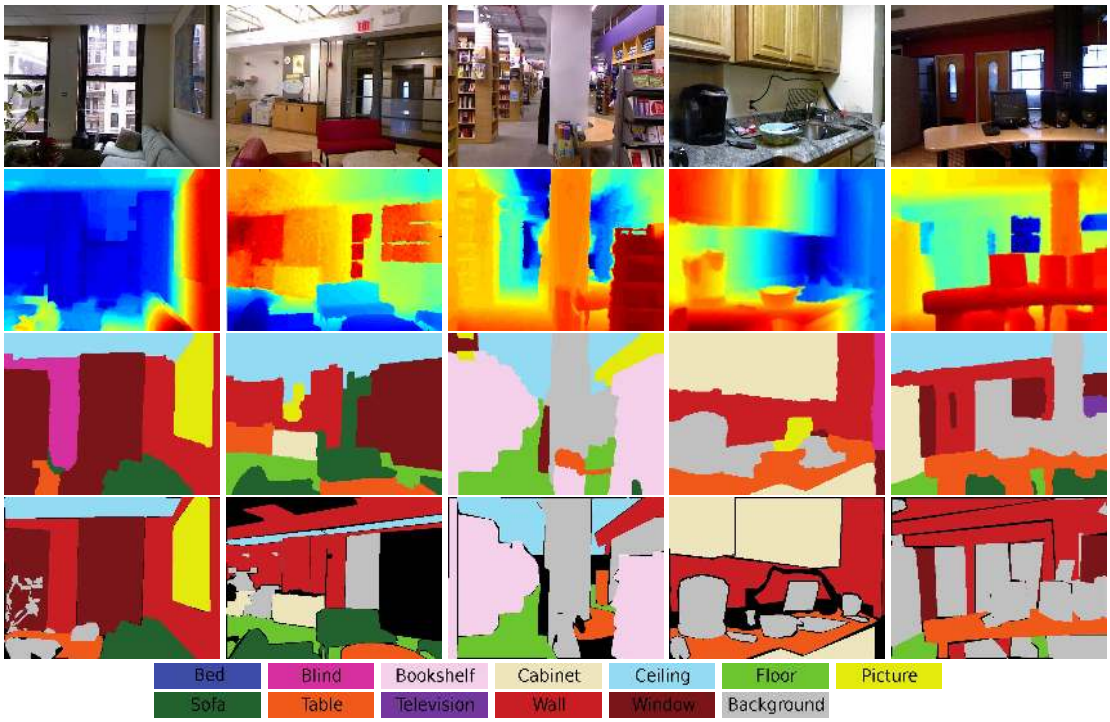


Figure 8. Examples of 13-class semantic labeling on the NYU Depth dataset. The four rows are (1) color frame, (2) depth frame, (3) result and (4) groundtruth. We do well in many challenging cases with complex scene layouts. Meanwhile there is clearly room for improvement.

(a) (b) (c) (d) (e)

Figure 9. An illustration of using segmentation trees. (a) The fourth example above. (b-d) Superpixel classifications at three tree layers (2^{nd},5^{th},9^{th}), colored as above and overlaid with segmentation. We show the SVM outputs (linear scores, *not* a histogram, of 13 classes) at a *wall* point A, which is misclassified in (b) due to over-segmentation and in (d) due to under-segmentation. (e) Treepath classification, where we combine the SVM outputs from all the layers to classify the bottom layer, capturing both fine-scale structures (missing in (d)) and coarse-scale structures (missing in (b)). An MRF on (e) further enforces the smoothness of labels to produce the final results in Fig. 8.

data, may continue to work well despite its simplicity.

Our study confirmed that indoor scenes are more diverse and difficult than outdoor scenes, and the labeling problem is far from solved even with RGB-D. We systematically studied RGB-D features and showed that their combinations largely improve scene labeling accuracy. Meanwhile, our technique is generic and performs well on outdoor scenes where dense depth is not readily available. We plan to investigate RGB-D specific techniques in future work.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. PAMI*, 2010. 2, 3

[2] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *NIPS*, 2010. 1, 2, 3, 4

[3] L. Bo, X. Ren, and D. Fox. Depth Kernel Descriptors for Object Recognition. In *IROS*, 2011. 1, 2, 3, 5

[4] L. Bo and C. Sminchisescu. Efficient match kernels between sets of features for visual recognition. In *NIPS*, 2009. 2, 4

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. PAMI*, 23(11):1222–1239, 2001. 4

[6] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *CVPR*, 2010. 2

[7] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008. 4

[8] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009. 1, 2, 3, 5, 6, 8

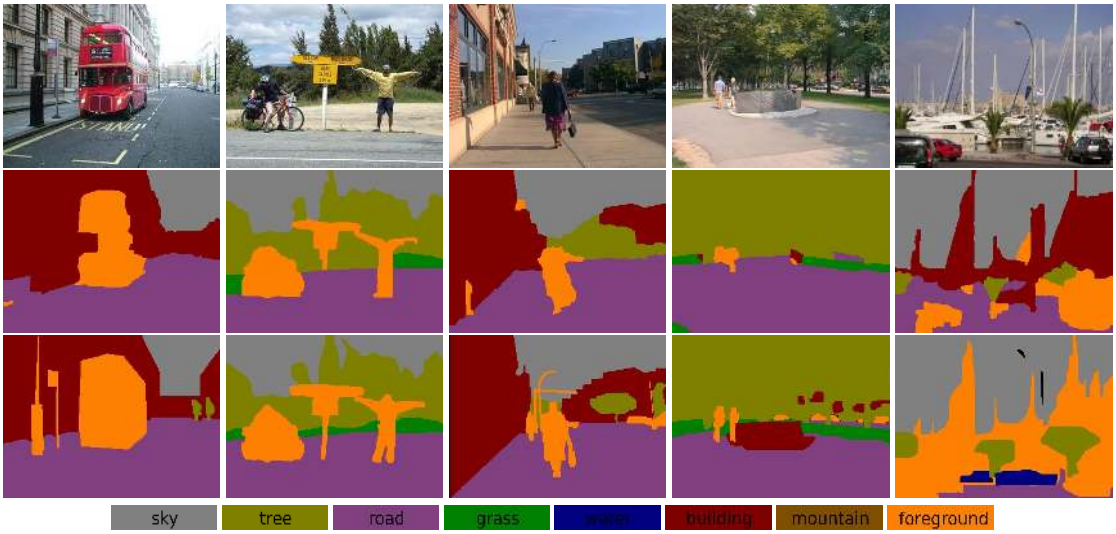


Figure 10. Examples of 8-class labeling results on the Stanford Background dataset. We do well in many challenging cases with large scale and appearance variations. The last example is interesting as discussed in [8]: we still confuse boat masts with building, but our segmentation is more accurate. (More examples can be found in the supplemental material.)

[9] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller. Multi-class segmentation with relative location prior. *Int'l. J. Comp. Vision*, 80(3):300–316, 2008. 2

[10] X. He, R. Zemel, and M. Carreira-Perpinan. Multiscale conditional random fields for image labelling. In *CVPR*, volume 2, pages 695–702, 2004. 2

[11] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *CVPR*, 2009. 1, 2

[12] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *International Symposium on Experimental Robotics (ISER)*, 2010. 1, 2

[13] D. Hoiem, A. Efros, and M. Hebert. Geometric context from a single image. In *ICCV*, 2005. 1, 2

[14] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *ISMAR*, pages 559–568, 2011. 1, 2

[15] H. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *NIPS*, 2011. 1, 2

[16] M. Kumar and D. Koller. Efficiently selecting regions for scene understanding. In *CVPR*, 2010. 2, 5

[17] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *ICCV*, pages 1150–1157, 2003. 2

[18] L. Ladicky, C. Russell, P. Kohli, and P. Torr. Associative hierarchical crfs for object class image segmentation. In *CVPR*, pages 739–746, 2009. 2

[19] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *ICRA*, pages 1817–1824, 2011. 1, 2

[20] D. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *CVPR*, pages 2136–2143, 2009. 1, 2

[21] J. Lim, P. Arbeláez, C. Gu, and J. Malik. Context by region ancestry. In *ICCV*, pages 1978–1985, 2009. 1, 2, 4

[22] Microsoft Kinect. http://www.xbox.com/en-us/kinect. 1, 2

[23] D. Munoz, J. Bagnell, and M. Hebert. Stacked hierarchical labeling. In *ECCV*, pages 57–70, 2010. 1, 2, 4, 5

[24] PrimeSense. http://www.primesense.com/. 2

[25] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *CVPR*, 2009. 1, 2

[26] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, volume 1, pages 10–17, 2003. 1

[27] B. Russell, W. Freeman, A. Efros, J. Sivic, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, volume 2, pages 1605–1614, 2006. 2

[28] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, volume 2, page 3, 2011. 1, 2

[29] 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, 2

[30] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *IEEE Workshop on 3D Representation and Recognition (3dRR)*, 2011. 1, 2, 3, 5, 6

[31] R. Socher, C. Lin, A. Ng, and C. Manning. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 2011. 2, 5

[32] J. Tighe and S. Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. In *ECCV*, pages 352–365, 2010. 2, 5

[33] L. Yang, P. Meer, and D. Foran. Multiple class segmentation using a unified framework over mean-shift patches. In *CVPR*, 2007. 1, 2