Label Propagation from ImageNet to 3D Point Clouds

Yan Wang, Rongrong Ji, and Shih-Fu Chang
Department of Electrical Engineering, Columbia University
{yanwang,rrji,sfchang}@ee.columbia.edu

Abstract

Recent years have witnessed a growing interest in understanding the semantics of point clouds in a wide variety of applications. However, point cloud labeling remains an open problem, due to the difficulty in acquiring sufficient 3D point labels towards training effective classifiers. In this paper, we overcome this challenge by utilizing the existing massive 2D semantic labeled datasets from decade-long community efforts, such as ImageNet and LabelMe, and a novel "cross-domain" label propagation approach. Our proposed method consists of two major novel components, Exemplar SVM based label propagation, which effectively addresses the cross-domain issue, and a graphical model based contextual refinement incorporating 3D constraints. Most importantly, the entire process does not require any training data from the target scenes, also with good scalability towards large scale applications. We evaluate our approach on the well-known Cornell Point Cloud Dataset, achieving much greater efficiency and comparable accuracy even without any 3D training data. Our approach shows further major gains in accuracy when the training data from the target scenes is used, outperforming state-of-the-art approaches with far better efficiency.

1. Introduction

Coming with the popularity of Kinect sensors and emerging 3D reconstruction techniques [1][2][3][4], we are facing an increasing amount of 3D point clouds. Such massive point cloud data has shown great potential for solving several fundamental problems in computer vision and robotics, for example, route planning and face analysis.

While the existing work mainly focuses on building better point clouds [1][2][3][4], point-wise semantic labeling remains an open problem. Its solution, however, can bring a breakthrough in a wide variety of computer vision and robotics research, with great potential in human-computer interface, 3D object indexing and retrieval, as well as 3D scene understanding and object manipulation in robotics. Exciting applications such as self-driving vehicles

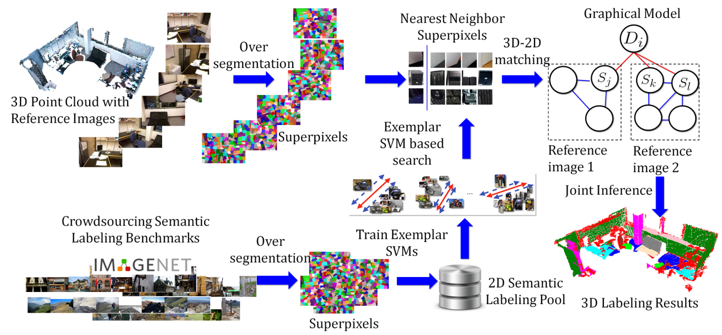

Figure 1. The framework of search based label propagation from ImageNet to point clouds.

can also be expected, in which inferring the 3D object labels can bring more comprehensive understanding for navigation and decision making. Another instance comes from semantic-aware augmented reality, bringing realistic interactions between virtual and physical objects.

Challenges. While important, labeling 3D point cloud is not an easy task at all. Following 2D semantic labeling, the state-of-the-art solutions [5][6][7][8][9][10] train point-wise label classifiers based on visual and 3D geometric features, and optionally refined with spatial contexts. However, their bottleneck lies in the difficulty to design effective 3D features, while a 3D feature variant to rotation, translation, scaling and illumination like 2D SIFT does is still missing. Meanwhile, it is also hard to extend 2D features to 3D given critical operations for 2D feature extraction such as convolution is no longer valid for point clouds.

Another fundamental challenge comes from the lack of sufficient point cloud labels for training, which, in turn has been shown as a key factor towards successful 2D image labeling [11][12][13]. This factor, as highly aware by the computer vision community, has led to a decade-long effort in building large-scale labeling datasets, showing large benefits for 2D image segmentation, labeling, classification and object detection [11][12][13][14]. However, limited efforts are conducted for point cloud labeling benchmarks. To the best of the authors' knowledge, the existing labeled point cloud or RGB-D datasets [15][16] are incomparable to the 2D ones, in terms of either scale or coverage. This causes even state-of-the-art point cloud labeling algorithms only touch data from well controlled environments, with similar training and testing conditions [5][6][7].

Inspirations. Manual point cloud labeling is certainly one solution to the lack of sufficient training data. However, it requires intensive human labor, especially considering the difficulty in labeling 3D points. Even given sufficient point cloud labels, the effective 3D feature design still remains open. But turning to the 2D side, with such massive pixel-wise image labels at hand, is it possible to "propagate" or "transfer" such labels from images to point clouds? This approach, if possible, solves the training data insufficiency, while not requiring the intensive point cloud labeling, and also gets around the open problem in designing effective 3D feature and geometric representation.

To achieve this goal, we propose to exploit the *reference images* required for point cloud constructions as a "bridge". Turning to a label propagation perspective, if we can link query regions to regions in the dataset with the right label as a graph formulation, this point cloud or reference image labeling problem can be easily solved by propagating labels from labeled nodes to unlabeled nodes along the edges. This idea is also inspired from the recent endeavors in search based mask transfer learning, which has shown great potential to deal with the "cross-domain" issue in both object detection and image segmentation [14][17][18]. While requiring good data coverage, this is more and more practical with the increasingly "dense" sampling of our world as images, for instance there are over 10M images in ImageNet [12], over 100K segments in LabelMe [11], and over 500K segmented images in ImageNet-Segment[14]. Furthermore, such search based propagation can be performed in parallel by nature, with high scalability towards big data.

Other than memory based approaches, it is certainly possible to directly train a classifier [19] from other sources [11][12]. However, this solution lacks the generality among different datasets, thus is not practical for our situation.

Approach. We design two key operations to propagate external image labels to point clouds, namely "Search based Superpixel Labeling" and "3D Contextual Refinement", as outlined in Figure 1.

Search based Superpixel Labeling: Given the massive pixel-wise image labels from external sources such as ImageNet [12] or LabelMe [11], we first use Mean Shift to over-segment individual images into "superpixels", and then propagate their labels onto the visually similar superpixels in the reference images of point clouds. We accomplish this by using *Exemplar SVMs* rather than the naive nearest neighbor search, because the latter is not robust enough against the "data bias" issue, e.g., the photometric condition changes between training and testing sets. More specifically, we first train linear Support Vector Machines (SVMs) for individual "exemplar" superpixels in the external image collection, use them to retrieve the robust k Nearest Neighbors (kNN) for each superpixel from the reference images, and then collect their labels for future fusion. Note this is comparably efficient to naive kNN search by exploiting the high independence and efficiency of the linear SVMs.

3D Contextual Refinement: We then aggregate superpixel label candidates to jointly infer the point cloud labels. Similar to the existing works in image labeling, we exploit the intra-image spatial consistency to boost the labeling accuracy. In addition, and more importantly, 3D contexts are further modeled to capture the inter-image superpixel consistency. Both contexts are integrated into a graphical model to seek for a joint optimal among the superpixel outputs with Loopy Belief Propagation.

The rest of this paper is organized as follows: Section 2 introduces our search based superpixel labeling. Section 3 introduces our 3D contextual refinement. We detail the experimental comparisons in Section 4 and discuss related work in Section 5, with conclusions in Section 6.

2. Search based Superpixel Labeling

Notations. We denote a point cloud as a set of 3D points $\mathcal{P} = \{\mathbf{p}_i\}$, each of which is described with its 3D coordinates and RGB colors $\{x_i, y_i, z_i, R_i, G_i, B_i\}$. $\mathcal{P}$ is built from R reference images $\mathcal{I}_R = \{I_r\}_{r=1}^{R}$ using methods such as Structure-from-Motion [1] or Simultaneous Localization And Mapping (SLAM) [3]. We also have an external superpixel labeling pool consisting of superpixels with ground truth labels $\mathcal{S} = \{S_i, l_i\}_{i=1}^{N}$. Our goal is to assign each $\mathbf{p}_i$ a semantic label l from an exclusive label set $\mathcal{L}$, as propagated from the labeling pool $\mathcal{S}$[1].

Search based Label Propagation. In our approach, 2D image operations are performed in the unit of "superpixels", produced by over-segmentation [20]. Note that we do not leverage randomly sampled rectangles used in recent works in search based segmentation [14][21] or object detection [18], to ensure label consistency among pixels in each region, as widely assumed for superpixels [19][2].

For every superpixel S_q to be labeled in the reference images, we aim to find the most "similar" superpixels in $\mathcal{S}$, whose label will then be propagated and fused to S_q. To achieve this goal, a straightforward solution is to directly find the k nearest neighbors in $\mathcal{S}$, which results in the following objective function:

$$k\text{NN}(S_q) = \arg\min_{S_i \in \mathcal{S}}^{k} D(S_i, S_q), \tag{1}$$

in which $\arg\min_{S_i}^{k}$ denotes the top k superpixels with the

[1] In practice, our external superpixel labeling pool comes from ImageNet [12], as detailed in Section 4, while other image/region labeling datasets can also be integrated.

[2] Techniques like objectness detectors [22] can be further integrated to boost the accuracy and efficiency.

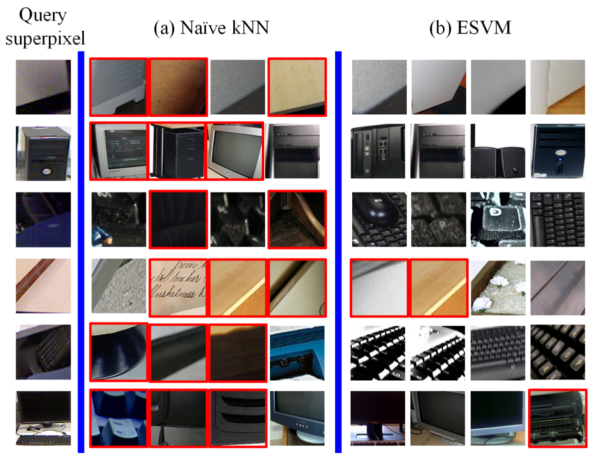

Figure 2. Examples of search based label propagation from ImageNet [12] to Cornell Point Cloud Dataset [6]: (a) outputs from Naive kNN, (b) outputs from ESVMs. We can see Naive kNN has lots of false positives (with red borders), e.g., it outputs *printer* and *table* for the input of *wall*. But ESVM performs much more robust. For each result, we show not only the superpixel but also its surroundings for clarity.

least distance D from S_q, and $D(\cdot,\cdot)$ is the metric used to measure the distance among superpixels, e.g., Euclidean distance in the feature space.

Propagation with Exemplar SVM. As pointed out in [18], nearest neighbor search with Euclidean distance cannot capture the intrinsic visual similarity between superpixels, while on the other hand training a label classifier is too sensitive to the training data, with large generalization error against propagation.

To address this issue, Exemplar SVM (ESVM) [18] is introduced to build a robust metric. For every superpixel extracted from the labeling pool $S_i \in \mathcal{S}$, we train a linear SVM to identify its visually similar superpixels. The Exemplar SVM for S_i is trained to optimize the margin of the classification boundaries:

$$\begin{aligned} &\arg\min_{\mathbf{w}_i, b_i} \frac{1}{2}\|\mathbf{w}_i\|_2^2 + C^+ \sum_{\{j|y_j>0\}} \xi_j + C^- \sum_{\{j|y_j<0\}} \xi_j, \\ &\quad \text{s.t. } y_j(\mathbf{w}_i^T \mathbf{x}_j + b_i) \geq 1 - \xi_j, \forall j, \end{aligned} \tag{2}$$

where $\mathbf{x}_j$ is the visual features extracted from S_j.

To further guarantee the matching robustness, every superpixel S_i is translated and rotated to expand to more positive examples for training. And the negative examples are subsampled from other superpixels. Larger penalties C^+ are given for false negatives to balance the boundary. However, even with this C setting, if a superpixel other than but very similar to the exemplar appears in the negative set, it will significantly degenerate the performance with an ill-trained SVM, while this is not studied in object detection [18]. To address this issue, we add an extra constraint that

---

Algorithm 1: Building Superpixel Labeling Pool.

Input: A set of superpixels with ground truth labels $\mathcal{S} = \{S_i, l_i\}_{i=1}^N$
for Superpixel $S_i \in \mathcal{S}$ **do**
  Train an Exemplar SVM for each superpixel with Equation 2 and hard negative mining
end
Learn a reranking function by solving Equation 5
Output: A set of Exemplar SVMs with reranking weights and associated labels $\{\mathbf{w}_i, b_i, w_i^{(r)}, b_i^{(r)}, l_i\}$

---

only superpixels with different labels from S_i can be chosen as negative examples.

Hard Negative Mining. Different from regular Exemplar SVMs only able to find nearly identical instances, we set a small C in the training process for better generality, allowing examples not exactly the same as the exemplar also have positive scores. However, this may increase the number of false positives with different labels. To address this problem, given the decision boundary is only determined by the "hard" examples (the support vectors), we introduce the hard negative mining to constrain the decision boundary:

1. Apply the ESVM trained from S_i on the training data, collecting the prediction scores $\{s_j\}$
2. Add the false positives $\{S_j | s_j > 0, l(S_j) \neq l(S_i)\}$ into the negative examples and launch another round of SVM training
3. Repeat the first two steps until no new hard examples are found or a preset iteration number is reached.

By the combination of small C and supervised hard negative mining with labels, we achieve a balance of generality and sensitivity. Figure 2 shows a comparison between ESVM and naive kNN search. We can see that although naive kNN usually outputs visually similar instances, it is not robust against false positives, while ESVM does better in label robustness, with more sensitivity on the difference of labels.

To label the superpixel S_q in the reference images, we find the superpixels with the k strongest responses from their ESVMs as its k nearest neighbors in $\mathcal{S}$, i.e.,

$$k\text{NN}(S_q) = \arg\max_{S_i \in \mathcal{S}}^{k} F_i(\mathbf{w}_i^T \mathbf{x}(S_q) + b_i). \tag{3}$$

Here F_i is the ranking function as introduced below.

Learning Reranking Functions. While we use the relative ranking of SVMs' prediction scores for kNN search, this relative order is still not constrained in the training process. We address this issue by learning a linear function for each SVM for reranking, which minimizes the ranking error *among* different SVMs, i.e.,

$$F_i(x) = w_i^{(r)} \cdot x + b_i^{(r)}. \tag{4}$$

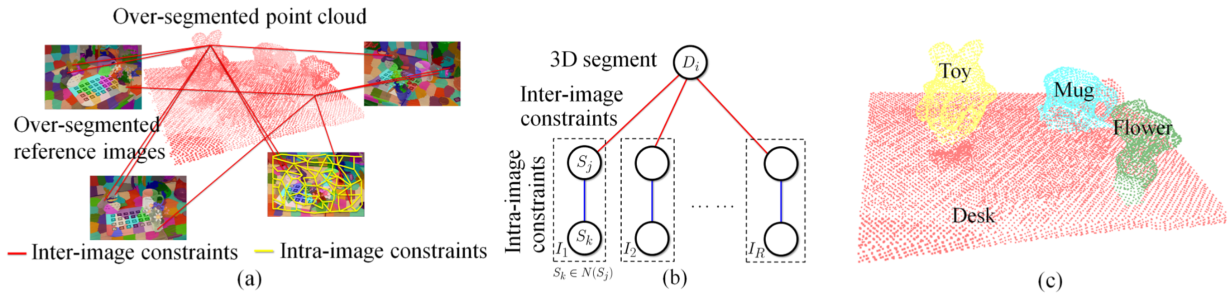

Figure 3. Construction of the graphical model. (a) shows how to construct nodes and edges from an example point cloud. (b) shows the abstract representation of the graphical model, where $N(S_j)$ denotes the spatial adjacent superpixels of S_j in the reference images. And (c) shows the expected output after optimizing on the model. For clarity, only part of the connections are plotted in (a).

This linear mapping does not modify the relative order in each ESVM, but rescales and pushes the ESVMs jointly to make their prediction scores comparable. This differs from the calibration step in [18] that transforms the decision boundary of each SVM *independently* for object detection.

More specifically, given a held-out superpixels validation set with ground truth labels $\mathcal{S}_H = \{S_q\}_{q=1}^{H}$, we first apply the ESVMs to get prediction scores $\{s_q\}_{q=1}^{H}$, and then collect SVMs with positive scores for reranking, only some of which have the same label with S_q. Here we aim to learn the function F_i for each ESVM in $\mathcal{S}$, making superpixels with the same label as S_q have larger score than others, formulated as a structured learning-to-rank problem [23].

$$\begin{aligned} &\min \sum_i \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum \xi_{i,j,k} \\ &\text{s.t. for every query } q_i, \\ &\quad \mathbf{w}^T\mathbf{\Phi}(q_i,s_j) > \mathbf{w}^T\mathbf{\Phi}(q_i,s_k) + 1 - \xi_{i,j,k} \\ &\quad \forall l(S_j) = l(S_{q_i}), l(S_k) \neq l(S_{q_i}) \\ &\quad \xi_{i,j,k} \geq 0. \end{aligned} \tag{5}$$

Here $\mathbf{w} = \{(w_i, b_i)\}_{i=1}^{n}$. $\mathbf{\Phi}(q_i, s_j)$'s $(2j-1)$th and $2j$th dimensions are s_j and 1 respectively, to encode the weights and scores into a single vector. Although this problem is not convex or differentiable, its upper bound can be optimized efficiently with a cutting plane algorithm [24]. Algorithm 1 shows the training procedures of our label propagation.

3. 3D Contextual Refinement

Given the k nearest neighbors of each superpixel in the reference images, the next step is to label the point cloud $\{l(\mathbf{p}_i) \mid \mathbf{p}_i \in \mathcal{P}\}$ by backprojecting and fusing their labels. Considering a point $\mathbf{p}_i$ in $\mathcal{P}$ may appear in different reference images with different viewing angles, our prediction exploits both the intra-image 2D context and the inter-image 3D context, as shown in Figure 3. Note different from traditional contextual refinement approaches [5][6][25], our approach does not require any labeled 3D training data.

Graphical Model. First the point cloud is over-segmented based on the smoothness and continuity [5], producing a 3D segment set $\{D_i\}$ as shown in Figure 3 (a). Second, with the transform matrices $\{M_i\}$ from local 2D coordinates in reference images $\{I_i\}_{i=1}^{R}$ to the global 3D coordinates, segments in $\{D_i\}$ are matched to the reference images, each resulting in a 2D region $S_{M_j}(D_i)$. If this projected region shares enough portion with some superpixel $\{S_i\}$ from this reference image, we connect an edge between $\{S_i\}$ and $S_{M_j}(D_i)$, as shown as red links in Figure 3 (a). Spatially adjacent superpixels within one reference image are also connected, as shown as yellow links in Figure 3 (a). Then, an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is built with 3D segments and 2D superpixels as nodes $\mathcal{V} = \{D_i\} \cup \{S_q\}_{q=1}^{M}$ and the connections mentioned above as edges $\mathcal{E}$. Figure 3 (b) shows the corresponding graphical model.

For every node v, we adopt its label $l(v)$ as the variables on the graphical model, and define the potential function to enforce both intra-image and inter-image consistency, as detailed later. In the end, the semantic labels $\mathbf{L}$ for 3D segments $\{D_i\}$ is inferred by minimizing the potential function of the graphical model:

$$\arg\min_{\mathbf{L}\in\mathcal{L}^n} \sum_{v\in\mathcal{V}} \log\psi_d\big(l(v)\big) + \lambda \sum_{(v_1,v_2)\in\mathcal{E}} \log\psi_s\big(l(v_1), l(v_2)\big), \tag{6}$$

in which $\mathcal{L}$ is the label set, n is the number of nodes in the graphical model, and λ is a constant that weights different potential components. The potential function consists of the data term ψ_d and the smoothing term ψ_s.

The assigned labels of 2D superpixels are encouraged to be the same as their results from label propagation. Therefore the data term is defined as,

$$\psi_d\big(l(S_i)\big) = \exp\Big(-p_{S_i}\big(l(S_i)\big)\Big), \tag{7}$$

where $p_{S_i}(\cdot)$ is the label distribution among kNNs of S_i retrieved using Equation 3.

Intra-Image Consistency. To encode the intra-image consistency in the reference images, neighbor superpixels

Algorithm 2: Search based Label Propagation.

Input: Point cloud $\mathcal{P}$ with reference images $\mathcal{I}_R$, and superpixel propagation pool $\mathcal{S}$
Do over-segmentation on both $\mathcal{P}$ and $\mathcal{I}_R$
for Superpixel $S_q \in \mathcal{I}_R$ **do**
  Use Equation 3 to retrieve kNN from $\mathcal{S}$
end
Use the over-segmentation structure of $\mathcal{P}$ and $\mathcal{I}_R$, and the kNN of S_q to construct the graphical model as introduced in Section 3
Solve the graphical model by minimizing Equation 6 with Loopy Belief Propagation
Output: 3D-segment-wise semantic labels of $\mathcal{P}$

are encouraged to have related labels, defined by the intra-image smoothing term $\psi_{s,2\text{D}}$.

$$\psi_{s,2\text{D}}\big(l(S_i), l(S_j)\big) = p\big(l(S_i), l(S_j)\big), \tag{8}$$

in which $p(\cdot,\cdot)$ is the co-occurrence probability learned from the superpixel labeling pool $\mathcal{S}$.

Inter-Image Consistency. To make the 3D labeling results consistent among reference images, we further define the inter-image smoothing term as,

$$\psi_{s,3\text{D}}\big(l(S_i), l(D_j)\big) = \begin{cases} 1 & l(S_i) = l(D_j) \\ c & l(S_i) \neq l(D_j), \end{cases} \tag{9}$$

in which $c > 1$ is a constant. Overall, we have a more specific potential function design.

$$\begin{aligned} \log \psi(\mathbf{L}) = & \sum_{S_q \in \mathcal{I}_R} \log \psi_d\big(l(S_q)\big) \\ & + \lambda_1 \sum_{(S_i,S_j)\in\mathcal{E}, S_i,S_j \in \mathcal{I}_R} \log \psi_{s,2\text{D}}\big(l(S_i), l(S_j)\big) \\ & + \lambda_2 \sum_{(S_i,D_j)\in\mathcal{E}} \log \psi_{s,3\text{D}}\big(l(S_i), l(D_j)\big). \end{aligned} \tag{10}$$

Similar with above, λ_1 and λ_2 denote the weights for different potential parts. We use Loopy Belief Propagation (LBP) to find a local minima of ψ. Algorithm 2 outlines the overall procedure based on the potential design above.

Integrating Other Contexts. If 3D point cloud training data with ground truth label is also available, we can further integrate stronger context into our graphical model. Such contexts may include 3D relative positions (e.g., a *table* may appear under a *book* but should not under *floor*), and normal vector (e.g. a *wall* must be vertical and *floor* must be horizontal), etc., as investigated in [5]. However, this requirement puts stronger dependence on training data thus decreases the generality.

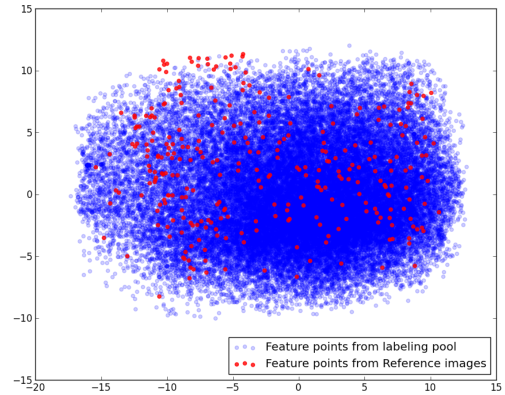

Figure 4. Superpixel distribution of the labeling pool (blue) and reference images (red) in the (dimension reduced) feature space.

4. Experimental Validations

Data Collection. To build the superpixel labeling pool, we collect superpixels from ImageNet [12], which provides object detection ground truth as bounding boxes. We first over-segment the image using Mean Shift [20], and then select the superpixels sharing enough area with the bounding boxes of some object of interest (e.g. *wall*, *floor*). These superpixels are then added in to the superpixel labeling pool with their corresponding labels.

To evaluate our algorithm, the Cornell's indoor dataset [16] is adopted, which contains 24 office scenes and 28 home scenes, constructed from Kinect sensor and RGBD-SLAM (http://openslam.org/rgbdslam). Each scene consists of 3D points with 3D coordinates, RGB values, semantic labels, and reference images used for the RGBDSLAM construction. Following [5], we take labels present in ≥ 5 scenes (point clouds) for evaluation. We merge too specific labels while they do not occur in the labeling pool (e.g., *chairBackRest*, *chairBase*, and *chairBack* to *chair*). As a result, in the office scene we use labels {*wall, floor, table, chair, monitor, printer, keyboard, cpu, book, paper*}, and in the home scene we use labels {*wall, floor, table, chair, sofa, bed, quilt, pillow, shelfRack, laptop, book*}[3].

Rationality Checking. We illustrate the rationality of our approach by visualizing the superpixel labeling pool (blue dots) and the superpixels from the reference images (red dots) in a 2D space mapped from the feature space with Principal Component Analysis, as Figure 4 shows. We can observe a good coverage in the feature space, although the collection conditions of Cornell Point Cloud dataset are certainly different from ImageNet. On the other hand, the coverage is still not perfect, calling for more advanced techniques other than naive kNN search such as ESVM.

Evaluation Protocol and Baselines. We use average classification accuracy, that is, the average percentage of the

[3] Different from [5], these labels are only used for testing, *not* involved in our training process at all.

Table 1. Visual features for label propagation

Superpixel Feature	**Dimension**
HoG	9×12
Average HSL values	3×12
Difference of Gaussian	1×12
Laplacian	1×12
Edge detector (with different angles)	5×12
4×4 Walsh Hadamard kernels	16

correctly classified points among all the point clouds, as our protocol to evaluate both the 2D superpixel labeling and the 3D point labeling. We compare our approach to the following baselines: (1) Naive kNN search based propagation; (2) ESVM based propagation; (3) ESVM based propagation with contextual refinement; (4) The state-of-the-art work by Anand et al. [5], which uses 3D training data known to be similar with the test data.
For Baseline (1) and (2), with the lack of inter-image optimization, we use majority voting to get the 3D labels. And for Baseline (4), given our approach does not use any local 3D shape or geometry, for the fairness of comparison, we adopt the accuracy only using the visual appearance reported in [6]. But note our approach is complementary to 3D geometry based methods, it is easy to add more features and contexts as introduced in Section 3. Baseline (1) to (3) use our superpixel label propagation pool for ESVM training, and all the Cornell Point Cloud Dataset for testing.

Implementation Details. In terms of visual features, we use Histogram of Gradient (HoG) feature [26] (with 4×3 grids), average HSL values, texture features, and 4×4 Walsh Hadamard kernels [27] to represent each superpixel. A complete list of features is shown in Table 1.

We adopt LibSVM[4] for Exemplar SVM training and testing, with $C^- = 0.01$ and $C^+ = 0.05$. And the training examples are generated from the original superpixel with five levels of translation and rotation. For every superpixel, we collect 10,000 negative examples and do five rounds of hard negative mining. For reranking function learning, we use SVM[rank5], with $C = 200$ and 10-fold cross-validation. In terms of superpixel labeling pool, we collect about 28K superpixels for each type of scenes (office or home).

Propagation Accuracy. Figure 6 shows the average accuracy of 3D labeling in both office and home scenes, with comparison among different baselines. We can see ESVM, even without contextual refinement, performs well and outperforms naive kNN with a large gap. In addition there is a performance gain after incorporating contexts, making our method (Baseline (3)) comparable with state-of-the-art method [6] in classification accuracy, without requiring any specific knowledge about the target scene. Note it is also easy to integrate geometric features in our approach. In terms of efficiency, our methods only requires less than 20

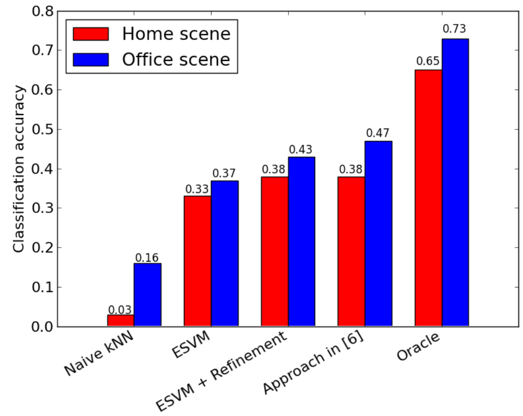

Figure 6. Point cloud labeling accuracy of different baselines in both office and home scenes.

Table 2. Efficiency of our approach (ESVM + Context)

Training	**Training** (parallel)	**Testing** (parallel)
43.2 s/superpixel	3.6 s/superpixel	18.2 s/point cloud

seconds for one point cloud, while the approach in [6] requires 18 minutes to finish in average. In the training stage, training an ESVM in a PC with a Core i7-970 CPU costs three to four seconds. More details about efficiency are shown in Table 2. Figure 5 shows several examples of results from different baselines, with ground truth labels.

Oracle rate. We also test our oracle rate, i.e., ESVM with contextual refinement trained with labeled reference images in Cornell Point Cloud Dataset. The accuracy of the oracle rate is also shown in Figure 6. The performance is evaluated with a four-fold cross-validation setting with Cornell Point Cloud Dataset for training and testing. We can see with a fair experiment setting, i.e., using the same training data and type of features (visual feature), our approach can outperform state-of-the-art method with only visual appearance features. And Figure 7 shows the confusion matrices of office scenes and home scenes. Because some classes are affected by extreme shooting conditions such as *paper* is often over-exposed, we cannot extract meaningful features from these classes, thus cannot distinguish them from other classes such as *wall* well. This indicates the limit of pure visual approaches for 3D point cloud labeling. But we are not proposing to substitute geometric features and contexts with our approach. This observation, on the contrary, denotes that our approach is able to provide complementary information with shape based methods, considering easy 3D features like normal vector can distinguish *paper* from *wall*. We can expect even higher performance by integrating geometric features and contexts as mentioned in Section 3.

5. Related Work

Semantic Labeling with Contextual Optimization. Semantic labeling of 2D images is a long-standing problem in computer vision. State-of-the-art approaches usu-

[4]http://csie.ntu.edu.tw/ cjlin/libsvm/
[5]http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

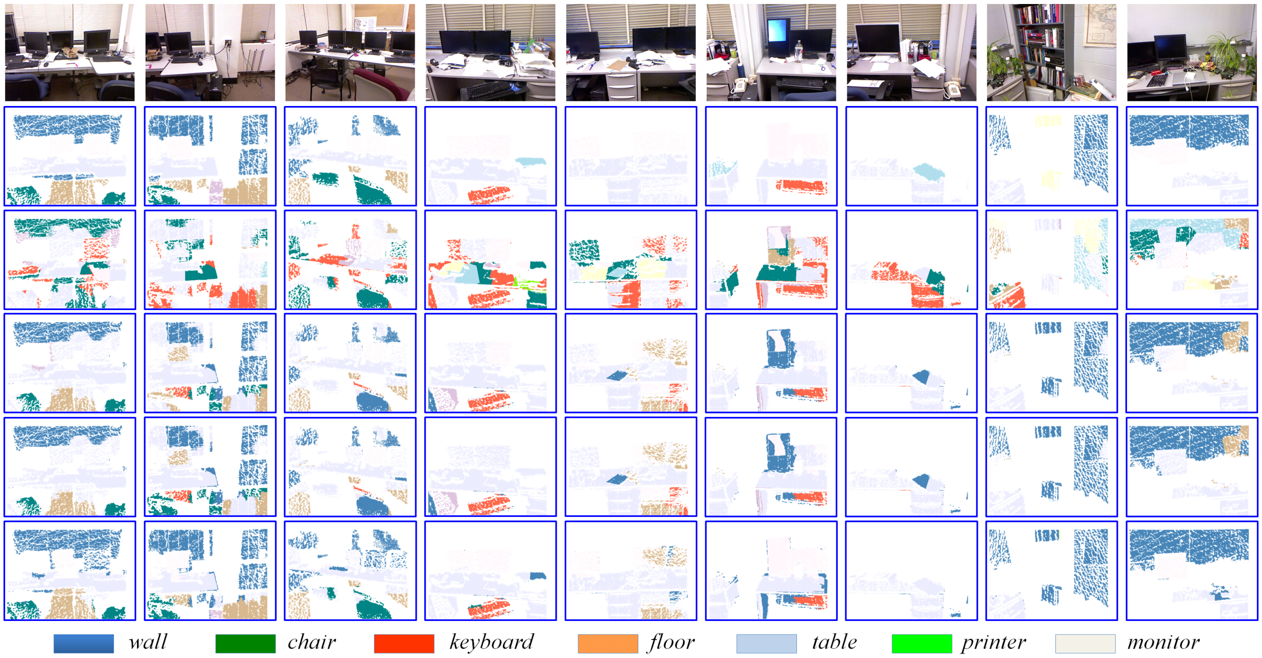

Figure 5. Example results of point cloud labeling on Cornell dataset [16]. To demonstrate labeling results with more details, reference images from multiple views are provided. The rows are, reference images, ground truth, labeling result from Naive kNN, ESVM, ESVM with refinement, and our oracle performance.

	wall	table	floor	monitor	chair	printer	cpu	book	keyboard	paper
wall	0.95	0.03	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00
table	0.13	0.83	0.02	0.01	0.00	0.00	0.00	0.00	0.00	0.00
floor	0.18	0.04	0.77	0.00	0.01	0.00	0.00	0.00	0.00	0.00
monitor	0.29	0.09	0.09	0.53	0.00	0.00	0.01	0.00	0.00	0.00
chair	0.11	0.09	0.12	0.01	0.66	0.00	0.00	0.00	0.00	0.00
printer	0.47	0.39	0.01	0.00	0.00	0.14	0.00	0.00	0.00	0.00
cpu	0.30	0.17	0.03	0.01	0.00	0.02	0.46	0.00	0.00	0.00
book	0.28	0.46	0.03	0.00	0.03	0.04	0.00	0.17	0.00	0.00
keyboard	0.10	0.14	0.26	0.00	0.05	0.00	0.00	0.00	0.44	0.00
paper	0.25	0.61	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13

	wall	table	floor	chair	book	bed	sofa	quilt	pillow	laptop	shelfRack
wall	0.98	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
table	0.22	0.69	0.06	0.00	0.00	0.01	0.01	0.01	0.00	0.00	0.00
floor	0.15	0.03	0.80	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.00
chair	0.41	0.15	0.17	0.24	0.00	0.02	0.00	0.02	0.00	0.00	0.00
book	0.52	0.42	0.00	0.00	0.00	0.00	0.00	0.07	0.00	0.00	0.00
bed	0.14	0.03	0.04	0.00	0.00	0.78	0.00	0.00	0.00	0.00	0.00
sofa	0.22	0.03	0.07	0.00	0.00	0.02	0.65	0.00	0.00	0.00	0.00
quilt	0.35	0.07	0.18	0.00	0.00	0.01	0.03	0.36	0.00	0.00	0.01
pillow	0.49	0.13	0.21	0.00	0.00	0.08	0.08	0.00	0.00	0.00	0.00
laptop	0.01	0.41	0.30	0.00	0.00	0.29	0.00	0.00	0.00	0.00	0.00
shelfRack	0.14	0.41	0.17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.28

Figure 7. Confusion matrix of both office and home scenes if we have training data from the target scenes.

ally incorporate context with independently predicted labels (for each pixel or region) to get spatially consistent results [25][28][29]. When integrating the contextual information, Conditional Random Field (CRF) is often used [28][29], while some work also makes context a feature and encodes it in the independent classifiers [25].

Some of the recent works in 3D semantic labeling also follow this scheme, either under a structured SVM framework [5][6] or using CRFs [9]. Similarly, there is also work use features to incorporate the spatial context into classifiers [7]. Although good performance is reported, such approaches, no matter 2D or 3D ones, require the training and testing data being from similar collection settings, thus prevents its practical applications on 3D point clouds, where large scale training data is not available and hard to label. We address this problem by seeking help from existing massive 2D datasets, with a novel labeling approach inspired from mask transfer.

Semantic Labeling with Mask Transfer. Another branch of labeling work comes from the rising endeavors in transfer learning, i.e., to intelligently obtain certain knowledge from different yet related sources with metadata propagation [30], showing promising performance in various tasks such as scene understanding [31], segmentation [14], and 3D object detection [32]. In 3D semantic labeling, there is also work adopting online synthesized data for label transfer [8][10]. Its principle lies in identifying the nearest neighbors in the reference data collection, following by transferring the corresponding metadata from the neighbors to the query target. However, traditional search based mask transfer is typically deployed between datasets within the same domain (e.g. from 2D images to 2D images), which does not fit our scenario involving domain changes. We address this with robust search using Exemplar SVMs and incorporating 3D context to ensure a robust fusion from 2D superpixels to point clouds.

Search by Exemplar SVM. Exemplar SVM [17][18] targets at combining the above parametric classifiers and non-parametric, search-based model. To this end, an SVM is trained for each instance, e.g., image or superpixel, the ensemble of which is then used to identify the nearest neighbors of the target instance. Since the discriminatively trained classifier is able to detect the most unique features for each instance, Exemplar SVM has shown promising performance in object detection [18] and cross-domain retrieval [17]. However, it is not easy to directly extend Exemplar SVMs trained on 2D images/superpixels to 3D points, which demands a comprehensive distance metric rather than making binary decisions (is/is not the given instance). Our approach, as detailed in Section 2, handles it with a jointly optimized reranking step using structured prediction [23].

6. Conclusion

How to deal with the semantic labeling problem on the rapidly growing point cloud data is an emerging challenge with a wide variety of practical applications. To the best of the authors' knowledge, this is the first work trying to overcome the key difficulty in the lack of sufficient 3D training data by exploiting existing 2D data. In this work, we propose a novel 2D-to-3D search based label propagation approach to address this issue. More specially, we use an Exemplar SVM based scheme to transfer the massive 2D image labels from ImageNet to point clouds, with a structured SVM based reranking functions design. Our second contribution is proposing a graphical model to integrate both the intra-image and inter-image spatial context in and among reference images to fuse individual superpixel labels onto 3D points. Experiments over popular datasets validate our advantages, with comparable accuracy and superior efficiency to the direct and fully supervised 3D point labeling state of the arts, even *without* any point cloud labeling ground truth.

References

[1] N. Snavely, S. M. Seitz, and R. Szeliski. Photo Tourism: Exploring Image Collections in 3D. *SigGraph*, 2006.

[2] S. Izadi, D. Kim, O. Hilliges, et al. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. *UIST*, 2011.

[3] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. *AAAI*, 2002.

[4] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski. Towards Internet-scale Multi-view Stereo. *CVPR*, 2010.

[5] H. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic Labeling of 3D Point Clouds for Indoor Scenes. *NIPS*, 2011.

[6] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena. Contextually Guided Semantic Labeling and Search for 3D Point Clouds. *IJRR*, 2012.

[7] X. Xiong, D. Munoz, J. Bagnell, and M. Hebert. 3-d scene analysis via sequenced predictions over points and regions. *ICRA*, 2011.

[8] L. Nan, K. Xie, and A. Sharf. A Search-Classify Approach for Cluttered Indoor Scene Understanding. *SigGraph Asia*, 2012.

[9] E. Kalogerakis, A. Hertzmann, and K. Singh. Learning 3D Mesh Segmentation and Labeling. *ToG*, 2010.

[10] K. Lai and D. Fox Object Recognition in 3D Point Clouds Using Web Data and Domain Adaptation *IJRR*, 2010.

[11] B. Russell, A. Torralba, K. P. Murphy, and W. Freeman. LabelMe: A Database and Web-Based Tool for Image Annotation. *IJCV*, 2008.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. *CVPR*, 2009.

[13] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. SUN Database: Large-scale Scene Recognition from Abbey to Zoo. *CVPR*, 2010.

[14] D. Kuettel, M. Guillaumin, and V. Ferrari. Segmentation Propagation in ImageNet. *ECCV*, 2012.

[15] NYU Indoor Depth Dataset. http://cs.nyu.edu/ silberman/datasets/.

[16] Cornell Point Cloud Dataset. http://pr.cs.cornell.edu/sceneunderstanding/data/data.php.

[17] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros. Data-driven Visual Similarity for Cross-domain Image Matching. *SigGraph*, 2011.

[18] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of Exemplar-SVMs for Object Detection and Beyond. *ICCV*, 2011.

[19] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. *IJCV*, 2009.

[20] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *TPAMI*, 2002.

[21] T. Deselaers, B. Alexe, and V. Ferrari. Weakly Supervised Localization and Learning with Generic Knowledge. *IJCV*, 2012.

[22] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? *CVPR*, 2010.

[23] T. Joachims. Optimizing Search Engines Using Clickthrough Data *KDD*, 2002.

[24] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun Large Margin Methods for Structured and Interdependent Output Variables *JMLR*, 2005.

[25] D. Munoz, J. Bagnell, and M. Hebert. Stacked Hierarchical Labeling. *ECCV*, 2010.

[26] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. *CVPR*, 2005.

[27] G. Ben-Artzi, H. Hel-Or, and Y. Hel-Or. The gray-code filter kernels. *TPAMI*, 2007.

[28] K. Murphy, A. Torralba, and W. Freeman. Using the forest to see the trees: a graphical model relating features, objects and scenes. *NIPS*, 2003

[29] R. Fulton and D. Koller. Decomposing a Scene into Geometric and Semantically Consistent Regions. *CVPR*, 2009.

[30] S. J. Pan and Q. Yang. A Survey on Transfer Learning. *TKDE*, 2010.

[31] C. Liu, J. Yuen, and A. Torralba. Nonparametric Scene Parsing via Label Transfer. *TPAMI*, 2011.

[32] A. Patterson, P. Mordohai, and K. Daniilidis. Object Detection from Large-scale 3-D Datasets Using Bottom-up and Top-down Descriptors. *ECCV*, 2008.