3D ShapeNets for 2.5D Object Recognition and Next-Best-View Prediction

Zhirong Wu[†⋆] **Shuran Song**[†] **Aditya Khosla**[‡] **Xiaoou Tang**[⋆] **Jianxiong Xiao**[†]

[†]Princeton University [‡]MIT [⋆]CUHK

Abstract

3D shape is a crucial but heavily underutilized cue in object recognition, mostly due to the lack of a good generic shape representation. With the recent boost of inexpensive 2.5D depth sensors (e.g. Microsoft Kinect), it is even more urgent to have a useful 3D shape model in an object recognition pipeline. Furthermore, when the recognition has low confidence, it is important to have a fail-safe mode for object recognition systems to intelligently choose the best view to obtain extra observation from another viewpoint, in order to reduce the uncertainty as much as possible. To this end, we propose to represent a geometric 3D shape as a probability distribution of binary variables on a 3D voxel grid, using a Convolutional Deep Belief Network. Our model naturally supports object recognition from 2.5D depth map and also view planning for object recognition. We construct a large-scale 3D computer graphics dataset to train our model, and conduct extensive experiments to study this new representation.

1 Introduction

Since the establishment of computer vision as a field five decades ago, 3D geometric shape is considered to be one of the most important cues in object recognition. Even though there are many theories about 3D representation [4, 18], the success of 3D-based methods is largely limited to instance recognition, using model-based keypoint matching [20, 24]. For object category recognition, 3D shape is not used in any state-of-the-art recognition method (e.g. [9, 15]), mostly due to the lack of a good generic representation for 3D geometric shapes. Furthermore, the recent boost of inexpensive 2.5D depth sensors, such as Microsoft Kinect, Google Project Tango, Apple PrimeSense and Intel RealSense, has led to a renewed interest in 2.5D object recognition from depth maps. Because the depth from these sensors is very reliable, 3D shape can play a more important role in recognition. It becomes more urgent to have a good 3D shape model in an object recognition pipeline.

On the other hand, object recognition is sometimes quite difficult, even for humans. It is possible that we cannot confidently recognize an object from a particular viewpoint and we have to resolve to another view to gather more observation for recognizing an object. This situation is even more common for computer vision. Automatic object recognition systems today fail frequently [27], and we desire a robust system that can recover from errors automatically. In particular, as shown in Figure 2, if a robot cannot identify an object confidently from a given view, a fail-safe mode is to allow the robot to move and observe the object from another viewpoint, in order to reduce the uncertainty for recognition. This naturally raises the question for view planning: which next view is the best for helping the robot to discriminate the object category?

To study shape representation and view planning for recognition, we propose to represent a geometric 3D shape as a probabilistic distribution of binary variables on a 3D voxel grid. This model, which we name 3D ShapeNets, uses a powerful Convolutional Deep Belief Network (Figure 1) to learn the

[†]This work was done when Zhirong Wu was a visiting student at Princeton University.

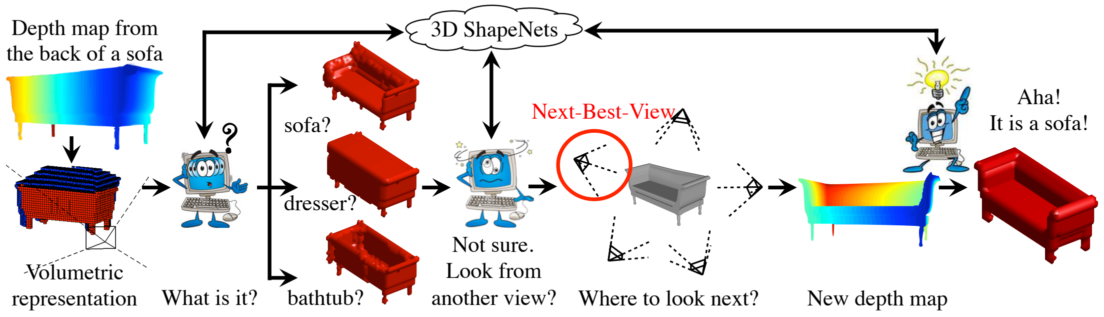


Figure 1: **2.5D Object Recognition and Next-Best-View Prediction using 3D ShapeNets.** Given a depth map of an object (e.g. from RGB-D sensors), we convert the depth map into a volumetric representation and identify the observed surface and free space. Conditioned on these observed voxels, we use our 3D ShapeNets model to identify the object category. If the recognition is ambiguous, we use our 3D ShapeNets model to predict which next view is the best view that has the greatest potential to reduce the recognition uncertainty. Then, a new view is selected and a new depth map is observed. We integrate both views to the volumetric representation and use our 3D ShapeNets to recognize the category. If the uncertainty is still high, the same process will be repeated.

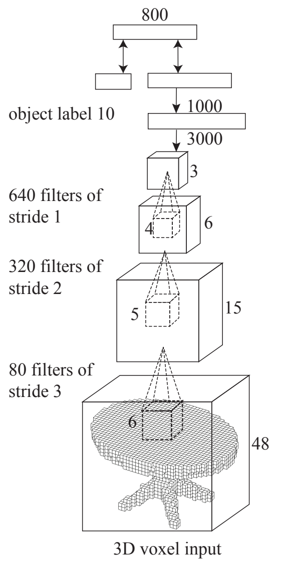
complex joint distribution of all 3D voxels from the data automatically. To train this deep model, we also construct a large scale high quality object dataset, called ModelNet, which includes 127,915 3D computer graphics CAD models. Extensive experiments show that we can use these CAD models for training, to recognize objects in single-view 2.5D depth images and hallucinate the missing parts of the shape of an object. Last but not least, our model can also predict the next-best-view in view planning for object recognition.

1.1 Related Works

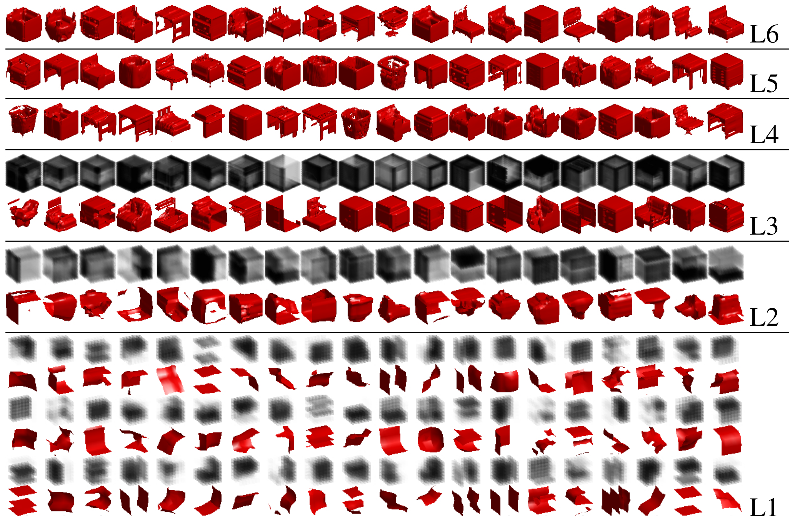
Researchers have built deep model for 2D shapes: most notably DBN [12] to generate handwritten digits and ShapeBM [8] to generate horses etc. Samples from these model are able to capture intra-class varieties. We also desire this generative ability but we are interested in modeling complex shapes in 3D. For deep learning on RGB-D images, [23] built a convolutional-recursive neural network. Although their algorithm runs on depth maps, they didn't explicitly build a model in 3D. Instead, we try to have a model to learn a shape distribution over a voxel grid. To deal with the difficulty of high resolution voxels, we apply the convolution technique in lower layers. [17] used a same technique but their deep model is primary for unsupervised feature extraction[1], which means a separate discriminative classifier is being trained afterward on top of that. Here, we use a single convolutional DBN framework to do reconstruction and recognition.

Unlike static object recognition by a single image, active object recognition allows the sensor to gain more information by moving to new view points. Therefore, the Next-Best-View problem [21] of how to do view planning based on current observation arises. Previous work of [13, 7] considered only color information while we use depth information since we are more interested in object shapes. [6] approached the problem without having prior knowledge of object appearance distributions. [1, 2] implemented the idea into real world robots, but they assumed there is only one object associated with each class so their problem boiled down to instance level and no intra-class variance was considered. Same with our algorithm, [7] also use mutual information to decide the NBV. However, the only extracted a single scalar (mean gray value of the picture) as feature representation, so most of their experiments are preliminary. We consider this problem to the voxel level so that we can tell how voxels in a 3D region would contribute to the reduction of recognition uncertainty. This is not possible without a powerful model directly operated on voxels.

[1]The model is precisely a convolutional DBM where all the connections are undirected.



(a) Architecture of our 3D ShapeNets. For simplicity, we only show one filter for each convolutional layer.



(b) Data-driven visualization: For each neuron, we average the top 100 training examples with highest responses (>0.99) and crop the volume inside the receptive field. The averaged result is visualized by transparency in 3D (Gray) and by the average surface obtained from the zero-crossing (Red). We can see that 3D ShapeNets is able to capture complex structures in 3D space, from low-level surfaces and corners at L1, to objects parts at L2 and L3, and whole objects at L4 and above.

Figure 2: **3D ShapeNets.** Architecture and sample visualization from different layers.

2 3D ShapeNets: A Convolutional Deep Belief Network for 3D Shapes

To study shape representation and view planning for recognition, we propose to represent a geometric 3D shape as a probability distribution of binary variables on a 3D voxel grid. Each 3D mesh is represented as a binary tensor: 1s indicate the voxels on or inside the mesh surface, and 0s indicate the voxels outside the mesh (i.e. empty space). The grid size in our experiments is $48 \times 48 \times 48$.

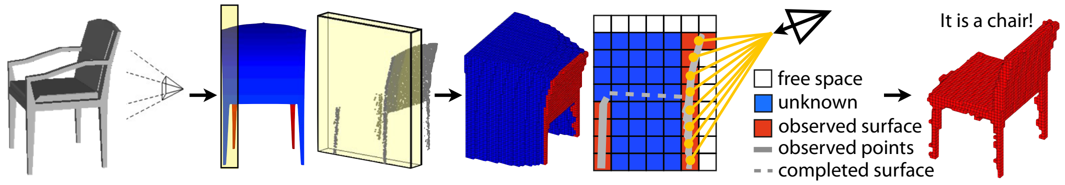
To represent the probability distribution of these binary variables for 3D shapes, we designed a Convolutional Deep Belief Network (CDBN). Deep Belief Network (DBN) [12] is a powerful probabilistic model for binary variables that is typically used for 2D images where they modeled the joint probabilistic distribution over pixels and labels. However, adapting the model from 2D pixel data to 3D voxel data is non-trivial. A 3D voxel volume with reasonable resolution (say $48 \times 48 \times 48$) would have the same dimension of a high-res image (332×332). A fully connected DBN would result in a huge number of parameters that are intractable to train effectively. Therefore, we propose to use convolution to reduce model parameters by weight sharing. But different from typical convolutional deep learning model (e.g. [17]), we don't introduce any kind of pooling in hidden layers, because although pooling may give us invariance properties for recognition, it will also give us more uncertainty during reconstruction, which is important for next-best-view prediction.

The energy of a convolutional layer in our model is defined as:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_f \sum_j \left(h_j^f \left(W^f * v \right)_j + c^f h_j^f \right) - \sum_l b_l v_l \tag{1}$$

where v_l denotes each visible unit, h_j^f denotes each hidden unit in a feature channel f, W^f denotes the convolutional filter. The "$*$" sign represents convolution operation. In this energy definition, each visible unit v_l is associated with a unique bias term b_l to facilitate reconstruction, and all hidden units $\{h_j^f\}$ in the same convolution channel share the same bias term c^f. We also allow convolution stride as in [15].

A 3D shape is represented as a $42 \times 42 \times 42$ voxel grid with 3 extra cells of empty space on both direction for padding to reduce convolution border artifacts. The labels are presented as standard one of K softmax variables. All of our training data are manually aligned to the same direction. The final architecture of our model is in Figure 2(a). The first layer has 80 filters of size 6 and stride 3;



(1) object (2) depth & point cloud (3) volumetric representation (4) recognition & completion

Figure 3: **View-based 2.5D Object Recognition.** (1) illustrates that a depth map taken from a physical object in the 3D world. (2) shows the depth image captured from the back of the chair. A slice is used for visualization. (3) shows the profile of the slice and different types of voxels. The surface voxels of the chair $\mathbf{x}_o$ are in red, and the occluded voxels $\mathbf{x}_u$ are in blue. (4) shows the recognition and shape completion result, conditioned on the observed free space and surface.

the second layer has 320 filters of size 5 and stride 2 (i.e. each filter has $80 \times 5 \times 5 \times 5$ parameters); the third layer has 640 filters of size 4; each convolution filter is connected to all the feature channels in the previous layer; the following two layers are standard fully connected RBM with 3000 and 1000 hidden units; the last layer takes the input as a combination of multinomial label variables and Bernoulli feature variables. The top layer forms an associate memory and all the other layer connections are directed top-down.

We train the deep model in a layer-wise pre-training fashion followed by a generative fine-tuning procedure. During pre-training, the lower four layers are trained using standard Contrastive Divergence [11], while the top layer is trained more carefully using FPCD [25]. Once the lower layer is learned, the weights are fixed and the hidden activations are fed into the next layer as input. Our fine-tuning procedure is similar to wake sleep algorithm [12] except that we keep the weight tied. In the wake phase, we propagate the data bottom up and use the activations to collect the positive learning signal; In the sleep phase, we maintain a persistent chain on the topmost layer and propagate the data top down to collect the negative learning signal. This fine-tuning procedure mimics the recognition and generation behavior of the model and works well in practice. We find that this overall generative fine-tuning is critical for shape completion performance. Some examples of the learned filters are shown in Figure 2(b).

During pre-training of the first layer, we collect learning signal only to receptive fields which are non-empty. Because of the nature of data, empty spaces occupy a large portion of the whole volume, which has no information for RBM and would distract the learning. Our experiment shows that ignoring those learning signals during gradient computation results in more meaningful filters. During pre-training of the first two layers, we add sparsity regularization to encourage the mean activity of hidden unit over training samples to a small constant (we follow the method of [16].) During pre-training of the topmost RBM where the joint distribution of labels and high-level abstractions are learned, we duplicate the label units by a factor of 10 to increase the significancy of the labels.

3 View-based 2.5D Object Recognition

After training the CDBN, the model learns the joint distribution $p(\mathbf{x}, y)$ of voxel data $\mathbf{x}$ and object category label $y \in \{1, \cdots, K\}$. Although the model is trained on complete 3D shapes, it is able to recognition objects in a single-view 2.5D depth map (e.g. from RGB-D sensors). As shown in Figure 3, the 2.5D depth map is firstly converted into a volumetric representation. We categorize each voxel as free space, surface or occluded, depends on whether it is in front of, on, or behind the visible surface (the depth value) from the depth map. The first two types of voxels are observed, and the occluded voxels are regarded as missing data. The testing data is represented in the form $\mathbf{x} = (\mathbf{x}_o, \mathbf{x}_u)$ where $\mathbf{x}_o$ contains the observed free space and surface voxel, and $\mathbf{x}_u$ represent the unknown voxels. Recognizing the object category is to estimate $p(y|\mathbf{x}_o)$.

We approximate the posterior distribution $p(y|\mathbf{x}_o)$ by Gibbs sampling. The sampling procedure is as follows. We first initialize $\mathbf{x}_u$ to random value and propagate the data $\mathbf{x} = (\mathbf{x}_u, \mathbf{x}_o)$ bottom up to sample for a label y from $p(y|\mathbf{x}_u, \mathbf{x}_o)$. Then the high level signal is propagated down to sample for voxels $\mathbf{x}$. We clamp the observation $\mathbf{x}_o$ to this sample $\mathbf{x}$ and do another bottom up pass. This up-down sampling procedure runs for about 200 iterations and we can get the shape completion result $\mathbf{x}$ and its corresponding label y. The above sampling procedure runs in parallel for a lot of particles, and gives a variety of completion results corresponding to different classes.

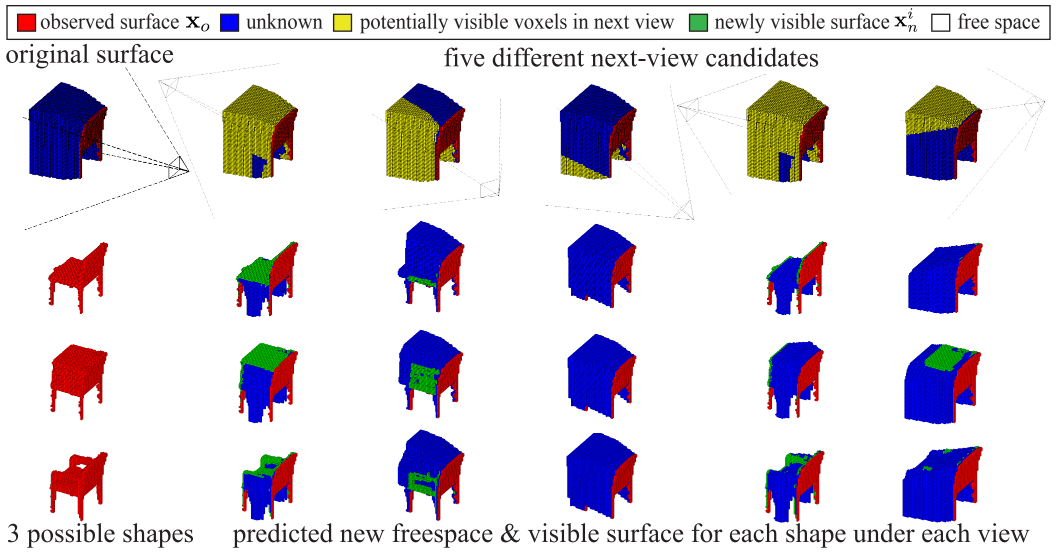

🟥 observed surface $\mathbf{x}_o$
🟦 unknown
🟨 potentially visible voxels in next view
🟩 newly visible surface $\mathbf{x}_n^i$
□ free space
original surface
five different next-view candidates
3 possible shapes
predicted new freespace & visible surface for each shape under each view

Figure 4: **Next-Best-View Prediction.** [Row1 Col1]: the observed and unknown voxels from a single view. [Row2-4 Col1]: three possible completion samples generated by condition on $(\mathbf{x}_o, \mathbf{x}_u)$ [Row1 Col2-6]: five possible camera positions $\mathbf{V}^i$, front top, left-sided, tilted bottom, front, top. [Row2-4 Col2-6]: predict what will happen from each view and each sample using rendering.

4 Next-Best-View Prediction for Recognition

Object recognition is sometimes very challenging. However, if it is allowed to observe the object from another view point when the first recognition fails, we may be able to largely reduce the recognition uncertainty. Our model is able to predict and choose which of such next views is the best for discriminating the object category.

The inputs of our next-best-view system are observed voxels $\mathbf{x}_o$ of an unknown object captured by a depth camera from a single view, and a finite list of next-view candidates $\{\mathbf{V}^i\}$ represents camera rotation and translation in 3D. An algorithm chooses the next-view from the list that has the highest potential to reduce the recognition uncertainty most. Note that during this view planning process, we do not gain any improvement on confidence of $p(y|\mathbf{x}_o = x_o)$ since we observe no new data.

The original recognition uncertainty is measured by the entropy of y conditioned on the observed $\mathbf{x}_o$,

$$H = H\left(p(y|\mathbf{x}_o = x_o)\right) = -\sum_{k=1}^{K} p(y = k|\mathbf{x}_o = x_o)\log p(y = k|\mathbf{x}_o = x_o) \tag{2}$$

where the conditional probability $p(y|\mathbf{x}_o = x_o)$ can be approximated as before by sampling from $p(y, \mathbf{x}_u|\mathbf{x}_o = x_o)$ and marginalizing $\mathbf{x}_u$.

When the camera is moved to another view $\mathbf{V}^i$, some of the previously unobserved voxels $\mathbf{x}_u$ may become observed, subjective to its actual shape. Different views $\mathbf{V}^i$ will cost different visibility of these unobserved voxels $\mathbf{x}_u$. A view with the potential to see distinctive parts of objects (e.g. arms of chairs) may be a better next view. But since the actual shape is partially unknown[2], we will hallucinate that region from our model. As shown in Figure 4, conditioning on $\mathbf{x}_o = x_o$, we can sample many shapes to generate hypotheses of the actual shape, and then render each hypothesis to obtain the depth maps observed from different view $\mathbf{V}^i$. In this way, we can simulate the new depth maps for different views on different samples and compute the benefit on reducing recognition uncertainty.

Mathematically, let $\mathbf{x}_n^i = \text{Render}(\mathbf{x}_u, \mathbf{x}_o, \mathbf{V}^i) \setminus \mathbf{x}_o$ to denote the **new** observed voxels (both free space and surface) in the next view $\mathbf{V}^i$. We have $\mathbf{x}_n^i \subseteq \mathbf{x}_u$, and they are unknown variables that will be marginalized in the following equation. Then the potential recognition uncertainty for $\mathbf{V}^i$ is

[2]If the 3D shape is fully observed, adding more views will not help to reduce the recognition uncertainty in any algorithm purely based on 3D shapes, including our 3D ShapeNets.

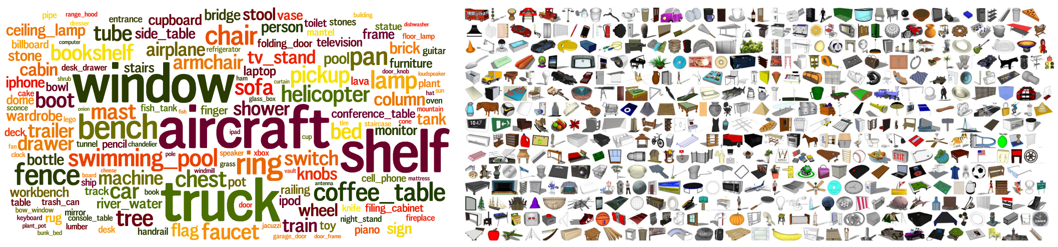


Figure 5: **ModelNet dataset.** Left: visualization of the ModelNet dataset based on the number of images in each category. Larger font size indicates more instances in the corresponding category. Right: examples of 3D models from different categories.

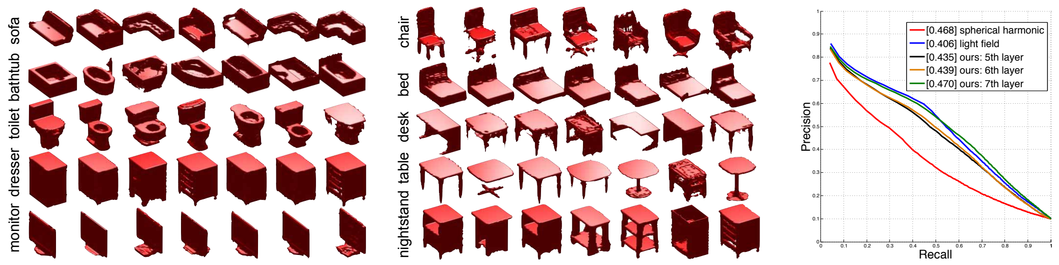


Figure 6: **Results.** Left: Example shapes generated by sampling our 3D ShapeNets for each category. Right: Precision-recall curves and average precision [in brackets] for 3D mesh retrieval.

measured by this conditional entropy,

$$H_i = H\left(p(y|\mathbf{x}_n^i, \mathbf{x}_o = x_o)\right) = \sum_{\mathbf{x}_n^i} p(\mathbf{x}_n^i|\mathbf{x}_o = x_o) H(y|\mathbf{x}_n^i, \mathbf{x}_o = x_o). \tag{3}$$

The above conditional entropy could be calculated by first sampling enough $\mathbf{x}_u$ from $p(\mathbf{x}_u|\mathbf{x}_o = x_o)$, do the 3D rendering to obtain 2.5D depth map in order to get $\mathbf{x}_n^i$ from $\mathbf{x}_u$, and then take each $\mathbf{x}_n^i$ to calculate $H(y|\mathbf{x}_n^i = x_n^i, \mathbf{x}_o = x_o)$ as before.

According to information theory, the reduction of entropy $H - H_i = I(y; \mathbf{x}_n^i|\mathbf{x}_o = x_o) \geq 0$ is the mutual information between y and $\mathbf{x}_n^i$ conditioned on $\mathbf{x}_o$. This meets our intuition that observing more data will always potentially reduce the uncertainty. With this definition, our view planning algorithm is simply to choose the view that can maximize this mutual information,

$$\mathbf{V}^* = \arg\max_{\mathbf{V}^i} I(y; \mathbf{x}_n^i|\mathbf{x}_o = x_o). \tag{4}$$

Our view planning scheme can be naturally extended for a sequence of view planning steps. After deciding the best candidate to move for the first frame, we physically move the camera there and capture the other object surface from that view. The object surfaces from all previous history are merged together as our new observation $\mathbf{x}_o$, and then run our view planning scheme again.

5 Princeton ModelNet: A Large-scale 3D CAD Object Dataset

Training a 3D shape model that captures intra-class variance requires a large collection of 3D shapes. Previous CAD datasets (e.g. [22]) are limited both by category varieties and data size per category. Therefore, we construct a new large scale 3D CAD model data set named ModelNet.

To construct ModelNet, we download 3D models from Google 3D Warehouse by querying object category names. We query categories that are common object categories in SUN database [26] with no less than 20 object instances per category, and remove ones with too few searching results, resulting in 585 remaining categories. We also include models from Princeton Shape Benchmark [22]. After downloading, we remove models not belonging to their labelled categories. This step is done in Amazon Mechanical Turk, in which turkers are shown a sequence of thumbnails of the models and answer "Yes" or "No" to whether their category label is correct. The authors then manually check each 3D model and remove irrelevant objects in each CAD model (e.g, floor, thumbnail image, person standing next to the object), so that each mesh model contains only one object belongs to the labelled category. We also discard unrealistic (overly simplified models or ones that only contain images of the object) and duplicated meshes. Comparing with [22], which consists of 6670

	bathtub	bed	chair	desk	dresser	monitor	nightstand	sofa	table	toilet	all
[23] Depth	0.000	0.729	**0.806**	0.100	**0.466**	0.222	0.343	**0.481**	**0.415**	0.200	0.376
ICP	**0.142**	0.445	0.322	**0.125**	0.200	0.333	0.625	0.253	0.155	0.200	0.280
3D ShapeNets	**0.142**	0.500	0.685	0.100	0.366	**0.500**	**0.719**	0.277	0.377	**0.700**	**0.437**
[23] RGB	**0.142**	**0.743**	0.766	0.150	0.266	0.166	0.218	0.313	0.376	0.200	0.334
[23] RGBD	0.000	0.743	0.693	**0.175**	**0.466**	0.388	0.468	**0.602**	**0.441**	0.500	**0.448**

Table 1: **Accuracy for View-based 2.5D Recognition on NYU dataset [19]**. The first three rows are algorithms that use only depth information. The last two rows are algorithms that use color information. Our 3D ShapeNets performs the best among all depth-based algorithms, and very close to [23] that used both RGB color and depth.

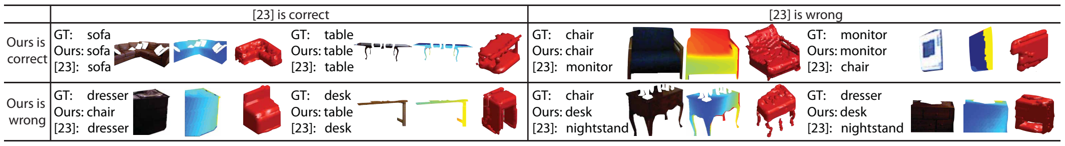

[23] is correct
[23] is wrong
Ours is correct
Ours is wrong
GT: sofa Ours: sofa [23]: sofa
GT: table Ours: table [23]: table
GT: chair Ours: chair [23]: monitor
GT: monitor Ours: monitor [23]: chair
GT: dresser Ours: chair [23]: dresser
GT: desk Ours: table [23]: desk
GT: chair Ours: desk [23]: nightstand
GT: dresser Ours: desk [23]: nightstand

Figure 7: **Success and failure examples of 2.5D Kinect-based object recognition on NYU dataset [19]**. In each example, we show the color, the depth map, and the completed shape by 3D ShapeNets.

models in 161 categories, our new dataset which is 19 times larger contains 127,915 3D meshes in 585 categories. Example of major object categories and dataset statistics are shown in Figure 5.

6 Experiments

To have the same categories with NYU Depth V2 dataset [19], we choose 10 common indoor object categories (see Figure 6) from ModelNet with 4899 CAD models. 3899 models are used for training and 1000 models (100 per category) are used for testing in Section 6.1. Pre-training and fine-tuning each took about two days on a Intel XEON E5-2690 CPU and a NVIDIA K40c GPU. Figure 6 shows some shapes sampled from our trained model.

6.1 3D Shape Classification and Retrieval

Deep learning has been widely used as a feature extraction technique. Here, we are also interested in how well the feature learned from 3D ShapeNets compared with other state-of-the-art 3D mesh features. After training the 3D ShapeNets, we use the top three layer activations as our feature separately for evaluation. When propagating the data upward, we set the labels to zeros so the features do not explicitly contain label information.

For comparison, we choose Light Field descriptor [5] (4700 dimensions) and Spherical Harmonic descriptor [14] (28672 dimensions), which performed best among all descriptors [22]. 3D classification and retrieval experiments are conducted to evaluate our features. For classification, we use linear SVM to train classifiers for each feature to calculate the classification accuracy on the testing set. Our 3D ShapeNets feature achieves an accuracy of 86.5%, 83.7% and 82.0% for the 5th, 6th and 7th layer, while Light Field [5] achieves 86.1% and Spherical Harmonic [14] achieves 82.0%. For retrieval, similarity is measured by the L2 distance of shape descriptors between any testing pairs. Given a query from the testing set, a ranked list of remaining testing data is returned according to the similarity measure. The retrieval performance is evaluated by a precision recall curve in Figure 6. Both experiments show that our 3D ShapeNets is able to learn features comparable to state-of-the-art hand-crafted features.

6.2 View-based 2.5D Recognition

To evaluate 3D ShapeNets for 2.5D depth-based object recognition tasks, we set up an experiment on NYU RGBD dataset with Kinect depth maps [19]. We choose the same 10 categories from NYU and create each testing example by cropping the 3D point cloud from the 3D bounding boxes. Since the model is trained with aligned data while the pose of testing set are arbitrary, we need to firstly estimate the object pose. To do this, we run our recognition algorithm on every possible pose, and choose the one whose completion samples have the highest free energy[3] as the correct pose [10].

As a baseline method, we match the testing point cloud to each our 3D mesh models using Iterated Closest Point method [3] and use the first top 10 matches to predict the labels. We also compare our

[3]Free energy of RBM: $F(x) = -\log\left(\sum_h e^{-E(x,h)}\right)$, the negative logarithm of unnormalized probability.

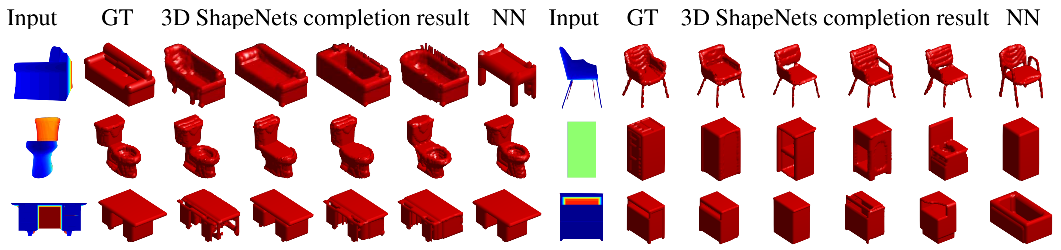

Input
GT
3D ShapeNets completion result
NN
Input
GT
3D ShapeNets completion result
NN

Figure 8: **Shape Completion.** From left to right: input depth map from a single view, ground truth shape, shape completion result (4 examples), nearest neighbor result (1 example).

	bathtub	bed	chair	desk	dresser	monitor	nightstand	sofa	table	toilet	all
Ours	**0.8**	**0.9**	**0.8**	**0.8**	0.2	**1.0**	**0.8**	**0.7**	**1.0**	**0.7**	**0.77**
Max Visibility	0.6	**0.9**	**0.8**	0.6	**0.3**	**1.0**	**0.8**	**0.7**	**1.0**	**0.7**	0.74
Furthest Away	0.3	0.7	**0.8**	0.7	**0.3**	**1.0**	0.7	0.5	0.9	0.6	0.65
Random Selection	0.3	**0.9**	**0.8**	0.6	**0.3**	**1.0**	0.5	0.4	0.9	**0.7**	0.64

Table 2: **Comparison of Different Next-Best-View Selections Based on Recognition Accuracy from Two Views.** Based on the algorithms' choice, we obtain the actual depth map for the next view and recognize the objects using two views by our 3D ShapeNets to compute the accuracies.

result with [23] which is the state-of-the-art deep learning model applied on RGB-D data. To train and test their model, the 2D bounding box is obtained by projecting the 3D bounding box to the image plane, and the object segmentation is also used to extract features. 1390 instances are used to train the algorithm of [23], and the other 495 instances are used for testing all three methods. As shown in Table 1, by only using the depth information, our algorithm can predict a 3D pose and achieve a similar recognition accuracy as [23] which use both RGB color and depth. Figure 7 shows the visualization for success and failure cases.

6.3 Next-Best-View Prediction

For our view planning strategy, computation of the term $p(\mathbf{x}_n^i|\mathbf{x}_o = x_o)$ is critical. When the observation $\mathbf{x}_o$ is ambiguous, samples drawn from $p(\mathbf{x}_n^i|\mathbf{x}_o = x_o)$ should have varieties across different categories. When the observation is rich, samples should be limited to very few categories. Since $\mathbf{x}_n^i$ is the surface of the completions, we could just test the shape completion performance $p(\mathbf{x}_u|\mathbf{x}_o = x_o)$. In Figure 8, our results give reasonable shapes across different categories. We also match the nearest neighbor in the training set to show that our algorithm is not just memorizing the shape and it can generalize well.

To evaluate our view planning strategy, we use CG models from the test set to create synthetic rendering of depth maps. We evaluate the accuracy by running our 3D ShapeNets model on the integration depth maps of both the first view and the selected second view. A good view-planning strategy will result in a better recognition accuracy. Note that next-best-view selection is always coupled with the recognition algorithm. We prepare three baseline methods for comparison : 1) random selection among the candidate views. 2) choose the view with the highest new visibility (yellow voxels, NBV for reconstruction). 3) choose the view which is furthest away with previous view (based on camera center distance). In our experiment, we generate 8 view candidates randomly distributed on the sphere of the object, pointing to the region near the object center. And we randomly choose 100 testing examples (10 for each category) from our testing set. Table 2 reports the recognition accuracy of different view planning strategies with the same recognition 3D ShapeNets. We can see that our entropy based strategy is the best for selecting new views.

7 Conclusion

To study 3D shape representation for objects, we propose a convolutional deep belief network to represent a geometric 3D shape as a probability distribution of binary variables on a 3D voxel grid. We construct a large-scale 3D CG dataset to train our model, and use it to recognize objects in a single-view 2.5D depth map (e.g. from popular RGB-D sensors). Besides it outperforms state-of-the-art algorithms in our experiments, it also supports next-best-view planning for object recognition. Future work includes constructing a large-scale Kinect-based 2.5D dataset so that we can train 3D ShapeNets with all categories from ModelNet and properly evaluate it using this 2.5D dataset.

References

[1] N. Atanasov, B. Sankaran, J. Le Ny, T. Koletschka, G. J. Pappas, and K. Daniilidis. Hypothesis testing framework for active object detection. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4216–4222. IEEE, 2013.

[2] N. Atanasov, B. Sankaran, J. L. Ny, G. J. Pappas, and K. Daniilidis. Nonmyopic view planning for active object detection. *arXiv preprint arXiv:1309.5401*, 2013.

[3] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *PAMI*, 1992.

[4] I. Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 1987.

[5] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, 2003.

[6] F. Deinzer, J. Denzler, and H. Niemann. Viewpoint selection–planning optimal sequences of views for object recognition. In *Computer analysis of images and patterns*, pages 65–73. Springer, 2003.

[7] J. Denzler and C. M. Brown. Information theoretic sensor data selection for active object recognition and state estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(2):145–157, 2002.

[8] S. M. A. Eslami, N. Heess, and J. Winn. The shape boltzmann machine: a strong model of object shape. In *CVPR*, 2012.

[9] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2010.

[10] G. Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 2010.

[11] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 2002.

[12] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 2006.

[13] Z. Jia, Y.-J. Chang, and T. Chen. Active view selection for object and pose recognition. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 641–648. IEEE, 2009.

[14] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Symposium on Geometry Processing*, 2003.

[15] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[16] H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area v2. In *NIPS*, 2007.

[17] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Unsupervised learning of hierarchical representations with convolutional deep belief networks. *Communications of the ACM*, 2011.

[18] J. L. Mundy. Object recognition in the geometric era: A retrospective. In *Toward category-level object recognition*. 2006.

[19] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[20] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *IJCV*, 2006.

[21] W. Scott, G. Roth, and J.-F. Rivest. View planning for automated 3d object reconstruction inspection. *ACM Computing Surveys*, 2003.

[22] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The princeton shape benchmark. In *Shape Modeling Applications*, 2004.

[23] R. Socher, B. Huval, B. Bhat, C. D. Manning, and A. Y. Ng. Convolutional-recursive deep learning for 3d object classification. In *NIPS*. 2012.

[24] J. Tang, S. Miller, A. Singh, and P. Abbeel. A textured object recognition pipeline for color and depth image data. In *ICRA*, 2012.

[25] T. Tieleman and G. Hinton. Using fast weights to improve persistent contrastive divergence. In *ICML*, 2009.

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

[27] P. Zhang, J. Wang, A. Farhadi, M. Hebert, and D. Parikh. Predicting failures of vision systems. In *CVPR*, 2014.