Unsupervised learning through one-shot image-based shape reconstruction

Dinesh Jayaraman
UT Austin
dineshj@cs.utexas.edu

Ruohan Gao
UT Austin
gao@cs.utexas.edu

Kristen Grauman
UT Austin
grauman@cs.utexas.edu

Abstract

Objects are three-dimensional entities, but visual observations are largely 2D. Inferring 3D properties from individual 2D views is thus a generically useful skill that is critical to object perception. We ask the question: can we learn useful image representations by explicitly training a system to infer 3D shape from 2D views? The few prior attempts at single view 3D reconstruction all target the reconstruction task as an end in itself, and largely build category-specific models to get better reconstructions. In contrast, we are interested in this task as a means to learn generic visual representations that embed knowledge of 3D shape properties from arbitrary object views. We train a single category-agnostic neural network from scratch to produce a complete image-based shape representation from one view of a generic object in a single forward pass. Through comparison against several baselines on widely used shape datasets, we show that our system learns to infer shape for generic objects including even those from categories that are not present in the training set. In order to perform this "mental rotation" task, our system is forced to learn intermediate image representations that embed object geometry, without requiring any manual supervision. We show that these learned representations outperform other unsupervised representations on various semantic tasks, such as object recognition and object retrieval.

1. Introduction

While visual perception relies largely on 2D observations, the visual world and objects in particular are inherently three dimensional entities. Inferring 3D geometry from 2D views is therefore a necessary component of object perception.

Cognitive psychologists have found strong evidence supporting this view. In a seminal series of discoveries [38], Shepard and collaborators observed that humans attempting to determine if two views portrayed the same abstract 3D shape spent time that was linearly proportional to the 3D angular rotation between those views. Further, they showed that the time taken was independent of whether the 3D shape underwent simple in-plane rotations in the picture plane, or more complex out-of-plane rotations. These findings are striking. In particular, they tell us that humans may explicitly form mental representations of 3D shape from individual 2D views, and suggest that the act of mentally rotating such representations is integral to registering object views, and by extension, to object perception and recognition.

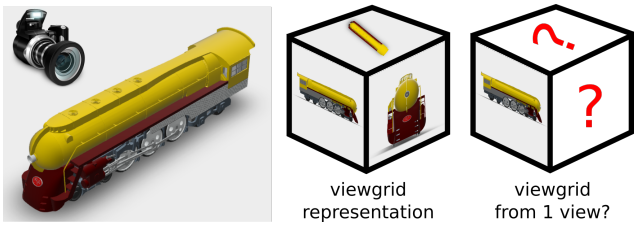


Figure 1. One-shot image-based shape reconstruction: A 3D shape is represented by its "viewgrid'— views from evenly spaced viewpoints. Given one 2D view of an arbitrary shape, we train a deep network to produce the remaining views in the viewgrid. Objects are inherently 3D entities and lifting standard 2D views of objects to 3D is a generically useful skill. So we ask: does training for viewgrid reconstruction as above induce transferable representations in the network?

Inspired by this, we ask the question: can we learn image representations encoding 3D geometry from 2D views, for use in computer vision systems? We propose to do this by training a system for the same target task as was set for the humans in Shepard *et al.*'s study: mental rotation.

More concretely, we set up the task of single view image-based shape reconstruction. Given one 2D view of an object from an arbitrary viewpoint, the system must output views corresponding to arbitrary rotations from the current viewpoint.

With an infinite number of views from all around an object, classic geometric methods allow full 3D shape recovery [17]. These methods can operate with very limited object understanding, as they are completely agnostic to the semantics of the object, and work for arbitrary shapes that might not even represent real-world objects. With real

world objects however, 3D understanding is possible from much sparser observations by using cues such as semantics, shading, etc. This suggests the use of learning-based approaches for reconstruction.

Vision researchers have recently begun to investigate single view 3D reconstruction by employing deep learning methods [10, 49, 7, 13, 35, 46, 26, 42, 40]. However, these few prior attempts (1) all target reconstruction as an end task in itself, (2) largely build category-specific models for common categories (e.g. chairs, cars, faces), and (3) rely on deep neural networks pretrained heavily on manually supervised classification tasks to yield better reconstructions. In contrast, we wish to investigate this task as a means to learn generic visual representations that embed knowledge of 3D shape properties from arbitrary views of arbitrary unseen objects. As suggested above, we hypothesize that downstream vision tasks such as recognition would benefit from an image representation that "lifts" 2D views of objects to inferred 3D shapes.

To this end, we train a category-agnostic deep feedforward neural network from scratch, to produce a complete image-based shape representation given a single view of a generic object, in one forward pass. In order to perform this one-shot reconstruction task, the network must learn a representation that encodes knowledge of the full 3D object shape. Therefore, after training our deep neural network model on this unsupervised task, we extract representations from intermediate layers in the model to use as input features for recognition tasks.

Through experiments on widely used shape datasets and comparison against various baselines, we validate that (1) our system successfully learns the training task of category-agnostic image-based shape reconstruction, generalizing even to categories that were not seen in the training set, and (2) the representations learned in the process are generically useful, in particular transferring well to the semantic tasks of object recognition and retrieval. Our results establish the promise of explicitly targeting 3D understanding as a means to learn generically useful visual representations.

2. Related Work

Geometric view synthesis For many years, new view synthesis methods focused purely on geometry. In image-based rendering, rather than explicitly construct a 3D model, new views are generated directly from multiple 2D views [25]. This usually entails computing point correspondences between views then warping pixels appropriately to obtain intermediate views, as determined by projective or multi-view geometry [37, 2]. Image-based models for object shape have also been developed, where silhouette images are (implicitly) intersected to carve a visual hull [31, 28], possibly with class-specific regularization [15].

Learning 2D-3D relationships Recently, and particularly with the advances in high capacity deep neural networks, there is great interest in instead *learning* the connection between a view and its underlying 3D shape. The problem is being tackled on two main fronts: image-based and volumetric.

In image-based work, the goal is to infer an image as a function of a specified transformation or viewpoint. When provided with two 2D views, methods can learn to predict intermediate views—as, for example, in transforming autoencoders [18], deep stereo [11], deep morphing [23], and the learned similarity functions of [8]. Alternatively, when the input consists of a single view, methods can learn to render the observed object from new camera poses. A tensor completion approach organizes views by people's body poses and camera viewpoints in order to infer "missing" novel views [6]. The inverse graphics network of [27] generates new images of an object with varying pose and lighting, having learned a disentangled representation with a deep convolutional neural network (CNN). Related to this, recurrent convolutional encoder-decoder networks are explored for rendering rotated objects from a single input image [47]. Training with synthetic models, a generative CNN can produce images with specified object types, viewpoints, colors, etc. [10]. A learning objective based on "appearance flow" allows a CNN to learn how to map pixels in the input to its proper destination in a new target view [49]. The main restriction of such methods are their category-specificity. Thus far, view synthesis models are trained for a particular object category at a time (e.g., cars, chairs, people). In contrast, we consider *generalizing* synthesis across objects. To our knowledge, ours are the first view synthesis results for unseen categories.

In volumetric approaches, the goal is instead to map a view(s) directly to a 3D representation of the object, such as a voxel occupancy grid. 3D recurrent networks with LSTM can predict the voxel grid from one or more views [7], while a combination of a 3D generative model and 2D image embedding network supports voxel prediction and 3D model retrieval [13]. Some recent solutions show how to circumvent the need for 3D ground truth in training, such as the space carving-inspired solution of [46] and the generative approach of [35]. While most efforts study synthetic 3D object models (e.g., leveraging CAD datasets), recent work also ventures into real-world natural imagery to learn mappings from instance segmentation and viewpoints to 3D meshes and depth maps [26]. Aside from voxels, inferring depth maps with CNNs allows mesh reconstruction [40] and inferring keypoint skeletons offers a compact representation of 3D object structure [42]. Unlike these methods, we target 2D view synthesis, an implicit representation of the 3D shape. Furthermore, nearly all past volumetric work is also object-specific, while we strive to learn generic representa-

tions. An out-of-category test in [46] shows both the potential and challenges for 3D outputs. Finally, we treat the one-view reconstruction task as a means towards learning unsupervised visual representations, rather than as an end in itself.

Recognition of 3D objects Though 2D object models dominate recognition in recent years (e.g., as evidenced in challenges like PASCAL, COCO, ImageNet), there is growing interest in grounding object models in their 3D structures and shapes. Recent contributions in large-scale data collections are fostering such progress [43, 4, 44], and researchers are developing models to integrate volumetric and multiview approaches effectively [33, 39], as well as new ideas for relating 3D properties (pose, occlusion boundaries) to 2D recognition schemes [45].

Unsupervised representation learning While supervised "pre-training" of CNNs with large labeled datasets is useful [14], it comes at a high supervision cost and there are limits to its transferability to tasks unlike the original labeled categories. As an increasingly competitive approach, researchers are investigating *unsupervised* feature learning [9, 32, 20, 1, 29, 48, 41, 12, 22]. An emerging theme is to identify "pretext" tasks in which during training the network targets an objective for which supervision is inherently free. For example, features tasked with being predictive of contextual layout [9, 32], egomotion [20, 1], colorization [29], or feature slowness [41, 12, 22] often simultaneously embed basic visual concepts useful for recognition. In this spirit, our approach can be seen as a new way to force the visual learner to pick up on foundational cues. In particular, our method can be thought of as generalizing this family of learning methods to 3D information, where the full image-based shape reconstruction is withheld from the system, except for one view.

While prior work considers 3D egomotion [20, 1] for feature learning, it focuses on unattached view pairs [20, 1] and enforces view predictability in the feature space that is being learned [20]. This makes optimization difficult by setting up a moving target regression problem with degenerate solutions that could potentially lead to loss of important information. Such solutions must be carefully avoided using contrastive losses as detailed in [20]. Instead, we focus on the pixel space, where the optimization problem reduces to straightforward fixed-target regression to the ground truth reconstruction. Further, this allows easy interpretation of the trained network, since the learned features are naturally visualized via output reconstructions.

Active object recognition When an agent can move around, it has the chance to be active about its motions, such that fewer views suffice to perform successful recognition. Active recognition work relates to our idea, in that they typically employ models for "hallucinating" an unseen view to reason about its information value [36, 34, 43, 24, 21]. However, active models are almost always object-specific, and rather than a sequence of views our method learns to infer all views jointly from a single input.

3. Approach

Our goal is to train a system to recover a full image-based shape reconstruction of an object after observing one view of it from an arbitrary viewpoint. This task of "mentally rotating" the object from its observed viewpoint to arbitrary relative poses requires developing 3D understanding from single 2D views, which is crucial for many vision tasks. Through training on this image-based shape reconstruction task, we want to eventually learn generically useful visual representations that embed this 3D understanding and apply those representations to recognition and retrieval tasks.

3.1. Task setup

During training, we first evenly sample views from the viewing sphere around each object. To do this, we begin by selecting a set S_{az} of M camera azimuths $S_{az} = \{360°/M, 720°/M, \dots 360°\}$ centered around the object. Then we select a set S_{el} of N camera elevations $S_{el} = \{0°, \pm 180°/(N-1), \pm 360°/(N-1), \cdots \pm 90°\}$. We now sample all $M \times N$ views of every object corresponding to the cartesian product $S = S_{az} \times S_{el}$ of azimuth and elevation positions: $\{\boldsymbol{y}(\boldsymbol{\theta_i}) : \boldsymbol{\theta_i} \in S\}$.[1] Note that each $\boldsymbol{\theta_i}$ is an elevation-azimuth pair, and represents one position in the viewing grid S.

Now, with these evenly sampled views, the one-shot image-based shape reconstruction task can be formulated as follows. Suppose the observed view is at an unknown camera position $\boldsymbol{\theta}$ sampled from our viewing grid set S of camera positions. The system must learn to predict the views $\boldsymbol{y}(\boldsymbol{\theta'})$ at position $\boldsymbol{\theta'} = \boldsymbol{\theta} + \boldsymbol{\delta_i}$ for all $\boldsymbol{\delta_i} \in S$. Because of the even sampling over the full viewing sphere, $\boldsymbol{\theta'}$ is itself also in our original viewpoint set S, so we have already acquired supervision for all views that our system must learn to predict.

Our training phase may be thought of as representing a setting where a robotic visual agent learns visual perception from scratch by inspecting a large number of objects. It can move its camera to arbitrary viewpoints around each object as it does so, to acquire its own supervision. At test time, as a first goal, it must be able to observe only one view and hallucinate the effects of all camera displacements from that view. Eventually, the goal is to transfer knowledge acquired from training on this unsupervised reconstruction task to recognition tasks.

[1] omitting object indices throughout to simplify notation.

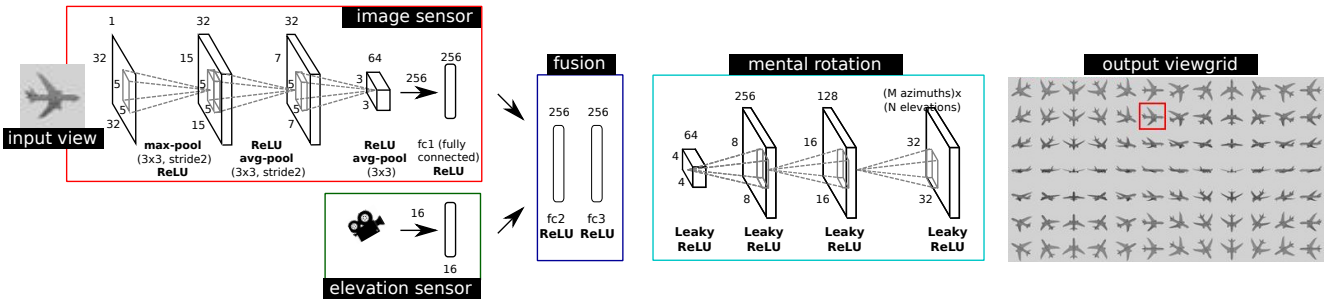


Figure 2. Architecture of our system. A single view of an object (top left) and the corresponding camera elevation (bottom left) are processed independently in "image sensor" and "elevation sensor" neural net modules, before fusion in a "fusion module". The output code, which embeds the 3D shape of the object aligned to the observed view, is now processed in a deconvolutional "mental rotation" decoder. The final output is a sequence of images representing systematically shifted viewpoints relative to the observed view.

3.2. Shape reconstruction system architecture

To tackle this one-shot shape reconstruction task, we model the system as a deep feed-forward neural network. Our network architecture naturally splits into four modular sub-networks with different functions: an elevation sensor module, an image sensor module, a fusion module, and finally, a mental rotation module. Together, the elevation sensor, image sensor, and fusion modules process the observation and proprioceptive camera elevation information to produce a single feature vector that encodes the full object model. The mental rotation module then acts as a decoder — it processes this code through a series of learned deconvolutions to produce the desired image-based shape reconstruction at its output. Fig 2 presents this architecture in detail.

Encoder: First, the image sensor module embeds the observed view through a series of convolutions and a fully connected layer into a vector. In parallel, information about the camera elevation angle is processed through the elevation sensor module. Note that the object pose is not fully known — while camera elevation can be determined from gravity cues, there is no way to determine the azimuth.

The outputs of the image and elevation sensor modules are concatenated and passed through a fusion module which jointly processes the information from the image and elevation sensors to produce a $D = 256$-dimensional output "code" vector, which embeds knowledge of object shape. To sum up, the function of the encoder is to "lift" a 2D view to a single vector representation of the full 3D object shape.

Decoder: The output of the fusion module is now processed through another fully connected layer to increase its dimensionality before reshaping into a sequence of small 4×4 feature maps. These maps are then iteratively upsampled through a series of learned deconvolutional layers. The final output of the mental rotation module is a sequence of MN output maps $\{\hat{\boldsymbol{y}}_{\boldsymbol{i}} : i = 1, \ldots M \times N\}$ of same height and width as the input image. Together these MN maps represent the system's output viewgrid.

The complete architecture, together with more detailed specifications, is visualized in Fig 2. Our convolutional encoder-decoder [30] neural network architecture is similar to [40, 46, 49]. The primary focus of our work is, however, very different. We view one-shot reconstruction as a path to good image representations that lift 2D views to 3D.

There is one final detail of the pipeline to be dealt with — what is the correspondence between the individual views in the target viewgrid and the system's output maps? Recall that at test time, the system will be presented with a single view of a novel object, from an unknown viewpoint (again elevation known, azimuth unknown). How then can it know the correct viewpoint coordinates for the viewgrid it must produce? It instead produces viewgrids aligned with the *observed* viewpoint at the azimuthal coordinate origin. Azimuthal rotations of a given viewgrid all form an equivalence class. In other words, circularly shifting the 7×12 viewgrid in Fig 1 by one column will produce a different, but entirely valid viewgrid representation of the same airplane object.

To optimize the entire pipeline, we regress to the target viewgrid $\boldsymbol{y}$, which is available for each training object. Since our output viewgrids are aligned by the observed view, we must accordingly shift the target viewgrid before performing regression. This leads to the following minimization objective:

$$\mathcal{L} = \sum_{i=1}^{M \times N} \|\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{y}(\boldsymbol{\theta} + \boldsymbol{\delta}_{\boldsymbol{i}})\|^2, \tag{1}$$

where we omit the summation over the training set to keep the notation simple. Each output map $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is thus penalized for deviation from a specific *relative* shift from the observed viewpoint $\boldsymbol{\theta}$.[2]

[2]Since our system has access to elevation, we zero out the elevation coordinate of $\boldsymbol{\theta}$, i.e., the system is trained to produce viewgrids that are perfectly aligned with the target viewgrid in elevation.

At test time, the full image-based shape representation is recovered in one shot, i.e., in a single forward pass through the neural network. This is different from iterative mental rotation by learning to directly hallucinate only some elemental rotations, and then composing them together, as in [7]. Directly producing the full viewgrid has several advantages. In particular, it is efficient, and it avoids "drift" degradation from iterated forward passes. Most critically to our end goal of unsupervised feature learning, this one-shot reconstruction task enforces that the encoder must capture the *full* 3D object shape from observing just one 2D view.

3.3. Training

Models are initialized with parameters drawn from a uniform distribution between -0.1 and +0.1. The mean squared error loss is then optimized through standard minibatch stochastic gradient descent (batch size 32, momentum 0.9) via backpropagation. For our method and all baselines, we optimize the learning rate hyperparameter on validation data. Training terminates when the loss on the validation set begins to rise.

3.4. Unsupervised features for recognition

While we have thus far described our formulation for training a deep neural network for image-based shape reconstruction from single views, our hypothesis is that representations trained in this manner will facilitate high-level visual recognition tasks. This is motivated by the fact that in order to solve the reconstruction task effectively, the network must implicitly learn to "lift" 2D views of objects to inferred 3D shapes. A full 3D shape representation has many attractive properties for generic visual tasks. For instance, pose invariance is desirable for recognition, but is a hard problem in 2D views. This becomes trivial in a 3D representation, since different poses correspond to simple transformations in the 3D space.

Suppose that the visual agent has learned a model as above for image-based shape reconstruction by inspecting 3D shapes. If it is now presented with new dataset of class-labeled training images for a categorization task, we may transfer the 3D knowledge acquired in the one-view reconstruction task to this new task.

Specifically, for each new class-labeled image, we directly represent it in the feature space represented by an intermediate layer in our deep neural network trained for reconstruction. These features are then input to a generic machine learning pipeline, such as a k-nearest neighbor classifier, that is to be trained for the categorization task.

Recall that the output of the fusion module in Fig 2, which is the fc3 feature vector, is trained to encode 3D shape. In our experiments, we test the usefulness of features from fc3 and its two immediate preceding layers, fc2, and fc1, for solving object classification and retrieval tasks.

4. Experiments

We evaluate our approach for two tasks. First, we assess its performance on the image-based shape completion task that it is trained on. Next, we evaluate the strength of the features learned within the model for semantic recognition tasks.

4.1. Datasets

Dataset→	ModelNet		ShapeNet	
Methods↓/ Data→	seen classes	unseen classes	seen classes	unseen classes
Camera elevations	0,±30°,±60°,±90°		0,±30°,±60°	
Camera azimuths	0,30°,60°,...,330°		0,45°,±315°	
View size	32×32		32×32	
Categories	30	10	30	25
Training models	5,852	-	11,532	-
Validation models	312	-	1,681	-
Testing models	1,248	726	3,569	641

Table 1. Dataset statistics

We test our method on two publicly available datasets: ModelNet [44] and ShapeNet [5]. Both of these datasets provide large numbers of manually generated 3D models, with class labels. For each object model, we render 32×32 grayscale views from a grid of viewpoints that is evenly sampled over the viewing sphere centered on the object.

ModelNet [44] has 3D CAD models that are downloaded from the Web, and then manually aligned and categorized. ModelNet comes with two standard subsets: ModelNet-10 and ModelNet-40, with 10 and 40 object classes respectively. The 40 classes in ModelNet-40 include the 10 classes in ModelNet-10. We use the 10 ModelNet-10 classes as our unseen classes, and the other 30 ModelNet-40 classes as seen classes. We use the standard train-test split, and set aside 20% of seen class test set models as validation data. Table 1 shows more details.

ShapeNet [5] contains a large number of models organized into semantic categories under the WordNet taxonomy. All models are consistently aligned to fixed canonical viewpoints. We use the standard ShapeNetCore-v2 subset which contains models from 55 diverse categories. Of these, we select the 30 largest categories as seen categories, and the remaining 25 categories are unseen. We use the standard train-test split. Further, since different categories have highly varying numbers of object instances, we limit each category in our seen class training set to 500 models at most, to prevent the training from being dominated by models of a small number of very common categories. Table 1 shows more details.

4.2. Image-based shape reconstruction

First, we train and test our method on the one-shot image-based shape reconstruction task. For both datasets, the system is trained on the seen classes training set. To set the learning rate and determine termination criteria during

Dataset→	ModelNet		ShapeNet	
Methods↓/ Data→	seen classes	unseen classes	seen classes	unseen classes
	MSE×1000	MSE×1000	MSE×1000	MSE×1000
Avg view	13.514	15.956	14.793	16.394
Avg viewgrid	12.954	15.725	14.334	15.942
GT category avg view	11.006	-	12.279	-
GT category avg viewgrid	8.891	-	9.374	-
Ours w. canonical alignment	4.689	10.440	5.879	9.021
Ours	**3.718**	**7.005**	**4.656**	**6.811**

Table 2. Quantitative results for one-shot image-based shape completion. Results are reported in MSE on images normalized to lie in $[0, 1]$. Lower is better. Per-category results are shown in the appendix in Sec 6.1. "GT category" methods do not work on unseen classes since they rely on knowledge of the category, so those entries are left blank.

deep network training, we employ the seen classes validation set. The trained model is subsequently tested on both seen and unseen class test sets.

To quantitatively evaluate our method, we measure the per-pixel mean squared deviation of the viewgrid produced by our method from the ground truth viewgrid. We compare our method on both seen and unseen classes, against several baselines:

- **Avg view:** This baseline simply predicts, at each viewpoint in the viewgrid, the average of all views observed in the training set *over all viewpoints*.
- **Avg viewgrid:** Both ModelNet and ShapeNet have consistently aligned models, so there are significant biases that can be exploited by a method that has access to this canonical alignment information. This baseline aims to exploit this bias by predicting, at each viewpoint in the viewgrid, the average of all views observed in the training set *at that viewpoint*. Note that our system does not actually have access to this alignment information, so it cannot exploit this bias.
- **GT category avg view:** This baseline represents a model that has perfectly learned classification. Given an arbitrary object from some ground truth category, this baseline predicts, at each viewpoint, the average of all views observed in the training set for that category.
- **GT category avg viewgrid:** This baseline is the same as GT category avg view, but has knowledge of canonical alignments too, so it produces the average of views observed at each viewpoint over all models in that category in the training set.
- **Ours w. canonical alignment:** Our method does not see canonical viewgrid alignments during training, since it is trained to produce views at various *relative* displacements from the observed view, *i.e.*, its target is a viewgrid that has the current view at its origin. This is realistic, since such knowledge is unlikely to be available to an agent observing objects in the real world. However, for the sake of evaluation, we also evaluate a baseline that has access to canonical viewpoints. Concretely, at training time, this baseline is trained to produce a canonically aligned viewgrid from observing one view. Rather than optimizing the objective in Eq (1), this baseline optimizes:

$$\mathcal{L} = \sum_{i=1}^{M \times N} (\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{y}(\boldsymbol{\delta}_{\boldsymbol{i}}))^2, \tag{2}$$

  so that each output map $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ of the system is now assigned to a specific coordinate in the canonical viewgrid axes, rather than a specific relative shift from the observed viewpoint $\boldsymbol{\theta}$, as in Eq (1).

Table 2 shows the mean squared error results.[3] "Avg viewgrid" and "GT category avg viewgrid" improve by large margins over "Avg view" and "GT category avg view" respectively. This shows that viewgrid alignment biases can be useful for reconstruction in ModelNet and ShapeNet. Recall however that while these baselines can exploit the bias, our approach cannot; it knows only the elevation as sensed from gravity, so it cannot, for instance, learn to memorize and align an average viewgrid.

Despite this, our approach not only outperforms these methods by large margins but even outperforms its variant "Ours w. canonical alignment" that does use alignment biases in the data. The margin is especially large on unseen class subsets of both ModelNet and ShapeNet, where using alignment biases of seen class categories strongly hurts the performance of "Ours w. canonical alignment". Why is "Ours w. canonical alignment" weaker? Recall that it is trained to produce canonically aligned viewgrids from observing one view, but like our method, it also does not have access to the absolute azimuth of the observed view. If "Ours w. canonical alignment" were learning to do something simple like produce the average viewgrid for an inferred category (similar to "GT category avg viewgrid"),

[3] Per-category results are shown in the appendix in Sec 6.1, and Fig 6 and 7.

then targeting canonically aligned viewgrids would be a clear advantage.[4] But to exploit instance-specific details from the observed view effectively, it must perform the additional step of inferring the position of the observed view in the canonical viewgrid, which can be difficult i.e. even if it infers shape correctly, it may struggle to produce correct canonically aligned viewgrids. It is therefore, easier and more natural to represent shape with respect to the observed viewpoint, as in "Ours".

Fig 5 shows some example viewgrids generated by our method. In (row 1, panel 1) from ModelNet and (row 3, panel 4) from ShapeNet, our method exhibits the ability to infer shape from shading cues for simple objects. In (row 1, panel 3), the system manages to reconstruct an object shape from what appears to be an impossible viewpoint to fully infer the shape, indicating that it effectively exploits the semantic structure in ModelNet to make educated guesses. In (row 2, panel 2), the system observes a very ambiguous viewpoint that could be any one of four different views at the same azimuth. In response to this ambiguity, it attempts to play it safe to minimize MSE loss by averaging over possible outcomes, producing blurry views.

Overall, these results establish that our approach successfully learns one single *unified category-agnostic* one-shot shape reconstruction model that handles not only views of shapes from a large number of generic categories that are represented in its training set, but also shapes from unseen categories.

4.2.1 Influence of observed view position on viewgrid reconstruction error

We now break down the errors summarized in Tab 2 in the paper on the basis of the dependence of those errors on the observed view. Specifically, in both ModelNet and ShapeNet, models are manually aligned to some canonical starting positions (which are deliberately withheld from our approach). This means that we can meaningfully attempt to understand which positions in the viewgrid, when observed, led to higher or lower reconstruction errors for the full viewgrid for each category. In other words, which views are more or less informative to our one-shot image-based shape reconstruction approach?

Fig 3 show results for a few ModelNet seen classes. For each class, a heatmap of mean-square errors is overlaid over the average viewgrid for that class. The first map corresponds to all 30 ModelNet seen classes, and the accompanying colorbar illustrates how lower MSEs correspond to darker, colder, blue colors, and higher MSEs to lighter, warmer, yellow colors.

[4] assuming seen categories. For unseen categories, such a strategy is infeasible.

Studying these heatmaps reveals some intuitive and interesting trends. Across all datasets, perfectly aligned viewing positions from where only a single or a small number of faces of an object are visible have yellowish, lighter highlights indicating high reconstruction errors conditioned on those views, *i.e*. lower informativeness. See for instance, the "bench" heatmap in Fig 3, which has characteristic yellowish horizontal and vertical stripes running across the viewgrid corresponding to elevations and azimuths that only reveal a small number of faces of the object. Top and bottom views, sampled at -90°and +90°in ModelNet, are consistently uninformative, since very different shapes can have very similar overhead projections. Shapes that have narrow linear projections along some directions tend to present very little information from the corresponding views. See the horizontal view of the airplane and the keyboard (middle row in the corresponding viewgrids, corresponding to zero azimuth) in Fig 3.

Overall, these trends largely agree with our intuitive notions of which views are most informative for 3D understanding, and serve as evidence that our method learns to perform reconstruction by observing meaningful and appropriate cues.

4.3. Unsupervised feature evaluation

Our system is trained end-to-end from scratch for the task of predicting all unobserved viewpoints given only one view of an object. We now test whether it learns generically useful visual representations in the process.

4.3.1 Nearest neighbor classification

First, as described in Sec 3.4, we extract features from various layers in the network (fc1, fc2, fc3 in Fig 2) and use them as inputs in a k-nearest neighbor classifier trained for categorization of individual object views. We run this experiment on both seen and unseen class subsets on both ModelNet and ShapeNet. In each case, we use 1000 samples per class in the training set, and set $k = 5$. See Sec 6.2 for results varying training set size.

We compare our features against a variety of baselines:

- **Pixels:** For this baseline the 32×32 image is vectorized and used directly as a feature vector.
- **Random weights:** A network with identical architecture to ours and initialized with the same scheme is used to extract features with no training.
- **DrLIM [16]:** This is a commonly used unsupervised feature learning approach. During training, DrLIM attempts to learn an invariant feature space by mapping features of views of the same training object close to

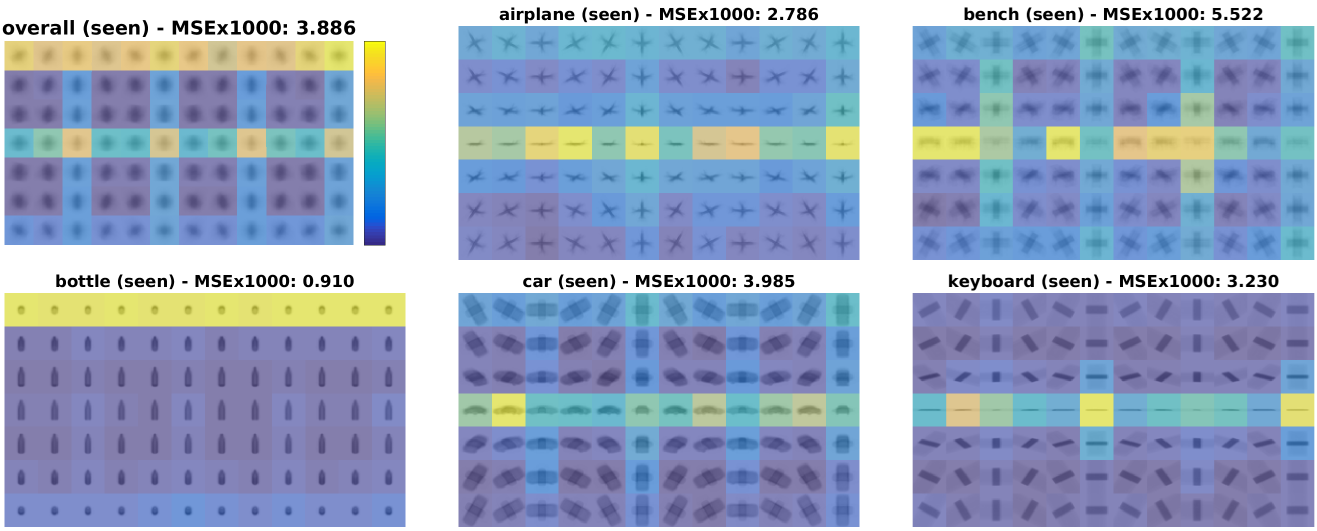


Figure 3. ModelNet reconstruction MSE for a few sample seen classes, conditioned on observed view (best observed in pdf at high resolution). Warmer, yellower colors correspond to high MSE (bad) and cooler, bluer colors correspond to low MSE (good) as shown in the color bar for the first heatmap. See Sec 4.2.1.

Datasets→	ModelNet-seen classes (30 cls)						ModelNet-unseen classes (10 cls)					
Layers→	fc1		fc2		fc3		fc1		fc2		fc3	
Methods↓/Metrics→	accuracy	mAP	accuracy	mAP	accuracy	mAP	accuracy	mAP	accuracy	mAP	accuracy	mAP
Pixels	52.5	58.4	52.5	58.4	52.5	58.4	60.7	67.1	60.7	67.1	60.7	67.1
Random weights	49.6	55.9	48.4	55.1	48.1	54.9	59.4	65.8	58.0	65.0	58.0	64.7
DrLIM [16]	57.4	63.3	55.2	61.5	52.8	59.1	64.9	70.9	63.7	69.7	61.8	68.1
Autoencoder [19, 3, 30]	52.5	58.5	52.5	58.6	52.4	58.6	60.8	67.7	59.6	66.9	60.1	67.0
Egomotion [1]	55.3	61.4	56.1	62.1	55.8	62.1	63.9	70.1	64.1	70.4	65.0	70.9
Ours w. canonical alignment	63.9	69.2	64.0	69.3	63.4	68.6	69.6	74.9	69.4	74.7	69.1	74.4
Ours	**65.2**	**70.6**	**65.0**	**70.4**	**64.7**	**69.9**	**71.2**	**76.1**	**70.4**	**75.8**	**70.0**	**75.0**

Table 3. Nearest neighbor classification with features from our model vs. baselines using identical architectures, on the ModelNet dataset. Results are reported as % accuracy and % mean average precision (mAP). Results with varying training dataset size reported in Sec 6.2.

each other, and pushing features of views of different training objects far apart from one another.

- **Autoencoder [19, 3, 30]:** A network is trained to observe an input view from an arbitrary viewpoint and produce exactly that same view as its output (compared to our method which produces the full viewgrid, including views from *other* viewpoints too). A long line of work has attempted to learn unsupervised visual representations through autoencoders [19, 3, 30]. For this method, we use an architecture identical to ours except at the very last deconvolutional layer, where, rather than producing $N \times M$ output maps, it predicts just one map corresponding to the observed view itself.
- **Egomotion [1]:** Like our method, this baseline also exploits camera motion to learn unsupervised representations. While our method is trained on the task of predicting all rotated views given a starting view, [1] trains on the task of predicting the camera rotation between a given pair of images. In our implementation of this baseline, we trained a model to predict 8 classes of rotations, corresponding to the immediately adjacent viewpoints in the viewgrid for a given view (3×3 neighborhood).
- **Ours w. canonical alignment:** Also used in the previous experiment, this baseline is a variant of our method that is trained to produce canonically aligned viewgrids at training time, rather than views that are *relatively* displaced from the observed view.

All models are trained with identical architectures to ours until fc3, to allow fair comparison. Fig 4 shows detailed specifications of network architectures used in our method and baselines.

Recall that our models are trained to observe camera elevations together with views, as shown in Fig 2. While this is plausible in a real world setting where an agent may know its camera elevation angle from gravity cues, for fair comparison with our baselines, we do not use location inputs when evaluating our unsupervised features. Instead, we feed in camera elevation 0° to the location module for all views.

Datasets→	ShapeNet-seen classes (30 cls)						ShapeNet-unseen classes (25 cls)					
Layers→	fc1		fc2		fc3		fc1		fc2		fc3	
Methods↓/Metrics→	accuracy	mAP	accuracy	mAP	accuracy	mAP	accuracy	mAP	accuracy	mAP	accuracy	mAP
Pixels	43.1	49.3	43.1	49.3	43.1	49.3	44.9	51.1	44.9	51.1	44.9	51.1
Random weights	39.6	46.5	38.4	45.7	38.5	45.5	39.7	46.1	39.6	46.5	39.5	46.1
DrLIM [16]	47.5	52.2	45.2	50.9	45.6	49.9	47.1	52.1	47.2	53.1	45.9	53.1
Autoencoder [19, 3, 30]	43.7	50.0	43.9	50.1	44.3	50.3	46.0	51.8	46.0	52.0	45.5	51.7
Egomotion [1]	49.0	54.7	47.7	53.5	47.0	53.0	49.7	55.4	48.3	54.2	48.2	54.1
Ours w. canonical alignment	56.9	62.0	56.9	62.1	56.3	61.5	54.5	59.6	54.5	59.5	54.1	59.3
Ours	**57.5**	**62.4**	**57.7**	**62.7**	**57.3**	**62.3**	**54.7**	**59.9**	**54.8**	**60.1**	**54.7**	**59.9**

Table 4. Nearest neighbor classification with features from our model vs. baselines using identical architectures, on the ShapeNet dataset. Accuracy and mean AP reported in %. Results with varying training dataset size reported in Sec 6.2.

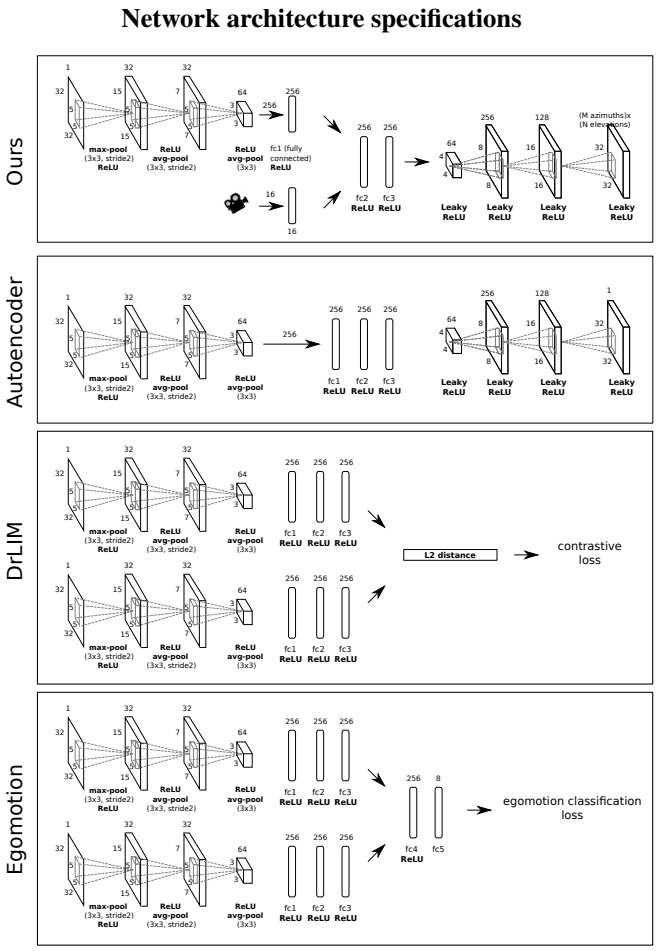


Figure 4. Architecture of our system and various baselines (best observed in pdf at high resolution).

Table 3 and Table 4 show the results for ModelNet and ShapeNet respectively. Trends across fc1, fc2, and fc3 are all very similar, and all three layers' representations appear to be approximately equally discriminative for this task for each method. Among the baselines, all unsupervised learning methods outperform "Pixels" and "Random weights", as expected. The two strongest baselines are "Egomotion" and "DrLIM". Recall that "Egomotion" is especially relevant to our approach as it also has access to relative camera motion information. However, our approach exploits this information much more effectively. "Ours" and "Ours w. canonical alignment" both strongly outperform all prior approaches, and of the two, "Ours" marginally outperforms "Ours w. canonical alignment".

As seen from our results, "Autoencoder" features perform very poorly. Interestingly, our method — which can be thought of as a generalized autoencoder in 3D that maps one view to a full image-based shape reconstruction — learns much stronger representations. This suggests that our system benefits strongly from the incentive to learn not just an identity mapping, as in the autoencoder, but to acquire 3D understanding.

4.3.2 Object category retrieval

Aside from these nearest neighbor classification experiments, we also perform object category retrieval experiments to test the usefulness of our features for a different high-level task. A retrieval system is presented with individual object views as queries, and the task is to fetch the closest views from the training set in the learned feature space. We compare against the same baselines as for the nearest neighbor experiments. For this task, we present a query image from the test set, and retrieve the closest images in the training set, as measured by Euclidean distance in the learned feature space. For representations that encode semantics, these closest images would belong to the same category as the query, so we evaluate various representations on their ability to retrieve images from the same class as the query.[5]

We measure top-1, top-5, and top-20 accuracies. We separately test fc1, fc2, and fc3 features from our method and the baselines, as in the classification experiments in Sec 4.3 in the paper. Results are shown in Tab 5, for ModelNet and ShapeNet for both seen and unseen classes.

[5]For the unseen class experiments, images were retrieved from the corresponding unseen class training set, disjoint from the test set from which queries were drawn.

Trends are very similar to those observed for classification in the paper. Ours and Ours w. canonical alignment once again easily outperform all baselines. Performance for all methods remain roughly similar at all three feature layers, except DrLIM. DrLIM is strongest at fc1, and weakens at fc2 and still further at fc3, indicating that the invariance enforced by DrLIM at fc3 (See Fig 4) leads to loss of discriminativeness.

An interesting trend related to generalization to unseen classes is observed most clearly in the ShapeNet experiments, where the number of classes among seen and unseen classes is comparable (30 and 25 respectively), so that the numbers are roughly comparable among seen and unseen class experiments. Note how features from the Autoencoder baseline has much lower accuracies on unseen classes than on seen classes, demonstrating lack of generalization. Our one-shot reconstruction-based approaches have very similar accuracies on seen and unseen classes, thus establishing that they learn generalizable features. DrLIM and Egomotion accuracies are also comparable for seen and unseen classes, but significantly lower than our method.

5. Conclusions

We have proposed an approach to mentally rotate a view of any object to arbitrary viewpoints, recovering a full image-based shape reconstruction in one shot, from a single view. Through experiments on two recent, widely used and publicly available shape datasets, we have validated that our approach learns a *single model* that can produce good shape reconstructions for a variety of objects from across many categories including categories that are not seen during training time. This is in contrast to most prior art for single-view 3D reconstruction, which learns category-specific models.

Further, our approach is well-suited to learn generically useful unsupervised image features that can represent 3D shape information from observing just a single 2D view, since it is trained on the task of producing a full viewgrid in one shot. Through experimental comparison against various unsupervised learning techniques, we have validated that our approach learns generic visual representations that transfer well to semantic tasks like object recognition and image retrieval, outperforming representative state of the art approaches. Our results establish the promise of explicitly targeting 3D understanding as a means of learning generically useful visual representations.

References

[1] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *ICCV*, 2015.

[2] S. Avidan and A. Shashua. Novel view synthesis in tensor space. In *CVPR*, 1997.

[3] Y. Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[4] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[5] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[6] C.-Y. Chen and K. Grauman. Inferring unseen views of people. In *CVPR*, 2014.

[7] C. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstrution. In *ECCV*, 2016.

[8] W. Ding and G. Taylor. "mental rotation" by optimizing transforming distance. In *NIPS Workshop*, 2014.

[9] C. Doersch, A. Gupta, and A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

[10] A. Dosovitskiy, J. Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In *CVPR*, 2015.

[11] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2015.

[12] R. Gao, D. Jayaraman, and K. Grauman. Object-centric representation learning from unlabeled videos. In *ACCV*, 2016.

[13] R. Girdhar, D. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016.

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[15] K. Grauman, G. Shakhnarovich, and T. Darrell. A bayesian approach to image-based visual hull reconstruction. In *CVPR*, 2003.

[16] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality Reduction by Learning an Invariant Mapping. *CVPR*, 2006.

[17] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[18] G. Hinton, A. Krizhevsky, and S. Wang. Transforming auto-encoders.

[19] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[20] D. Jayaraman and K. Grauman. Learning image representations tied to ego-motion. In *ICCV*, 2015.

ModelNet retrieval results

Datasets→	ModelNet-seen classes (30 cls)								
Layers→		fc1			fc2			fc3	
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
Pixels	55.2	46.9	36.6	55.2	46.9	36.6	55.2	46.9	36.6
Random weights	51.6	44.2	34.9	50.0	42.9	34.2	50.4	42.8	34.0
DrLIM [16]	58.8	51.1	41.7	56.7	48.8	39.7	53.8	46.3	37.1
Autoencoder [19, 3, 30]	54.8	46.5	37.5	55.1	46.7	37.5	55.5	47.3	37.7
Egomotion [1]	56.6	48.8	39.6	57.6	49.6	39.9	57.4	49.6	39.7
Ours w. canonical alignment	64.3	57.9	49.5	64.4	58.2	**50.2**	63.5	57.7	**50.0**
Ours	**65.6**	**59.2**	**49.7**	**64.9**	**58.3**	49.5	**65.5**	**58.4**	49.7

Datasets→	ModelNet-unseen classes (10 cls)								
Layers→		fc1			fc2			fc3	
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
Pixels	60.3	53.9	44.5	60.3	53.9	44.5	60.3	53.9	44.5
Random weights	60.0	52.6	44.3	57.9	51.7	43.5	57.9	51.1	43.0
DrLIM [16]	64.7	58.5	49.5	63.9	56.8	47.6	60.7	54.0	44.8
Autoencoder [19, 3, 30]	60.5	54.7	45.5	60.0	54.3	45.5	60.6	54.0	45.1
Egomotion [1]	63.4	57.4	48.2	64.2	57.3	48.2	63.8	57.4	48.3
Ours w. canonical alignment	69.0	63.9	**56.3**	68.8	63.8	**56.5**	68.4	63.3	**56.3**
Ours	**69.8**	**64.7**	55.9	**69.2**	**63.9**	55.5	**68.5**	**64.0**	55.2

ShapeNet retrieval results

Datasets→	ShapeNet-seen classes (30 cls)								
Layers→		fc1			fc2			fc3	
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
Pixels	42.8	36.5	28.6	42.8	36.5	28.6	42.8	36.5	28.6
Random weights	38.6	32.9	26.4	37.5	31.9	25.8	36.4	32.0	25.7
DrLIM [16]	47.1	40.2	31.3	45.9	39.6	30.2	43.3	37.1	28.9
Autoencoder [19, 3, 30]	42.5	36.8	29.6	42.6	37.1	29.8	42.3	37.4	29.5
Egomotion [1]	48.9	42.1	33.3	47.5	40.6	32.1	47.2	40.1	31.7
Ours w. canonical alignment	**56.1**	**50.8**	**44.0**	**57.3**	**51.6**	**45.2**	**56.9**	**51.4**	**43.6**
Ours	56.0	50.2	42.4	55.8	50.5	42.8	55.3	49.6	42.2

Datasets→	ShapeNet-unseen classes (25 cls)								
Layers→		fc1			fc2			fc3	
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
Pixels	44.6	40.4	33.6	44.6	40.4	33.6	44.6	40.4	33.6
Random weights	26.6	20.8	15.4	25.9	21.0	15.3	25.7	20.4	15.2
DrLIM [16]	48.1	41.5	36.1	48.3	41.8	34.7	44.9	41.1	34.1
Autoencoder [19, 3, 30]	30.7	23.9	17.0	31.1	23.9	17.3	30.5	23.9	17.2
Egomotion [1]	49.9	43.6	37.1	49.4	42.8	36.3	48.7	42.5	36.1
Ours w. canonical alignment	53.7	49.1	**42.9**	53.4	**49.6**	**43.5**	52.8	49.1	**43.6**
Ours	**54.9**	**49.3**	42.7	**55.1**	49.5	42.7	**54.7**	**49.2**	42.4

Table 5. **Above:** Retrieval experiments on ModelNet (1000 training samples per class). Seen class (top) and unseen class (bottom) results. Results reported as top-1, top-5 and top-20 accuracies. (Higher is better.) **Below:** Retrieval experiments on ShapeNet (1000 training samples per class). Seen class (top) and unseen class (top) results. Results reported as top-1, top-5 and top-20 accuracies. (Higher is better.)

[21] D. Jayaraman and K. Grauman. Look-ahead before you leap: End-to-end active recognition by forecasting the effect of motion. In *ECCV*, 2016.

[22] D. Jayaraman and K. Grauman. Slow and steady feature analysis: Higher order temporal coherence in video. In *CVPR*, 2016.

[23] D. Ji, J. Kwon, M. McFarland, and S. Savarese. Deep view morphing. In *arXiv*, 2017.

[24] E. Johns, S. Leutenegger, and A. Davison. Pairwise decomposition of image sequences for active multi-view recognition. In *CVPR*, 2016.

[25] S. B. Kang. A survey of image-based rendering techniques. *Videometrics SPIE Intl Symp on Elec Imag: Science and Technology*, 1999.

[26] A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Category-specific object reconstruction from a single image. In *CVPR*, 2015.

[27] T. Kulkarni, W. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *NIPS*, 2015.

[28] K. Kutulakos and S. Seitz. A theory of shape by space carving. *IJCV*, 2000.

[29] G. Larsson, M. Maire, and G. Shakhnarovich. Colorization as a proxy task for visual understanding. In *CVPR*, 2017.

[30] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*, pages 52–59. Springer, 2011.

[31] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan. Image-based visual hulls. In *SIGGRAPH*, 2000.

[32] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016.

[33] C. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, 2016.

[34] V. Ramanathan and A. Pinz. Active object categorization on a humanoid robot. In *VISAPP*, 2011.

[35] D. Rezende, S. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. Unsupervised learning of 3d structure from images.

[36] B. Schiele and J. Crowley. Transinformation for active object recognition. In *ICCV*, 1998.

[37] S. Seitz and C. Dyer. View morphing. In *SIGGRAPH*, 1996.

[38] R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171:701–703, 1971.

[39] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural netowrks for 3d shape recognition. In *ICCV*, 2015.

[40] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Multi-view 3d models from single images with a convolutional network. In *ECCV*, 2016.

[41] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015.

[42] J. Wu, T. Xue, J. Lim, Y. Tian, J. Tenenbaum, A. Torralba, and W. Freeman. Single image 3d interpreter network. In *ECCV*, 2016.

[43] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.

[44] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.

[45] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Data-driven 3d voxel patterns for object category recognition. In *CVPR*, 2015.

[46] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *NIPS*, 2016.

[47] J. Yang, S. Reed, M.-H. Yang, and H. Lee. Weakly supervised disentangling with recurrent transformations in 3d view synthesis. In *NIPS*, 2015.

[48] R. Zhang, P. Isola, and A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *arXiv*, 2016.

[49] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. Efros. View synthesis by appearance flow. In *ECCV*, 2016.

6. Appendix

6.1. Category-wise reconstruction results

We showed quantitative reconstruction results on ModelNet seen/unseen class and ShapeNet seen/unseen class datasets in Sec 4.2 and Tab 2 in the paper.

To break down those results in more detail, we present per-pixel mean squared errors for each category individually in Fig 6 and 7, arranged in sorted order for seen and unseen classes separately for each dataset.

As can be seen, while the distribution of errors among unseen classes is shifted towards higher errors from the seen classes, there is a significant overlap *i.e.*, many unseen classes have lower reconstruction errors than many seen classes, indicating significant generalization from seen to unseen classes.

6.2. Nearest neighbor classification with varying training set size

Nearest neighbor classification results are presented in the paper in Sec 4.3, as a means of evaluating the unsupervised features learned by our one-shot reconstruction method, for discriminativeness. Nearest neighbor classifiers are often sensitive to the size and constitution of the training set. We test the stability of the results in Tab 3 and 4 in the paper. To do this, we sample multiple training sets of varying sizes ranging from 50 to 1000 samples per class (50, 100, 200, 300, ... 1000), and report accuracies for k-nearest neighbor classification (k=5 as in the paper) with each training set.

These are presented for both seen and unseen class subsets of both ModelNet and ShapeNet, in Fig 8. We observe that curves corresponding to different methods rarely overlap as the training set is varied, establishing the stability of the results in Tab 3 and 4 in the paper. In particular, our one-shot reconstruction-based approaches continue to produce the most discriminative features at all training set sizes. fc1, fc2, and fc3 trends all continue to be similar.

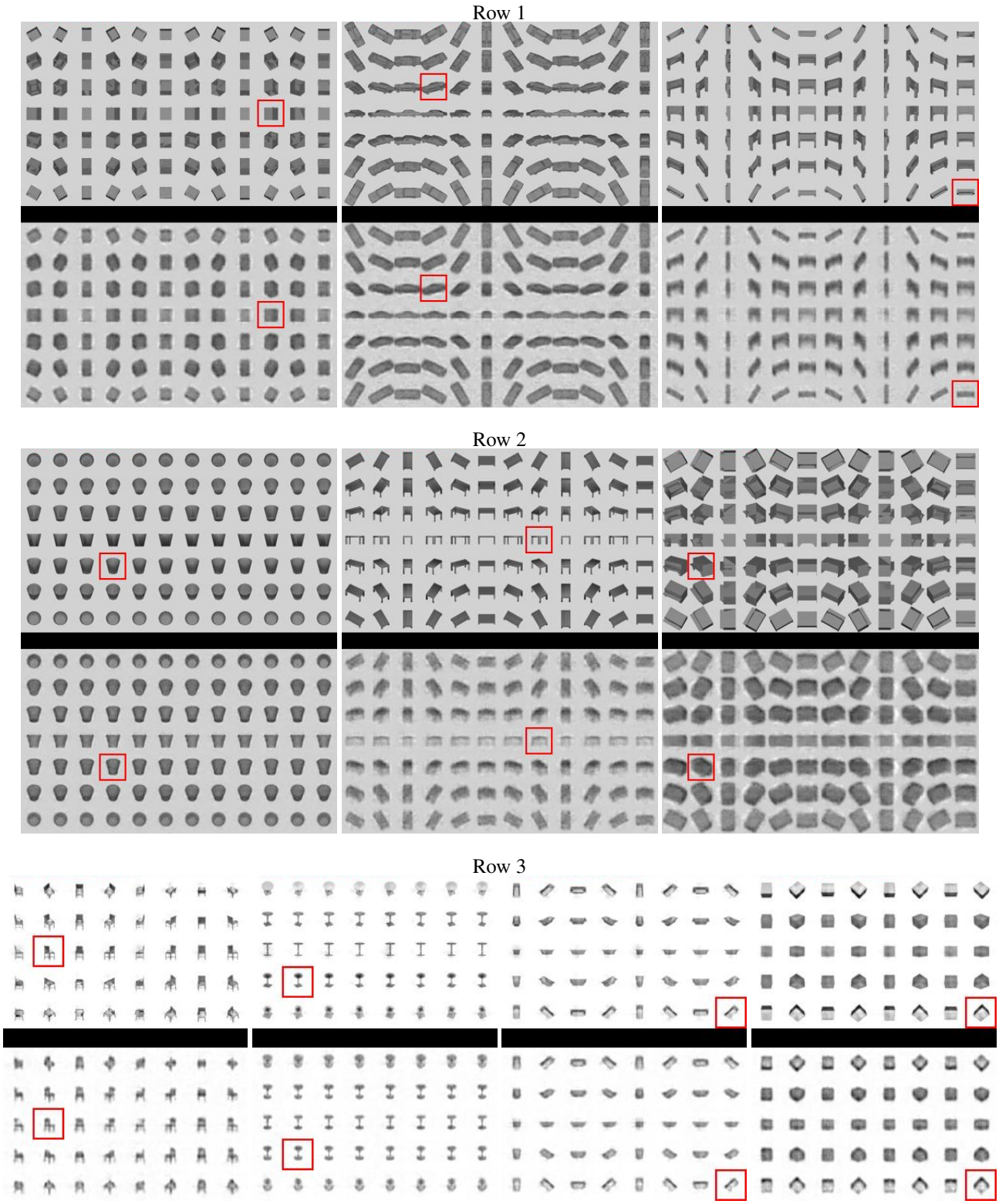


Figure 5. One-shot image-based shape reconstructions from single view. In each panel, ground truth viewgrids are shown at the top, the observed view is marked with a red box, and our method's reconstructions are shown at the bottom. Top two rows show ModelNet results, and bottom row shows ShapeNet results. (Best seen in pdf at high resolution.) In (row 1, panel 1) from ModelNet and (row 3, panel 4) from ShapeNet, our method exhibits the ability to infer shape from shading cues for simple objects. In (row 1, panel 3), the system manages to reconstruct an object shape from what appears to be an impossible viewpoint to fully infer the shape, indicating that it effectively exploits the semantic structure in ModelNet to make educated guesses. In (row 2, panel 2), the system observes a very ambiguous viewpoint that could be any one of four different views at the same azimuth. In response to this ambiguity, it attempts to play it safe to minimize MSE loss by averaging over possible outcomes, producing blurry views.

ModelNet category-wise MSE

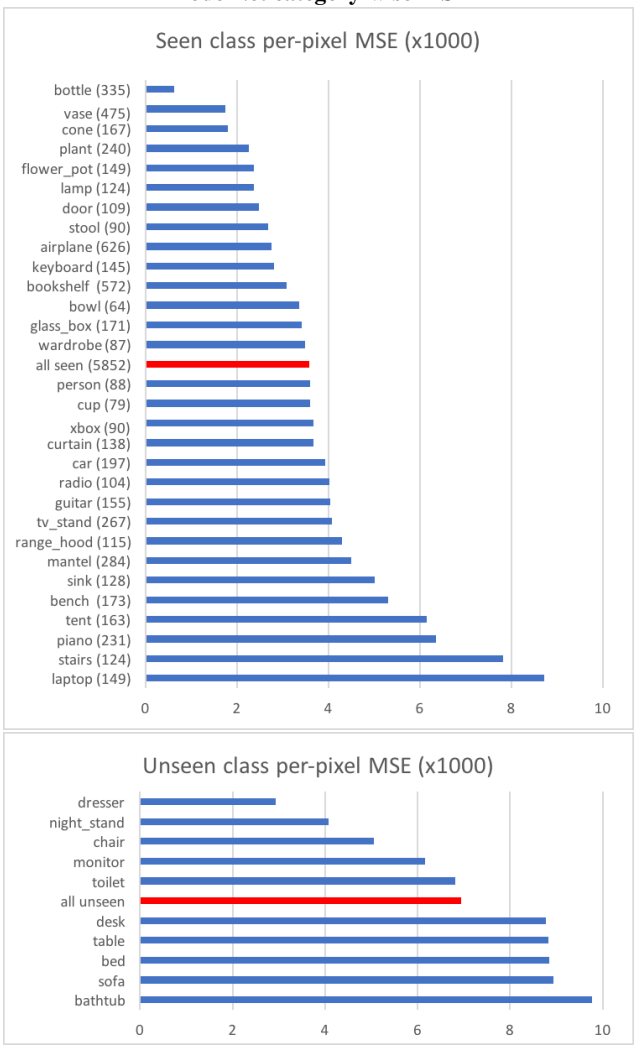


Figure 6. ModelNet seen (top) and unseen (bottom) class reconstruction performance in MSE per class (lower is better). Seen classes are not evenly represented in the training set, so the number of training samples per class is indicated in parentheses next to the corresponding class name. See Sec 6.1.

ShapeNet category-wise MSE

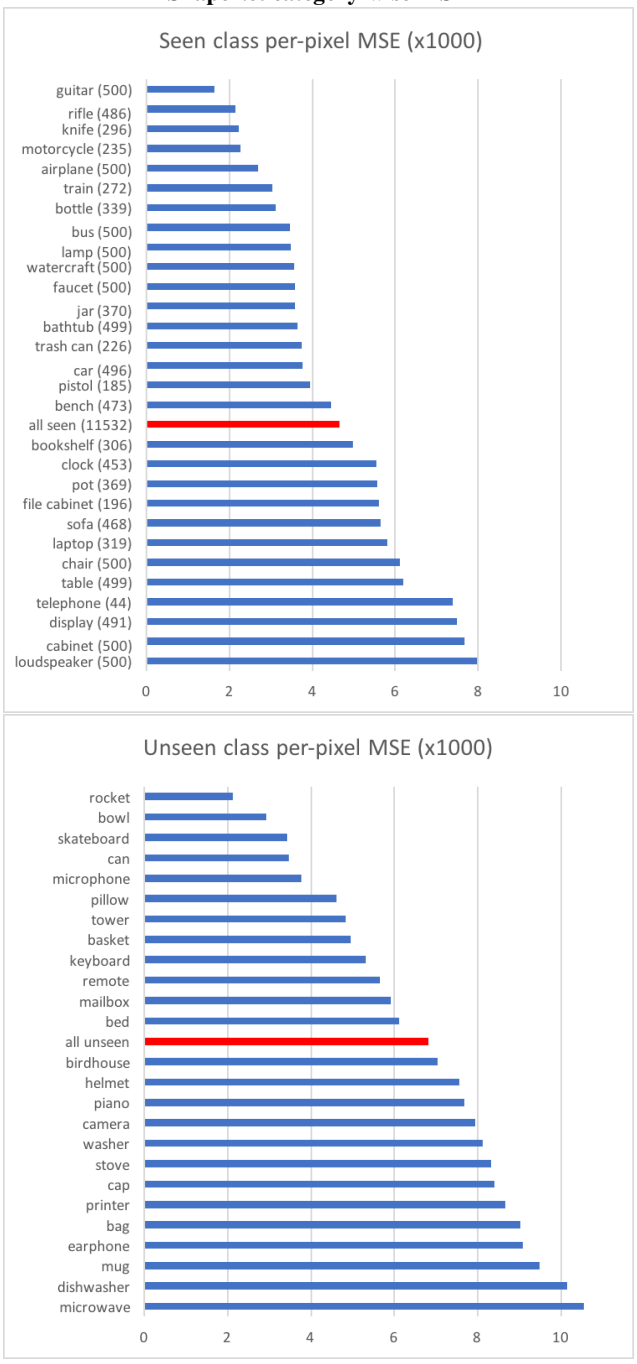
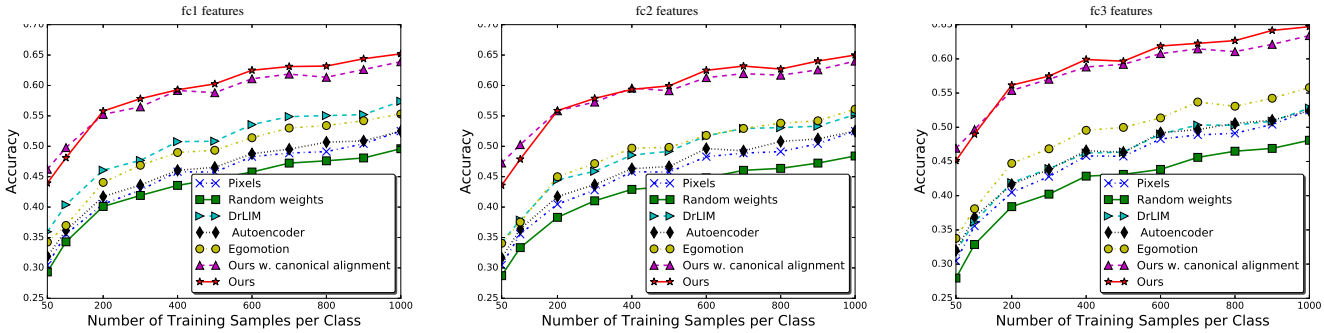
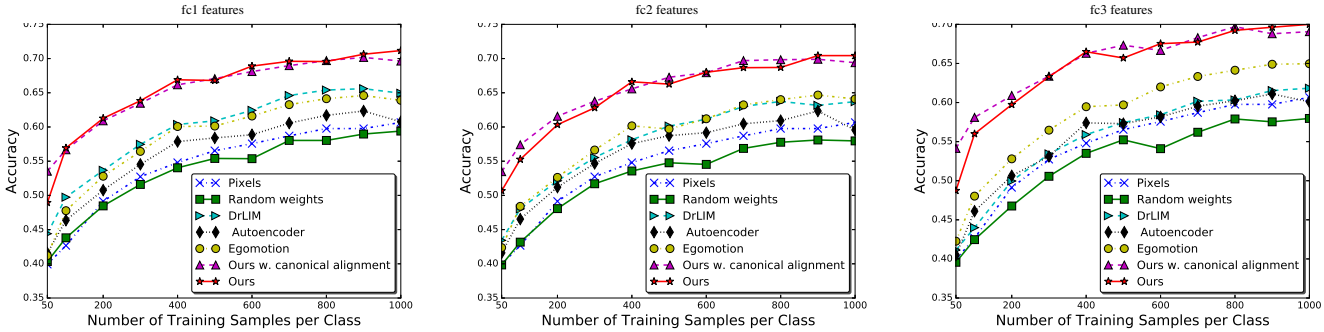


Figure 7. ShapeNet seen (top) and unseen (bottom) class reconstruction performance in MSE per class (lower is better). Seen classes are not evenly represented in the training set, so number of training samples per class are indicated in parentheses next to the class names. See Sec 6.1.

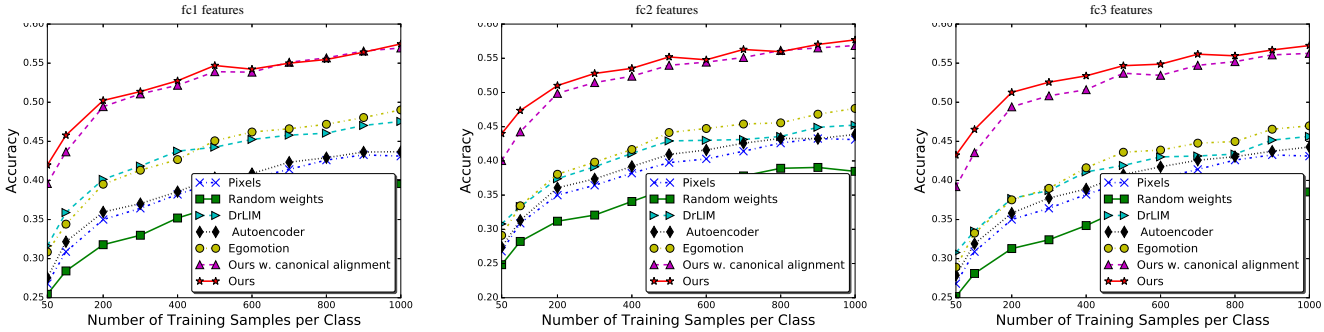
ModelNet seen classes: k-NN classification results with varying training set sizes



ModelNet unseen classes: k-NN classification results with varying training set sizes



ShapeNet seen classes: k-NN classification results with varying training set sizes



ShapeNet unseen classes: k-NN classification results with varying training set sizes

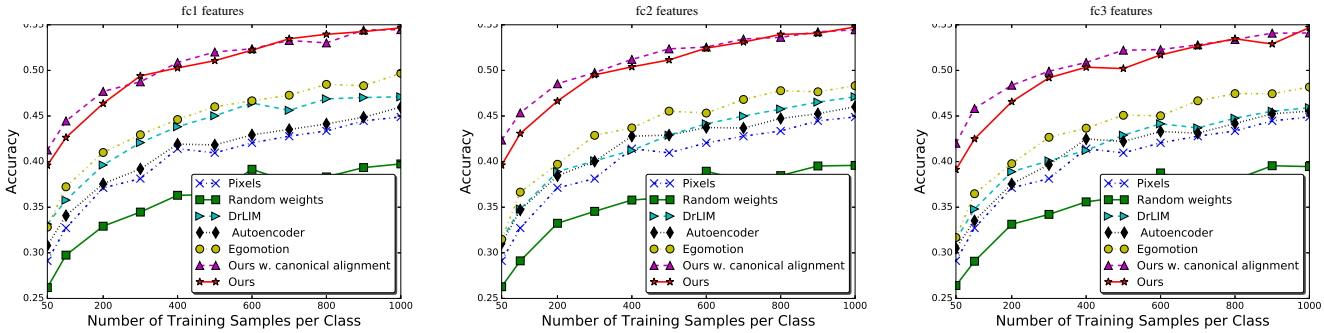


Figure 8. ModelNet and ShapeNet seen and unseen classes k-NN classification, varying training set size. See Sec 6.2.