This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

Multi-view Convolutional Neural Networks for 3D Shape Recognition

Hang Su Subhransu Maji Evangelos Kalogerakis Erik Learned-Miller
University of Massachusetts, Amherst
{hsu,smaji,kalo,elm}@cs.umass.edu

Abstract

A longstanding question in computer vision concerns the representation of 3D shapes for recognition: should 3D shapes be represented with descriptors operating on their native 3D formats, such as voxel grid or polygon mesh, or can they be effectively represented with view-based descriptors? We address this question in the context of learning to recognize 3D shapes from a collection of their rendered views on 2D images. We first present a standard CNN architecture trained to recognize the shapes' rendered views independently of each other, and show that a 3D shape can be recognized even from a single view at an accuracy far higher than using state-of-the-art 3D shape descriptors. Recognition rates further increase when multiple views of the shapes are provided. In addition, we present a novel CNN architecture that combines information from multiple views of a 3D shape into a single and compact shape descriptor offering even better recognition performance. The same architecture can be applied to accurately recognize human hand-drawn sketches of shapes. We conclude that a collection of 2D views can be highly informative for 3D shape recognition and is amenable to emerging CNN architectures and their derivatives.

1. Introduction

One of the fundamental challenges of computer vision is to draw inferences about the three-dimensional (3D) world from two-dimensional (2D) images. Since one seldom has access to 3D object models, one must usually learn to recognize and reason about 3D objects based upon their 2D appearances from various viewpoints. Thus, computer vision researchers have typically developed object recognition algorithms from 2D features of 2D images, and used them to classify new 2D pictures of those objects.

But what if one does have access to 3D models of each object of interest? In this case, one can directly train recognition algorithms on 3D features such as voxel occupancy or surface curvature. The possibility of building such classifiers of 3D shapes directly from 3D representations has recently emerged due to the introduction of large 3D shape repositories, such as 3D Warehouse, TurboSquid, and Shapeways. For example, when Wu et al. [37] introduced the ModelNet 3D shape database, they presented a classifier for 3D shapes using a deep belief network architecture trained on voxel representations.

While intuitively, it seems logical to build 3D shape classifiers directly from 3D models, in this paper we present a seemingly counterintuitive result – that by building classifiers of 3D shapes from 2D image renderings of those shapes, we can actually *dramatically outperform* the classifiers built directly on the 3D representations. In particular, a convolutional neural network (CNN) trained on a fixed set of rendered views of a 3D shape and only provided with a *single* view at test time increases category recognition accuracy by a remarkable 8% (77% $\rightarrow$ 85%) over the best models [37] trained on 3D representations. With more views provided at test time, its performance further increases.

One reason for this result is the relative efficiency of the 2D versus the 3D representations. In particular, while a full resolution 3D representation contains all of the information about an object, in order to use a voxel-based representation in a deep network that can be trained with available samples and in a reasonable amount of time, it would appear that the resolution needs to be significantly reduced. For example, 3D ShapeNets use a coarse representation of shape, a $30\times30\times30$ grid of binary voxels. In contrast a single projection of the 3D model of the same input size corresponds to an image of 164×164 pixels, or slightly smaller if multiple projections are used. Indeed, there is an inherent trade-off between increasing the amount of explicit depth information (3D models) and increasing spatial resolution (projected 2D models).

Another advantage of using 2D representations is that we can leverage (i) advances in image descriptors [22, 26] and (ii) massive image databases (such as ImageNet [9]) to pre-train our CNN architectures. Because images are ubiquitous and large labeled datasets are abundant, we can learn a good deal about generic features for 2D image categorization and then fine-tune to specifics about 3D model projections. While it is possible that some day as much 3D

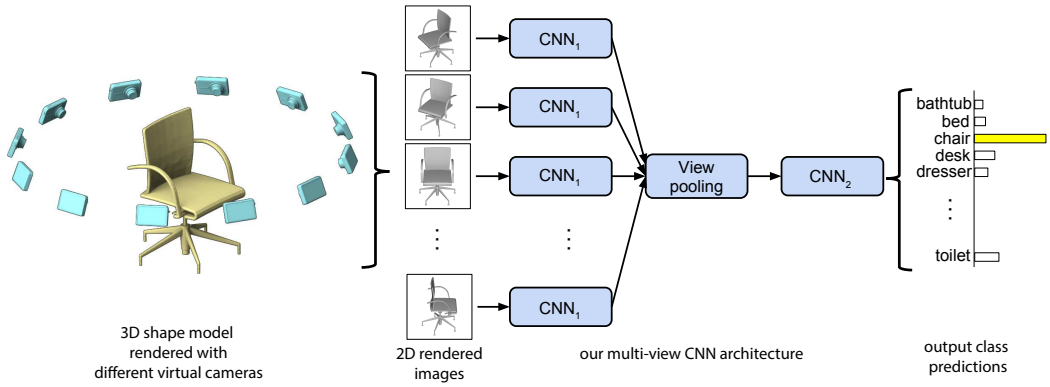

Figure 1. Multi-view CNN for 3D shape recognition (illustrated using the 1st camera setup). At test time a 3D shape is rendered from 12 different views and are passed thorough CNN_1 to extract view based features. These are then pooled across views and passed through CNN_2 to obtain a compact shape descriptor.

training data will be available, for the time being this is a significant advantage of our representation.

Although the simple strategy of classifying views independently works remarkably well (Sect. 3.2), we present new ideas for how to "compile" the information in multiple 2D views of an object into a compact object descriptor using a new architecture called *multi-view CNN* (Fig. 1 and Sect. 3.3). This descriptor is at least as informative for classification (and for retrieval is slightly more informative) than the full collection of view-based descriptors of the object. Moreover it facilitates efficient retrieval using either a similar 3D object or a simple hand-drawn sketch, without resorting to slower methods that are based on pairwise comparisons of image descriptors. We present state-of-the-art results on 3D object classification, 3D object retrieval using 3D objects, and 3D object retrieval using sketches (Sect. 4).

Our multi-view CNN is related to "jittering" where transformed copies of the data are added during training to learn invariances to transformations such as rotation or translation. In the context of 3D recognition the views can be seen as jittered copies. The multi-view CNN learns to combine the views instead of averaging, and thus can use the more informative views of the object for prediction while ignoring others. Our experiments show that this improves performance (Sect. 4.1) and also lets us visualize informative views of the object by back-propagating the gradients of the network to the views (Fig. 3). Even on traditional image classification tasks multi-view CNN can be a better alternative to jittering. For example, on the sketch recognition benchmark [11] a multi-view CNN trained on jittered copies performs better than a standard CNN trained with the same jittered copies (Sect. 4.2).

Pre-trained CNN models, data, and the complete source code to reproduce the results in the paper are available at http://vis-www.cs.umass.edu/mvcnn.

2. Related Work

Our method is related to prior work on shape descriptors for 3D objects and image-based CNNs. Next we discuss representative work in these areas.

Shape descriptors. A large corpus of shape descriptors has been developed for drawing inferences about 3D objects in both the computer vision and graphics literature. Shape descriptors can be classified into two broad categories: *3D shape descriptors* that directly work on the native 3D representations of objects, such as polygon meshes, voxel-based discretizations, point clouds, or implicit surfaces, and *view-based descriptors* that describe the shape of a 3D object by "how it looks" in a collection of 2D projections.

With the exception of the recent work of Wu et al. [37] which learns shape descriptors from the voxel-based representation of an object through 3D convolutional nets, previous 3D shape descriptors were largely "hand-designed" according to a particular geometric property of the shape surface or volume. For example, shapes can be represented with histograms or bag-of-features models constructed out of surface normals and curvatures [15], distances, angles, triangle areas or tetrahedra volumes gathered at sampled surface points [25], properties of spherical functions defined in volumetric grids [16], local shape diameters measured at densely sampled surface points [4], heat kernel signatures on polygon meshes [2, 19], or extensions of the SIFT and SURF feature descriptors to 3D voxel grids [17]. Developing classifiers and other supervised machine learning algorithms on top of such 3D shape descriptors poses a number of challenges. First, the size of organized databases with annotated 3D models is rather limited compared to image datasets, *e.g.*, ModelNet contains about 150K shapes (its 40 category benchmark contains about 4K shapes). In contrast, the ImageNet database [9] already includes tens of millions of annotated images. Second, 3D shape descriptors tend to

be very high-dimensional, making classifiers prone to overfitting due to the so-called 'curse of dimensionality'.

On the other hand view-based descriptors have a number of desirable properties: they are relatively low-dimensional, efficient to evaluate, and robust to 3D shape representation artifacts, such as holes, imperfect polygon mesh tesselations, noisy surfaces. The rendered shape views can also be directly compared with other 2D images, silhouettes or even hand-drawn sketches. An early example of a view-based approach is the work by Murase and Nayar [24] that recognizes objects by matching their appearance in parametric eigenspaces formed by large sets of 2D renderings of 3D models under varying poses and illuminations. Another example, which is particularly popular in computer graphics setups, is the LightField descriptor [5] that extracts a set of geometric and Fourier descriptors from object silhouettes rendered from several different viewpoints. Alternatively, the silhouette of an object can be decomposed into parts and then represented by a directed acyclic graph (shock graph) [23]. Cyr and Kimia [8] defined similarity metrics based on curve matching and grouped similar views, called aspect graphs of 3D models [18]. Eitz et al. [12] compared human sketches with line drawings of 3D models produced from several different views based on local Gabor filters, while Schneider et al. [30] proposed using Fisher vectors [26] on SIFT features [22] for representing human sketches of shapes. These descriptors are largely "hand-engineered" and some do not generalize well across different domains.

Convolutional neural networks. Our work is also related to recent advances in image recognition using CNNs [20]. In particular CNNs trained on the large datasets such as ImageNet have been shown to learn general purpose image descriptors for a number of vision tasks such as object detection, scene recognition, texture recognition and fine-grained classification [10, 13, 28, 7]. We show that these deep architectures can be adapted to specific domains including shaded illustrations of 3D objects, line drawings, and human sketches to produce descriptors that have superior performance compared to other view-based or 3D shape descriptors in a variety of setups. Furthermore, they are compact and efficient to compute. There has been existing work on recognizing 3D objects with CNNs [21] using two concatenated views (binocular images) as input. Our network instead learns a shape representation that aggregates information from any number of input views without any specific ordering, and always outputs a compact shape descriptor of the same size. Furthermore, we leverage both image and shape datasets to train our network.

Although there is significant work on 3D and 2D shape descriptors, and estimating informative views of the objects (or, aspect graphs), there is relatively little work on learning to combine the view-based descriptors for 3D shape recognition. Most methods resort to simple strategies such as performing exhaustive pairwise comparisons of descriptors extracted from different views of each shape, or concatenating descriptors from ordered, consistent views. In contrast our multi-view CNN architecture learns to recognize 3D shapes from views of the shapes using image-based CNNs but in the *context* of other views via a view-pooling layer. As a result, information from multiple views is effectively accumulated into a single, compact shape descriptor.

3. Method

As discussed above, our focus in this paper is on developing view-based descriptors for 3D shapes that are trainable, produce informative representations for recognition and retrieval tasks, and are efficient to compute.

Our view-based representations start from multiple views of a 3D shape, generated by a rendering engine. A simple way to use multiple views is to generate a 2D image descriptor per each view, and then use the individual descriptors directly for recognition tasks based on some voting or alignment scheme. For example, a naïve approach would be to average the individual descriptors, treating all the views as equally important. Alternatively, if the views are rendered in a reproducible order, one could also concatenate the 2D descriptors of all the views. Unfortunately, aligning a 3D shape to a canonical orientation is hard and sometimes ill-defined. In contrast to the above simple approaches, an aggregated representation combining features from multiple views is more desirable since it yields a single, compact descriptor representing the 3D shape.

Our approach is to learn to combine information from multiple views using a unified CNN architecture that includes a view-pooling layer (Fig. 1). All the parameters of our CNN architecture are learned discriminatively to produce a single compact descriptor for the 3D shape. Compared to exhaustive pairwise comparisons between single-view representations of 3D shapes, our resulting descriptors can be directly used to compare 3D shapes leading to significantly higher computational efficiency.

3.1. Input: A Multi-view Representation

3D models in online databases are typically stored as polygon meshes, which are collections of points connected with edges forming faces. To generate rendered views of polygon meshes, we use the Phong reflection model [27]. The mesh polygons are rendered under a perspective projection and the pixel color is determined by interpolating the reflected intensity of the polygon vertices. Shapes are uniformly scaled to fit into the viewing volume.

To create a multi-view shape representation, we need to setup viewpoints (virtual cameras) for rendering each mesh.

We experimented with two camera setups. For **the 1st camera setup**, we assume that the input shapes are upright oriented along a consistent axis (*e.g.*, z-axis). Most models in modern online repositories, such as the 3D Warehouse, satisfy this requirement, and some previous recognition methods also follow the same assumption [37]. In this case, we create 12 rendered views by placing 12 virtual cameras around the mesh every 30 degrees (see Fig. 1). The cameras are elevated 30 degrees from the ground plane, pointing towards the centroid of the mesh. The centroid is calculated as the weighted average of the mesh face centers, where the weights are the face areas. For **the 2nd camera setup**, we do not make use of the assumption about consistent upright orientation of shapes. In this case, we render from several more viewpoints since we do not know beforehand which ones yield good representative views of the object. The renderings are generated by placing 20 virtual cameras at the 20 vertices of an icosahedron enclosing the shape. All cameras point towards the centroid of the mesh. Then we generate 4 rendered views from each camera, using 0, 90, 180, 270 degrees rotation along the axis passing through the camera and the object centroid, yielding total 80 views.

We note that using different shading coefficients or illumination models did not affect our output descriptors due to the invariance of the learned filters to illumination changes, as also observed in image-based CNNs [20, 10]. Adding more or different viewpoints is trivial, however, we found that the above camera setups were already enough to achieve high performance. Finally, rendering each mesh from all the viewpoints takes no more than ten milliseconds on modern graphics hardware.

3.2. Recognition with Multi-view Representations

We claim that our multi-view representation contains rich information about 3D shapes and can be applied to various types of tasks. In the first setting, we make use of existing 2D image features directly and produce a descriptor for each view. This is the most straightforward approach to utilize the multi-view representation. However, it results in multiple 2D image descriptors per 3D shape, one per view, which need to be integrated somehow for recognition tasks.

Image descriptors. We consider two types of image descriptors for each 2D view: a state-of-the-art "hand-crafted" image descriptor based on Fisher vectors [29] with multi-scale SIFT, as well as CNN activation features [10].

The Fisher vector image descriptor is implemented using VLFeat [36]. For each image multi-scale SIFT descriptors are extracted densely. These are then projected to 80 dimensions with PCA, followed by Fisher vector pooling with a Gaussian mixture model with 64 components, square-root and ℓ_2 normalization.

For our CNN features we use the VGG-M network from [3] which consists of mainly five convolutional layers $\text{conv}_{1,\ldots,5}$ followed by three fully connected layers $\text{fc}_{6,\ldots,8}$ and a softmax classification layer. The penultimate layer fc_7 (after ReLU non-linearity, 4096-dimensional) is used as image descriptor. The network is pre-trained on ImageNet images from 1k categories, and then fine-tuned on all 2D views of the 3D shapes in training set. As we show in our experiments, fine-tuning improves performance significantly. Both Fisher vectors and CNN features yield very good performance in classification and retrieval compared with popular 3D shape descriptors (*e.g.*, SPH [16], LFD [5]) as well as 3D ShapeNets [37].

Classification. We train one-vs-rest linear SVMs (each view is treated as a separate training sample) to classify shapes using their image features. At test time, we simply sum up the SVM decision values over all 12 views and return the class with the highest sum. Alternative approaches, *e.g.*, averaging image descriptors, lead to worse accuracy.

Retrieval. A distance or similarity measure is required for retrieval tasks. For shape $\mathbf{x}$ with n_x image descriptors and shape $\mathbf{y}$ with n_y image descriptors, the distance between them is defined in Eq. 1. Note that the distance between two 2D images is defined as the ℓ_2 distance between their feature vectors, *i.e.* $\|\mathbf{x}_i - \mathbf{y}_j\|_2$.

$$d(\mathbf{x}, \mathbf{y}) = \frac{\sum_j \min_i \|\mathbf{x}_i - \mathbf{y}_j\|_2}{2n_y} + \frac{\sum_i \min_j \|\mathbf{x}_i - \mathbf{y}_j\|_2}{2n_x} \tag{1}$$

To interpret this definition, we can first define the distance between a 2D image $\mathbf{x}_i$ and a 3D shape $\mathbf{y}$ as $d(\mathbf{x}_i, \mathbf{y}) = \min_j \|\mathbf{x}_i - \mathbf{y}_j\|_2$. Then given all n_x distances between $\mathbf{x}$'s 2D projections and $\mathbf{y}$, the distance between these two shapes is computed by simple averaging. In Eq. 1, this idea is applied in both directions to ensure symmetry.

We investigated alternative distance measures, such as minimun distance among all $n_x \cdot n_y$ image pairs and the distance between average image descriptors, but they all led to inferior performance.

3.3. Multi-view CNN: Learning to Aggregate Views

Although having multiple separate descriptors for each 3D shape can be successful for classification and retrieval compared to existing 3D descriptors, it can be inconvenient and inefficient in many cases. For example, in Eq. 1, we need to compute all $n_x \times n_y$ pairwise distances between images in order to compute distance between two 3D shapes. Simply averaging or concatenating the image descriptors leads to inferior performance. In this section, we focus on the problem of learning to aggregate multiple views in order

Method	Training Config.			Test Config.	Classification (Accuracy)	Retrieval (mAP)
	Pre-train	Fine-tune	#Views	#Views		
(1) SPH [16]	-	-	-	-	68.2%	33.3%
(2) LFD [5]	-	-	-	-	75.5%	40.9%
(3) 3D ShapeNets [37]	ModelNet40	ModelNet40	-	-	77.3%	49.2%
(4) FV	-	ModelNet40	12	1	78.8%	37.5%
(5) FV, 12×	-	ModelNet40	12	12	84.8%	43.9%
(6) CNN	ImageNet1K	-	-	1	83.0%	44.1%
(7) CNN, f.t.	ImageNet1K	ModelNet40	12	1	85.1%	61.7%
(8) CNN, 12×	ImageNet1K	-	-	12	87.5%	49.6%
(9) CNN, f.t.,12×	ImageNet1K	ModelNet40	12	12	88.6%	62.8%
(10) MVCNN, 12×	ImageNet1K	-	-	12	88.1%	49.4%
(11) MVCNN, f.t., 12×	ImageNet1K	ModelNet40	12	12	89.9%	70.1%
(12) MVCNN, f.t.+metric, 12×	ImageNet1K	ModelNet40	12	12	89.5%	**80.2**%
(13) MVCNN, 80×	ImageNet1K	-	80	80	84.3%	36.8%
(14) MVCNN, f.t., 80×	ImageNet1K	ModelNet40	80	80	**90.1**%	70.4%
(15) MVCNN, f.t.+metric, 80×	ImageNet1K	ModelNet40	80	80	**90.1**%	79.5%

* f.t.=fine-tuning, metric=low-rank Mahalanobis metric learning

Table 1. Classification and retrieval results on the ModelNet40 dataset. On the top are results using state-of-the-art 3D shape descriptors. Our view-based descriptors including Fisher vectors (FV) significantly outperform these even when a single view is available at test time (#Views = 1). When multiple views (#Views=12 or 80) are available at test time, the performance of view-based methods improve significantly. The multi-view CNN (MVCNN) architecture outperforms the view-based methods, especially for retrieval.

to synthesize the information from all views into a single, compact 3D shape descriptor.

We design the multi-view CNN (MVCNN) on top of image-based CNNs (Fig. 1). Each image in a 3D shape's multi-view representation is passed through the first part of the network (CNN_1) separately, aggregated at a view-pooling layer, and then sent through the remaining part of the network (CNN_2). All branches in the first part of the network share the same parameters in CNN_1. We use element-wise maximum operation across the views in the view-pooling layer. An alternative is element-wise mean operation, but it is not as effective in our experiments. The view-pooling layer can be placed anywhere in the network. We show in our experiments that it should be placed close to the last convolutional layer (conv_5) for optimal classification and retrieval performance. View-pooling layers are closely related to max-pooling layers and maxout layers [14], with the only difference being the dimension that their pooling operations are carried out on. The MVCNN is a directed acyclic graphs and can be trained or fine-tuned using stochastic gradient descent with back-propagation.

Using fc_7 (after ReLU non-linearity) in an MVCNN as an aggregated shape descriptor, we achieve higher performance than using separate image descriptors from an image-based CNN directly, especially in retrieval (62.8% $\rightarrow$ 70.1%). Perhaps more importantly, the aggregated descriptor is readily available for a variety of tasks, *e.g.*, shape classification and retrieval, and offers significant speed-ups against multiple image descriptors.

An MVCNN can also be used as a general framework to integrate perturbed image samples (also known as data jittering). We illustrate this capability of MVCNNs in the context of sketch recognition in Sect. 4.2.

Low-rank Mahalanobis metric. Our MVCNN is fine-tuned for classification, and thus retrieval performance is not directly optimized. Although we could train it with a different objective function suitable for retrieval, we found that a simpler approach can readily yield a significant retrieval performance boost (see row 12 in Tab. 1). We learn a Mahalanobis metric W that directly projects MVCNN descriptors $\phi \in \mathbb{R}^d$ to $W\phi \in \mathbb{R}^p$, such that the ℓ_2 distances in the projected space are small between shapes of the same category, and large otherwise. We use the large-margin metric learning algorithm and implementation from [32], with $p < d$ to make the final descriptor compact ($p = 128$ in our experiments). The fact that we can readily use metric learning over the output shape descriptor demonstrates another advantage of using MVCNNs.

4. Experiments

4.1. 3D Shape Classification and Retrieval

We evaluate our shape descriptors on the Princeton ModelNet dataset [1]. ModelNet currently contains 127,915

3D CAD models from 662 categories.[1] A 40-class well-annotated subset containing 12,311 shapes from 40 common categories, ModelNet40, is provided on the ModelNet website. For our experiments, we use the same training and test split of ModelNet40 as in [37].[2]

Our shape descriptors are compared against the 3D ShapeNets by Wu *et al*. [37], the Spherical Harmonics descriptor (SPH) by Kazhdan *et al*. [16], the LightField descriptor (LFD) by Chen *et al*. [5], and Fisher vectors extracted on the same rendered views of the shapes used as input to our networks.

Results on shape classification and retrieval are summarized in Tab. 1. Precision-recall curves are provided in Fig. 2. Remarkably the Fisher vector baseline with just a single view achieves a classification accuracy of 78.8% outperforming the state-of-the-art learned 3D descriptors (77.3% [37]). When all 12 views of the shape are available at test time (based on our first camera setup), we can also average the predictions over these views. Averaging increases the performance of Fisher vectors to 84.8%. The performance of Fisher vectors further supports our claim that 3D objects can be effectively represented using view-based 2D representations. The trends in performance for shape retrieval are similar.

Using our CNN baseline trained on ImageNet in turn outperforms Fisher vectors by a significant margin. Fine-tuning the CNN on the rendered views of the training shapes of ModelNet40 further improves the performance. By using all 12 views of the shape, its classification accuracy reaches 88.6%, and mean average precision (mAP) for retrieval is also improved to 62.8%.

Our MVCNN outperforms all state-of-the-art descriptors as well as the Fisher vector and CNN baselines. With fine-tuning on the ModelNet40 training set, our model achieves 89.9% classification accuracy, and 70.1% mAP on retrieval using the first camera setup. If we do not make use of the assumption about consistent upright orientation of shapes (second camera setup), the performance remains still intact, achieving 90.1% classification accuracy and 70.4% retrieval mAP. MVCNN constitutes an absolute gain of 12.8% in classification accuracy compared to the state-of-the-art learned 3D shape descriptor [37] (77.3% $\rightarrow$ 90.1%). Similarly, retrieval mAP is improved by 21.2% (49.2% $\rightarrow$ 70.4%). Finally, learning a low-rank Mahalanobis metric improves retrieval mAP further while classification accuracy remains almost unchanged, and the resulting shape descriptors become more compact ($d = 4096, p = 128$).

We considered different locations to place the view-pooling layer in the MVCNN. Performance is not very sensitive among the later few layers (conv$_4$$\sim$fc$_7$); however any location prior to conv$_4$ decreases classification accuracies significantly. We find conv$_5$ offers slightly better accuracies ($\sim$1%), and thus use it for all our experiments.

[1] As of 09/24/2015.

[2] Based on our correspondence with the authors of [37], for each category the first 80 shapes in the "train" folder (or all shapes if there are fewer than 80) should be used for training, while the first 20 shapes in the "test" folder should be used for testing.

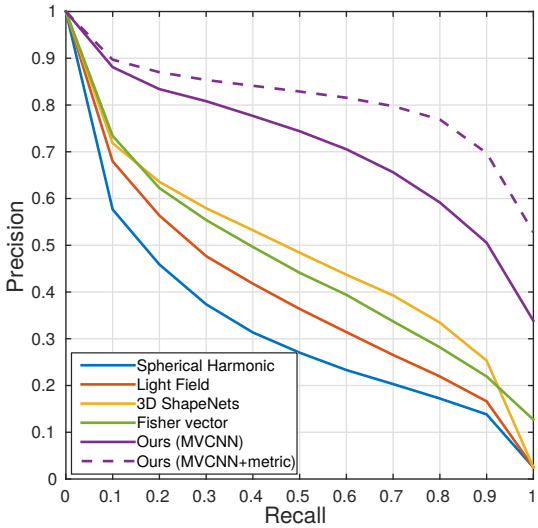

Figure 2. Precision-recall curves for various methods for 3D shape retrieval on the ModelNet40 dataset. Our method significantly outperforms the state-of-the-art on this task achieving 80.2% mAP.

Saliency map among views. For each 3D shape S, our multi-view representation consists of a set of K 2D views $\{I_1, I_2 \dots I_K\}$. We would like to rank pixels in the 2D views w.r.t. their influence on the output score F_c of the network (*e.g*. taken from fc$_8$ layer) for its ground truth class c. Following [33], saliency maps can be defined as the derivatives of F_c w.r.t. the 2D views of the shape:

$$[w_1, w_2 \dots w_K] = \left[\frac{\partial F_c}{\partial I_1}\bigg|_S, \frac{\partial F_c}{\partial I_2}\bigg|_S \dots \frac{\partial F_c}{\partial I_K}\bigg|_S \right] \quad (2)$$

For MVCNN, w in Eq. 2 can be computed using back-propagation with all the network parameters fixed, and can then be rearranged to form saliency maps for individual views. Examples of saliency maps are shown in Fig. 3.

4.2. Sketch Recognition: Jittering Revisited

Given the success of our aggregated descriptors on multiple views of a 3D object, it is logical to ask whether aggregating multiple views of a 2D image could also improve performance. Here we show that this is indeed the case by exploring its connection with data jittering in the context of sketch recognition.

Data jittering, or data augmentation, is a method to generate extra samples from a given image. It is the process of perturbing the image by transformations that change its appearance while leaving the high-level information (class

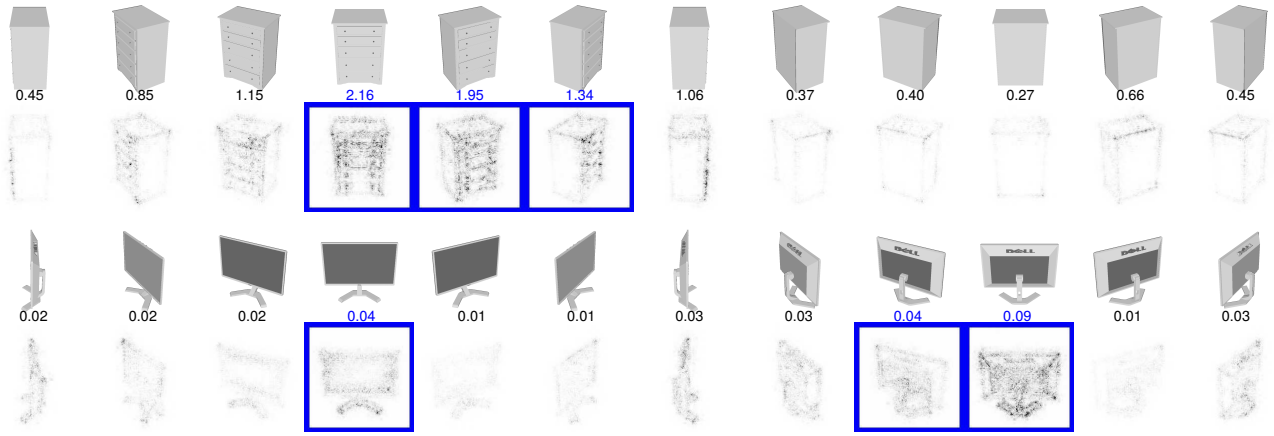

Figure 3. Top three views with the highest saliency are highlighted in blue and the relative magnitudes of gradient energy for each view is shown on top. The saliency maps are computed by back-propagating the gradients of the class score onto the image via the view-pooling layer. Notice that the handles of the dresser and of the desk are the most discriminative features. (Figures are enhanced for visibility.)

label, attributes, *etc*.) intact. Jittering can be applied at training time to augment training samples and to reduce overfitting, or at test time to provide more robust predictions. In particular, several authors [20, 3, 35] have used data jittering to improve the performance of deep representations on 2D image classification tasks. In these applications, jittering at training time usually includes random image translations (implemented as random crops), horizontal reflections, and color perturbations. At test time jittering usually only includes a few crops (*e.g*., four at the corners, one at the center and their horizontal reflections). We now examine whether we can get more benefit out of jittered views of an image by using the same feature aggregation scheme we developed for recognizing 3D shapes.

The human sketch dataset [11] contains 20,000 hand-drawn sketches of 250 object categories such as airplanes, apples, bridges, *etc*. The accuracy of humans in recognizing these hand-drawings is only 73% because of the low quality of some sketches. Schneider and Tuytelaars [30] cleaned up the dataset by removing instances and categories that humans find hard to recognize. This cleaned dataset (SketchClean) contains 160 categories, on which humans can achieve 93% recognition accuracy. Using SIFT Fisher vectors with spatial pyramid pooling and linear SVMs, Schneider and Tuytelaars achieved 68.9% recognition accuracy on the original dataset and 79.0% on the SketchClean dataset. We split the SketchClean dataset randomly into training, validation and test set,[3] and report classification accuracy on the test set in Tab. 2.

With an off-the-shelf CNN (VGG-M from [3]), we are able to get 77.3% classification accuracy without any network fine-tuning. With fine-tuning on the training set, the accuracy can be further improved to 84.0%, significantly surpassing the Fisher vector approach. These numbers are achieved by using the penultimate layer (fc_7) in the network as image descriptors and linear SVMs.

Although it is impractical to get multiple views from 2D images, we can use jittering to mimic the effect of views. For each hand-drawn sketch, we do in-plane rotation with three angles: $-45°$, $0°$, $45°$, and also horizontal reflections (hence 6 samples per image). We apply the two CNN variants (regular CNN and MVCNN) discussed earlier for aggregating multiple views of 3D shapes, and get 85.5% (CNN w/o view-pooling) and 86.3% (MVCNN w/ view-pooling on fc_7) classification accuracy respectively. The latter also has the advantage of a single, more compact descriptor for each hand-drawn sketch.

With a deeper network architecture (VGG-VD, a network with 16 weight layers from [34]), we achieve 87.2% accuracy, further advancing the classification performance, and closely approaching human performance.

4.3. Sketch-based 3D Shape Retrieval

Due to the growing number of online 3D repositories, many approaches have been investigated to perform efficient 3D shape retrieval. Most online repositories (*e.g*. 3D Warehouse, TurboSquid, Shapeways) provide only text-based search engines or hierarchical catalogs for 3D shape retrieval. However, it is hard to convey stylistic and geometric variations using only textual descriptions, thus sketch-based shape retrieval [38, 31, 12] has been proposed as an alternative for users to retrieve shapes with an approximate sketch of the desired 3D shape in mind. Sketch-based retrieval is challenging since it involves two heterogeneous data domains (hand-drawn sketches and 3D shapes), and sketches can be highly abstract and visually different from target 3D shapes. Here we demonstrate the potential strength of MVCNNs in sketch-based shape retrieval.

For this experiment, we construct a dataset containing

[3]The dataset does not come with a standard training/val/test split.

Method	Aug.	Accuracy
(1) FV [30]	-	79.0%
(2) CNN M	-	77.3%
(3) CNN M, fine-tuned	-	84.0%
(4) CNN M, fine-tuned	6×	85.5%
(5) MVCNN M, fine-tuned	6×	86.3%
(6) CNN VD	-	69.3%
(7) CNN VD, fine-tuned	-	86.3%
(8) CNN VD, fine-tuned	6×	86.0%
(9) MVCNN VD, fine-tuned	6×	**87.2%**
(10) Human performance	n/a	93.0%

Table 2. Classification results on SketchClean. Fine-tuned CNN models significantly outperform Fisher vectors [30] by a significant margin. MVCNNs are better than CNN trained with data jittering. The results are shown with two different CNN architectures – VGG-M (row 2-5) and VGG-VD (row 6-9).

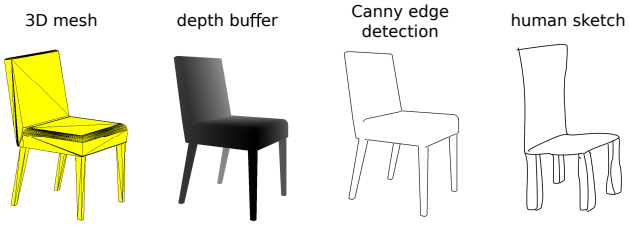

Figure 4. Line-drawing style rendering from 3D shapes.

193 sketches and 790 CAD models from 10 categories existing in both SketchClean and ModelNet40. Following [12], we produce renderings of 3D shapes with a style similar to hand-drawn sketches (see Fig. 4). This is achieved by detecting Canny edges on the depth buffer (also known as z-buffer) from 12 viewpoints. These edge maps are then passed through CNNs to obtain image descriptors. Descriptors are also extracted from 6 perturbed samples of each query sketch in the manner described in Sect. 4.2. Finally we rank 3D shapes w.r.t. "average minimum distance" (Eq. 1) to the sketch descriptors. Representative retrieval results are shown in Fig. 5.

We are able to retrieve 3D objects from the same class with the query sketch, as well as being visually similar, especially in the top few matches. Our performance is 36.1% mAP on this dataset. Here we use the VGG-M network trained on ImageNet without any fine-tuning on either sketches or 3D shapes. With a fine-tuning procedure that optimizes a distance measure between hand-drawn sketches and 3D shapes, *e.g.*, by using a Siamese Network [6], retrieval performance can be further improved.

5. Conclusion

While the world is full of 3D shapes, as humans at least, we understand that world is mostly through 2D images. We have shown that using images of shapes as inputs to modern learning architectures, we can achieve performance better than any previously published results, including those that operate on direct 3D representations of shapes.

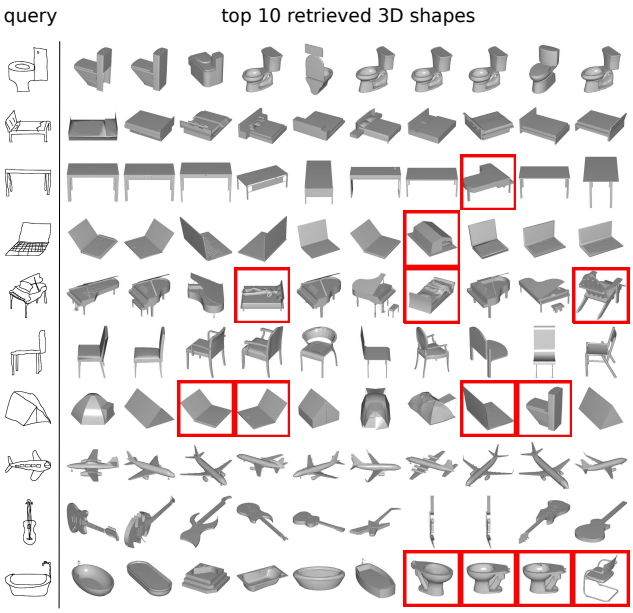

Figure 5. Sketch-based 3D shape retrieval examples. Top matches are shown for each query, with mistakes highlighted in red.

While even a näive usage of these multiple 2D projections yields impressive discrimination performance, by building descriptors that are aggregations of information from multiple views, we can achieve compactness, efficiency, and better accuracy. In addition, by relating the content of 3D shapes to 2D representations like sketches, we can retrieve these 3D shapes at high accuracy using sketches, and leverage the implicit knowledge of 3D shapes contained in their 2D views.

There are a number of directions to explore in future work. One is to experiment with different combinations of 2D views. Which views are most informative? How many views are necessary for a given level of accuracy? Can informative views be selected on the fly?

Another obvious question is whether our view aggregating techniques can be used for building compact and discriminative descriptors for real-world 3D objects from multiple views, or automatically from video, rather than merely for 3D polygon mesh models. Such investigations could be immediately applicable to widely studied problems such as object recognition and face recognition.

Acknowledgements We thank Yanjie Li for her help on rendering meshes. We thank NVIDIA for their generous donation of GPUs used in this research. Our work was partially supported by NSF (CHS-1422441).

References

[1] The Princeton ModelNet. http://modelnet.cs.princeton.edu/. [Online; accessed March 2015]. 5

[2] A. Bronstein, M. Bronstein, M. Ovsjanikov, and L. Guibas. Shape Google: Geometric words and expressions for invariant shape retrieval. *ACM Trans. Graph.*, 30, 2011. 2

[3] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *Proc. BMVC*, 2014. 4, 7

[4] S. Chaudhuri and V. Koltun. Data-driven suggestions for creativity support in 3D modeling. *ACM Trans. Graph.*, 29(6), 2010. 2

[5] D. Chen, X. Tian, Y. Shen, and M. Ouhyoung. On visual similarity based 3D model retrieval. *Proc. Eurographics*, 22(3):223–232, 2003. 3, 4, 5, 6

[6] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proc. CVPR*, 2005. 8

[7] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proc. CVPR*, 2014. 3

[8] C. M. Cyr and B. B. Kimia. A similarity-based aspect-graph approach to 3D object recognition. 57(1), 2004. 3

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. CVPR*, 2009. 1, 2

[10] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. *CoRR*, abs/1310.1531, 2013. 3, 4

[11] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Trans. Graph.*, 31(4):44:1–44:10, 2012. 2, 7

[12] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa. Sketch-based shape retrieval. *ACM Trans. Graph.*, 31(4), 2012. 3, 7, 8

[13] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. CVPR*, 2014. 3

[14] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. *ArXiv e-prints*, Feb. 2013. 5

[15] B. K. P. Horn. Extended gaussian images. *Proc. of the IEEE*, 72(12):1671–1686, 1984. 2

[16] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3D shape descriptors. *Proc. Symposium of Geometry Processing*, 2003. 2, 4, 5, 6

[17] J. Knopp, M. Prasad, G. Willems, R. Timofte, and L. Van Gool. Hough transform and 3D SURF for robust three dimensional classification. In *Proc. ECCV*, 2010. 2

[18] J. J. Koenderink and A. J. Van Doorn. The singularities of the visual mapping. *Biological cybernetics*, 24(1):51–59, 1976. 3

[19] I. Kokkinos, M. Bronstein, R. Litman, and A. Bronstein. Intrinsic shape context descriptors for deformable shapes. In *Proc. CVPR*, 2012. 2

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*. 2012. 3, 4, 7

[21] Y. LeCun, F. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proc. CVPR*, 2004. 3

[22] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. ICCV*, 1999. 1, 3

[23] D. Macrini, A. Shokoufandeh, S. Dickinson, K. Siddiqi, and S. Zucker. View-based 3-D object recognition using shock graphs. In *Proc. ICPR*, volume 3, 2002. 3

[24] H. Murase and S. K. Nayar. Visual learning and recognition of 3-D objects from appearance. 14(1), 1995. 3

[25] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape distributions. *ACM Trans. Graph.*, 21(4), 2002. 2

[26] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *Proc. ECCV*, 2010. 1, 3

[27] B. T. Phong. Illumination for computer generated pictures. *Commun. ACM*, 18(6), 1975. 3

[28] A. S. Razavin, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *DeepVision workshop*, 2014. 3

[29] J. Sanchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with the Fisher vector: Theory and practice. 2013. 4

[30] R. G. Schneider and T. Tuytelaars. Sketch classification and classification-driven analysis using Fisher vectors. *ACM Trans. Graph.*, 33(6):174:1–174:9, Nov. 2014. 3, 7, 8

[31] T. Shao, W. Xu, K. Yin, J. Wang, K. Zhou, and B. Guo. Discriminative sketch-based 3D model retrieval via robust shape matching. In *Computer Graphics Forum*, volume 30. Wiley Online Library, 2011. 7

[32] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher vector faces in the wild. In *Proc. BMVC*, 2013. 5

[33] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013. 6

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 7

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 7

[36] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008. 4

[37] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D ShapeNets: A deep representation for volumetric shape modeling. In *Proc. CVPR*, 2015. 1, 2, 4, 5, 6

[38] S. M. Yoon, M. Scherer, T. Schreck, and A. Kuijper. Sketch-based 3D model retrieval using diffusion tensor fields of suggestive contours. In *Proc. International Conference on Multimedia*, 2010. 7