RotationNet for Joint Object Categorization and Unsupervised Pose Estimation from Multi-View Images

Asako Kanezaki, *Member, IEEE*, Yasuyuki Matsushita, *Senior Member, IEEE*, and Yoshifumi Nishida, *Member, IEEE*

Abstract—We propose a Convolutional Neural Network (CNN)-based model "RotationNet," which takes multi-view images of an object as input and jointly estimates its pose and object category. Unlike previous approaches that use known viewpoint labels for training, our method treats the viewpoint labels as latent variables, which are learned in an unsupervised manner during the training using an unaligned object dataset. RotationNet uses only a partial set of multi-view images for inference, and this property makes it useful in practical scenarios where only partial views are available. Moreover, our pose alignment strategy enables one to obtain view-specific feature representations shared across classes, which is important to maintain high accuracy in both object categorization and pose estimation. Effectiveness of RotationNet is demonstrated by its superior performance to the state-of-the-art methods of 3D object classification on 10- and 40-class ModelNet datasets. We also show that RotationNet, even trained without known poses, achieves comparable performance to the state-of-the-art methods on an object pose estimation dataset. Furthermore, our object ranking method based on classification by RotationNet achieved the first prize in two tracks of the 3D Shape Retrieval Contest (SHREC) 2017. Finally, we demonstrate the performance of real-world applications of RotationNet trained with our newly created multi-view image dataset using a moving USB camera.

Index Terms—Object recognition, 3D shape retrieval, viewpoint estimation, multi-task learning, convolutional neural network

1 Introduction

Object classification accuracy can be enhanced by the use of multiple different views of a target object [3], [25]. Recent remarkable advances in image recognition and collections of 3D object models enabled learning of multi-view representations of objects in various categories. However, in real-world scenarios, objects can often only be observed from limited viewpoints due to occlusions, which makes it difficult to rely on multi-view representations that are learned with the whole circumference. The desired property for the real-world object classification is that, when a viewer observes a partial subset (≥ 1 images) of the full multi-view images of an object, it should recognize from which directions it observes the target object to correctly infer the category of the object. It has been understood that if the viewpoint is known the object classification accuracy can be improved. Likewise, if the object category is known, that helps infer the viewpoint. As such, object classification and viewpoint estimation is a tightly coupled problem, which can benefit from their joint estimation.

We propose a new Convolutional Neural Network (CNN) model that we call *RotationNet*, which takes multi-view images of an object as input and predicts its pose and object category (Fig. 1). RotationNet outputs viewpoint-specific category likelihoods corresponding to all pre-defined discrete viewpoints for each image input, and then selects the object pose that maximizes the integrated object category likelihood. Whereas, at the training phase, RotationNet uses a complete set of multi-view images of an object captured from all the pre-defined viewpoints, for inference it is able to work with only a partial set of all the multi-view images – a single image at minimum – as input. In addition, RotationNet does not require the multi-view images to be provided at once but allows their sequential input and updates of the target object's category likelihood. This property is suitable for applications that require on-the-fly classification with a moving camera.

The most representative feature of RotationNet is that it treats viewpoints where training images are observed as *latent* variables during the training (Fig. 3). This enables unsupervised learning of object poses using an unaligned object dataset; thus, it eliminates the need of preprocessing for pose normalization that is often sensitive to noise and individual shape differences. Our method automatically determines the basis axes of objects based on their appearance during the training and achieves not only intra-class but also inter-class object pose alignment (Fig. 2). Inter-class

- *A. Kanezaki and Y. Nishida are with the National Institute of Advanced Industrial Science and Technology (AIST)2-4-7, Aomi, Koto-ku, Tokyo 135-0064, Japan. E-mail: {kanezaki.asako, y.nishida}@aist.go.jp, yasumat@ist.osaka-u.ac.jp.*
- *Y. Matsushita is with the Osaka University1-5, Yamadaoka, Suita-shi, Osaka 565-0871, Japan.*

Manuscript received 18 May 2018; revised 18 Apr. 2019; accepted 7 June 2019. Date of publication 14 June 2019; date of current version 3 Dec. 2020. (Corresponding author: Asako Kanezaki.)
Recommended for acceptance by B. Ommer.
Digital Object Identifier no. 10.1109/TPAMI.2019.2922640

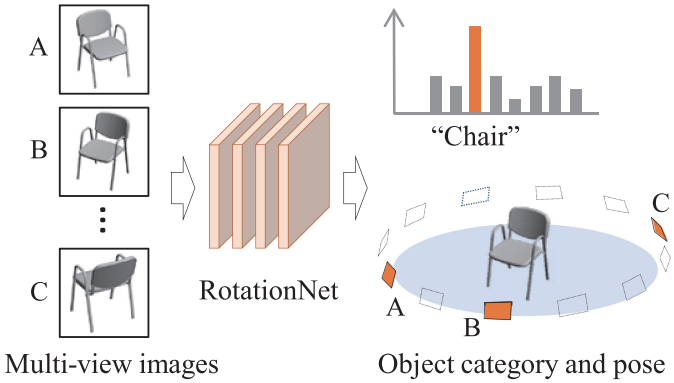


Fig. 1. Illustration of the proposed method *RotationNet*. RotationNet takes a partial set (≥ 1 images) of the full multi-view images of an object as input and predicts its object category by rotation, where the best pose is selected to maximize the object category likelihood. Here, viewpoints from which the images are observed are jointly estimated to predict the pose of the object.

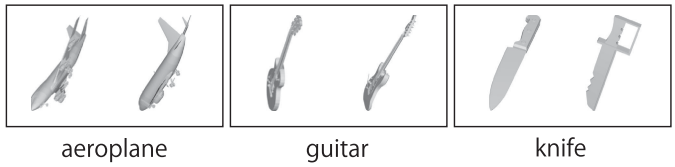


Fig. 2. Illustration of inter-class alignment of 3D models. Objects are placed in the direction of ever-increasing appearance. Inter-class alignment enables comparison among different categories in the pose where image appearance is similar to each other, which emphasizes differences in different categories.

pose alignment is important to deal with joint learning of object pose and category, because the importance of object classification lies in emphasizing differences in different categories when their appearances are similar. Without inter-class pose alignment, it may become an ill-posed problem to obtain a model to distinguish, e.g., a car and a bus if the side view of a car is compared with the frontal view of a bus.

Our main contributions are described as follows. We first show that RotationNet outperforms the current state-of-the-art classification performance on 3D object benchmark datasets consisting of 10- and 40-categories by a large margin (Table 5). Next, even though it is trained without the ground-truth poses, RotationNet achieves superior performance to previous works on an object pose estimation dataset. We also show that our model generalizes well to a real-world image dataset that was newly created for the general task of multi-view object classification. Furthermore, RotationNet shows outstanding performance on the 3D object retrieval task, both with CAD models and RGB-D scans of real objects, which are demonstrated on the 3D Shape Retrieval Contest (SHREC) 2017. Finally, we train RotationNet with the new dataset named MIRO[1] and demonstrate the performance of real-world applications using a moving USB camera.

2 Related Work

There are two main approaches for the CNN-based 3D object classification: voxel-based and 2D image-based approaches. The earliest work on the former approach is 3D ShapeNets [42], which learns a Convolutional Deep Belief Network that outputs probability distributions of binary occupancy voxel values. Latest works on similar approaches showcased improved performance [22], [23], [41]. Even when working with 3D objects, 2D image-based approaches are shown effective for general object recognition tasks. Su et al. [36] proposed multi-view CNN (MVCNN), which takes multi-view images of an object captured from surrounding virtual cameras as input and outputs the object's category label. Multi-view representations are also used for 3D shape retrieval [1]. Qi et al. [27] gives a comprehensive study on the voxel-based CNNs and multi-view CNNs for 3D object classification. Other than those above, point-based approach [12], [17], [26] is recently drawing much attention; however, the performance on 3D object classification is yet inferior to those of multi-view approaches.

Because MVCNN integrates multi-views in a view-pooling layer which lies in the middle of the CNN, it requires a complete set of multi-view images recorded from all the pre-defined viewpoints for object inference. Unlike MVCNN, our method is able to classify an object using a *partial* set of multi-view images that may be sequentially observed by a moving camera. Elhoseiny et al. [8] explored CNN architectures for joint object classification and pose estimation learned with multi-view images. Whereas their method takes a single image as input for its prediction, we mainly focus on how to aggregate predictions from multiple images captured from different viewpoints.

Viewpoint estimation is significant in its role in improving object classification. Better performance was achieved on face identification [49], human action classification [7], and image retrieval [38] by generating unseen views after observing a single view. These methods "imagine" the appearance of objects' unobserved profiles, which is innately more uncertain than using real observations. Sedaghat et al. [31] proposed a voxel-based CNN that outputs orientation labels as well as classification labels and demonstrated that it improved 3D object classification performance.

All the methods mentioned above assume known poses in training samples; however, object poses are not always aligned in existing object databases. Novotny et al. [24] proposed a viewpoint factorization network that utilizes relative pose changes within each sequence to align objects in videos in an unsupervised manner. Our method also aligns object poses via unsupervised viewpoint estimation, where viewpoints of images are treated as *latent* variables during the training. Here, viewpoint estimation is learned in an unsupervised manner to best promote the object categorization task. In such a perspective, our method is related to Zhou et al. [48], where view synthesis is trained as the "meta"-task to train multi-view pose networks by utilizing the synthesized views as the supervisory signal.

Although joint learning of object classification and pose estimation has been widely studied [2], [21], [29], [37], [46], inter-class pose alignment has drawn little attention. However, it is beneficial to share view-specific appearance information across classes to simultaneously solve for object

1. The dataset is publicly available on https://github.com/kanezaki/MIRO.

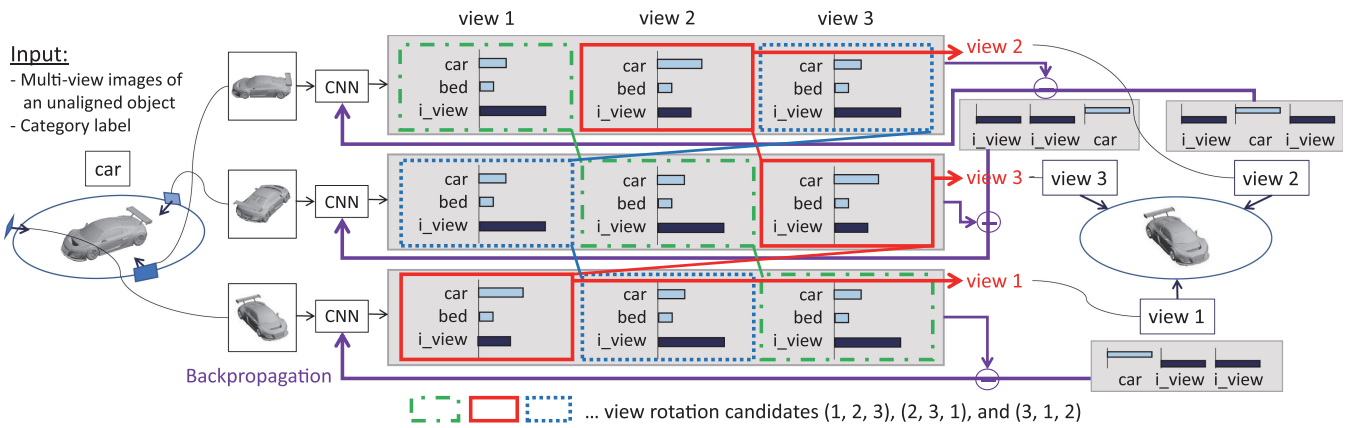


Fig. 3. Illustration of the training process of RotationNet, where the number of views M is 3 and the number of categories N is 2. A training sample consists of M images of an unaligned object and its category label y. For each input image, our CNN (RotationNet) outputs M histograms with $N+1$ bins whose norm is 1. The last bin of each histogram represents the "incorrect view" class, which serves as a weight of how likely the histogram does not correspond to each viewpoint variable. According to the histogram values, we decide which image corresponds to views 1, 2, and 3. There are three candidates for view rotation: (1, 2, 3), (2, 3, 1), and (3, 1, 2). For each candidate, we calculate the score for the ground-truth category ("car" in this case) by multiplying the histograms and selecting the best choice: (2, 3, 1) in this case. Finally, we update the CNN parameters in a standard back-propagation manner with the estimated viewpoint variables. Note that it is the same CNN that is being used.

classification and pose estimation. Kuznetsova et al. [19] pointed out this issue and presented a metric learning approach that shares visual components across categories for simultaneous pose estimation and class prediction. Our method also uses a model with view-specific appearances that are shared across classes; thus, it is able to maintain high accuracy for both object classification and pose estimation.

3 Proposed Method

3.1 Training and Inference

The training process of RotationNet is illustrated in Fig. 3. We assume that multi-view images of each training object instance are observed from all the pre-defined viewpoints. Let M be the number of the pre-defined viewpoints and N denote the number of target object categories. A training sample consists of M images of an object $\{\boldsymbol{x}_i\}_{i=1}^{M}$ and its category label $y \in \{1, \ldots, N\}$. We attach a viewpoint variable $v_i \in \{1, \ldots, M\}$ to each image $\boldsymbol{x}_i$ and set it to j when the image is observed from the jth viewpoint, i.e., $v_i \leftarrow j$. In our method, only the category label y is given during the training whereas the viewpoint variables $\{v_i\}$ are *unknown*, namely, $\{v_i\}$ are *treated as latent variables that are optimized in the training process*.

RotationNet is defined as a differentiable multi-layer neural network $R(\cdot)$. The final layer of RotationNet is the concatenation of M softmax layers, each of which outputs the category likelihood $P(\hat{y}_i \,|\, \boldsymbol{x}_i, v_i = j)$ where $j \in \{1, \ldots, M\}$ for each image $\boldsymbol{x}_i$. Here, $\hat{y}_i$ denotes an estimate of the object category label for $\boldsymbol{x}_i$. For the training of RotationNet, we input the set of images $\{\boldsymbol{x}_i\}_{i=1}^{M}$ simultaneously and solve the following optimization problem:

$$\max_{R, \{v_i\}_{i=1}^{M}} \prod_{i=1}^{M} P(\hat{y}_i = y \,|\, \boldsymbol{x}_i, v_i). \tag{1}$$

The parameters of R and latent variables $\{v_i\}_{i=1}^{M}$ are optimized to output the highest probability of y for the input of multi-view images $\{\boldsymbol{x}_i\}_{i=1}^{M}$.

Now, we describe how we design $P(\hat{y}_i \,|\, \boldsymbol{x}_i, v_i)$ outputs. First of all, the category likelihood $P(\hat{y}_i = y \,|\, \boldsymbol{x}_i, v_i)$ should become close to one when the estimated v_i is correct; in other words, the image $\boldsymbol{x}_i$ is truly captured from the v_ith viewpoint. Otherwise, in the case that the estimated v_i is incorrect, $P(\hat{y}_i = y \,|\, \boldsymbol{x}_i, v_i)$ may not necessarily be high because the image $\boldsymbol{x}_i$ is captured from a different viewpoint. As described above, we decide the viewpoint variables $\{v_i\}_{i=1}^{M}$ according to the $P(\hat{y}_i = y \,|\, \boldsymbol{x}_i, v_i)$ outputs as in (1). In order to obtain a stable solution of $\{v_i\}_{i=1}^{M}$ in (1), we introduce an "incorrect view" class and append it to the target category classes. Here, the "incorrect view" class plays a similar role to the "background" class for object detection tasks, which represents negative samples that belong to a "non-target" class. Then, RotationNet calculates $P(\hat{y}_i \,|\, \boldsymbol{x}_i, v_i)$ by applying softmax functions to the $(N+1)$-dimensional outputs, where $\sum_{\hat{y}_i=1}^{N+1} P(\hat{y}_i \,|\, \boldsymbol{x}_i, v_i) = 1$. Note that $P(\hat{y}_i = N+1 \,|\, \boldsymbol{x}_i, v_i)$, which corresponds to the probability that the image $\boldsymbol{x}_i$ belongs to the "incorrect view" class for the v_ith viewpoint, indicates how likely it is that the estimated viewpoint variable v_i is incorrect.

Based on the above discussion, we substantiate (1) as follows. Letting $P_i = \left[p_{j,k}^{(i)}\right] \in \mathbb{R}_{+}^{M \times (N+1)}$ denote a matrix composed of $P(\hat{y}_i \,|\, \boldsymbol{x}_i, v_i)$ for all the M viewpoints and $N+1$ classes, the target value of P_i in the case that v_i is correctly estimated is defined as follows:

$$p_{j,k}^{(i)} = \begin{cases} 1 & (j = v_i \text{ and } k = y) \text{ or } (j \neq v_i \text{ and } k = N+1) \\ 0 & (\text{otherwise}). \end{cases} \tag{2}$$

In this way, (1) can be rewritten as the following cross-entropy optimization problem:

$$\max_{R, \{v_i\}_{i=1}^{M}} \sum_{i=1}^{M} \left(\log p_{v_i,y}^{(i)} + \sum_{j \neq v_i} \log p_{j,N+1}^{(i)} \right). \tag{3}$$

If we fix $\{v_i\}_{i=1}^{M}$ here, the above can be written as a subproblem of optimizing R as follows:

$$\max_{R} \sum_{i=1}^{M} \left(\log p^{(i)}_{v_i,y} + \sum_{j \neq v_i} \log p^{(i)}_{j,N+1} \right), \tag{4}$$

where the parameters of R can be iteratively updated via standard back-propagation of M softmax losses. Since $\{v_i\}_{i=1}^{M}$ are not constant but latent variables that need to be optimized during the training of R, we employ alternating optimization of R and $\{v_i\}_{i=1}^{M}$. More specifically, in every iteration, our method determines $\{v_i\}_{i=1}^{M}$ according to P_i obtained via forwarding of (fixed) R, and then update R according to the estimated $\{v_i\}_{i=1}^{M}$ by fixing them.

The latent viewpoint variables $\{v_i\}_{i=1}^{M}$ are determined by solving the following problem:

$$\begin{aligned} &\max_{\{v_i\}_{i=1}^{M}} \sum_{i=1}^{M} \left(\log p^{(i)}_{v_i,y} + \sum_{j \neq v_i} \log p^{(i)}_{j,N+1} \right) \\ &= \max_{\{v_i\}_{i=1}^{M}} \sum_{i=1}^{M} \left(\log p^{(i)}_{v_i,y} + \sum_{j=1}^{M} \log p^{(i)}_{j,N+1} - \log p^{(i)}_{v_i,N+1} \right) \\ &= \max_{\{v_i\}_{i=1}^{M}} \prod_{i=1}^{M} \frac{p^{(i)}_{v_i,y}}{p^{(i)}_{v_i,N+1}}, \end{aligned} \tag{5}$$

in which the conversion used the fact that $\sum_{j=1}^{M} \log p^{(i)}_{j,N+1}$ is constant w.r.t. $\{v_i\}_{i=1}^{M}$. Because the number of candidates for $\{v_i\}_{i=1}^{M}$ is limited, we calculate the evaluation value of (5) for all the candidates and take the best choice. The decision of $\{v_i\}_{i=1}^{M}$ in this way emphasizes view-specific features for object categorization, which contributes to the self-alignment of objects in the dataset.

In the inference phase, RotationNet takes as input M' $(1 \leq M' \leq M)$ images of a test object instance, either simultaneously or sequentially, and outputs M' probabilities. Finally, it integrates the M' outputs to estimate the category of the object and the viewpoint variables as follows:

$$\left\{ \hat{y}, \{\hat{v}_i\}_{i=1}^{M'} \right\} = \arg\max_{y, \{v_i\}_{i=1}^{M'}} \prod_{i=1}^{M'} \frac{p^{(i)}_{v_i,y}}{p^{(i)}_{v_i,N+1}}. \tag{6}$$

Similarly to the training phase, we decide $\{\hat{v}_i\}_{i=1}^{M'}$ according to the outputs $\{P_i\}_{i=1}^{M'}$. Thus RotationNet is able to estimate the pose of the object as well as its category label.

3.2 Viewpoint Setups for Training

While choices of the viewpoint variables $\{v_i\}_{i=1}^{M'}$ can be arbitrary, we consider two setups in this paper, with and without an upright orientation assumption, similarly to MVCNN [36]. The former case is often useful with images of real objects captured with one-dimensional turning tables, whereas the latter case is rather suitable for unaligned 3D models. We also consider the third case that is also based on the upright orientation assumption (as the first case) but with multiple elevation levels. We illustrate the three viewpoint setups in Fig. 4.

3.2.1 Case (i): With Upright Orientation

In the case where we assume upright orientation, we fix a specific axis as the rotation axis (e.g., the z-axis), which defines the upright orientation, and then place viewpoints at intervals of the angle θ around the axis, elevated by ϕ (set to 30 degree in this paper) from the ground plane. We set $\theta = 30^\circ$ by default, which yields 12 views for an object ($M = 12$). We define that "view $m+1$" is obtained by rotating the view position "view m" by the angle θ about the z-axis. Note that the view obtained by rotating "view M" by the angle θ about the z-axis corresponds to "view 1." We assume the sequence of input images is consistent with respect to a certain direction of rotation in the training phase. For instance, if v_i is m $(m < M)$, then v_{i+1} is $m+1$. Thus the number of candidates for all the viewpoint variables $\{v_i\}_{i=1}^{M}$ is M.

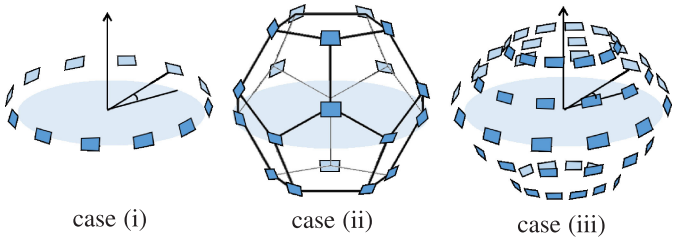


Fig. 4. Illustration of three viewpoint setups considered in this work. A target object is placed on the center of each circle.

3.2.2 Case (ii): W/o Upright Orientation

In the case where we do not assume upright orientation, we place virtual cameras on the $M = 20$ vertices of a dodecahedron encompassing the object. This is because a dodecahedron has the largest number of vertices among regular polyhedra, where viewpoints can be completely equally distributed in 3D space. Unlike case (i), where there is a unique rotation direction, there are three different patterns of rotation from a certain view, because three edges are connected to each vertex of a dodecahedron. Therefore, the number of candidates for all the viewpoint variables $\{v_i\}_{i=1}^{M}$ is 60 $(= 3M)$.[2]

3.2.3 Case (iii): With Upright Orientation and Multiple Elevation Levels

This case is an extension of case (i). Unlike case (i) where the elevation angle is fixed, we place virtual cameras at intervals of ϕ in $[-90^\circ, 90^\circ]$. There are $M = M_a \times M_e$ viewpoints, where $M_a = \frac{360^\circ}{\theta}$ and $M_e = \frac{180^\circ}{\phi} + 1$. As with the case (i), the number of candidates for all the viewpoint variables $\{v_i\}_{i=1}^{M}$ is M_a due to the upright orientation assumption.

3.3 Self-Alignment During Training

The most characteristic property of RotationNet is the ability to align objects in datasets during its training. We consider the situation that the reference axis of one object instance in a training dataset differs from another. In most cases, the basis axes of 3D models in such datasets are calculated using techniques such as principal component analysis (PCA), which is known to be sensitive to noise and individual differences in shapes. In contrast, we decide the basis axes of objects based on their appearance in an *unsupervised* manner rather than calculating them during preprocessing. This self-alignment phenomenon is achieved in (5) where the best choice of viewpoint variables $\{v_i\}_{i=1}^{M}$ are

2. A dodecahedron has 60 orientation-preserving symmetries.

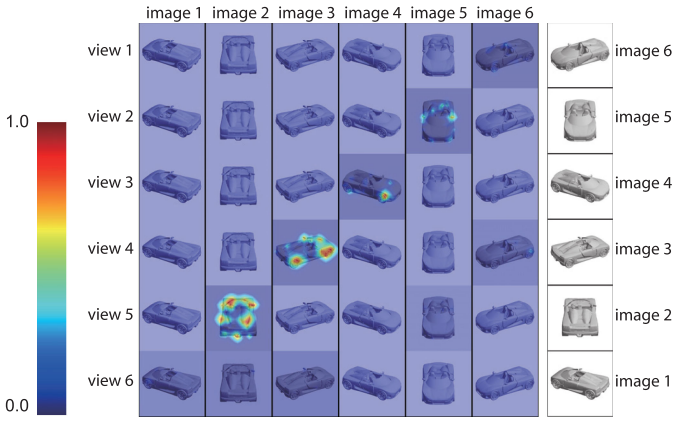


Fig. 5. Visualization of responses to each viewpoint. Images with the maximum responses to each viewpoint are shown right.

determined based on object categorization performance. Intuitively, an object instance in inference is rotated to the pose where it appears most likely that the object is correctly categorized. In this manner, after several iterations of forward and backward process of RotationNet updates, it achieves view-wise image representations that are reactive to specific object pose. Fig. 5 shows important areas (responses) in test images of a car corresponding to each viewpoint, where we set M to 6 in case (i) in this example. We used the interpretable explanations technique [11] to visualize the responses of RotationNet to each viewpoint. A high response value indicates a high probability that the image belongs to a car category and is observed from the respective viewpoint. Note that the response of some images (image #1 and image # 6, in this case) are less salient than other images. This explains our motivation to use multi-view images for improving object categorization and pose estimation.

Figs. 6 and 7 show the state transition of the inter- and intra-class object pose alignment that is automatically achieved during the training of RotationNet with ModelNet40 [42]. They depict the variation of the average images generated by concatenating multi-view images in order of their predicted viewpoint variables. The figures correspond to the cases (i): with upright orientation and (ii): w/o upright orientation, respectively. We can see that the variance of average images decreases together with the variance of object poses. The red dotted lines show the mean variance of average images of all the 40 classes, whereas the blue lines show the variance of average images of the "chair" class.

The images of test object instances in the "chair" class with the same predicted viewpoint variable are shown in the right of the figures. In both cases (i) and (ii), where the latter case is more interesting because of its difficulty, the chairs with initial random poses gradually get aligned in the same direction after several hundreds of training iterations. Moreover, the average images of all the 40 classes with the same predicted viewpoint variable shown in red boxes indicate that not only the intra-class alignment but also the inter-class alignment is achieved. The alignment is less obvious in the red boxes of Fig. 7; however, it is confirmed that this does not harm the object classification accuracy.

3.4 Viewpoint Augmentation

Similar to MVCNN [36] and any other 3D object classification method that considers discrete variance of rotation, RotationNet has the limitation that each image should be observed from one of the pre-defined viewpoints. In other words, RotationNet does not guarantee to classify images captured from a novel viewpoint, correctly. In order to mitigate the limitation, we introduce viewpoint augmentation either or both in the training and inference phases. This is available either with 3D object models or densely captured continuous images (e.g., video frames). We prepare multiple sets of multi-view images captured with different camera system orientations. For the viewpoint augmentation in training, we train a single RotationNet model with a mixture of multiple sets of multi-view images captured with different camera system orientation. For the viewpoint augmentation in inference, we apply the same RotationNet model to all the sets of multi-view test images captured

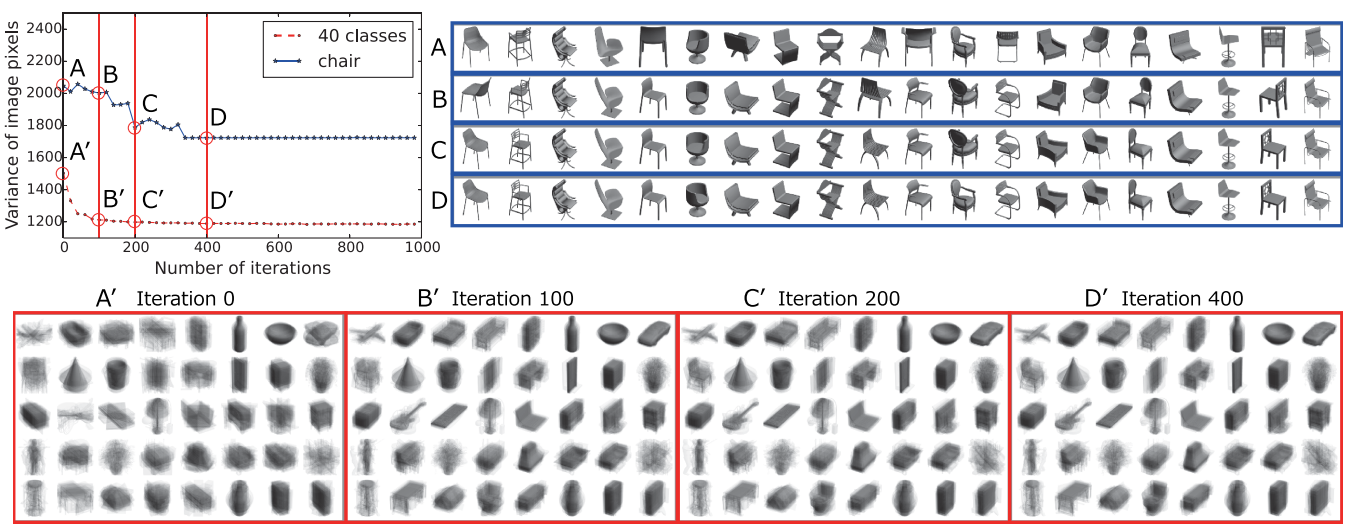


Fig. 6. Variance change of the average images generated by concatenating multi-view images (case (i)) in order of their predicted viewpoint variables. Average images of 40 classes and images of the "chair" class with the same predicted viewpoint variable in iterations 0, 100, 200, and 400 are shown in red boxes and blue boxes, respectively.

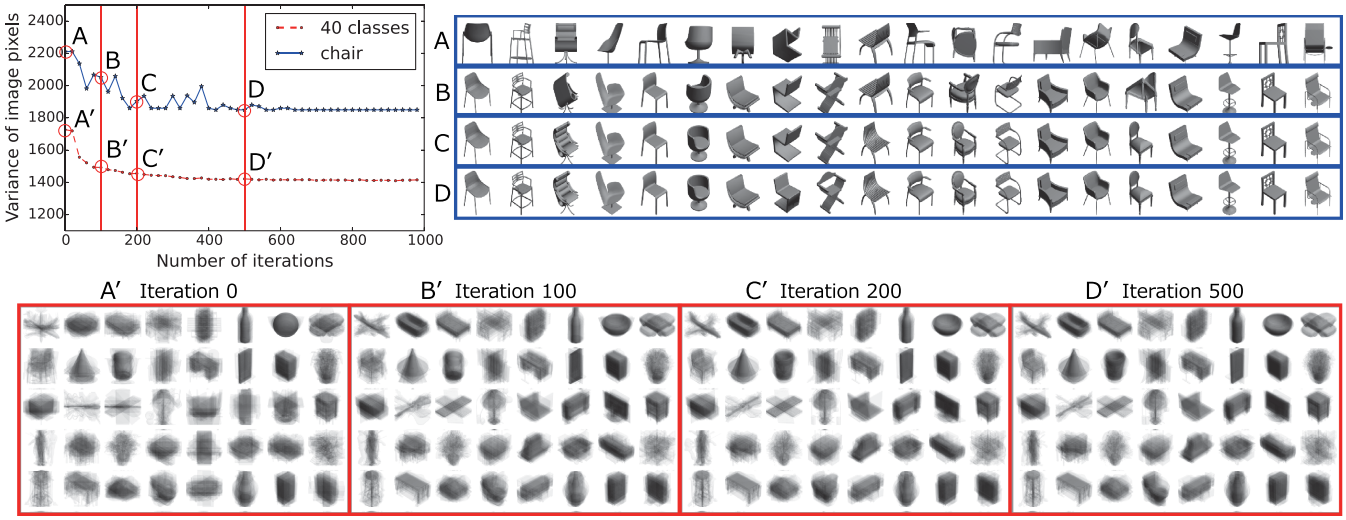


Fig. 7. Variance change of the average images generated by concatenating multi-view images (case (ii)) in order of their predicted viewpoint variables. Average images of 40 classes and images of the "chair" class with the same predicted viewpoint variable in iterations 0, 100, 200, and 500 are shown in red boxes and blue boxes, respectively.

with different camera system orientations, and then the predictions with the highest scores among different camera system orientations are taken as the final results. As described in Section 3.2.2, the number of pose candidates we investigate is 60 in case (ii): w/o upright orientation. Therefore, letting N_{tr} and N_{te} respectively denote the numbers of different camera system orientations, $60N_{tr}/60N_{te}$ different poses of objects are taken into consideration in the training/testing phase in case (ii). Even though the poses are discrete, we achieve a high chance that one of the investigated poses in the testing phase is close to the pose of any object in the training set. The effect of viewpoint augmentation is examined with the ShapeNetCore55 dataset [5] in Section 4.4.2, where we set N_{tr} and N_{te} to 11.

4 Experiments

In this section, we show the results of the experiments with 3D model benchmark datasets (Section 4.1), a real image benchmark dataset captured with a one-dimensional turning table (Section 4.2), and our new dataset consisting of multi-view real images of objects viewed with two rotational degrees of freedom (Section 4.3). We also describe our submitted results as well as additional experiments on the 3D Shape Retrieval Contest (SHREC) 2017 in Section 4.4. Finally, we demonstrate a real-world application of RotationNet using a moving USB camera (Section 4.5). The baseline architecture of our CNN is based on AlexNet [18], which requires less memory than the VGG-M network architecture that MVCNN [36] used. To train RotationNet, we fine-tune the weights pre-trained using the ILSVRC 2012 dataset [28]. We used classical momentum SGD with a learning rate of 0.0005 and a momentum of 0.9 for optimization.

As a baseline method, we also fine-tuned the pre-trained weights of a standard AlexNet CNN that only predicts object categories. To aggregate the predictions of multi-view images, we summed up all the scores obtained through the CNN. This method can be recognized as a modified version of MVCNN [36], where the view-pooling layer is placed after the final softmax layer. We chose average pooling for the view-pooling layer in this setting of the baseline, because we observed that the performance was better than that with max pooling. We also implemented MVCNN [36] based on the AlexNet architecture with the original view-pooling layer for a fair comparison.

4.1 Experiment on 3D Model Datasets

We first describe the experimental results on two 3D model benchmark datasets, ModelNet10 and ModelNet40 [42]. ModelNet10 consists of 4,899 object instances in 10 categories, whereas ModelNet40 consists of 12,311 object instances in 40 categories. First, we show the change of object classification accuracy versus the number of views used for prediction in cases (i) and (ii) with ModelNet40 and ModelNet10, respectively, in Figs. 10a and 10b and Figs. 10d and 10e. Here, we randomly shuffled the observation order of the multi-view images of each object 120 times, and then we computed the average classification accuracies of 120 trials. For fair comparison, we used the same training and test split of ModelNet40 as in [42] and [36]. We prepared multi-view images (i) with the upright orientation assumption and (ii) without the upright orientation assumption using the rendering software published in [36]. In Figs. 10a and 10d, which show the results with ModelNet40, we also draw the scores with the original MVCNN using Support Vector Machine (SVM) reported in [36]. Interestingly, as we focus on the object classification task whereas Su et al. [36] focused more on object retrieval task, we found that the baseline method with late view-pooling is better in most cases than the original MVCNN with the view-pooling layer in the middle. The baseline method does especially well with ModelNet10 in case (i) (Fig. 10b), where it achieves the best performance among the methods. With ModelNet40 in case (i) (Fig. 10a), RotationNet achieved a comparable result with MVCNN when we used all the 12 views as input. In case (ii) (Figs. 10d and 10e), where we consider three-axis rotation, RotationNet demonstrated superior performance to other methods. Only with three views, it showed

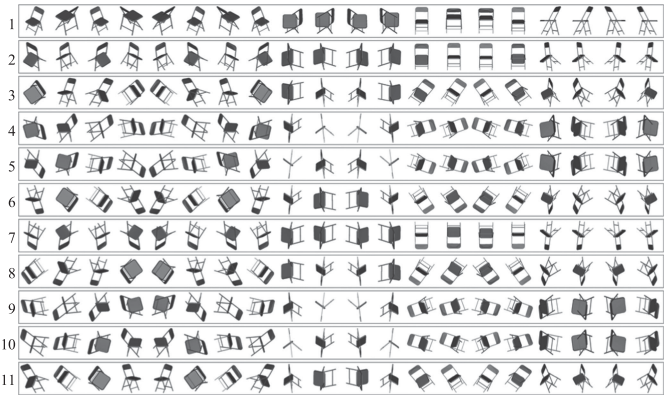


Fig. 8. Exemplar multi-view images of a chair captured in case (ii) with 11 different camera system orientations. Numbers in the left indicate the camera system orientation IDs.

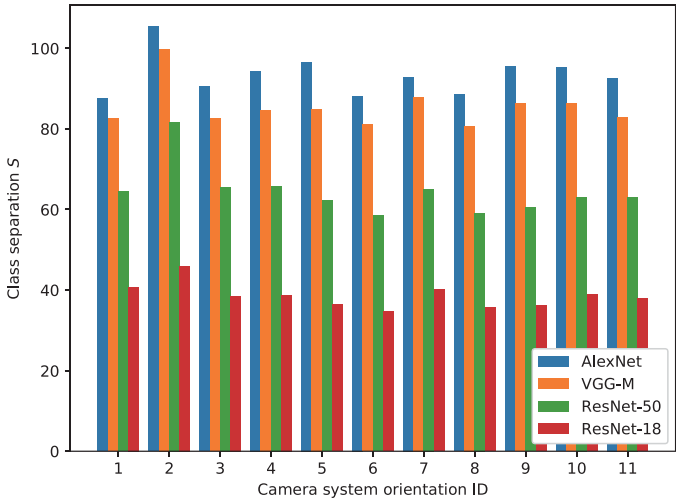


Fig. 9. Class separation S for different camera system orientations on ModelNet10.

comparable performance to that of MVCNN with a full set (80 views) of multi-view images.

Next, we investigate the performance of RotationNet with four different architectures: AlexNet [18], VGG-M [6], ResNet-50 [13], and ResNet-18 [13]. We tested the performance of RotationNet with 11 different camera system orientations. Fig. 8 shows exemplar multi-view images of a chair in ModelNet40 dataset captured in case (ii) with the 11 camera system orientations. Although "aligned" ModelNet40 dataset has been recently released, we used the original "unaligned" ModelNet40 dataset in our work. Camera system orientations are first rotated by 36 degree about the x-axis, and then rotated by $t \times 36^\circ$ $(t = 0, \ldots, 9)$ about the y-axis. In this way, different camera system orientations can capture different object profiles. Table 1 shows the comparison of classification accuracy (%) on ModelNet40 and ModelNet10 with the different camera system orientations. Surprisingly, the performance difference among different architectures is marginal compared to the difference caused by different camera system orientations. It indicates that the placement of viewpoints is the most important factor in multiview-based 3D object classification. The best scores with each architecture among the orientations are shown in bold. As shown here, the best camera system orientation is consistent across different architectures: the second one for ModelNet10 and the fourth one for ModelNet40. It indicates that multi-view object classification is greatly improved by observing appropriate aspects of objects. In addition, Table 1 shows that the best performance on the validation set (which we extracted from the training split of ModelNet40) was achieved with the same camera system orientation as the test set. Therefore, it is possible to obtain the best RotationNet model by selecting the one that best classifies a validation set among different camera system orientations.

Since the second camera system orientation for ModelNet10 provided significantly better results than the other orientations, we measured the class separation S in Fisher's linear discriminant analysis [10]. We extracted the features before the final fully-connected layer of the four networks pretrained on ImageNet. Letting C, $\boldsymbol{\mu}$, $\boldsymbol{\mu}_i$, Σ_b, and Σ_w denote the number of classes (i.e., 10 in this case,) the mean of all samples, the mean of samples in the ith class, the between-class covariance, and the covariance of the whole dataset, the class separation $S_{\boldsymbol{w}}$ in a direction $\boldsymbol{w}$ can be computed as follows:

$$S_{\boldsymbol{w}} = \frac{\boldsymbol{w}^\top \Sigma_\mathrm{b} \boldsymbol{w}}{\boldsymbol{w}^\top \Sigma_\mathrm{w} \boldsymbol{w}}, \quad \text{s.t.} \quad \Sigma_\mathrm{b} = \frac{1}{C} \sum_{i=1}^{C} (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^\top. \tag{7}$$

If $\boldsymbol{w}$ is an eigenvector of $\Sigma_\mathrm{w}^{-1}\Sigma_\mathrm{b}$, $S_{\boldsymbol{w}}$ equals the corresponding eigenvalue. Therefore, we compute S as the summation of all eigenvalues of $\Sigma_\mathrm{w}^{-1}\Sigma_\mathrm{b}$. Fig. 9 shows S for different camera

TABLE 1
Comparison of Classification Accuracy (%) with Different Camera System Orientations.

Dataset	Archit.	Camera system orientation ID											Mean
		1	2	3	4	5	6	7	8	9	10	11	
ModelNet40 - val	AlexNet	93.03	94.33	95.14	**95.54**	92.63	92.46	92.38	92.95	92.79	92.79	92.87	93.35 ± 1.06
ModelNet40 - test	AlexNet	93.03	94.04	95.22	**96.39**	93.35	92.99	92.79	93.15	92.99	93.31	93.48	93.70 ± 1.07
	VGG-M	93.64	95.91	96.07	**97.37**	94.12	93.80	94.08	94.25	93.68	94.41	94.17	94.68 ± 1.16
	ResNet-50	93.76	96.64	95.91	**96.92**	94.08	93.68	94.45	94.29	94.45	94.29	94.00	94.77 ± 1.10
	ResNet-18	92.63	95.62	94.65	**96.03**	92.87	92.42	93.03	92.71	92.83	92.63	93.52	93.54 ± 1.23
ModelNet10 - test	AlexNet	94.16	**97.58**	93.94	94.38	94.60	93.61	94.05	93.94	94.38	94.71	94.38	94.52 ± 1.01
	VGG-M	94.38	**98.46**	94.05	94.93	94.38	94.38	94.38	94.49	94.82	94.49	94.27	94.82 ± 1.17
	ResNet-50	94.82	**97.80**	94.49	94.49	94.16	94.60	94.38	94.60	94.27	94.71	94.49	94.80 ± 0.96
	ResNet-18	94.27	**98.90**	95.15	94.82	95.04	94.82	95.04	94.82	95.15	94.71	95.04	95.25 ± 1.18

The best scores with each RotationNet architecture among the orientations are shown in bold.

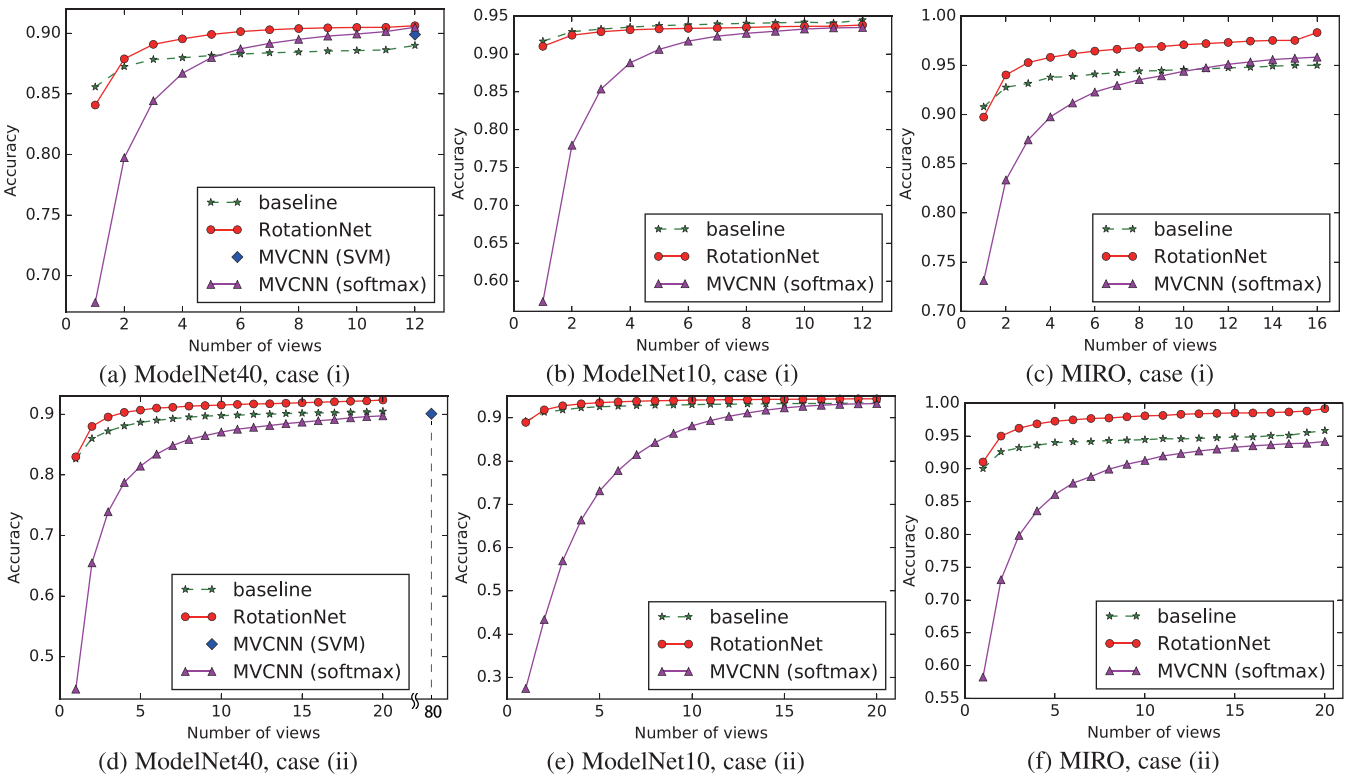


Fig. 10. Classification accuracy versus number of views used for prediction. From left to right are shown the results on ModelNet40, ModelNet10, and our new dataset MIRO. The results in case (i) are shown in top and those in case (ii) are shown in bottom. See Table 5 for an overall performance comparison to existing methods on ModelNet40 and ModelNet10.

system orientations. Regardless to the network architecture, S for the second camera system orientation is larger than that for the other orientations. This indicates that measuring S (before training a network) enables to roughly estimate how successful the camera orientation is, though the ranking does not necessarily reflect the resulting classification accuracies.

Finally, we summarize the comparison of classification accuracy (per class and per instance) on ModelNet40 and ModelNet10 to existing 3D object classification methods in Table 5. RotationNet (with VGG-M architecture for ModelNet40 and with ResNet-18 architecture for ModelNet10) significantly outperformed existing methods. Note that we reported the maximum accuracy among the aforementioned 11 rotation trials. The time required for training the 11 networks on a single GPU was approximately 3 to 5 days, which is relatively greater than competing methods, e.g., one day for MVCNN-New [39] and a few hours for VoxNet [23] and PointNet [26]. More detailed comparison to those methods is shown in Table 6. Note that MVCNN-New [39] indicates MVCNN with VGG-11 architecture. Even though the number of parameters is greater than that of VoxNet and PointNet, RotationNet with AlexNet architecture is the most efficient with respect to memory usage.

TABLE 2
Accuracy of Classification and Viewpoint Estimation (%) in Case (i) with RGBD

Algorithm	class	view
MVCNN (softmax)	86.08	-
Baseline	88.73	-
Fine-grained, T=300	81.23	26.94
Fine-grained, T=4000	76.95	31.96
RotationNet	**89.31**	**33.59**

4.2 Experiment on a Real Image Benchmark Dataset

Next, we describe the experimental results on a benchmark RGBD dataset published in [20], which consists of real images of objects on a one-dimensional rotation table. This dataset contains 300 object instances in 51 categories. Although it contains depth images and 3D point clouds, we used only RGB images in our experiment. We applied the upright orientation assumption (case (i)) in this experiment, because the bottom faces of objects on the turning table were not recorded. We picked out 12 images of each object instance with the closest rotation angles to $\{0°, 30°, \ldots, 330°\}$. In the training phase, objects are self-aligned (in an unsupervised manner) and the viewpoint variables for images are determined. To predict the pose of a test object instance, we predict the discrete viewpoint that each test image is observed, and then refer the most frequent pose value among those attached to the training samples predicted to be observed from the same viewpoint. We used the training/test split provided by the official site.[3]

Table 2 summarizes the classification and viewpoint estimation accuracies. Note that the reported viewpoint estimation accuracy indicates the classification accuracy of 12 discrete viewpoints. The baseline method and MVCNN are

3. http://rgbd-dataset.cs.washington.edu/dataset/rgbd-dataset_eval/testinstance_ids.txt

TABLE 3
Accuracy of Classification and Viewpoint Estimation (%) in Case (i) with MIRO

Algorithm	class	view
MVCNN (softmax)	95.83	-
Baseline	95	-
Fine-grained, T=800	92.76	56.72
Fine-grained, T=4000	91.35	58.33
RotationNet	**98.33**	**85.83**

TABLE 4
Accuracy of Classification and Viewpoint Estimation (%) in Case (ii) with MIRO

Algorithm	class	view
MVCNN (softmax)	94.17	-
Baseline	95.83	-
Fine-grained, T=1100	94.21	70.63
Fine-grained, T=2600	93.54	72.38
RotationNet	**99.17**	**75.67**

TABLE 5
Comparison of Classification Accuracy (%)

	ModelNet40		ModelNet10
Algorithm	Per Instance	Per Class	Per Instance
RotationNet	**97.37**	**96.29**	**98.90**
MVCNN-New [39]	95.0	92.4	-
MVCNN-MultiRes [27]	93.8	91.4	-
Dominant Set Clustering [40]	93.8	-	-
Kd-Networks [17]	91.8	-	94.0
VRN-single [4]	91.33	-	93.61
FusionNet [14]	90.80	-	93.11
Pairwise [16]	90.70	-	92.80
PANORAMA-NN [32]	90.7	-	91.1
DeepSets [44]	90.3	-	-
MVCNN [36]	90.10	90.10	-
ORION [31]	89.7	-	93.80
PointNet [26]	89.2	86.2	-
LightNet [47]	88.93	-	93.94
FPNN [22]	88.4	-	-
Multiple Depth Maps [45]	87.8	-	91.5
ECC [34]	87.4	83.2	90.8
VoxNet [23]	85.9	83.0	-
3DShapeNets [42]	84.7	77.3	-
Geometry Image [35]	83.9	-	88.4
3D-GAN [41]	-	83.30	-
GIFT [1]	83.10	-	92.35
Beam Search [43]	81.26	81.26	88
DeepPano [33]	77.63	-	85.45
PointNet [12]	-	-	77.6

RotationNet achieved the state-of-the-art *performance both with ModelNet40 and ModelNet10.*

TABLE 6
Comparison of the Number of Parameters, Memory Usage (During Training with a Batch Size 64), and Classification Accuracy on ModelNet40

Algorithm	# Params	Memory [GB]	Accuracy [%]
MVCNN-New [39]	128.9M	10.0	95.0 (92.4)
PointNet [26]	3.5M	4.4	89.2 (86.1)
VoxNet [23]	**1.4M**	2.0	85.6 (81.4)
RotationNet (AlexNet)	60.2M	**1.8**	96.4 (94.9)
RotationNet (VGG-M)	102.2M	5.3	**97.4 (96.3)**
RotationNet (ResNet-50)	24.2M	7.1	96.9 (94.8)
RotationNet (ResNet-18)	11.6M	2.5	96.0 (93.6)

Numbers in brackets show per-class accuracy. Numbers for MVCNN-New, PointNet, and VoxNet are derived from [39].

TABLE 7
Comparison on Object Instance/Category Recognition and Pose Estimation on RGBD Dataset

	Instance (%)	Category (%)	Avg. Pose (%)
Lai et al. [21]	78.40	94.30	53.50
Zhang et al. [46]	74.79	93.10	61.57
Bakry et al. [2]	80.10	94.84	76.63
Elhoseiny et al. [8]	-	97.14	79.30
Ours - single view	90.44	96.55	78.67
Ours - 12 views	**97.45**	**99.51**	**81.17**

not able to estimate viewpoints because they are essentially viewpoint invariant. As another baseline approach to compare, we learned a CNN with AlexNet architecture that outputs 612 $(= 51 \times 12)$ scores to distinguish both viewpoints and categories, which we call "Fine-grained." Here, T denotes the number of iterations that the CNN parameters are updated in the training phase. As shown in Table 2, the classification accuracy with "Fine-grained" decreases while its viewpoint estimation accuracy improves as the iteration grows. We consider this is because the "Fine-grained" classifiers become more and more sensitive to intra-class appearance variation through training, which affects the categorization accuracy. In contrast, RotationNet demonstrated the best performance in both object classification and viewpoint estimation, although the ground-truth poses are not given to RotationNet during the training.

Table 7 shows the object instance/category recognition as well as pose estimation accuracy comparison to existing methods. Here, we report the pose estimation accuracy on the same training/test split and the metric used in [21] (and thus the numbers are different from Table 2). RotationNet with a single image input performs comparable to Elhoseiny et al. [8]. Interestingly, when we estimate object instance/category and pose using 12 views altogether, both accuracies are remarkably improved.

4.3 Experiment on a 3D Rotated Real Image Dataset

We describe the experimental results on our new dataset "Multi-view Images of Rotated Objects (MIRO)" in this section. We used Ortery's 3D MFP studio[4] to capture multi-view images of objects with 3D rotations. The RGBD benchmark dataset [20] has two issues for training multi-view based CNNs: insufficient number of object instances per category (which is a minimum of two for training) and inconsistent cases to the upright orientation assumption. There are several cases where the upright orientation assumption is actually invalid; the attitudes of object instances against the rotation axis are inconsistent in some

4. https://www.ortery.com/photography-equipment/3d-modeling/

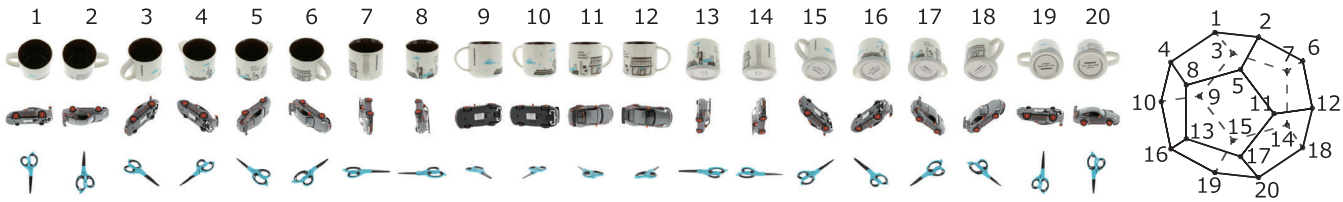


Fig. 11. Exemplar multi-view images of objects in our new dataset MIRO. There are 20 images for each object instance in case (ii).

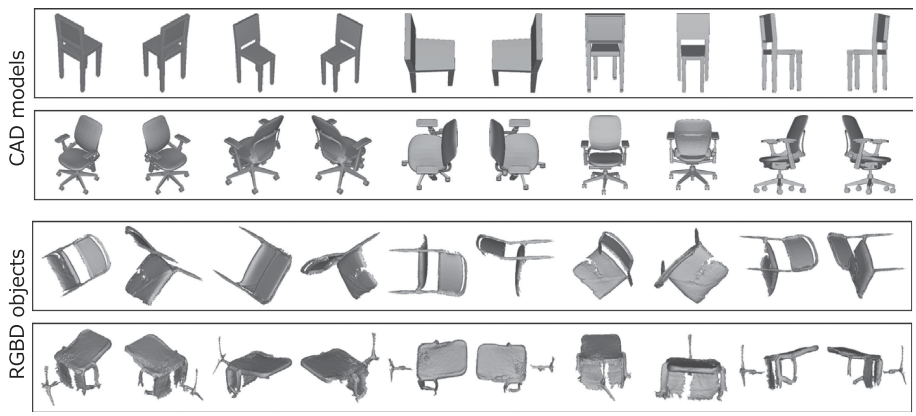


Fig. 12. Exemplar chair images of the CAD models and the RGB-D scans in SHREC'17 track 1.

object categories. Also, this dataset does not include the bottom faces of objects on the turning table. Our MIRO dataset includes 10 object instances per object category. It consists of 120 object instances in 12 categories in total. We captured each object instance with $M_e = 10$ levels of elevation angles and 16 levels of azimuth angles to obtain 160 images. For our experiments, we used 16 images ($\theta = 22.5°$) with $0°$ elevation of an object instance in case (i). We carefully captured all the object instances in each category to have the same upright direction in order to evaluate performance in the case (i). For case (ii), we used 20 images observed from the 20 vertices of a dodecahedron encompassing an object. Fig. 11 shows examples of multi-view images of objects in case (ii).

Figs. 10c and 10f show the object classification accuracy versus the number of views used for the prediction in case (i) and case (ii), respectively. In both cases, RotationNet clearly outperforms both MVCNN and the baseline method when the number of views is larger than 2. We also tested the "Fine-grained" method that outputs $(192 = 12 \times 16)$ scores in case (i) and $(240 = 12 \times 20)$ scores in case (ii) to distinguish both viewpoints and categories, and the overall results are summarized in Tables 3 and 4. The reported viewpoint estimation accuracy indicates the classification accuracy of 16 and 20 discrete viewpoints for case (i) and (ii), respectively. Similar to the results with an RGBD dataset described above, there is a trade-off between object classification and viewpoint estimation accuracies in the "Fine-grained" approach. RotationNet achieved the best performance in both object classification and viewpoint estimation, which demonstrates the strength of the proposed approach.

4.4 Experiment on 3D Shape Retrieval

In this section, we describe our results using RotationNet on the 3D Shape Retrieval Contest (SHREC) 2017. We participated in two tracks of the SHREC'17: (Track 1) "RGB-D Object-to-CAD Retrieval Contest" and (Track 3) "Large-scale 3D Shape Retrieval from ShapeNet Core55 Challenge". The former track tackles the retrieval of CAD models with noisy RGB-D objects as input. The latter track also aims at the retrieval of CAD models, but with the inputs of CAD models. We treated rendered multi-view images of CAD models and RGBD objects (without color) in the same manner in both tracks, where we won the best scores among all teams. We describe our results on the track 1 in Section 4.4.1 and the track 3 in Section 4.4.2. We also show additional results using viewpoint augmentation on ShapeNetCore55 dataset [5] in Section 4.4.2.

4.4.1 Track 1: RGB-D to CAD Retrieval

The dataset provided in the SHREC'17 track 1 contains 1,667 RGB-D objects and 3,308 CAD models in ShapeNet [5] in 20 categories. The RGB-D objects are split into training, validation, and test set (query) with the split ratio of $50/25/25$ percent. The retrieval is performed using the RGB-D objects as query and the CAD models as targets.[5] The RGB-D objects are acquired from a consumer-grade depth camera, so that the reconstructed 3D models are noisy and partial, due to occlusion. Note that the specific CAD model corresponding to each query RGB-D object does not exist. A retrieved CAD model is considered correct if it belongs to the same *category* as the query RGB-D object, regardless of the similarity in shape or texture. Fig. 12 shows exemplar chair images of the CAD models and the RGB-D objects in the dataset. We rendered both the CAD models and the RGB-D objects in case (ii) viewpoint setting (i.e., w/o upright orientation assumption).

We trained our RotationNet model using the provided 3,308 CAD models, and then we fine-tuned the model using the RGB-D objects in the training set. Interestingly, our

5. Target models are allowed to be used for training a system.

TABLE 8
Evaluation Results on the Test Set in SHREC'17 Track 1 (Quoted from [15])

Team	Run	Training	Domain	Precision	Recall	F1	mAP	NDCG	Tier-1	Tier-2
Kanezaki **(RotationNet)**	single	Y	View-based	0.792	0.792	0.792	0.792	0.792	0.792	0.792
Kanezaki **(RotationNet)**	thresh	Y	View-based	0.793	0.799	0.794	0.794	0.796	0.794	0.794
Kanezaki **(RotationNet)**	1000	Y	View-based	**0.820**	**0.820**	**0.820**	**0.833**	**0.805**	**0.824**	**0.824**
Chiang	3dcnn	Y	Full 3D	0.769	0.769	0.769	0.749	0.774	0.769	0.769
Chiang	mvcnn	Y	View-based	0.727	0.727	0.727	0.710	0.735	0.727	0.727
Chiang	fuse	Y	-	0.759	0.759	0.759	0.746	0.763	0.759	0.759
Chiang	mvcnn_triplet	Y	View-based	0.672	0.672	0.672	0.649	0.714	0.672	0.672
Truong	2D	Y	View-based	0.740	0.740	0.740	0.740	0.740	0.740	0.740
Truong	3D	Y	Full 3D	0.487	0.487	0.487	0.487	0.487	0.487	0.487
Li		N	View-based	0.105	0.320	0.145	0.062	0.476	0.120	0.100
Tashiro		N	View-based	0.141	0.472	0.198	0.149	0.552	0.188	0.144

TABLE 9
Evaluation Results on the Test Set in SHREC'17 Track 3 (Quoted from [30])

		micro					macro				
Dataset	Method	P@N	R@N	F1@N	mAP	NDCG@N	P@N	R@N	F1@N	mAP	NDCG@N
Normal	Kanezaki **(RotationNet)**	0.810	0.801	**0.798**	**0.772**	**0.865**	0.602	0.639	**0.590**	**0.583**	0.656
	Zhou	0.786	0.773	0.767	0.722	0.827	0.592	0.654	0.581	0.575	**0.657**
	Tatsuma	0.765	**0.803**	0.772	0.749	0.828	0.518	0.601	0.519	0.496	0.559
	Furuya	**0.818**	0.689	0.712	0.663	0.762	**0.618**	0.533	0.505	0.477	0.563
	Thermos	0.743	0.677	0.692	0.622	0.732	0.523	0.494	0.484	0.418	0.502
	Deng	0.418	0.717	0.479	0.540	0.654	0.122	**0.667**	0.166	0.339	0.404
	Li	0.535	0.256	0.282	0.199	0.330	0.219	0.409	0.197	0.255	0.377
	Mk	0.793	0.211	0.253	0.192	0.277	0.598	0.283	0.258	0.232	0.337
	SHREC16-Su (MVCNN)	0.770	0.770	0.764	0.735	0.815	0.571	0.625	0.575	0.566	0.640
	SHREC16-Bai	0.706	0.695	0.689	0.640	0.765	0.444	0.531	0.454	0.447	0.548
Perturbed	Furuya	**0.814**	0.683	0.706	0.656	0.754	**0.607**	0.539	**0.503**	**0.476**	**0.560**
	Tatsuma	0.705	**0.769**	0.719	0.696	**0.783**	0.424	0.563	0.434	0.418	0.479
	Zhou	0.660	0.650	0.643	0.567	0.701	0.443	0.508	0.437	0.406	0.513
	Kanezaki **(RotationNet)**	0.655	0.652	0.636	0.606	0.702	0.372	0.393	0.333	0.327	0.407
	Deng	0.412	0.706	0.472	0.524	0.642	0.120	**0.659**	0.164	0.329	0.395
	Li	0.496	0.234	0.258	0.172	0.303	0.199	0.373	0.179	0.215	0.336
	Mk	0.690	0.012	0.020	0.009	0.043	0.546	0.052	0.052	0.047	0.109
	SHREC16-Bai	0.678	0.667	0.661	0.607	0.735	0.414	0.496	0.423	0.412	0.518
	SHREC16-Su (MVCNN)	0.632	0.613	0.612	0.535	0.653	0.405	0.484	0.416	0.367	0.459
	RotationNet + aug. (train, test)	0.732	0.734	**0.723**	**0.698**	0.777	0.504	0.530	0.481	0.464	0.535
	RotationNet + aug. (test)	0.698	0.707	0.691	0.662	0.751	0.434	0.478	0.412	0.399	0.476
	RotationNet + aug. (train)	0.664	0.658	0.639	0.613	0.694	0.405	0.433	0.357	0.349	0.421

Methods are ranked by the average of the micro and macro mAP. The best scores with each metric are shown in bold, whereas the second best scores are shown with underlines. We include additional results of RotationNet with viewpoint augmentation on "perturbed" dataset separately below the SHREC'17 results reported in [30]. RotationNet without viewpoint augmentation achieved the best mAP scores with "normal" dataset, whereas RotationNet with viewpoint augmentation (both in training and testing phases) outperformed all the SHREC'17 results with "perturbed" dataset, with respect to the average mAP.

preliminary experiments showed that the pre-training of RotationNet with CAD models improved the classification accuracy of RGB-D objects. It indicates that RotationNet has a high potential to transfer category knowledge obtained from CAD models to real objects.

For each query, we constructed a retrieval set in three different approaches described below.

single. We predict a single category label for each query and construct a retrieval set with all the target models in the same category.

thresh. We rank categories by the classification scores for each query. If the score of a category is higher than a certain ratio of the maximum score, we add the target models in the category to the retrieval set. We set the threshold to 0.1. In the case that the scores of all the categories (except for the predicted category) are lower than 10 percent of the score of the predicted category, we obtain exactly the same retrieval set as "single" case.

1,000. In order of decreasing classification scores, we add the target models in the same category to the retrieval set for each query. We stop adding a target model when the number of the models in the retrieval set reaches to 1,000.

Table 8 shows evaluation results on the test set in SHREC'17 track 1 (quoted from [15]).[6] RotationNet with "1,000" retrieval strategy achieved the best scores in all the

6. http://people.sutd.edu.sg/~saikit/projects/sceneNN/shrec17/evaluation/

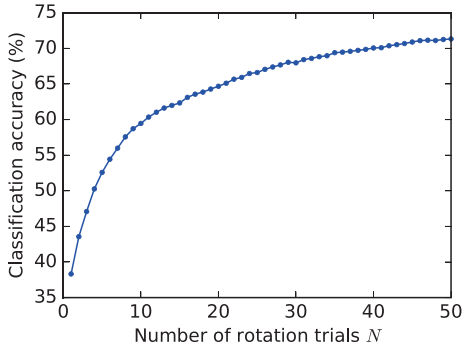


Fig. 13. Classification accuracy (%) on ShapeNetCore55. We rotated a test model N times and used the maximum scores.

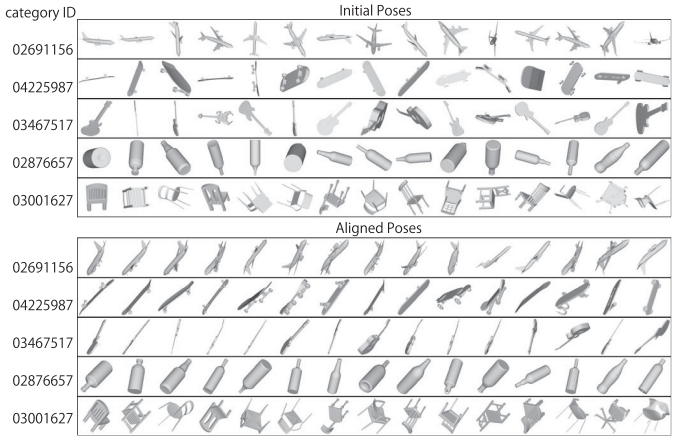


Fig. 14. Validation samples of the "perturbed" SHREC'17 dataset in initial poses and aligned poses using RotationNet.

metrics. Note that other teams "Chiang" and "Truong" also used CNNs, whereas "Li" and "Tashiro" performed feature matching using bag-of-words or shape descriptors. Please refer to [15] for more details.

4.4.2 Track 3: Large-Scale Shape Retrieval

The dataset provided in the SHREC'17 track 3 is named ShapeNetCore55, which contains a total of 51,162 3D models categorized into 55 WordNet synsets. Two different versions of the dataset is provided: "normal" dataset, where all model data is consistently aligned, and "perturbed" dataset, where each model has been randomly rotated by a uniformly sampled rotation in $SO(3)$. Participants were asked to return a ranked list of (up to 1,000) retrieved models for each model in the testing set, where the target models to be retrieved were all models in that set, including the query model itself. As is the case in track 1 (Section 4.4.1), a retrieved model is considered correct if it belongs to the same *category* as the query model. In the same manner as track 1 (Section 4.4.1), we trained RotationNet in case (ii) with "train" data and classified all the models of "test" data. For each query, we constructed a retrieval set containing all the samples with the same predicted category label in the order of descending scores of the predicted category.

Table 9 shows evaluation results on the test set in SHREC'17 track 3 (quoted from [30]). RotationNet achieved the best performance in "normal" dataset, whereas the point set-based approach of Furuya has the best overall performance for "perturbed" dataset. It indicates that point-based approach is more robust to random rotation than multi-view image based approaches. However, we argue that adding more images captured from different viewpoints improve the robustness to rotation. After the SHREC'17 submission period, we conducted additional experiments of RotationNet with the viewpoint augmentation either or both in the training and inference phases, as described in Section 3.4. In the same manner as Section 4.1, we trained/tested RotationNet with the 11 camera system orientations, which are nearly equally distributed in 3D space. Table 9 also shows the results of RotationNet with the viewpoint augmentation on "perturbed" dataset separately below the SHREC'17 results reported in [30]. The viewpoint augmentation either in the training or the testing phase improve the retrieval performance for the "perturbed" dataset, where the latter has more gain than the former. This implies that taking many viewpoints in consideration for object inference is a key issue to improve object categorization performance. Fig. 14 shows validation samples in five categories in the "perturbed" dataset. The fifteen object instances classified with highest likelihoods per category are shown. For each object instance in the latter half of the figure, the orientation with the highest category likelihood among the 11 camera system orientations is selected, and then the object is rotated (i.e., aligned) according to the orientation and captured from a certain consistent viewpoint. It is notable that the assignment of the viewpoints is determined by RotationNet. Although the alignment is not completely consistent and some objects are upside down, the objects' appearance is correlated to each other. Therefore, this brings a great improvement on the categorization performance. Eventually, RotationNet with the viewpoint augmentation both in the training and the testing phases outperformed all the SHREC'17 results with "perturbed" dataset, with respect to the average mAP.

We further examined the influence of the viewpoint augmentation in inference. We trained our RotationNet model with "normal" training dataset and tested the classification of "perturbed" validation dataset. Fig. 13 shows the classification accuracy (%) where we randomly rotated a test model (in the "perturbed" validation dataset) N times and used the maximum scores for the final inference. The classification accuracy without viewpoint augmentation (i.e., when $N = 1$) was 38 percent whereas it was 90 percent for "normal" validation set, which means our model is rather sensitive to pre-defined viewpoints. However, as shown in Fig. 13, the accuracy increases if we randomly rotate the test model N times. This is because adding more views for inference increases the probability of observing objects from the viewpoints that are close to the pre-defined viewpoints, which tend to give higher object category likelihood than unseen views.

4.5 Demonstration of Real-World Application of RotationNet Trained with the MIRO Dataset

Finally, we demonstrate the performance of RotationNet for real-world applications. Here, we tackle the application of object recognition and pose estimation on real images with cluttered backgrounds captured by a moving USB camera. For training, we used our MIRO dataset with the viewpoint setup case (iii), where all the outputs for images with 10

Fig. 15. Object instances self-aligned as a result of training RotationNet. The last instance in each category is a 3D CAD model.

levels of elevation angles are concatenated, which enables RotationNet to distinguish 160 viewpoints. We added rendered images of a single 3D CAD model (whose upright orientation is manually assigned) to each object class, which were trained together with MIRO dataset. Then we obtained successful alignments between a CAD model and real images for all the 12 object classes (Fig. 15).

The MIRO dataset is created in a controlled lighting system with a white background. Also, the target objects are placed on the center of images in the MIRO dataset. To improve the robustness to background clutters and translation of objects, we augmented the training images in two manners: background synthesis and random shift. We randomly crop a square region from an image randomly selected from the ILSVRC 2012 validation dataset [28]. Then we synthesize a MIRO image that is removed the white background with the random vertical and horizontal shift ranging from 0 to 56 pixels. We augmented the training dataset in this way from 20 thousands to 2 million.

Fig. 16 shows exemplar objects recognized using a USB camera. We estimated relative camera poses by LSD-SLAM [9] to integrate predictions from multiple views in sequence. The results obtained using multiple views (shown in the third and sixth rows) are consistently more accurate than those using a single view (shown in the second and fifth rows). It is worth noting that not only object classification but also pose estimation performance is improved by using multiple views. The classification and pose estimation run faster than 30 fps with a Titan X GPU.

5 Discussion

We proposed RotationNet, which jointly estimates object category and viewpoint from each single-view image and aggregates the object class predictions obtained from a partial set of multi-view images. In our method, object instances are automatically aligned in an unsupervised manner with both inter-class and intra-class structures based on their appearance during the training. In the experiment using 3D object benchmark datasets ModelNet40 and ModelNet10, RotationNet significantly outperformed the state-of-the-art methods based on voxels, point clouds, and multi-view images. RotationNet is also able to achieve comparable performance to MVCNN [36] with 80 different multi-view images using only a couple of view images, which is important for real-world applications. Another contribution is that we developed a publicly available new dataset named MIRO. Using this dataset and RGBD object benchmark dataset [20], we showed that RotationNet even outperformed supervised learning based approaches in a pose estimation task. We consider that our pose estimation performance benefits from view-specific appearance information shared across classes due to the inter-class self-alignment.

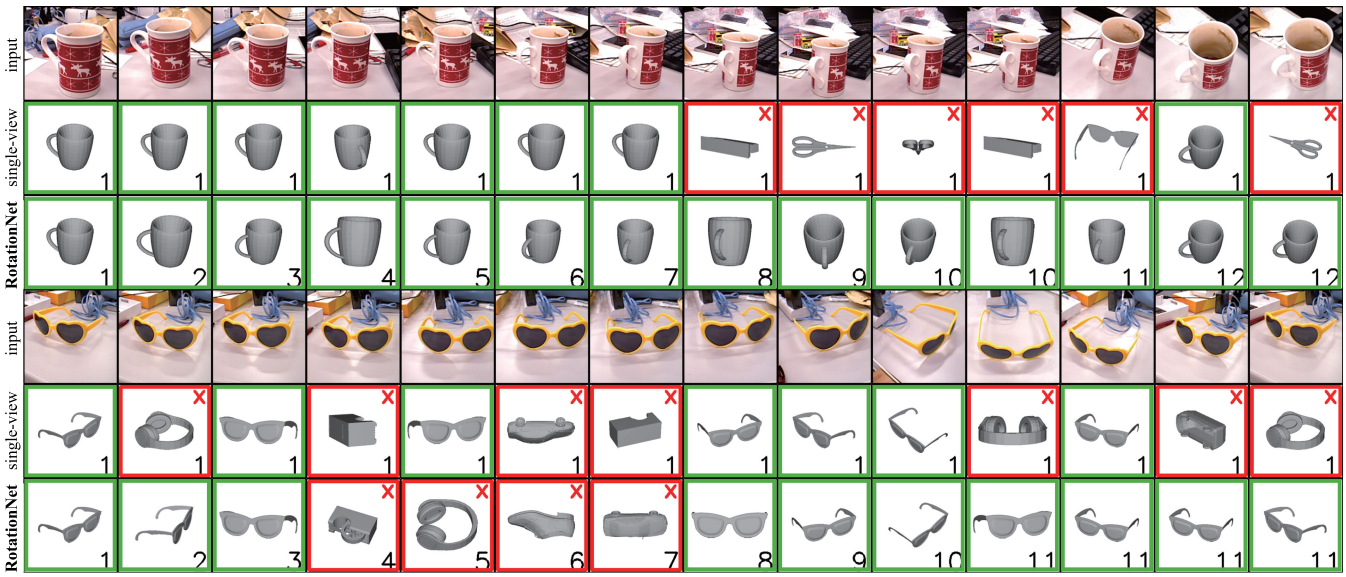


Fig. 16. Exemplar objects recognized using a USB camera. The second and fifth rows show 3D models in the estimated category and pose from a single view, whereas the third and sixth rows show those estimated using multiple views. The number in each image indicates the number of views used for predictions. Failure cases are shown in red boxes.

Multi-view based approaches have drawbacks in dealing with unseen viewpoints because 2D image representation depends on discrete viewpoints. In order to mitigate the limitation, we introduced viewpoint augmentation either or both in the training and inference phases. Either with 3D object models or densely captured continuous images (e.g., video frames), this is available by preparing multiple sets of multi-view images captured with different camera system orientations. We trained a single RotationNet model using augmented training images with 11 different camera system orientations, which brought a three percent improvement in mAP in the "SHREC'17 Large-Scale Shape Retrieval" task. This indicates that a better classification model can be achieved with consideration of more viewpoints. Taking many viewpoints in consideration for the inference phase is even more effective and achieved 14 percent boost in mAP. Eventually, with the viewpoint augmentation both in the training and inference phases, RotationNet outperformed all the existing methods including point-based approaches for the "perturbed" ShapeNetCore55 dataset in the SHREC'17 track. This is an important insight into 3D object recognition both with synthetic models and real scanned objects.

Acknowledgments

This project is supported by the New Energy and Industrial Technology Development Organization (NEDO). The authors would like to thank Hiroki Matsuno for his support in developing the HoloLens application.

References

[1] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. J. Latecki, "GIFT: A real-time and scalable 3D shape search engine," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5023–5032.

[2] A. Bakry and A. Elgammal, "Untangling object-view manifold for multiview recognition and pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 434–449.

[3] H. Borotschnig, L. Paletta, M. Prantl, and A. Pinz, "Appearance-based active object recognition," *Image Vis. Comput.*, vol. 18, no. 9, pp. 715–727, 2000.

[4] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and discriminative voxel modeling with convolutional neural networks," *arXiv preprint arXiv:1608.04236*, 2016.

[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:1512.03012, 2015.

[6] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proc. Brit. Mach. Vis. Conf.*, 2014. [Online]. Available: http://dx.doi.org/10.5244/C.28.6

[7] C.-Y. Chen and K. Grauman, "Inferring unseen views of people," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 2011–2018.

[8] M. Elhoseiny, T. El-Gaaly, A. Bakry, and A. Elgammal, "A comparative analysis and study of multiview CNN models for joint object categorization and pose estimation," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 888–897.

[9] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 834–849.

[10] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[11] R. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3449–3457.

[12] A. Garcia-Garcia, F. Gomez-Donoso, J. Garcia-Rodriguez, S. Orts-Escolano, M. Cazorla, and J. Azorin-Lopez, "PointNet: A 3D convolutional neural network for real-time object class recognition," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2016, pp. 1578–1584.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[14] V. Hegde and R. Zadeh, "FusionNet: 3D object classification using multiple data representations," arXiv:1607.05695, 2016. [Online]. Available: https://arxiv.org/abs/1607.05695

[15] B.-S. Hua, Q.-T. Truong, M.-K. Tran, Q.-H. Pham, A. Kanezaki, T. Lee, H. Chiang, W. Hsu, B. Li, Y. Lu, et al., "SHREC'17: RGB-D to CAD retrieval with ObjectNN dataset," in *Proc. Eurograph. Workshop 3D Object Retrieval*, 2017,pp. 25–32.

[16] E. Johns, S. Leutenegger, and A. J. Davison, "Pairwise decomposition of image sequences for active multi-view recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3813–3822.

[17] R. Klokov and V. Lempitsky, "Escape from cells: Deep Kd-networks for the recognition of 3D point cloud models," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 863–872.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[19] A. Kuznetsova, S. J. Hwang, B. Rosenhahn, and L. Sigal, "Exploiting view-specific appearance similarities across classes for zero-shot pose prediction: A metric learning approach," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 3523–3529.

[20] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 1817–1824.

[21] K. Lai, L. Bo, X. Ren, and D. Fox, "A scalable tree-based approach for joint object and pose recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2011, pp. 1474–1480.

[22] Y. Li, S. Pirk, H. Su, C. R. Qi, and L. J. Guibas, "FPNN: Field probing neural networks for 3D data," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 307–315.

[23] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 922–928.

[24] D. Novotny, D. Larlus, and A. Vedaldi, "Learning 3D object categories by looking around them," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5228–5237.

[25] L. Paletta and A. Pinz, "Active object recognition by view integration and reinforcement learning," *Robot. Auton. Syst.*, vol. 31, no. 1/2, pp. 71–86, 2000.

[26] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.

[27] C. R. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5648–5656.

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[29] S. Savarese and L. Fei-Fei, "3D generic object categorization, localization and pose estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[30] M. Savva, F. Yu, H. Su, A. Kanezaki, Z. Zhou, R. Yu, S. Bai, X. Bai, M. Aono, A. Tatsuma, S. Thermos, A. Axenopoulos, G. T. Papadopoulos, P. Daras, X. Deng, Z. Lian, B. Li, H. Johan, Y. Lu, and S. Mk, "SHREC'17 track large-scale 3D shape retrieval from ShapeNet core55," in *Proc. Eurograph. Workshop 3D Object Retrieval*, 2017.

[31] N. Sedaghat, M. Zolfaghari, and T. Brox, "Orientation-boosted voxel nets for 3D object recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2017.

[32] K. Sfikas, T. Theoharis, and I. Pratikakis, "Exploiting the PANORAMA representation for convolutional neural network classification and retrieval," in *Proc. Eurograph. Workshop 3D Object Retrieval*, 2017, pp. 1–7.

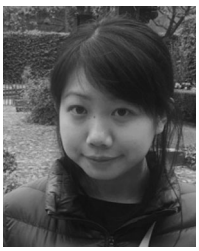
[33] B. Shi, S. Bai, Z. Zhou, and X. Bai, "DeepPano: Deep panoramic representation for 3-D shape recognition," *IEEE Signal Process. Lett.*, vol. 22, no. 12, pp. 2339–2343, Dec. 2015.

[34] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 29–38.

[35] A. Sinha, J. Bai, and K. Ramani, "Deep learning 3D shape surfaces using geometry images," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 223–240.

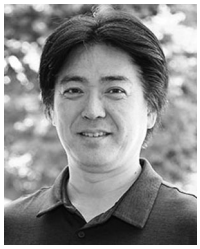
[36] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 945–953.

[37] H. Su, C. R. Qi, Y. Li, and L. J. Guibas, "Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2686–2694.
[38] H. Su, F. Wang, E. Yi, and L. J. Guibas, "3D-assisted feature synthesis for novel views of an object," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2677–2685.
[39] J.-C. Su, M. Gadelha, R. Wang, and S. Maji, "A deeper look at 3D shape classifiers," in *Proc. 2nd Workshop 3D Reconstruction Meets Semantics*, 2018, pp. 645–661.
[40] C. Wang, M. Pelillo, and K. Siddiqi, "Dominant set clustering and pooling for multi-view 3D object recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2017.
[41] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 82–90.
[42] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1912–1920.
[43] X. Xu and S. Todorovic, "Beam search for learning a deep convolutional neural network of 3D shapes," in *Proc. Int. Conf. Pattern Recognit.*, 2016, pp. 3506–3511.
[44] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 3391–3401.
[45] P. Zanuttigh and L. Minto, "Deep learning for 3D shape classification from multiple depth maps," in *Proc. IEEE Int. Conf. Image Process.*, 2017, pp. 3615–3619.
[46] H. Zhang, T. El-Gaaly, A. M. Elgammal, and Z. Jiang, "Joint object and pose recognition using homeomorphic manifold analysis," in *Proc. AAAI Conf. Artif. Intell.*, 2013, pp. 1012–1019.
[47] S. Zhi, Y. Liu, X. Li, and Y. Guo, "Toward real-time 3D object recognition: A lightweight volumetric CNN framework using multitask learning," *Comput. Graph.*, vol. 71, pp. 199–207, 2018.
[48] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6612–6619.
[49] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Multi-view perceptron: A deep model for learning face identity and view representations," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 217–225.



Asako Kanezaki received the BS, MS, and PhD degrees in information science and technology from the University of Tokyo, in 2008, 2010, and 2013, respectively. In 2010, she was a visiting researcher with Intelligent Autonomous Systems Group, Technische Universität München. From 2013 to 2016, she was an assistant professor with the University of Tokyo. She joined Living Intelligence Research Team of the National Institute of Advanced Industrial Science and Technology (AIST), in 2016 and is currently working as a senior researcher. Her research interests include object detection, 3D shape recognition, and robot applications. She is a member of the IEEE.

Yasuyuki Matsushita received the BS, MS, and PhD degrees in EECS from the University of Tokyo, in 1998, 2000, and 2003, respectively. From April 2003 to March 2015, he was with Visual Computing Group, Microsoft Research Asia. In April 2015, he joined Osaka University as a professor. His research area includes computer vision, machine learning, and optimization. He is on the editorial board of the *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, the *International Journal of Computer Vision (IJCV)*, the *IPSJ Journal of Computer Vision and Applications (CVA)*, the *Visual Computer Journal*, and the *Encyclopedia of Computer Vision*. He served/is serving as a program co-chair of PSIVT 2010, 3DIMPVT 2011, ACCV 2012, ICCV 2017, and a general co-chair for ACCV 2014 and ICCV 2021. He is a senior member of the IEEE.



Yoshifumi Nishida received the PhD degree from the Graduate School of Mechanical Engineering, University of Tokyo, in 1998. In 1998, he joined Intelligent System Division of Electrotechnical Laboratory, Ministry of International Trade and Industry (MITI), Japan. In 2001, he joined Digital Human Laboratory of the National Institute of Advanced Industrial Science and Technology (AIST). From 2005 to 2011, he was the project leader of the Strategic Basic Research Program (CREST) of Japan Science and Technology Agency (JST) . From 2013 to 2019, he was also a prime senior research scientist with AIST. From 2017, he is serving as a program manager of Social Technology Domain of SECOM Science and Technology Foundation. In April 2019, he joined Tokyo Institute of Technology as a professor. He is a member of the IEEE, Robotics Society of Japan, and the Japanese Society of Artificial Intelligence.