Supplementary Materials for OVCNet

Sainan Liu Vincent Nguyen Isaac Rehg Zhuowen Tu
University of California, San Diego
{sal131, vvn012, irehg, ztu}@ucsd.edu

In this supplementary material, we first show the 2D in-plane rotation ablation results and rotation-invariant analysis. Then we provide more details regarding our experiments and runtime analysis.

1. 2D In-plane Rotation Ablation Study

We evaluate ResNet18 with different angles of rotation augmentation and observe that the evaluation accuracy stops increasing as we provide denser angle augmentations as is shown in Figure 1.

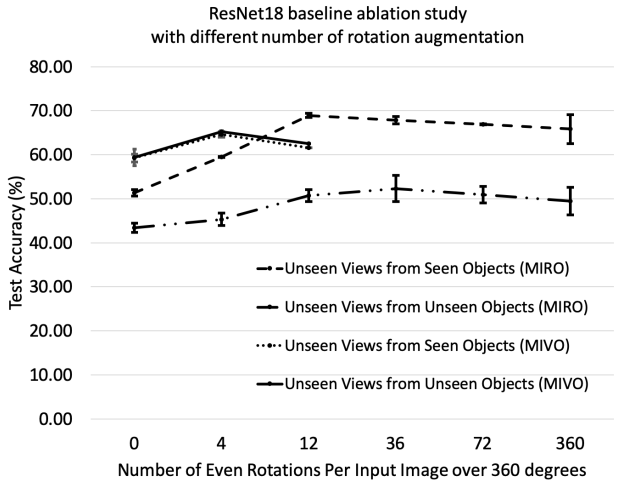

Figure 1. Rotational ablation study for ResNet18. X-axis: the number of rotation over 360 degrees; 0 means no rotation augmentation is applied to the original input view within the 2D plane. Means and standard deviations are reported over two repeats each. The test accuracy plateaus as the number of 2D in-plane rotations increases. The accuracy plateaus around 30 degrees for gMIRO and 90 degrees for gMIVO.

2. Rotation Invariant Analysis

For spherical CNNs, Cohen *et al*. has shown empirical support for rotation-invariant learning problems. Here we show in Figure 2 that the features of spherical CNNs (without any 3D rotation data augmentation) on the 3D reconstruction of a "bus" do demonstrate a certain level of rotation invariant property.

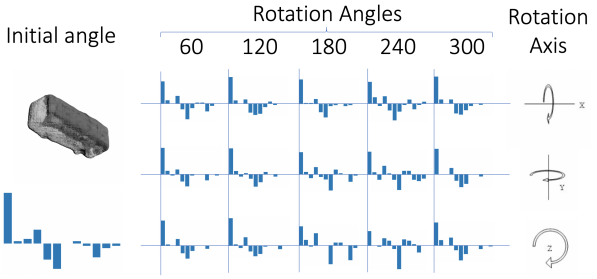

Figure 2. Demonstration of the **achieved** rotation-invariance property of spherical CNNs on 3D reconstruction of an object ("bus") instance. We train our model with spherical maps generated from a reconstructed 3D object in its initial orientation from GenRe. The reconstructed object (top) and features from the trained spherical maps (bottom) are shown in the left-most column. We then rotate the reconstructed object along 3 different axes over 5 different angle variations (in degrees) to generate a spherical map test set. The last column shows the axis of rotation.

3. Experiment details

With the gMIRO dataset, in the OC module (Figure 3) we train the spherical CNNs for 300 epochs, with the batch size 12, learning rate 0.1 (decay factor=10 for every 100 epochs for 300 epochs), and bandwidth 112; for the ResNet18 part, we use learning rate 0.1, batch size 10 for 500 epochs, and bandwidth of 3.

For the VC (3D) branch, we use 640 3D viewpoint augmentations with texture. We train ResNet18 for 500 epochs with batch size 512 and learning rate 0.01. For view selection, we use a nearest neighbor approach. The augmented image that is closest to the input viewpoint is used for evaluation on gMIRO dataset. When an attention selection layer is used, we first train a ResNet with 80% of the training data, and then 20% of the remaining training data is used to train a weighted or attention layer for selection. During the inference time, we use the ResNet model trained on the entire training set and the trained selection layer.

For the VC (2D) branch, we use online 2D in-plane augmentation with 30-degree rotations, and train ResNet18 for 1250 epochs with the batch size 512 and learning rate 0.01.

On the gMIVO dataset, for the OC branch, we train the spherical CNNs for 300 epochs, with the batch size 12, learning rate 0.1 (decay factor=10 for every 100 epochs for

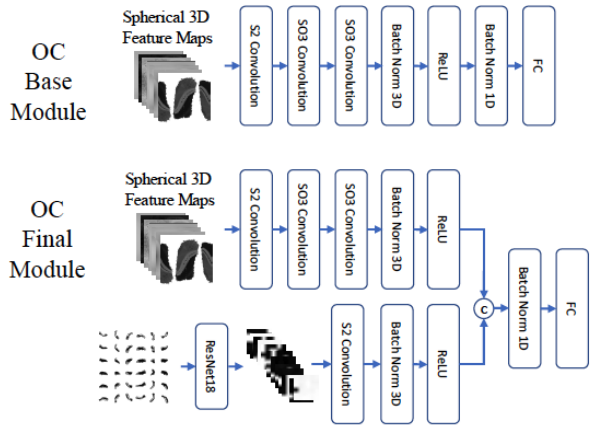

Figure 3. Network structure for OC baseline module vs. final OC module. c here means the concatenation of the two branches.

300 epochs), and bandwidth 112. For the VC (3D) branch, we use 160 3D viewpoint augmentations with texture, and train ResNet18 for 120 epochs with the batch size 2048 and learning rate 0.1 (decay factor=10 for every 50 epochs). For the VC (2D) module, we use an online 2D in-plane augmentation with 90-degree rotations. We train ResNet18 for 1250 epochs with the batch size 512 and learning rate 0.1 (decay factor=10 for every 350 epochs).

4. Runtime Analysis

Each ResNet18 consists of around 11 million trainable parameters, whereas the largest spherical CNN has about 1.4 million trainable parameters. The space complexity for training is approximately $O(Cn)$ where C equals 7 spherical signal maps + 3D to 2D projection of input view $\times$ 160/640 + 36 (with 3D-rotation augmentation) + 2D input view $\times$ 12 (with every 30 degree 2D-rotation augmentation). During testing, C equals 9 (7 spherical signal maps + 3D to 2D projection of input view $\times$ 1 (or 160 if using attention) + 36 + original input view $\times$ 1) spherical signal maps. The average inference time is always within minutes for the ResNet18s and spherical CNNs with the gMIRO dataset; it takes 10's of milliseconds for evaluation. It takes approximately a day to generate the reconstructions for all of the images from the gMIRO dataset with 3 Titan Xp GPUs. The texture estimation process uses a k-d tree structure for the nearest neighbor search, which takes on average $O(logV)$ in terms of time and $O(V)$ in terms of space. V here is the number of voxels in the volumetric representation ($128\times128\times128$) obtained from GenRe. It takes less than a second on average to process one object on a CPU. Both the texture mapping processes for images and the individual module training can be sped up by parallel processing.