Towards 3D Rotation Invariant Embeddings

Aditya Sanghi
Autodesk
aditya.sanghi@autodesk.com

Ara Danielyan
Autodesk
ara.danielyan@autodesk.com

Abstract

Obtaining rotation invariant embedding of a shape is very useful in many tasks such as shape comparison, classification and segmentation. In this work, we create a neural network architecture that creates a rotational invariant representation of a shape around a single axis in an unsupervised manner. We reconstruct the whole object so that the network does not learn trivial solutions for a specific task and contains all semantic information about the object. We show how this model performs better than existing models on unaligned datasets. In the process, we also manage to align objects in the same category.

1. Introduction

Currently, to efficiently perform several 3D tasks, we require shapes to be placed in canonical coordinate frame such that they are normalized for translation, scale and rotation. We would then like to capture features of these shapes which extract semantic properties such that they can generalize over several tasks. Ideally, we would like to obtain these features in an unsupervised manner without any manual labels. Furthermore, these features should be invariant to several transformations depending on the set of tasks.

One such set of transformations are affine transformations. For shapes at the global level, there are several effective ways to normalize for translation and scale. However, rotation remains a challenge. For example, when an powerful feature extractor like an autoencoder is trained on rotated objects the embeddings obtained are very sensitive to rotations. This is shown in Figure 1. Such sensitivity, will lead to inadequate performance on a task such as shape comparison.

In this work, we propose a neural net architecture which creates powerful features of rotated 3D shapes that are invariant to rotational transformations across a particular axis. We focus on one axis (z axis) because most 3D data in real world software is upright and orientated along that axis. The features are generated in an unsupervised manner so that the features do not over-fit to a particular task, and generalize over several downstream tasks. In the process of creating these features, the model manages to align all objects in a category.

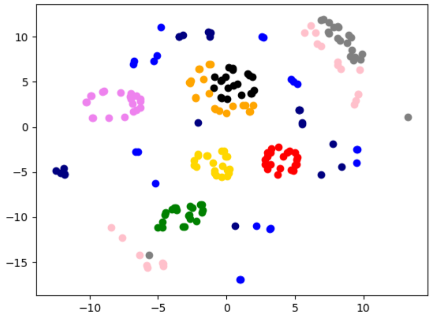

Figure 1. T-SNE clustering plot on embeddings of 10 distinct different objects randomly rotated 20 times around z axis. If we cluster these objects, the score is 0.851 as measured by AMI.

2. Related Work

There is a rich literature on getting rotational invariant features and representations. A lot of work is focused on local feature methods that have rotation invariance such as in [7], [12] and [11]. In deep learning methods, approaches such as [3] and [6] have been proposed which again use local features or interest points to get rotational invariant 3D descriptors. In shape analysis and at a global level, Kazhdan *et al.* [4] uses spherical harmonics to construct a rotation invariant representation of objects. Despite these representations being rotationally invariant, these methods do not reconstruct or use the actual 3D shape or scene, so it is possible that some important semantic information of shape or scene is lost in the process.

In 2D vision, a lot of methods have been used to achieve rotation equivariance by either constraining the filter structure, such as in Harmonic Networks [16] or by using filter orbits, such as in [2]. These ideas have been recently adopted by the 3D community, in works such as [14] and [15]. The work most similar to us is MVCNN [13] and Rot-

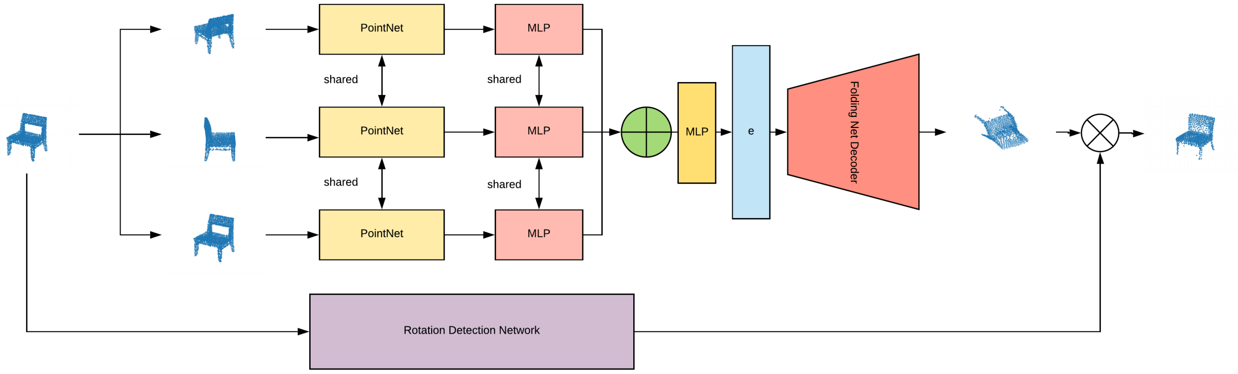

Figure 2. The architecture of the proposed network. It comprises of three sub-networks: Rotation Invariant Encoder, Folding Net based Decoder and Rotation Detection Network.

SO-Net [5]. However, most of these approaches are task specific, such as for classification or segmentation. It is not clear how these approaches can be expanded to unsupervised learning.

3. Rotation Invariant Embeddings

We assume the dataset we want to learn on is unaligned and rotated around a particular axis. The rotation group we consider is a discrete set of rotations from 0 to 360 degrees with quantization bins of 1 degree. For each shape, we create a rotation set, S, with a predetermined number of rotations, R (hyperparameter). We then feed the shape data point and it's rotation set to the model. If the original 3D shape is x and a rotation transformation is T_{ri}, the set of data point, S, is -

$$S = \{T_{r0}x, T_{r1}x, ..., T_{rR}x\} \tag{1}$$

Our model comprises of three neural nets: Rotation Invariant Encoder, Folding Net Decoder and a Rotation Detection Network. The overall architecture is shown in Figure 2. We use the encoder to obtain the rotational invariant representation by using a set invariant function over random rotations of the object, x. We then use this embedding and decode it to a shape with a learned orientation. To train the network in an unsupervised manner, we transform this learned shape to the original objects orientation using the Rotation Detection Network. The three nets are described in the below sections in more detail.

3.1. Rotation Invariant Encoder

The goal of the encoder is take the set of rotations, S, and create a vector embedding, e, which will be invariant to the order of the set. As the set represents the rotations of an object, we can obtain rotation invariant representation of the shape by using a set invariant function, such as the sum or max function. In principle, we need to aggregate over all possible rotations in the group. We approximate this by selecting small number of rotations, R, and summing over them.

First, we define a representation function, h, on each object in the set. We use the pointnet [10] based encoder which takes an orderless point cloud and creates the shape embedding. We apply h on each object in the set to produce orientation embeddings.

$$\{o_1, o_2, ..., o_R\} = \{h(T_{r0}x), h(T_{r1}x), ..., h(T_{rR}x)\} \tag{2}$$

We take all the orientation embeddings for the rotation set, S, and apply multilayer perceptron (MLP) layers, g. Finally, we take all the transformed orientation embeddings and apply the maxpool aggregation function. This ensures the encoder is approximately rotationally invariant and does not contain orientation information of the 3D shape. We also apply layers of MLPs, f, on this pooled representation to get the final invariant rotation embedding, e.

$$e = f(\max_{i=1,..,R}(g(o_i))) \tag{3}$$

3.2. Folding Net Decoder

We take the rotational invariant embedding, e, and use a decoder function, d, to produce a 3D shape, $\hat{p}$, with an unsupervised learned orientation. We will only produce one orientation for the shape so that the decoder network will produce the same orientation for different shapes in the same category. The intuition here is that it will be easier for the network to align all similar shapes rather then have to learn different orientations for each of them.

$$\hat{p} = d(e) \tag{4}$$

Method (train / test) + R	AMI (train)	AMI (test)
PN-FD (aligned / aligned)	0.772	0.739
PN-FD (unaligned / unaligned)	0.652	0.627
Ours (unaligned / unaligned) + 1	0.724	0.722
Ours (unaligned / unaligned) + 2	**0.753**	**0.726**

Table 1. **Clustering Results** Our model outperforms models trained and tested on unaligned objects by a significant margin and almost reaches the accuracy of model trained and tested on aligned objects.

The decoder we use is based on the folding net architecture [18] using two folding operations.

3.3. Rotation Detection Network

Rotation detection network, rn, takes the 3D shape data point, x, and detects the orientation of this data point with the respect to the 3D shape, $\hat{p}$. The network comprises of a pointnet encoder and MLP which takes in x and produces a normalized quaternion.

$$\hat{q} = rn(x) \tag{5}$$

$$\hat{x} = \hat{p}\hat{q} \tag{6}$$

The quaternion is multiplied to each point of the shape $\hat{p}$ as shown in Equation 6. Finally, we calculate the loss using Chamfer distance between x and $\hat{x}$ [1].

4. Experiments

4.1. Clustering

One of the goals of unsupervised learning is that the embeddings extracted from data can be easily clustered. We use the K-means algorithm in sklearn [9] with init parameter equalling k-means++, k set to 10 and n_inits set to 1000. We then use the adjusted mutual information metric (AMI) to see how the clusters aligns with the true labels. Our experiments use the ModelNet10 dataset [17]. We rotate the dataset around the z axis to create unaligned shapes. We use two baseline models. First, we train a pointnet encoder and folding net decoder based autoencoder (PN-FD) with aligned object. Note, for this case we train and test on aligned objects. The second baseline we use is the same model trained on unaligned objects. We also train our model on unaligned objects. We test the second baseline and our model on unaligned test set of ModelNet10. First, we test clustering of the training embedding with the AMI metric. Next, we use the k-means algorithm to predict on test embeddings and use the AMI metric. The results are reported in Table 1.

4.2. Rotation Invariance

To test if the representation changes with rotations, we select distinct 10 shapes from ModelNet10 which are unaligned and rotate them randomly 20 times around the z axis. We then randomize the order of all shapes and then use T-SNE [8] visualization as shown in Figure 3. We compare this with the PN-FD autoencoder trained on unaligned objects as shown in Figure 1. It can be clearly seen that our model makes better clusters and has a AMI score of 1 compared to 0.85 for the baseline model.

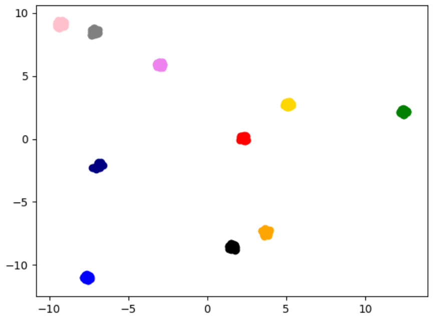

Figure 3. T-SNE clustering plot on embeddings created by our autoencoder. We get a score of 1.0 AMI. This is perfect clustering.

Method (train / test)	ModelNet10	ModelNet40
PN-FD (aligned / aligned)	92.07 %	86.99 %
PN-FD (unaligned / unaligned)	79.00 %	65.10 %
Ours (unaligned / unaligned)	**84.69** %	**74.43** %

Table 2. **Transfer Learning Results** Our model again outperforms models trained and tested on unaligned objects.

4.3. Alignment

We provided illustrations to show how our model aligns objects in the intermediate step, $\hat{p}$. This is shown in Figure 4. Based on empirical results, the model aligns objects in the same category. We use ModelNet10 dataset for these experiments.

4.4. Transfer Learning

To see the further effectiveness of our model we tested our model on transfer learning. First, we trained different models on Shapenet as specified earlier. Then, we obtained training embeddings of ModelNet10/40 from the autoencoder and trained a Linear SVM with default parameters in sklearn [9]. Then, we used the test embeddings and predicted the labels using the trained SVM. To make it fair for baseline models (trained on unaligned) we increased the dataset by R times. We set R at 2 for all tests in this section. The results are shown in Table 2.

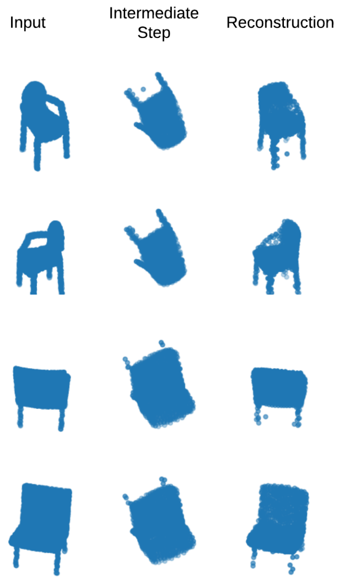

Figure 4. The objects are aligned to the same orientation in the intermediate step before changing to the desired input orientation.

5. Conclusion

In this work, we propose a neural network architecture that produces rotation invariant embeddings around an axis in an unsupervised manner. We showed how this model performs significantly better than current models trained and tested on unaligned objects over several experiments.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Representation learning and adversarial generation of 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2(3):4, 2017.

[2] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999, 2016.

[3] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 602–618, 2018.

[4] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, volume 6, pages 156–164, 2003.

[5] Jiaxin Li, Yingcai Bi, and Gim Hee Lee. Discrete rotation equivariance for point cloud recognition. *arXiv preprint arXiv:1904.00319*, 2019.

[6] Jiaxin Li and Gim Hee Lee. Usip: Unsupervised stable interest point detection from 3d point clouds. *arXiv preprint arXiv:1904.00229*, 2019.

[7] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[8] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[9] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[10] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, volume 11, page 2. Citeseer, 2011.

[12] Bastian Steder, Radu Bogdan Rusu, Kurt Konolige, and Wolfram Burgard. Narf: 3d range image features for object recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, volume 44, 2010.

[13] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[14] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

[15] Daniel Worrall and Gabriel Brostow. Cubenet: Equivariance to 3d rotation and translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 567–584, 2018.

[16] Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5028–5037, 2017.

[17] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[18] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018.