ROTATION-INVARIANT SPHERICAL AUTOENCODERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Omnidirectional images and spherical representations of $3D$ shapes cannot be processed with conventional 2D convolutional neural networks (CNNs) as the unwrapping leads to large distortion. Using fast implementations of spherical and $SO(3)$ convolutions, researchers have recently developed deep learning methods that are equivariant to 3D rotations, and thus better suited to operate on such data. In this paper, we consider the problem of unsupervised learning of rotation-invariant representations for spherical images. In particular, we design an autoencoder architecture consisting of S^2 and $SO(3)$ convolutional layers. As 3D rotations are often a nuisance factor, the latent space is constrained to be exactly invariant to them. As the rotation information is discarded in the latent space, we craft a novel rotation-invariant loss function for training the network. Extensive experiments on multiple datasets demonstrate the usefulness of the learned representations on clustering, retrieval and classification applications.

1 INTRODUCTION

Conventional deep learning techniques for regular images i.e. 2D convolutional neural networks (CNNs), are not suitable for processing spherical images. This is because, in order to employ the regular 2D CNN, the signal on the sphere must be first unwrapped onto a uniform two-dimensional grid, which introduces a large amount of distortion throughout the image. To this end, Cohen et al. (2018) and Esteves et al. (2018) independently designed convolutional neural networks that can operate directly on signals on spheres. Both these methods rely on defining a correlation function between signal on a sphere and a trainable kernel, which is also a signal on the same sphere. This can be achieved computationally efficiently using spherical harmonics. An important feature of this correlation operation is that it is *equivariant* to 3D rotations of the sphere. These spherical convolutional layers can then be stacked along with non-linearities to form deep neural networks for spherical signals. These networks have so far mainly been employed for classification problems where a spherical image is mapped to one of many predefined classes. Trained in a supervised fashion, they generally require a lot of labeled data.

In this paper, we propose an unsupervised framework for spherical images. In particular, we design a novel autoencoder architecture such that the latent features remain exactly invariant to any rotation of the input signal. This is because, in many applications, *the inputs do not have a canonical rotation, and the particular orientation that the input signal is measured in is a nuisance factor and should ideally be factored out*. However, as the latent space is rotation-invariant, the usual loss functions cannot be employed for training the proposed autoencoder. We design a novel loss function, based on the spherical cross-correlation between the estimated output and the desired output, which is also invariant to the relative orientation of the estimated output with respect to the desired output. Related work and necessary background are discussed in Appendices A and B.

2 ROTATION-INVARIANT AUTOENCODER FOR SPHERICAL IMAGES

Similar to the usual convolutional encoder for regular images, an input spherical image $f^{(0)} : S^2 \to \mathbb{R}^{C_0}$, where C_0 is the number of channels, is passed through multiple stacked convolutional layers each followed by non-linearities. The first convolutional layer performs a correlation on S^2, between $f^{(0)}$ and the filters $\{h_i^{(1),i=1,\dots,C_1}\}$, where $h_i^{(1)} : S^2 \to \mathbb{R}^{C_0}$. We perform spherical correlation between the input spherical image and the first layer filters, and apply a non-linearity producing the output of the first layer – a signal $f^{(1)} : SO(3) \to \mathbb{R}^{C_1}$. The rest of the convolutional layers in the encoder are $SO(3)$ correlations where the filters of layer $d, d > 1$, $\{h_i^{(d)}, i = 1, \dots, C_d\}$, where

$h_i^{(d)} : SO(3) \to \mathbb{R}^{C_{d-1}}$. After the input is fed through the D convolutional layers producing the feature map $f^{(D)} : SO(3) \to \mathbb{R}^{C_D}$. We use an integration layer to create the rotation-invariant $f_{inv} \in \mathbb{R}^D$ given by

$$f_{inv}(i) = \int_{SO(3)} f^{(D)}(X) dX, \quad i = 1, \ldots, D. \tag{1}$$

Once we have a rotation-invariant representation, we can use further fully connected layers to map to the latent rotation-invariant representation which we denote by $\mathbf{z} \in \mathbb{R}^N$.

The purpose of the decoder is to map the latent representation back to the original input space. The first layer of the decoder is a fully connected layer that maps $\mathbf{z}$ to a function f on $SO(3)$. We then use a series of $SO(3)$ convolutional layers, to map to $g : SO(3) \to \mathbb{R}^{C_0}$. Note that $SO(3)$ correlations produce outputs which are functions on $SO(3)$. In order to construct the desired function on S^2, we use another integration layer which integrates over γ: $\hat{f}(\alpha, \beta) = \int_{\gamma=0}^{2\pi} g(\alpha, \beta, \gamma) d\gamma$. Filters in the various S^2 and $SO(3)$ convolutional layers are the trainable parameters of the autoencoder, for which we also need to specify an appropriate loss function.

2.1 Proposed rotation-invariant loss function

As the baseline in our experiments, we employ the ubiquitously used L2 loss, denoted by L_b. However, this is expected to fail because all the rotation information is lost at the latent space. We propose a novel loss function that overcomes this disadvantage:

$$L(f, \hat{f}) = \int_{S^2} |f(x)|^2 dx + \int_{S^2} |\hat{f}(x)|^2 dx - 2 \max_{SO(3)} \{f \star \hat{f}\}, \tag{2}$$

where the max operator is over the $SO(3)$ function that is the output of the spherical cross-correlation operator $\star$. Essentially, the function finds the best rotation $R \in SO(3)$ such that the difference between $f(R^{-1}x)$ and $\hat{f}(x)$ is minimized. The key point to note is that the loss function is also rotation-invariant, as it solves for the optimal alignment before computing the usual squared error loss. That is, the spherical correlation function, $\star$, has the property that $\max_{SO(3)}\{f \star g\} = \max_{SO(3)}\{\mathcal{R}_1(f) \star \mathcal{R}_2(g)\}$, where $\mathcal{R}_1(f)$ and $\mathcal{R}_2(g)$ are arbitrarily rotated versions of f and g. This is crucial for obtaining useful gradients while training the rotation-invariant autoencoder. While training the network, we compute this loss over a mini-batch and average it, as is normally done. We can employ the Fourier correlation theorem to compute the above spherical cross-correlation efficiently. Note that (a) equation 2 is a natural generalization of the L2 loss that takes rotation-invariance into account and (b) we cannot use a *normalized* spherical cross-correlation as we want to reconstruct the inputs at the same energy level. The first two terms of equation 2 are required to ensure that the outputs have the same energy as the inputs. Variations like using a soft version of the max operator can also be considered, however we find that the above loss function supplies informative gradients for training.

3 Experimental results

We conduct experiments on the proposed rotation-invariant autoencoder using publicly available datasets. The goal of these experiments is not to achieve state-of-the-art performance on any dataset. Instead, it is to show that compared to a vanilla autoencoder trained without using the proposed loss function, the proposed rotation-invariant framework is significantly better in terms of reconstruction performance, as well as clustering and classification metrics for multiple application domains. This makes an important advance in invariant unsupervised representation learning for spherical signals. We present several experimental results on Spherical MNIST, SHREC17 and ModelNet40. Here we only present some results on Spherical MNIST and SHREC17 with the rest in the appendix.

3.1 Spherical MNIST image recognition

Dataset details: We use a spherical version of MNIST, called spherical MNIST, by Cohen et al. Cohen et al. (2018). It consists of 60000 training and 10000 test examples. The bandwidth of the signals is set to $b = 30$. This means that the size of the spherical signal is 60×60, i.e., the spherical signal is sampled at 60 values of $\alpha \in [0, 2\pi]$ and $\beta \in [0, \pi]$ using the Driscoll-Healy grid Driscoll & Healy (1994).

Train/test setting	Vanilla AE (L2 loss)	Proposed AE Rotation-invariant loss
NR/NR	18.24	22.46
R/R	10.35	22.52
NR/R	14.33	22.41

Table 1: Reconstruction results on Spherical MNIST, in terms of PSNR (dB), for different settings and variants of spherical autoencoders. We see that the proposed method outperforms the vanilla AE for all settings. We also observe that the proposed framework yield stable performance over all the settings as expected, as the latent representations are exactly rotation-invariant.

Method	Purity	Homogeneity	Completeness	Classification Accuracy (%)
Vanilla AE Train NR/Test R	0.37	0.27	0.29	86.58
Vanilla AE Train R/Test R	0.35	0.25	0.27	74.37
Rotation-invariant AE Train NR/Test R	**0.40**	**0.39**	**0.31**	**89.42**

Table 2: Clustering and classification results on Spherical MNIST. Clearly, the proposed method outperforms the other unsupervised baselines in all metrics. By comparison, our implementation of S^2CNN (described in the text) Cohen et al. (2018), which is a fully supervised deep classifier, yields 94% classification accuracy.

Training details: The exact architecture and hyperparameter values are given in Appendix C. We train two types of networks – (a) Vanilla AE (the baseline) Here, the spherical autoencoder trained using the regular L2 loss given by $L_{euc}(f, \hat{f}) = ||f - \hat{f}||_2^2$ and (b) Rotation-invariant AE trained using the proposed rotation-invariant loss function. These networks are trained for three different settings – (a) Train NR/Test NR: both the training and test sets are aligned and have the same orientation (b) Train R/Test R: Both the train and test set have random rotations applied to them and (c) Train NR/Test R: The training set is aligned while the test set has random rotations applied to it. That is, in this setting, the trained network is subject to unseen rotations at test time, and is the most challenging setting.

Results: First, we compare the various methods based on the reconstruction peak signal-to-noise ratio (PSNR) between the input spherical image and the reconstructed image for all the examples in the test set. The results are shown in Table 2. We easily see that the proposed autoencoder trained using the rotation-invariant loss function outperforms the vanilla autoencoder, trained with just the L2 loss, for all cases. It is especially important to note that, compared to Train NR/Test NR for the case of Train NR/Test R, the vanilla autoencoder (trained with the L2 loss) completely fails, while rotation-invariant autoencoder's performance is not affected.

Second, we gather the latent features for both the train and test images for the various settings using both the vanilla and the proposed rotation-invariant autoencoder. Then, we train a simple multi-class linear classifier using a single fully-connected layer to map from the 120-dimensional latent feature to a probability distribution over the 10 classes. We measure the classification accuracy on the held out test set. As shown in Table 2, we can see that the proposed framework is significantly better than the vanilla spherical autoencoder. We also compare our method with the purely supervised approach of Cohen et al. Cohen et al. (2018). For this, we construct a deeper spherical CNN classifier with a S^2 convolutional layer followed by a $SO(3)$ convolutional layer, the pooling layer and a fully connected layer. We see that the features obtained by unsupervised learning have a significant amount of discriminative information and perform only a little worse than the fully supervised technique. We also show evidence of rotation-invariance in Figure 2 and perform few-shot learning and t-SNE visualizations which are all given in Appendix C and clearly show the usefulness of the proposed method.

3.2 SHREC17 SHAPE RETRIEVAL AND RECOGNITION

Dataset details: The dataset consists Savva et al. (2017) of 51300 3D meshes belonging to 55 classes such as chairs, tables and airplanes. For the experiments, we use the variant of the dataset where both the training and test set are randomly rotated. The 3D meshes are centered at the origin and are projected onto a sphere around the mesh using ray casting. Rays are cast from the origin passing through the mesh and hitting the sphere. The distance between the point on the mesh and

(a) Input (ground-truth) spherical images

(b) Reconstructed spherical images

Figure 1: Visualization of the ground-truth spherical images (a) which are also the input to the autoencoder and the reconstructed images (b) from the proposed rotation-invariant autoencoder trained using the rotation-invariant loss function. Note that due to rotation-invariance in the latent space, the reconstruction can be achieved only up to a 3D rotation, as seen in the figure. The inputs are shown to be aligned only for the figure here for easy visualization. In reality, random rotations were applied to them before feeding into the network.

the sphere is recorded as the signal on the sphere, and forms the spherical representation of the 3D mesh. The Driscoll-Healy grid is used to sample the sphere with a bandwidth $b = 30$, i.e. the spherical signals can be stored as 60×60 arrays.

Autoencoder architecture and training details: The autoencoder architecture and traninig protocol are the same as in Spherical MNIST. We train two types of networks – (a) Vanilla AE (the baseline) Here, the spherical autoencoder trained using the regular L2 loss and (b) Rotation-invariant AE trained using the proposed rotation-invariant loss function.

Method	Purity	Homogeneity	Completeness	Classification Accuracy (%)
Vanilla AE	0.30	0.26	0.22	48.80
Rotation-invariant AE (proposed)	0.41	0.38	0.31	57.76

Table 3: Clustering and classification results on SHREC17. Clearly, the proposed method outperforms the other unsupervised baselines in all metrics. By comparison, our implementation of S^2CNN (with 3 conv layer) Cohen et al. (2018), which is a fully supervised deep classifier, yields 65.42% classification accuracy.

Type	Method	P@N	R@N	F1@N	mAP	NDCG
Supervised	S^2CNN Cohen et al. (2018)	0.701	0.711	0.699	0.676	0.756
Unsupervised	Vanilla AE	0.075	0.092	0.066	0.008	0.064
	Rotation-invariant AE (proposed)	0.351	0.361	0.335	0.215	0.345

Table 4: Shape retrieval results on SHREC17 compared to fully supervised method. We see that even the unsupervised features obtained using our proposed method with the rotation-invariant loss function yields good results, especially when compared to the vanilla spherical autoencoder with the L2 loss function. The results for the supervised method Savva et al. (2017) are taken from Cohen et al. (2018). For further comparison with more methods, more results on this dataset can be found in Esteves et al. (2019).

Results: First, we compute the latent features for both the training and test set 3D shape spherical representations. The latent features are then used to train a simple linear classifier. The results are shown in Table 3. We see that the proposed framework yields higher classification accuracy, compared to vanilla AE. Second, as in the case of Spherical MNIST, we conduct similar clustering experiments by performing k-means clustering with $k = 55$. We report purity, homogeneity and completeness in Table 3. We can clearly observe improved performance with the proposed method. Third, we conduct shape retrieval on the held out test set following the protocol in Savva et al. (2017). The results are reported in Table 4 for various retrieval metrics and comparison with fully supervised methods is also provided. We observe that the features obtained through the proposed method work reasonably well for shape retrieval.

REFERENCES

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pp. 40–49. PMLR, 2018.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

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

Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral cnn. In *ICML*, 2019.

Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *International Conference on Learning Representations*, 2018.

Pim de Haan, Maurice Weiler, Taco Cohen, and Max Welling. Gauge equivariant mesh cnns: Anisotropic convolutions on geometric graphs. *arXiv preprint arXiv:2003.05425*, 2020.

Samuel Dodge and Lina Karam. Understanding how image quality affects deep neural networks. In *2016 eighth international conference on quality of multimedia experience (QoMEX)*, pp. 1–6. IEEE, 2016.

James R Driscoll and Dennis M Healy. Computing fourier transforms and convolutions on the 2-sphere. *Advances in applied mathematics*, 15(2):202–250, 1994.

Carlos Esteves. Theoretical aspects of group equivariant neural networks. *arXiv preprint arXiv:2004.05154*, 2020.

Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. Learning so (3) equivariant representations with spherical cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 52–68, 2018.

Carlos Esteves, Yinshuang Xu, Christine Allen-Blanchette, and Kostas Daniilidis. Equivariant multi-view networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1568–1577, 2019.

Carlos Esteves, Ameesh Makadia, and Kostas Daniilidis. Spin-weighted spherical cnns. *Advances in Neural Information Processing Systems*, 33, 2020.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *arXiv preprint arXiv:2004.07780*, 2020.

Ian Goodfellow, Honglak Lee, Quoc Le, Andrew Saxe, and Andrew Ng. Measuring invariances in deep networks. *Advances in neural information processing systems*, 22:646–654, 2009.

Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *International conference on artificial neural networks*, pp. 44–51. Springer, 2011.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pp. 2017–2025, 2015.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *International Conference on Machine Learning*, pp. 2747–2755, 2018.

Risi Kondor, Zhen Lin, and Shubhendu Trivedi. Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. In *Advances in Neural Information Processing Systems*, pp. 10117–10126, 2018a.

Risi Kondor, Hy Truong Son, Horace Pan, Brandon Anderson, and Shubhendu Trivedi. Covariant compositional networks for learning graphs. *arXiv preprint arXiv:1801.02144*, 2018b.

Kaushik Koneripalli, Suhas Lohit, Rushil Anirudh, and Pavan Turaga. Rate-invariant autoencoding of time-series. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3732–3736. IEEE, 2020.

Peter Kostelec and Daniel Rockmore. Soft: So(3) fourier transforms. 2007.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 2012.

Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *Advances in neural information processing systems*, pp. 2539–2547, 2015.

Denis Kuzminykh, Daniil Polykovskiy, and Alexander Zhebrak. Extracting invariant features from images using an equivariant autoencoder. In *Asian Conference on Machine Learning*, pp. 438–453, 2018.

Suhas Lohit, Qiao Wang, and Pavan Turaga. Temporal transformer networks: Joint learning of invariant and discriminative time warping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12426–12435, 2019.

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

Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. In *International Conference on Learning Representations*, 2018.

Haggai Maron, Or Litany, Gal Chechik, and Ethan Fetaya. On learning sets of symmetric elements. *International Conference on Machine Learning*, 2020.

Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Deep learning with sets and point clouds. *International Conference on Learning Representations - Workshop track*, 2017.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pp. 3856–3866, 2017.

Manolis Savva, Fisher Yu, Hao Su, Asako Kanezaki, Takahiko Furuya, Ryutarou Ohbuchi, Zhichao Zhou, Rui Yu, Song Bai, Xiang Bai, Masaki Aono, Atsushi Tatsuma, S. Thermos, A. Axenopoulos, G. Th. Papadopoulos, P. Daras, Xiao Deng, Zhouhui Lian, Bo Li, Henry Johan, Yijuan Lu, and Sanjeev Mk. Large-scale 3d shape retrieval from shapenet core55: Shrec'17 track. In *Proceedings of the Workshop on 3D Object Retrieval*. Eurographics Association, 2017.

Zhixin Shu, Mihir Sahasrabudhe, Riza Alp Guler, Dimitris Samaras, Nikos Paragios, and Iasonas Kokkinos. Deforming autoencoders: Unsupervised disentangling of shape and appearance. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 650–665, 2018.

Ankita Shukla, Sarthak Bhagat, Shagun Uppal, Saket Anand, and Pavan Turaga. Product of orthogonal spheres parameterization for disentangled representation learning. *British Machine Vision Conference*, 2019.

Erik Henning Thiede, Truong Son Hy, and Risi Kondor. The general theory of permutation equivarant neural networks and higher order graph variational encoders. *arXiv preprint arXiv:2004.03990*, 2020.

Maurice Weiler, Fred A Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 849–858, 2018.

Daniel Worrall and Max Welling. Deep scale-spaces: Equivariance over scale. In *Advances in Neural Information Processing Systems*, pp. 7366–7378, 2019.

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*, pp. 5028–5037, 2017.

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*, pp. 1912–1920, 2015.

Wenju Xu, Guanghui Wang, Alan Sullivan, and Ziming Zhang. Towards learning affine-invariant representations via data-efficient cnns. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 904–913, 2020.

Yichong Xu, Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, and Zheng Zhang. Scale-invariant convolutional neural networks. *arXiv preprint arXiv:1411.6369*, 2014.

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*, pp. 206–215, 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pp. 3391–3401, 2017.

Binbin Zhang, Wen Shen, Shikun Huang, Zhihua Wei, and Quanshi Zhang. 3d-rotation-equivariant quaternion neural networks. *European Conference on Computer Vision*, 2020.

A Related Work

Before briefly reviewing some related work on invariant/equivariant deep learning, we first define equivariance and invariance with respect to a set of transformations $\mathcal{T}$. Consider, for simplicity, functions $f, g : L^2(\mathbb{X}) \to L^2(\mathbb{X})$. For all $x \in \mathbb{X}$, where $\mathbb{X}$ is some space on which the functions are defined. $f(\cdot)$ is invariant with respect to $\mathcal{T}$ if $\forall T \in \mathcal{T}, f \circ T(x) = f(x)$. Similarly, $g(\cdot)$ is equivariant with respect to $\mathcal{T}$ if $\forall T \in \mathcal{T}, g \circ T(x) = T \circ g(x)$.

A.1 Invariant representations using deep learning

While deep learning has afforded significant gains in various computer vision applications, it inevitably requires large amounts of training data and computational resources Krizhevsky et al. (2012), besides being sensitive to small perturbations in the input Dodge & Karam (2016). At the same time, overparameterized models, which are a mainstay of deep learning methods, are prone to learning mappings based on spurious correlations Arjovsky et al. (2019), also known as shortcuts Geirhos et al. (2020). Goodfellow et al. Goodfellow et al. (2009) showed that while regular CNNs do provide some translational invariance, invariance to other transformations cannot be guaranteed. To remedy this drawback, various approaches have been studied recently. These include *hybrid* model- and data-driven approaches such as spatial Jaderberg et al. (2015) and temporal transformers Lohit et al. (2019) which use specially designed modules to model nuisance factors, capsule networks for autoencoding by Hinton et al. Hinton et al. (2011) and for classification by Sabour et al. Sabour et al. (2017) which explicitly model pose variations, and other works which modify the convolutional layers for added scale Xu et al. (2014) and affine invariances Xu et al. (2020). In the realm of unsupervised representation learning, which is the focus of this paper, several disentangling methods have been proposed. In these methods, the elements of latent vectors are encouraged to represent disentangled factors of variation. That is, if an element of the latent vector, which encodes a particular factor of variation, is removed, the rest of the vector remains invariant to that factor. Examples in computer vision include the works by Kulkarni et al. Kulkarni et al. (2015) which disentangles factors of variation for face images, Shukla et al. Shukla et al. (2019) who propose using orthogonal latent spaces to encourage disentanglement, Shu et al. Shu et al. (2018) and Koneripalli et al. Koneripalli et al. (2020) who use spatial and temporal transformer modules in autoencoders in order to encourage affine-invariant representations for images and rate-invariant representations for human motion sequences. Permutation-invariant unsupervised representations for point-clouds have also been effective Achlioptas et al. (2018); Yang et al. (2018).

A.2 Equivariant neural networks

In addition to the above methods, a new promising class of architectures has emerged which, unlike previous methods, can *guarantee exact invariance* to certain transformations by design. The general architecture is as follows. There is a stack of *equivariant* layers which operate on the input, followed by a pooling/aggregation layer which promotes the representations to be fully *invariant*. In the case of classification architectures, the invariant features are then passed through the layers of a classifier head.

As mentioned in the introduction, the CNNs used for regular images are equivariant to 2D translations. Cohen and Welling Cohen & Welling (2016) described Group-CNNs which are equivariant to the action of discrete rotations on images. This idea has been applied for autoencoders by Kuzminykh et al. Kuzminykh et al. (2018). These ideas were then extended to the case of continuous rotations by Weiler et al. Weiler et al. (2018) and to both rotations and translations by Worrall et al. Worrall et al. (2017). Kondor and Trivedi Kondor & Trivedi (2018) showed theoretically that a convolutional structure is actually necessary, and not just sufficient, in order to guarantee equivariance to actions of compact groups on signals. The idea of CNNs has been extended for the case of spherical images by several authors. Cohen et al. Cohen et al. (2018) and Esteves et al. Esteves

et al. (2018) concurrently developed spherical CNNs based on spherical correlation layers, which are computed efficiently using the spherical and SO(3) Fourier transform (SOFT) Kostelec & Rockmore (2007). Kondor et al. proposed Clebsch-Gordan Nets Kondor et al. (2018a) which operated fully in the Fourier domain and is computationally more efficient. More recently, Esteves et al. Esteves et al. (2020) designed spin-weighted CNNs which are computationally more efficient compared to Cohen et al. (2018) and at the same time, do not sacrifice representation power unlike the isotropic filters used in Esteves et al. (2018). In this paper, we employ the layers proposed by Cohen et al. Cohen et al. (2018), however the ideas presented in this paper are applicable to these other architectures as well. Similar ideas have be used to design network layers equivariant to scaling and blurring operators on images Worrall & Welling (2019), rotation equivariance for point clouds Zhang et al. (2020), learning on sets Ravanbakhsh et al. (2017); Zaheer et al. (2017); Maron et al. (2020) and graphs Kondor et al. (2018b); Maron et al. (2018); Thiede et al. (2020), gauge equivariant transforms for spheres Cohen et al. (2019) and meshes de Haan et al. (2020). Existing literature on equivariant networks has mainly focused on supervised learning and classification problems. *In this paper, we propose a spherical CNN-based autoencoder for spherical images with a rotation-invariant latent space.* We now describe the background necessary to develop this framework.

B BACKGROUND: CORRELATIONS ON S^2 AND $SO(3)$

In this section, we provide a brief overview of some of the required mathematical preliminaries concerning correlations of signals defined on the unit sphere S^2 and the group of 3D rotations $SO(3)$. We direct the interested reader to the excellent article by Esteves Esteves (2020) for a longer treatment of these ideas. Briefly, by the unit sphere S^2, we mean the set of points $x \in \mathbb{R}^3$ such that $||x||_2^2 = 1$. This is a 2-dimensional space and can be equipped with a Riemannian manifold structure with the usual inner product as the metric. As such, the points on the sphere can be specified using a two-dimensional spherical co-ordinate system say $\alpha \in [0, 2\pi], \beta \in [0, \pi]$. The set of 3D rotations, denoted by $SO(3)$, is a three-dimensional space which can be specified using the popular convention of ZYZ Euler angles given by $\alpha \in [0, 2\pi], \beta \in [0, \pi], \gamma \in [0, 2\pi]$, which quantify the rotation around the Z-, X- and Z- axes respectively performed in order. Note that several such conventions exist and we can easily transform the coordinates from one system to another. Signals/images on S^2 and $SO(3)$ can be described as functions $f : S^2 \to \mathbb{R}$ and $g : SO(3) \to \mathbb{R}$ respectively. Also note that filters in the layers as well as the output feature maps are such functions.

We can now define a correlation – convolution, in deep learning parlance – between a spherical image f on the sphere with a filter h, which is also a function on the sphere as follows:

$$h \star f(R) = \int_{S^2} h(R^{-1}x) f(x) dx, \tag{3}$$

where $\star$ refers to S^2 correlation, $R \in SO(3)$ is a rotation matrix, dx is a rotation invariant measure of integration given by $d\alpha \sin(\beta) d\beta / 4\pi$. Note that (a) by this definition of correlation, the output of the correlation operator for two signals on S^2 is a signal on $SO(3)$. (b) this is, by no means, the only way of defining cross-correlation on a sphere. Equation equation 3 is followed in this paper following Cohen et al. Cohen et al. (2018). Esteves et al. Esteves et al. (2018) define a different correlation operator where the output is still a function on S^2. They show that their definition restricts the filters h in equation 3, which are eventually learned from data, to be isotropic/zonal filters. This can increase computational efficiency, but at the cost of reduced expressivity.

Similarly, for a function f and a filter h on $SO(3)$, the $SO(3)$ correlation is defined as

$$h \star f(R) = \int_{SO(3)} h(R^{-1}X) f(X) dX, \tag{4}$$

where $\star$ now denotes correlation on $SO(3)$, dX is the measure of integration given in ZYZ Euler angles by $d\alpha \sin(\beta) d\beta d\gamma / 8\pi^2$. Note that these definitions of correlations are equivariant to the rotation group $SO(3)$. That is, if the function f undergoes a rotation R, the correlation output undergoes the same rotation. Please refer Cohen et al. Cohen et al. (2018) and Kondor et al. Kondor et al. (2018a) for more details.

Now, we discuss how to compute these cross-correlations efficiently using spherical and $SO(3)$ Fourier transforms. Recall that we compute the Fourier coefficients for a signal on S^2 by projecting it onto the spherical harmonics, the elements of which are denoted by $Y_m^l(x)$, with two indices: the degree $l \geq 0$ and order m such that $-l \leq m \leq l$. The Fourier coefficients of a bandlimited signal f with a bandwidth of b is then given by

$$\hat{f}_m^l = \int_{S^2} f(x) \bar{Y}_m^l(x) dx. \tag{5}$$

It also follows that the inverse Fourier transform, i.e., the synthesis function is given by

$$f(x) = \sum_{l=0}^{b} \sum_{m=-l}^{l} \hat{f}_m^l Y_m^l(x). \tag{6}$$

Similar to the above, in order to compute the $SO(3)$ Fourier Transform of a signal defined on the group of rotations $SO(3)$, we project it onto the *irreducible representations* of the $SO(3)$ group, which are called Wigner-D matrices, the complex elements of which are denoted by $D_{m,n}^l$. $D_{m,n}^l$ has a dimension $2l+1 \times 2l+1$ and m, n index the rows and columns of the matrix. The Fourier coefficients of a signal f defined on $SO(3)$ with a bandwidth of b is given by

$$\hat{f}_{m,n}^l = \int_{SO(3)} f(x) \bar{D}_{m,n}^l(x) dx. \tag{7}$$

The inverse $SO(3)$ Fourier transform is given by

$$f(R) = \sum_{l=0}^{b} (2l+1) \sum_{m=-l}^{l} \sum_{n=-l}^{l} \hat{f}_{m,n}^l D_{m,n}^l(R). \tag{8}$$

Now that the required Fourier and inverse Fourier transforms have been defined, we can compute the correlation on S^2 between f and h using the correlation theorem:

$$\widehat{h \star f}^l = \hat{f}^l \hat{h}^{l\dagger}. \tag{9}$$

That is, we first compute the Fourier coefficients $\hat{f}^l$ and $\hat{h}^l$ for $l = 0, \ldots, b$, which are vectors. Then we compute the outer product which gives us the Fourier transform of the correlation operator. Finally, we compute the inverse $SO(3)$ Fourier transform which gives us the desired output of the correlation, which is a function on $SO(3)$.

In a similar fashion, we can compute the $SO(3)$ correlation between a function f and a filter h, both defined on $SO(3)$ using a very similar formula as in Equation equation 9, where we use the forward $SO(3)$ Fourier transform for computing $\hat{f}$ and $\hat{h}$, which are now block matrices. They are multiplied together and then we compute the inverse $SO(3)$ transform to yield the correlation output, also on $SO(3)$. In this paper, we use the excellent Pytorch/CUDA implementations of the above functions provided by Cohen et al. Cohen et al. (2018) for building our framework. However, the algorithms are still computationally expensive to scale to higher input resolutions and we believe this is an important direction for future research (rather than resorting to approximations of the sphere).

C SPHERICAL MNIST RESULTS

Autoencoder architecture and training details: The first layer of the encoder is an S^2 convolutional layer with 20 output channels and it reduces the bandwidth to $b = 12$. This is followed by a $SO(3)$ convolutional layer with 40 output channels which further reduces the bandwidth of the feature maps to 6. Both these layers use element-wise ReLU non-linearities. We then use the pooling layer defined in Equation equation 1 to create the rotation invariant representation. We set the latent dimension, $dim(\mathbf{z}) = 120$. In the decoder, we use a fully connected layer to map $\mathbf{z}$ to a function on $SO(3)$ with $b = 6$. This is followed by 2 $SO(3)$ convolutional layers which upsample

(a) Input images (b) Reconstructed images

Figure 2: This figure shows that all rotations of an input image result in identical reconstructions which are automatically aligned. This is as expected because the architecture and the loss function jointly guarantee rotation invariant reconstruction.

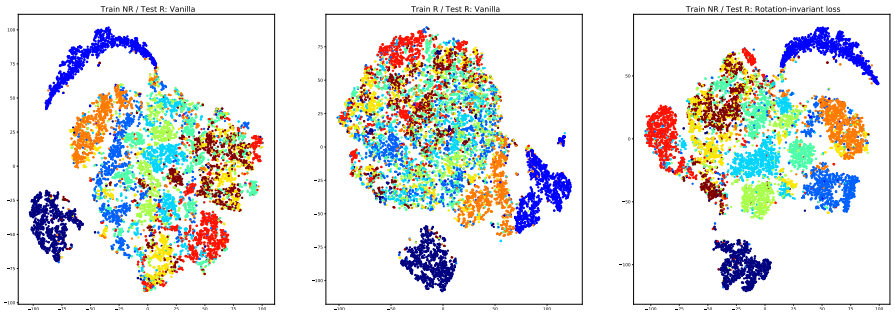

Figure 3: t-SNE visualization of test set latent features obtained from different spherical autoencoder variants we trained on Spherical MNIST for our experiments. Legend: dark blue = 0, blue = 1, light blue = 2, cyan =3, dark green = 4, light green = 5, yellow = 6, orange = 7, red = 8, brown = 9. As expected, we see a signficant overlap between the points for '6' and '9'.

the bandwidth to 12 and 30. The integration layer is used to produce the reconstructed image on the sphere. The autoencoders are trained for 20 epochs using Adam optimizer Kingma & Ba (2015) with an initial learning rate of 0.1 and a batch size of 32.

Additional results: We conduct a few-shot learning experiment to better illustrate the advantage of unsupervised feature learning. We see from Table 6 that when we have very few labeled examples for training classifiers, using the latent features of the proposed rotation-invariant autoencoder outperforms the supervised deep spherical CNN.

Third, we use the latent features for the various cases and perform both 2D visualization of the latent features using t-SNE Maaten & Hinton (2008) as well as k-means clustering with $k = 10$. The t-SNE plots are shown in Figure 3, which show that the clusters are easily observed to be more compact and homogeneous for the proposed method, compared to the vanilla autoencoder. Using the clusters from the k-means algorithm, we compute common clustering metrics – purity, homogeneity and completeness for the latent features and report them in Table 2, where we observe improved performance using the proposed method.

Effect of latent dimension In order to study the effect of the dimension of the latent space, we train autoencoders for the various settings with $dim(\mathbf{z}) = 60, 120, 240$ and record the reconstruction PSNRs on the test set. The results thus obtained are shown in Table 5.

$dim(\mathbf{z})$	Vanilla AE Train NR/Test NR	Vanilla AE Train R/Test R	Rot-inv AE Train NR/Test R
60	17.45	10.36	22.46
120	18.24	10.33	22.55
240	18.03	10.38	22.41

Table 5: Effect of dimension of the latent space measured in terms of PSNR (dB) on the test set. We see that the performance is quite stable with respect to $dim(\mathbf{z})$ and using 120 dimensions yields the best results.

Percentage of examples labeled	Deep S^2CNN (fully supervised)	Rot-inv AE features (unsupervised)
1	33.90%	**63.73%**
2	62.74%	**69.28%**
5	**80.82%**	74.76%
10	**86.32%**	79.71%
100	**95.74%**	89.42%

Table 6: Classification accuracy on the test set with few-shot learning. When there are few labeled examples, the unsupervised features can outperform the fully supervised deep S^2CNN.

D T-SNE VISUALIZATION FOR SHREC17

We use the latent features obtained from the vanilla AE and the proposed rotation-invariant AE and perform 2D visualization of the latent features using t-SNE Maaten & Hinton (2008). This is shown in Figure 4 in the appendix. We see more easily distinguishable clusters in the case of rotation-invariant AE.

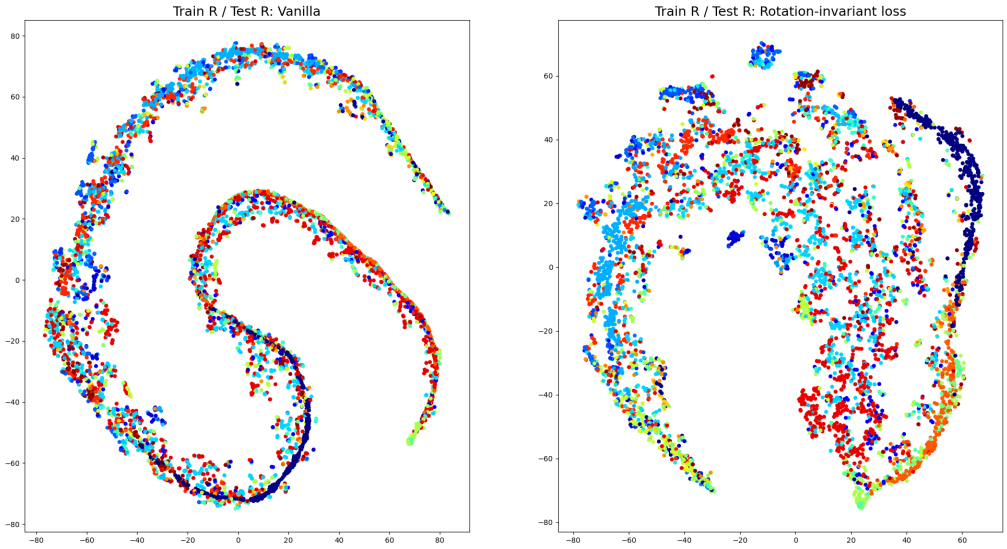

Figure 4: t-SNE visualization of test set latent features for the SHREC17 dataset. The clusters are more easily visible using the proposed method.

E SHAPE RECOGNITION ON MODELNET40

Dataset details: ModelNet40 Wu et al. (2015) consists of 3D CAD models belonging to 40 classes such as chair, sofa, nightstand and bathtub For the experiment, we choose the variant where both the training and test set are randomly rotated. In this version, the training set contains about $469,000$ examples and the test set contains about $30,000$ examples. As in the case of SHREC17, rays are cast from the origin passing through the mesh and hitting the sphere. The distance between the point on the mesh and the sphere is recorded as the signal on the sphere, and forms the spherical representation of the 3D mesh. As before Driscoll-Healy grid is used to sample the sphere with a bandwidth $b = 30$, i.e. the spherical signals can be stored as 60×60 arrays.

Autoencoder architecture and training details: The autoencoder architecture, hyperparameters and the training protocol is the same as in the case of SHREC17. We train two models – (a) the vanilla AE with the simple L2 loss and (b) the proposed AE with the rotation-invariant loss function.

Method	Classification accuracy (%)
Supervised – $\mathcal{S}^2$CNN Cohen et al. (2018)	57.04
Vanilla AE features	28.85
Rotation-invariant AE features	35.04

Table 7: Reconstruction results on Spherical MNIST, in terms of PSNR (dB), for different settings and variants of spherical autoencoders. We see that the proposed method outperforms the vanilla AE for all settings. We also observe that the proposed framework yield stable performance over all the settings as expected, as the latent representations are exactly rotation-invariant.

Results: We compute the latent features for both the training and test set 3D shape spherical representations. The latent features are then used to train a simple linear classifier. The results are shown in Table 7. We see that the proposed framework yields higher classification accuracy, compared to vanilla AE.