Permutation-Invariant Polar Harmonic Pooling for Point-based Neural Networks

Jaspreet Singh
JaSingh@lincoln.ac.uk
Grzegorz Cielniak
GCielniak@lincoln.ac.uk

Lincoln Institute for Agri-Food Technology
University of Lincoln, UK

Abstract

Point-based learning methods have recently gained significant attention for point cloud data analysis. These neural networks typically consist of a permutation-equivariant backbone network for feature extraction, a symmetric global pooling operation for feature aggregation, and a permutation-invariant classification or regression head. Most of the prior work has focused on improving the equivariant backbone while the global pooling operation crucial for effective feature summarization has remained relatively underexplored. In this paper, we introduce Polar Harmonic Pooling (PHP), a novel symmetric global pooling method that leverages polar harmonic transforms to aggregate point cloud features. Unlike traditional pooling methods that rely on simple statistical measures, PHP captures global frequency patterns across the point set. This approach yields a more expressive and compact representation by encoding both spatial structure and frequency-domain characteristics in polar coordinates. We evaluate PHP on both classification and regression tasks. Experimental results show that PHP enhances accuracy, reduces the number of parameters in the invariant classification or regression head, and helps reducing overfitting when compared to standard statistical pooling methods.

1 Introduction

The recent advancements in 3D reconstruction technologies and low-cost 3D sensing technologies have brought point cloud data to the forefront of computer vision and robotics applications. However, learning from point clouds remains challenging due to their unordered, irregular, sparse nature and, most importantly, high volume of points. Unlike deep convolutional neural networks (CNNs) which typically consist of a feature extraction block and a classification head for image classification, the point-based deep neural networks [2], designed for point cloud processing, generally include a permutation-equivariant feature extraction backbone, a symmetric global pooling layer, and a permutation-invariant classification head. A common design choice is to ensure that the backbone is permutation equivariant to handle the unordered nature of point clouds, use symmetric pooling to generate fixed size global representations, and apply an invariant network for the final classification. Notably, the point-based neural networks [3, 7, 9, 10, 15, 16, 18, 24, 27, 30] have established a foundational framework where the design of the global pooling layer significantly influences model performance and robustness.

While significant effort has gone into improving backbone architectures, including neighbourhood graph construction, attention mechanisms, and multi-resolution processing, the design of the global pooling layer is often oversimplified, typically relying on basic symmetric functions such as max or average pooling. Recent work [6] has highlighted that the choice of pooling strategy can have a significant impact on performance than backbone depth or width, especially in limited data settings. Nevertheless, most existing pooling techniques fail to capture rich frequency-domain information that is crucial for robust shape representation. In this work, we propose a novel Polar Harmonic Pooling (PHP) layer that leverages polar harmonic transforms to encode global structural features in a permutation-invariant manner. Unlike traditional pooling methods, PHP projects per-point features onto a set of orthogonal harmonic basis functions, enabling a richer and more expressive aggregation of information that captures both spatial and frequency-domain characteristics. The proposed pooling mechanism consistently enhances performance in both classification and regression tasks. Furthermore, PHP reduces overfitting, one of the major challenges in point-based methods, and reduces the number of learnable parameters in the permutation-invariant classification head. We integrate the PHP module into widely used point-based neural networks such as PointNet[15] and PointNet++[16], and validate its effectiveness across standard benchmark datasets including ModelNet40[26], ModelNet10, and Small Grain Cereal Biomass Prediction (SGCBP) [14].

Our main contributions are as follows:

1. We propose Polar Harmonic Pooling (PHP), a novel symmetric global pooling technique that leverages polar harmonic transforms to ensure full permutation invariance in point-based neural networks.

2. We validate the effectiveness of PHP on ModelNet and SGCBP datasets, showing consistent performance gains over traditional pooling methods in both classification and regression tasks.

3. We demonstrate that PHP improves generalization by consistently reducing overfitting, increasing accuracy, and decreasing the parameter count in the classification and regression head, when integrated into PointNet and PointNet++.

2 Related Work

In deep neural networks, pooling is a fundamental operation used to aggregate features from a collection of related inputs into a more compact, often fixed size, summary representation. For set-structured data like point clouds or graphs, where the order of elements is not inherent, permutation-invariant pooling is crucial to ensure that the network's output does not depend on the arbitrary ordering of the input elements. Neural networks for these data types typically consist of a permutation-equivariant backbone followed by a permutation-invariant global pooling layer to produce a representation for the entire set.
The classical pooling methods are computationally simple and widely used for aggregating symmetric global features into fixed size embeddings. These include global max pooling, which selects the maximum feature value across the input set, global average pooling (GAP), which computes the mean of features, and sum pooling. Generalized Mean (GeM) pooling is a generalization that includes both max and average pooling as special cases. These methods

are commonly used due to their simplicity. Covariance Pooling [1, 22]is a method used for aggregating features that captures richer second order statistics, unlike traditional first order pooling like average pooling. Radenovic et al. [17] introduces a novel trainable Generalized-Mean (GeM) pooling layer. This layer generalizes classical max and average pooling to aggregate features into a descriptor, demonstrating improved performance over standard non-trainable pooling methods. Janossy pooling [12] is a learnable permutation-invariant pooling method for variable size inputs. It expresses a permutation-invariant function by averaging a permutation-sensitive function applied to all possible reorderings of the input set, providing a flexible framework for feature aggregation. Kolouri et al.[5] employs linear Wasserstein embedding as its graph pooling method. This technique aggregates node embeddings into a fixed size representation by leveraging optimal transport principles to capture richer distributional information than simple pooling methods like average pooling. Mialon et al. [11] proposed Optimal Transport Kernel Embedding (OTKE) as a trainable pooling method for sets of features. This method aggregates a variable size set into a fixed size representation by using the optimal transport plan to align input features to a trainable reference set. Linear optimal transportation (LOT) [23] provides an isometric linear embedding. This embedding aggregates information (from particle approximations) by leveraging their optimal transport plan to a learned reference template. Pooling by sliced Wasserstein embedding (PSWE) [13] is a novel, trainable, permutation-invariant pooling method for aggregating sets of features into a fixed dimensional representation. It treats set elements as samples from a distribution and employs a Euclidean embedding for the sliced Wasserstein distance between the input set and a trainable reference. This approach captures richer distributional information than simpler pooling methods. Zhang et al. [29] developed featurewise sort pooling (FSPool) which is a differentiable, sorting based pooling method for aggregating sets of feature vectors. The core idea is sorting each feature across the elements of the set, ensuring permutation invariance while handling variable set sizes. FSPool performs a weighted sum over these sorted features to produce a fixed size representation. Skianis et al. [20] proposes a novel pooling method for variable size sets by comparing the input set to learned hidden sets using network flow algorithms like bipartite matching. This approach creates a permutation-invariant representation by capturing correspondences through matching, differing from simpler methods like sum or average pooling. Vaswani et al. [21] has shown that the attention function itself performs the role of pooling by aggregating information as a weighted sum of values based on query-key compatibility. Lee et al. [8] introduced the self-attention graph pooling (SAGPool) is a graph pooling method for GNNs that uses self-attention based on graph convolution. This approach considers both node features and graph topology to determine which nodes to retain, enabling the learning of hierarchical representations. Lee et al. [7] proposes a novel, trainable pooling method called Pooling by Multihead Attention (PMA). This method aggregates features by applying multihead attention on a set of learnable seed vectors, designed to better capture interactions among elements in variable size input sets compared to simple fixed pooling. Wang et al. [25] proposes two pooling methods: Global Multi-Head Attention and Multi-Resolution Multi-Head Attention. These methods use multi-head attention across the entire input sequence to aggregate features for speaker embedding. The multi-resolution variant introduces a temperature parameter for each head to vary the resolution of attentive weights, enhancing diversity and performance. Kothapalli et al. [6] performed extensive experiments comparing pairwise combinations of backbones and pooling layers, the top-performing pooling methods specifically when paired with the PointNet backbone are PSWE, and FSPool.

3 Proposed Method

In this section, we first discuss the computational aspects of polar harmonic transforms, and then discuss the formulation of the polar harmonic pooling operator as integrated within point-based neural networks.

3.1 Polar Harmonic Transforms

Let $f(i,j)$ be a two-dimensional real function defined on a discrete grid of size $M \times N$. The polar complex exponential transforms (PCETs) of order p and repetition q over $f(i,j)$ are defined as follows[9, 28]:

$$A_{pq} = \frac{4}{\pi MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [H_{pq}(x_i, y_j)]^* f(x_i, y_j), \tag{1}$$

where $[\cdot]^*$ denotes the complex conjugate function, $p \geq 0$, $|q| \leq p$ and $H_{pq}(x_i, y_j)$ represents the basis function, which combines a radial kernel with an angular component:

$$H_{pq}(x_i, y_j) = R_p(r_{x_i,y_j}) e^{iq\theta_{x_i,y_j}}. \tag{2}$$

The radial kernel function $R_p(r_{x_i,y_j})$ is defined as:

$$R_p(r_{x_i,y_j}) = e^{i2\pi p r^2_{x_i,y_j}}, \tag{3}$$

where the polar coordinates r_{x_i,y_j} and θ_{x_i,y_j} are defined as follows:

$$r_{x_i,y_j} = \sqrt{x_i^2 + y_j^2}, \quad \theta_{x_i,y_j} = \tan^{-1}\left(\frac{y_j}{x_i}\right).$$

Since the function $f(i,j)$ is discrete and defined over a grid of size $M \times N$ in the Cartesian domain. The following equation is used to map Cartesian grid to the normalized range $[-1,1]$ and the coordinates x_i and y_j are computed as follows[19]:

$$x_i = \frac{2i - M + 1}{D}, \quad y_j = \frac{2j - N + 1}{D}, \tag{4}$$

where $D = \sqrt{M^2 + N^2}$ ensures normalization.
The radial kernel $R_p(r)$ forms an orthogonal basis under the following condition:

$$\int_0^1 R_p(r)[R_{p'}(r)]^* r\, dr = \frac{1}{2}\delta_{pp'}, \tag{5}$$

where $\delta_{pp'}$ denotes the Kronecker delta function and $\delta_{pp'}$ is 1 if $p = p'$, 0 otherwise. Due to the orthogonality of the kernel functions, the original function $f(i,j)$ can be approximated or reconstructed using the inverse PCETs as follows:

$$\hat{f}(i,j) = \sum_{p=0}^{P} \sum_{q=0}^{P} A_{pq} H_{pq}(x_i, y_j), \tag{6}$$

where P denotes the maximum order of the PCETs.

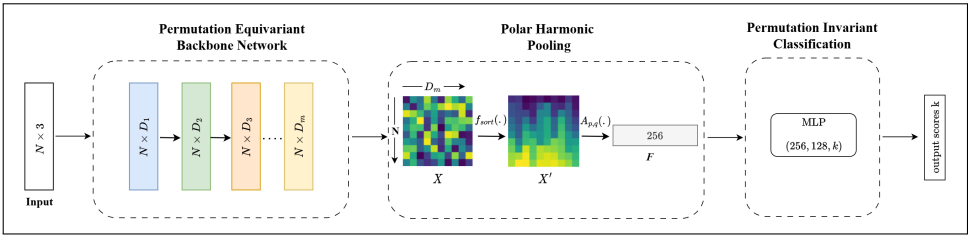


Figure 1: Polar Harmonic Pooling (PHP) pipeline illustrating the transformation and aggregation of point features in a point-based neural network.

3.2 Polar Harmonic Pooling

Let $X \in \mathbb{R}^{N\times D}$ represents a feature map generated by the backbone or feature extraction module of a point-based neural network, where N is the number of points and D is the feature dimension. The next step is to obtain a global representation of the feature map X using a symmetric pooling operation, with the goal of ensuring that the entire neural network remains permutation invariant.

We employ PCETs (Eq. (1)), a type of PHT, as a global pooling operation, referred to as Polar Harmonic Pooling (PHP), to extract permutation-invariant global features. The process begins with column-wise sorting of the input feature map X to impose a consistent ordering as given in the following equation.

$$X' = f_{\text{sort}}(X) = [f_{\text{sort}}(X_1), \ldots, f_{\text{sort}}(X_D)], \tag{7}$$

where $f_{\text{sort}}(\cdot)$ represents the sorting operation applied along each column (i.e., per feature dimension).

The next step is to compute PCETs over the sorted feature maps X' to obtain the global symmetric representation:

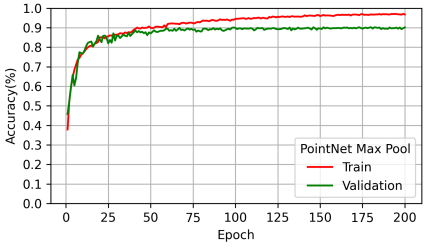
$$F = A_{p,q}(X'), \quad p = 0,1,\ldots,P; \quad q = 0,1,\ldots,P, \tag{8}$$

where $F \in \mathbb{R}^{P\times P}$ contains the PCET coefficients of order up to P. These coefficients are then flattened into a one-dimensional global feature vector and passed to the classification block for final prediction. The developed PHP module is shown in Fig. 1 for a point-based neural network, and the steps to compute PHP are shown in Algorithm 1.

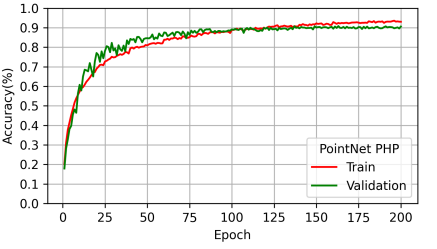
PHTs form an orthogonal basis and only a few low-order PHT coefficients are sufficient to capture the majority of the structural information in a signal, enabling compact and efficient encoding. Further, the PCET coefficients derived from PHTs can be interpreted as a two-dimensional Fourier decomposition, where the transformation is first applied in the radial direction, followed by the angular (circular) direction. This dual-domain encoding enables simultaneous analysis of spatial and frequency components in polar coordinates.

Algorithm 1 Polar Harmonic Pooling to Compute Global Symmetric Features

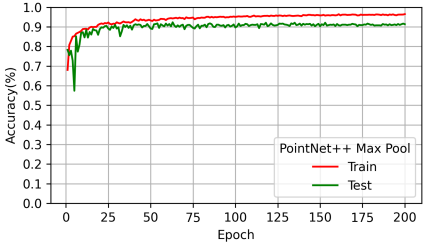
1: **Input:** Feature map $X \in \mathbb{R}^{N\times D}$, and PCET order P.
2: Sort the feature map X along the point dimension using Eq. (7) to obtain X'.
3: Computer PCETs over sorted feature map X' using Eq. (8) to obtain F.
4: Flatten $F \in \mathbb{R}^{P\times P}$ into a one dimensional feature vector.
5: **Output:** Flattened global feature vector F.



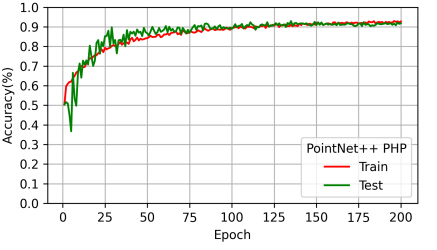
(a) PointNet with max pool



(b) PointNet with PHP



(c) PointNet++ with max pool



(d) PointNet++ with PHP

Figure 2: Train and test accuracies of models on ModelNet40 using max pooling and PHP over 200 epochs. PHP consistently achieves higher generalization performance with reduced overfitting compared to max pooling.

4 Experiments

In this section, we evaluate the effectiveness of our proposed Polar Harmonic Pooling (PHP) module by integrating it into two widely used point-based neural network architectures: PointNet [15] and PointNet++ [16]. We benchmark performance on classification using the ModelNet40[26] dataset, and regression using the Small Grain Cereal Biomass Prediction (SGCBP) [24] dataset. We compare the performance of PHP with the state-of-the-art max pooling method which is a very popular choice among the point-based neural networks and F-PSWE[6, 13].

Implementation Details We train both PointNet and PointNet++ (SSG variant) using the Adam optimizer with a learning rate of 0.001, a batch size of 32, and for 200 epochs. In our proposed model, the standard max pooling layer is replaced with the PHP module, with the PCET order $p = 16$, resulting in a flattened global feature vector of size 256. This feature vector is passed to a newly designed classification head comprising MLP layers with dimensions $(256, 128, k)$, where k denotes the number of output classes.

4.1 Classification

We evaluate our method on two benchmark datasets for 3D object classification: ModelNet40 and ModelNet10. ModelNet40 consists of 12,311 CAD models across 40 object categories, with 9,843 models used for training and 2,468 for testing. ModelNet10 is a smaller subset of ModelNet40, containing 3,991 training samples and 908 testing samples across 10 categories. For both datasets, we uniformly sample 1,024 points from each object and

normalize them to fit within a unit sphere. Importantly, only the raw 3D coordinates (x, y, z) of the sampled points are used in our experiments.

Table 1 demonstrates that PHP consistently outperforms both the conventional max pooling approach and the intermediate F-PSWE method across architectures and datasets. Specifically, PHP improves classification accuracy by 2.1% over max pooling and 1.6% over F-PSWE on ModelNet10 when used with PointNet. On ModelNet40, PHP also surpasses both baselines, with a 0.5% gain over max pooling and a 0.3% gain over F-PSWE. For the more expressive PointNet++ backbone, PHP yields consistent gains: 0.3% over max pooling and 0.3% over F-PSWE on ModelNet40, and 1.5% and 1.3% respectively on ModelNet10. These improvements highlight the capacity of PHP to better capture informative global structures, particularly benefiting models trained on smaller datasets such as ModelNet10.

Figure 2 shows a comparison of max pooling and PHP, illustrating the impact of PHP on feature aggregation on ModelNet40 dataset. The PHP-based variants (Fig. 2b and 2d) produce more spatially coherent and consistent representations than their max pooling counterparts (Fig. 2a and 2c). Notably, PHP helps reduce overfitting by emphasizing stable, frequency-aware global patterns rather than extreme feature responses. This is especially valuable in classification tasks where preserving geometric consistency across input variations is critical.

Table 1: Classification accuracies on ModelNet40 and ModelNet10 using Max pooling, F-PSWE, and PHP with PointNet and PointNet++.

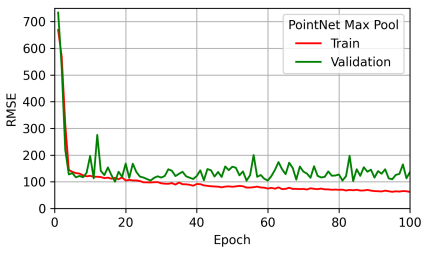
Pooling Technique	**ModelNet40**		**ModelNet10**	
	PointNet	**PointNet++**	**PointNet**	**PointNet++**
Max	89.2	90.6	90.6	92.2
F-PSWE	89.4	90.6	91.1	92.5
PHP (ours)	**89.7**	**90.9**	**92.7**	**93.8**

4.2 Regression

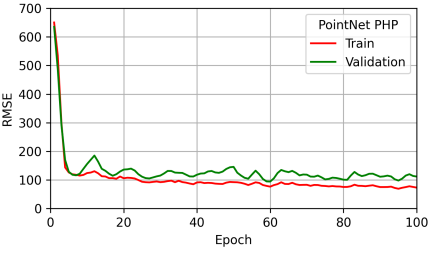
The SGCBP dataset includes 306 LiDAR-scanned point clouds of wheat and triticale plots grown under two sowing densities. Collected during two growth stages in a 2019 field trial in Yanco, Australia, each plot has corresponding ground-truth biomass measurements from destructive sampling. The dataset is split into 204 training and 102 testing samples and is publicly available to support 3D plant phenotyping research. We use SGCBP dataset to evaluate the performance of the proposed PHP method for regression task. In regression tasks, we adopt the Smooth $L1$ Loss and models are trained upto 100 epochs. As smooth $L1$ is less sensitive to outliers and avoids exploding gradients, making it particularly suitable for training on real-world noisy point cloud data.

$$\mathcal{L}_{\text{Smooth L1}}(x, y) = \begin{cases} 0.5(x-y)^2 & \text{if } |x-y| < 1, \\ |x-y| - 0.5 & \text{otherwise,} \end{cases} \tag{9}$$

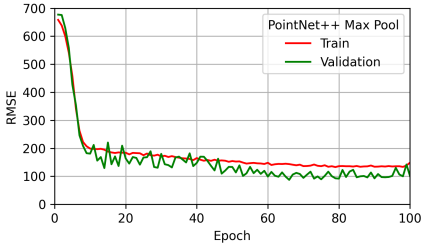
where x is the predicted value and y is the ground truth. Table 2 shows the root mean square error (RMSE) obtained using Max pooling, F-PSWE, and PHP. PHP consistently outperforms the standard Max pooling and the intermediate F-PSWE method across both the PointNet and PointNet++ architectures. Specifically, PHP reduces the RMSE from 100.2



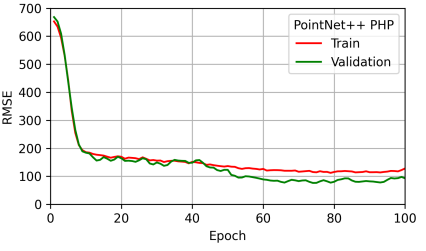
(a) PointNet with max pool



(b) PointNet with PHP



(c) PointNet++ with max pool



(d) PointNet++ with PHP

Figure 3: Train and test RMSE of models using max pooling and PHP over 200 epochs. PHP consistently achieves lower RMSE and demonstrates better generalization with reduced overfitting compared to max pooling.

to 93.57 in PointNet (a 6.63 decrease compared to Max and 5.1 compared to F-PSWE), and from 90.7 to 85.8 in PointNet++ (a 4.9 and 3.4 reduction, respectively). These improvements further demonstrate the effectiveness of PHP for biomass estimation—a regression task—by resulting in more accurate and stable predictions. Figure 3 compares max pooling and the proposed PHP across PointNet and PointNet++ architectures for regression task where the objetive is to reduce RMSE. PHP-based models (3b and 3d) produce smoother, more coherent representations than their max pooling counterparts (3a and 3c), again reflecting better preservation of global structure. These visual improvements indicate reduced overfitting. While max pooling can amplify local extremes and lead to instability, PHP captures broader spatial and frequency patterns, enhancing robustness especially valuable in regression tasks like biomass prediction or pose estimation.

Table 2: RMSE obtained on the SGCBP dataset using Max pooling, F-PSWE, and PHP with PointNet and PointNet++.

Pooling Technique	**PointNet**	**PointNet++**
Max	100.2	90.7
F-PSWE	98.67	89.2
PHP (ours)	**93.57**	**85.8**

4.3 Parameters and time complexity

To compare the number of learnable parameters required in the classification or regression head, we consider both standard max pooling and the proposed PHP. While max pooling requires 666,408 parameters, PHP reduces this to just 43,752, achieving a reduction of over 93%. This compact design not only enhances computational efficiency but also helps mitigate overfitting by limiting model capacity without compromising performance. The computational complexity of the proposed PHP layer is $\mathcal{O}(B \cdot P^2 \cdot D \cdot N)$, where B is the batch size, D is the feature dimension, N is the number of points, and P is the maximum order of the PCETs. This complexity arises from computing complex-valued inner products across each pair of harmonic orders (p,q), applied over the sorted feature map.

5 Conclusion

In this paper, we presented Polar Harmonic Pooling (PHP), a novel symmetric global pooling method for point-based neural networks. By leveraging polar harmonic transforms, PHP captures both spatial and frequency-domain features, enabling more expressive and stable global representations. Extensive experiments on classification and regression tasks demonstrate that PHP consistently outperforms conventional pooling methods, improving accuracy while reducing overfitting and model complexity. Notably, PHP reduces the number of parameters in the classification/regression head by over 93%, contributing to a more compact and efficient architecture. Although performance gains on larger datasets such as ModelNet40 are modest, PHP delivers significant improvements on smaller datasets such as ModelNet10 and SGCBP, highlighting its effectiveness in low-data regimes. These results establish PHP as a robust, general purpose, permutation-invariant pooling strategy that enhances generalization across tasks and architectures.

Acknowledgements

This study is partly supported by the NUE-Profits [grant ID: 10044243] project funded by Innovate UK.

References

[1] Dinesh Acharya, Zhiwu Huang, Danda Pani Paudel, and Luc Van Gool. Covariance pooling for facial expression recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 367–374, 2018.

[2] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[3] Miguel Dominguez, Rohan Dhamdhere, Atir Petkar, Saloni Jain, Shagan Sah, and Raymond Ptucha. General-purpose deep point cloud feature extractor. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 1972–1981. IEEE, 2018.

[4] Thai V. Hoang and Salvatore Tabbone. Fast generic polar harmonic transforms. *IEEE Transactions on Image Processing*, 23(7):2961–2971, 2014. doi: 10.1109/TIP.2014.2322933.

[5] Soheil Kolouri, Navid Naderializadeh, Gustavo K Rohde, and Heiko Hoffmann. Wasserstein embedding for graph learning. *arXiv preprint arXiv:2006.09430*, 2020.

[6] Abihith Kothapalli, Ashkan Shahbazi, Xinran Liu, Robert Sheng, and Soheil Kolouri. Equivariant vs. invariant layers: A comparison of backbone and pooling for point cloud classification. *arXiv preprint arXiv:2306.05553*, 2023.

[7] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019.

[8] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *International conference on machine learning*, pages 3734–3743. pmlr, 2019.

[9] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9397–9406, 2018.

[10] Xinhai Liu, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8778–8785, 2019.

[11] Grégoire Mialon, Dexiong Chen, Alexandre d'Aspremont, and Julien Mairal. A trainable optimal transport embedding for feature aggregation and its relationship to attention. *arXiv preprint arXiv:2006.12065*, 2020.

[12] Ryan L Murphy, Balasubramaniam Srinivasan, Vinayak Rao, and Bruno Ribeiro. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs. *arXiv preprint arXiv:1811.01900*, 2018.

[13] Navid Naderializadeh, Joseph F Comer, Reed Andrews, Heiko Hoffmann, and Soheil Kolouri. Pooling by sliced-wasserstein embedding. *Advances in Neural Information Processing Systems*, 34:3389–3400, 2021.

[14] Liyuan Pan, Liu Liu, Anthony G Condon, Gonzalo M Estavillo, Robert A Coe, Geoff Bull, Eric A Stone, Lars Petersson, and Vivien Rolland. Biomass prediction with 3d point clouds from lidar. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1330–1340, 2022.

[15] 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.

[16] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[17] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668, 2018.

[18] Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4548–4557, 2018.

[19] Chandan Singh and Jaspreet Singh. Multi-channel versus quaternion orthogonal rotation invariant moments for color image representation. *Digital Signal Processing*, 78: 376–392, 2018.

[20] Konstantinos Skianis, Giannis Nikolentzos, Stratis Limnios, and Michalis Vazirgiannis. Rep the set: Neural networks for learning set representations. In *International conference on artificial intelligence and statistics*, pages 1410–1420. PMLR, 2020.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[22] Qilong Wang, Jiangtao Xie, Wangmeng Zuo, Lei Zhang, and Peihua Li. Deep cnns meet global covariance pooling: Better representation and generalization. *IEEE transactions on pattern analysis and machine intelligence*, 43(8):2582–2597, 2020.

[23] Wei Wang, Dejan Slepčev, Saurav Basu, John A Ozolek, and Gustavo K Rohde. A linear optimal transportation framework for quantifying and visualizing variations in sets of images. *International journal of computer vision*, 101:254–269, 2013.

[24] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019.

[25] Zhiming Wang, Kaisheng Yao, Xiaolong Li, and Shuo Fang. Multi-resolution multi-head attention in deep speaker embedding. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6464–6468. IEEE, 2020.

[26] 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.

[27] Tiange Xiang, Chaoyi Zhang, Yang Song, Jianhui Yu, and Weidong Cai. Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 915–924, 2021.

[28] Pew-Thian Yap, Xudong Jiang, and Alex Chichung Kot. Two-dimensional polar harmonic transforms for invariant image representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1259–1270, 2009.

[29] Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Fspool: Learning set representations with featurewise sort pooling. *arXiv preprint arXiv:1906.02795*, 2019.

[30] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021.