Benchmarking 3D Point Cloud Classifiers in Synthetic to Real scenarios: A Cross-Domain Study with Qualitative Grad-CAM Insights

Stefano Barcio
Polytechnic University of Turin
s320174@studenti.polito.it

Luca Faieta
Polytechnic University of Turin
s323770@studenti.polito.it

Abstract

In recent years, significant progress has been made in the domain of 3D learning, particularly in the context of classification, detection and segmentation tasks on 3D point clouds. However, most existing literature at this time focuses on closed-set problems, while open-set classification models are much needed for their capability to reflect the open and dynamic nature of real-world environments, and they still suffer from a general lack of comprehensive and complete analysis. We aim to shed light on the underlying approach of some of these models in the context of out-of-distribution detection, to compare their advantages and limitations in an human friendly visualization. We start from results obtained by 3DOS paper benchmark and further investigate some of the proposed models. Our result is a both quantitative and visual analysis of models performance, leveraging a number of different evaluation strategies. In addition, we propose results from an adaptation of the well-known GradCam algorithm to 3D point clouds framework, to produce an human friendly visual explanation for decision processes of the proposed models. Code for our work can be found here.

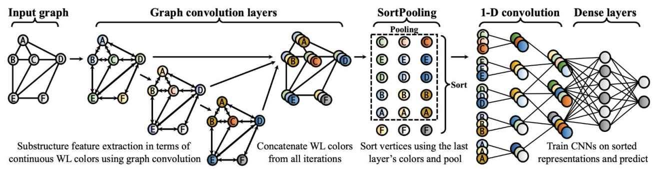
1. Introduction

Out-of-distribution (OOD) detection in the context of deep learning refers to the identification of data that significantly deviates from the training distribution. At test time, the introduction of a semantic shift in the data distribution can significantly impact a model's capability of correctly classifying samples, both in and out of distribution. OOD detection task is critical for ensuring the robustness and reliability of machine learning models, especially in safety-critical applications where this kind of misclassification can easily lead to potentially catastrophic behaviours. It is important to note that while OOD detection has been extensively studied in the realm of 2D data, the development and evaluation of effective technologies for corresponding 3D data-based models, e.g. point clouds, remain underexplored - this despite the growing importance of 3D data in various applications like autonomous driving and robotics.

In 2022, Alliegro et al. presented "3DOS: Towards 3D Open Set Learning – Benchmarking and Understanding Semantic Novelty Detection on Point Clouds" [1]. Their work stands as the first comprehensive study on 3D Open Set learning. They introduced 3DOS, a novel benchmark designed to address semantic novelty detection using 3D point clouds. The benchmark encompasses various settings with increasing difficulties, including in-domain scenarios (synthetic-to-synthetic, real-to-real) and cross-domain scenarios (synthetic-to-real). The main goal of our work is to further analyze 3DOS results in the synthetic-to-real scenario, picking some of the models they evaluated (specifically DGCNN [13] and PointNet++ [7]) and assessing their performance with a focus on OOD samples that are overconfidently misclassified as ID, and viceversa. Using this results as baselines, we then explore the possibility of fine-tuning a classifier head upon a richer semantic embedding given by very large pre-trained model as for example OpenShape [5]. We then proceed to perform the same kind of analysis stated before on the obtained model. Additionally, we propose a qualitative visual explanation of these models decisional behaviour, both for correct and erroneous classification results, using a version of the GradCam technique [8] specifically adapted to work with 3D point clouds. This final analysis brings a really interesting and easily readable explanation about the way these models actually process and classify data, and makes room for a number of possible suggestions to improve performances, that we propose in the final section of the paper.

2. Related Works

We split this section in three different subsections, grouping existing literature by their nature and usage in our work.


(a)

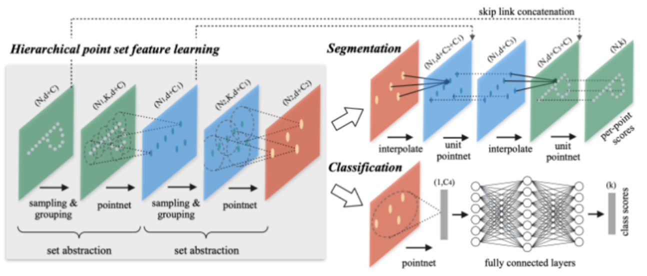

(b)

Figure 1. (a) DGCNN: An input graph is processed through graph convolution layers, sorted and pooled with SortPooling, then passed to CNNs for predictive modeling. Features are shown as colors. (b) PointNet++: Processes n points, transforms and aggregates features via max pooling, and outputs classification scores for m classes. The segmentation network combines global and local features for per-point scores. MLP denotes multi-layer perceptron with specified layer sizes.

2.1. Architectures

In our work we will mostly train and evaluate models already analyzed in 3DOS, nominally DGCNN and PointNet++. Those are neural network architectures specifically designed for processing point cloud data as shown in Figure 1. DGCNN dynamically constructs a graph based on the nearest neighbors in the feature space and applies convolutional operations on these graphs, allowing to captures local geometric structures. PointNet's strategy is similar, learning a spatial encoding of each point and aggregateing global point features. Its main contribution is the invariance to transformations on input data (e.g. rotations/translations in space). In the following section we test the implementation of three feature encoders proposed in OpenShape, a large architecture that integrates various data modalities (e.g., text, images, 3D shapes) into a unified multimodal embedding space, enabling rich and context-aware representations. In our work, we make use of a pre-trained version of this architecture, unlike DGCNN and PointNet which are trained from scratch.

2.2. Evaluation Methods - discriminative

3DOS proposes a number of evaluation methods belonging to different categories based on their strategies in assessing models performances. We refer only to discriminative and distance based methods. **MSP** [2] utilizes the maximum value of the softmax output probabilities from a neural network: samples with lower maximum softmax probabilities are considered as potential OOD data. **MLS** [12] operates on the same asssumption, but takes as input the raw confidences of the network rather then probability scores. **ODIN** [4] enhances MSP by incorporating temperature scaling and small perturbations to the input data, thus improving the separation between ID and OOD samples. **ReAct** [9] proposed to further increase this separability by rectifying the internal network activations. The **Energy** [6] based method transforms the output logits into an energy score, which is then used to identify OOD samples. Finally, **GradNorm** [3] exploits the gradient norms distinguishing signatures to amplify ID/OOD separability.

2.3. Evaluation Methods - distance based

These methods compute the position in the feature space of a given sample and then define a distance metric in the space. Then the label is assigned looking at the label of the nearest sample or computing the lowest distance from class prototypes. **CE(L2)** makes use of euclidean distances in the features space: OOD samples are distinguished by their higher distance from the nearest ID sample. **CosineProto** instead computes cosine similarity between given samples and prototypes generated for each class, following the same strategy as above for OOD detection.

3. 3DOS Benchmark

3.1. Baselines Preliminaries

Problem Formulation. As stated in the 3DOS publication, we draw from the training distribution p_s a labeled set $S = \{\mathbf{x}^{\mathrm{s}}, y^{\mathrm{s}}\}_{s=1}^{N}$ where known samples are defined by $y^s \in Y^s$. A model is trained on this distribution, and later a test set $T = \{\mathbf{x}^{\mathrm{t}}\}_{t=1}^{M}$ is drawn from distribution p_t and used to evaluate the model. Since we are addressing a cross-domain scenario with partial overlap between sets, for the test data labels $y^t \in Y^t$ it holds $Y^s \subset Y^t$. This means that test classes that are not contained in Y^t are defined as unknown. The goal of our model is to build a reliable model that outputs, for each sample in the test set, a normality score representing the probability of said set of belonging to any of the known classes. Additionally, it should also provide an output probability distribution over each of the known classes.

Performance Metrics. We evaluate models using two

Table 1. Results on the Synthetic to Real Benchmark track. Each column title indicates the chosen known class set, the other two sets serve as unknown.

	Pointnet						DGCNN					
Method	SR 1 (easy)		SR 2 (hard)		Avg		SR1(easy)		SR2 (hard)		Avg	
	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
MSP	79.5	83.6	68.2	91.2	73.9	78.8	73.1	89.3	62.0	89.0	67.55	89.1
MLS	80.8	77.8	66.5	90.5	73.7	75.8	70.9	93.2	63.6	87.5	67.2	90.4
ODIN	81.0	77.4	68.4	89.3	74.4	83.4	71.0	93.1	63.7	87.5	67.4	90.3
Energy	81.1	75.5	66.1	93.5	73.6	84.5	70.8	93.6	63.6	86.1	67.2	89.9
GradNorm	76.8	81.1	66.1	91.1	71.5	86.1	68.6	93.5	62.0	87.5	65.3	90.5
ReAct	80.8	78.1	66.3	92.4	73.6	85.3	71.3	93.0	63.8	87.1	67.5	90.0
Cosine proto	80.8	77.8	75.8	90.8	78.3	84.3	63.2	89.4	63.2	86.3	63.2	87.9
CE (L^2)	78.8	86.5	61.2	91.4	70.0	89.0	66.6	85.9	66.9	87.7	66.8	86.8

Figure 2. Visualization of the object categories in each of the sets of the Synthetic to Real Benchmark. SR1: chair, shelf, door, sink, sofa. SR2: bed, toilet, desk, table, display. SR3: bag, bin, box, pillow, cabinet.

main performance metrics: AUROC and FPR95. AUROC plots the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. It summarizes the ROC curve into a single value, with a score of 1 indicating perfect classification and a score of 0.5 suggesting the model performs no better than random guess. Higher AUROC values indicate better overall performance of the model.FPR95 (False Positive Rate at 95% True Positive Rate) measures the proportion of negative instances incorrectly classified as positive when the true positive rate is set to 95%. This metric is used to evaluate models in scenarios where maintaining a high true positive rate is crucial. Lower FPR95 values are better, indicating fewer false positives when the model correctly identifies 95% of positive instances.

Datasets. To realize the synthetic-to-real cross domain scenario, point clouds samples are taken from ModelNet40 [11] for training (synthetic 3D CAD models from 40 man-made object categories) and ScanObjectNN [14] for test (3D scans of real-world objects from 15 categories). Both datasets are amongst the best known collections for 3D classification tasks and provide a good benchmark for evaluating the generalization ability of models trained on synthetic data and tested on real-world data. Test/training splits, point sampling from clouds and hyperparameters selection are implemented following 3DOS indications to ensure consistency and comparability of results.

Benchmark Track. We define three distinct category sets: SR1, SR2, and SR3 as depicted in Fig. 2. The first two sets consist of matching classes from ModelNet40 and ScanObjectNN. In contrast, the third set (SR3) includes classes from ScanObjectNN that do not have a direct one-to-one correspondence with ModelNet40. We consider two scenarios where either SR1 or SR2 is used as the known set, while the other two sets are treated as unknown. For this track, models are trained on samples from the known classes set in ModelNet40 and evaluated on samples from both known and unknown classes in ScanObjectNN.

3.2. Baselines Results

Table 1 provides an overview of the results on the Synthetic to Real benchmark. We obtained results that are substantially comparable with the 3DOS benchmarks. The slightly deviations can be possibly justified with the different number of epochs in training (250 in our case against 500). Overall, for discriminative methods MSP proves to be the most robust baseline when considering both PointNet and DGCNN, even if by a smaller margin than in the 3DOS case study, and is generally more performing than more complex strategies with the exception of ODIN in the case of PointNet and ReAct for DGCNN. For distance-based methods we can confirm the superiority of CE(L2) for DGCNN while CosineProto definitely surpasses it in the case of PointNet. Considering these results plus the 3DOS conclusions, for the following analyses we will primarily focus CE(L2) and CosineProto along with MSP and MLS as baselines.

4. 3DOS Failure Cases

In the following section we will have a deeper look at cases in which the proposed architectures fail to correctly

Table 2. Number of incorrect predictions with high confidence class to class using MSP and MLS as evaluation methods

MSP (threshold=0.99)

		SR1					SR2				
predicted		chair	library	door	sink	sofa	bed	toilet	table+desk	monitor	OOD
SR1	chair	-	7	0	6	**67**	7	**43**	**44**	3	118
	library	0	-	0	0	0	0	0	1	1	7
	door	0	0	-	0	0	0	0	2	**41**	100
	sink	0	2	0	-	12	27	6	**61**	0	36
	sofa	0	0	0	**1**	-	0	0	0	0	0
SR2	bed	8	17	0	**41**	24	-	0	19	0	29
	toilet	4	1	0	3	2	0	-	0	0	2
	table+desk	**88**	32	0	6	6	24	5	-	0	58
	monitor	3	4	**202**	0	0	0	0	1	-	121

MLS (threshold=7)

		SR1					SR2				
predicted		chair	library	door	sink	sofa	bed	toilet	table+desk	monitor	OOD
SR1	chair	-	11	0	8	**73**	3	**44**	**55**	3	137
	library	0	-	0	0	0	0	0	0	0	7
	door	1	1	-	0	0	0	0	4	**45**	112
	sink	0	1	0	-	6	10	3	**47**	0	21
	sofa	0	0	0	0	-	0	0	0	0	0
SR2	bed	3	18	0	**33**	12	-	0	20	0	23
	toilet	2	0	0	2	1	0	-	0	0	1
	table+desk	**64**	26	0	2	3	29	5	-	3	94
	monitor	0	5	**208**	0	0	0	0	1	-	145

Table 3. Number of incorrect predictions with high confidence class to class using Euclidean distance and Cosine Similarities as evaluation methods

Euclidean distances (threshold=0.11)

		SR1					SR2				
predicted		chair	library	door	sink	sofa	bed	toilet	table+desk	monitor	OOD
SR1	chair	-	14	0	4	**67**	3	**44**	**45**	6	83
	library	0	-	0	0	0	0	0	3	12	5
	door	0	5	-	0	0	0	0	0	**72**	56
	sink	0	4	0	-	19	20	6	**86**	0	38
	sofa	0	0	0	1	-	1	0	0	0	0
SR2	bed	4	3	0	28	**46**	-	0	22	0	25
	toilet	10	5	0	1	2	0	-	0	0	8
	table+desk	114	**60**	0	5	30	16	11	-	0	68
	monitor	14	18	**76**	1	2	0	1	2	-	123

Cosine similarities(threshold=0.97)

		SR1					SR2				
predicted		chair	library	door	sink	sofa	bed	toilet	table+desk	monitor	OOD
SR1	chair	-	7	0	3	**50**	3	**40**	19	3	47
	library	0	-	0	0	0	0	0	0	7	8
	door	0	5	-	0	0	0	0	2	**100**	106
	sink	0	2	0	-	31	25	5	**85**	0	36
	sofa	0	0	0	2	-	2	0	1	0	2
SR2	bed	9	14	0	**43**	**50**	-	0	28	0	30
	toilet	6	0	0	2	3	0	-	0	0	2
	table+desk	**98**	29	0	4	12	22	10	-	2	71
	monitor	1	12	**138**	0	0	0	0	0	-	110

classify point clouds. The analysis is divided in two main scenarios: 1.) Incorrect predictions that are associated with really high values of confidences and 2.) Samples for which the score of the correct class is really low. This allows us to draw our first conclusions on the underlying features detection strategies for the point clouds in the test set, and on the possible causes that lead the networks to incorrect predictions. Thresholds used to select ”confidently” incorrect predictions are computed, separately for SR1 and SR2 and then averaged, by looking at the samples output distribution, and selecting values above the .75 percentile for case one, and below the .25 for case two.

4.1. Incorrect Predictions with High Confidence

In Table 2 and Table 3, we present the results of our experiments on both PointNet and DGCNN, evaluated using the selected methods from the previous section (MSP and MLS for discriminatives, and Euclidean/Cosine distances for distance-based methods). A general consistency in the nature of failure cases among all evaluated methods is evident, with confusion matrices showing similar patterns for the most misclassified class pairs. Many common misclassifications involve objects with similar geometric features, indicating that while both architectures effectively learn to extract regions of interest, they struggle to robustly differentiate between instances with shared patterns but different classes. For instance, the highest misclassification value across the table involves the monitor-door pair. The 3D point clouds used by our architectures lack color, relative dimensions, and other features that could help distinguish these categories in traditional image classification, making it challenging to differentiate between a door and a monitor based solely on shape, even for humans. Similarly, the high misclassification rates for chair-desk-table entries can be attributed to their similar legs and backs. It appears that the backrest is a crucial feature for our networks, which explains the significant misclassification between chairs and toilets (despite toilets not having legs). Beds are more frequently misclassified as sofas rather than chairs, likely due to their elongated shape. Finally, we observe that the classes with the highest rates of ID misclassifications are also the ones most frequently misclassified as out-of-distribution (OOD). This suggests that, in addition to the previously mentioned issues, there may be a problem of a generally weak feature embedding for certain categories, making them more prone to misclassification. To validate these intuitions, we need to verify that the regions we identified as plausible sources of error are indeed considered by the network when making predictions. We will conduct this qualitative analysis in the GradCam section of this work.

4.2. Low Correct Score Samples

Table 4 presents the number of samples for each class with a correct-class score lower than the computed threshold for each method. We observe a consistent trend across all methods, with some categories showing significantly higher numbers of low-confidence samples. Notably, classes that performed well in the previous analysis do not always maintain this trend. For example, sofas, which performed well previously, now show a higher number of low-confidence samples. This discrepancy is not contradictory, since the two extremes of the confidence spectrum convey

Table 4. Number of correct predictions with low confidence for each ID class with differents evaluations methods

method	MSP	MLS	Euclid.Dist.	Cosine Sim
chair	2	3	42	44
library	78	74	115	142
door	0	0	24	3
sink	20	17	39	29
sofa	214	220	94	96
bed	47	32	38	29
toilet	62	72	32	54
desk/table	81	85	104	94
monitor	7	8	23	20
thresh	0.27	2.43	0.1	0.93

different information: high confidence in incorrect predictions indicates the network misclassified an object with near certainty, seeing features of other categories in the sample. In contrast, low confidence in the correct class signifies a specific inability to extract typical features of the correct class from the sample. These results are coherent; for instance, the second table indicates a widespread difficulty for the network to assign high confidence to sofa samples, which aligns with Table 1, where only incorrect high-confidence samples are noted.

5. OpenShape Implementation

In this section, we explore the implementation of strategies proposed in the OpenShape publication within our framework. Specifically, we focus on three different versions of PointBERT backbone, one of the 3D point cloud backbones utilized in OpenShape. We leverage the pre-trained PointBERT models to extract features of the test set, aiming to evaluate their performance without additional fine-tuning. Subsequently, we attempt to enhance these models by building a dense classifier head, similar to those used on top of PointNet and DGCNN backbones.

5.1. OpenShape

OpenShape is a method designed to scale up 3D shape representation for open-world understanding by integrating multi-modal data, including text, images, and 3D shapes. It works in a multi-modal contrastive learning framework to align 3D shape embeddings with pre-trained CLIP embeddings of text and images, enabling robust cross-modal representation. The method is trained on a vast dataset of 876,000 shapes from four major public 3D datasets: ShapeNetCore, 3D-FUTURE, ABO, and Objaverse. The partial overlap of OpenShape's training set with our test set will be discussed in detail in subsequent sections. OpenShape has achieved superior performance in zero-shot 3D classification, with a notable zero-shot accuracy of 46.8% on the Objaverse-LVIS benchmark and 85.3% on ModelNet40, making it a suitable candidate for our study.

5.2. PointBERT

PointBERT is one of the backbones analyzed in OpenShape. It is a transformers-based architecture designed to extend the BERT-style pre-training strategy to 3D point cloud data. Inspired by BERT, which uses masked language modeling (MLM) for text, PointBERT introduces a masked point modeling (MPM) task to pre-train 3D point cloud Transformers. The process begins by dividing a point cloud into local patches using a discrete Variational AutoEncoder (dVAE), which generates discrete point tokens that encapsulate significant local information. These tokens are then randomly masked, and the model is trained to recover the original tokens from the masked locations. This approach allows the Transformer to learn robust representations of 3D shapes. PointBERT has demonstrated impressive accuracies in experiments conducted on our dataset, achieving 93.8% on ModelNet40 and 86.2% on ScanObjectNN, indicating its strong capability in handling diverse 3D data.

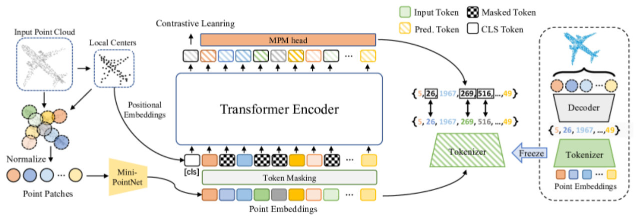


Figure 3. PointBERT architecture.

5.3. Experiments

Our first experiment in Table 5 confirms the results obtained in the PointBERT paper on the ScanObjectNN dataset. Out of the three proposed checkpoints, all evaluated on cosine similarities in the features space, two of them exhibits really high accuracies, with pointbert-VITB32 reaching up to 87.1% accuracies on the easy test set, and an average of 75.1%. Starting from this results, we train a classifier head alike the ones used in the previous sections and compare the results with the specialized CE embedding (CE L2) in 3DOS. Results are shown in Table N+1 and we can see that the proposed architecture outperforms DGCNN by an average of 5 points and shows comparable results with PointNet architecture.

5.4. Datasets Overlap and Conclusions

In our study, the checkpoints used for evaluation in Table 5 and Table 6 are partially trained on samples from the

Figure 4. GradCam architecture

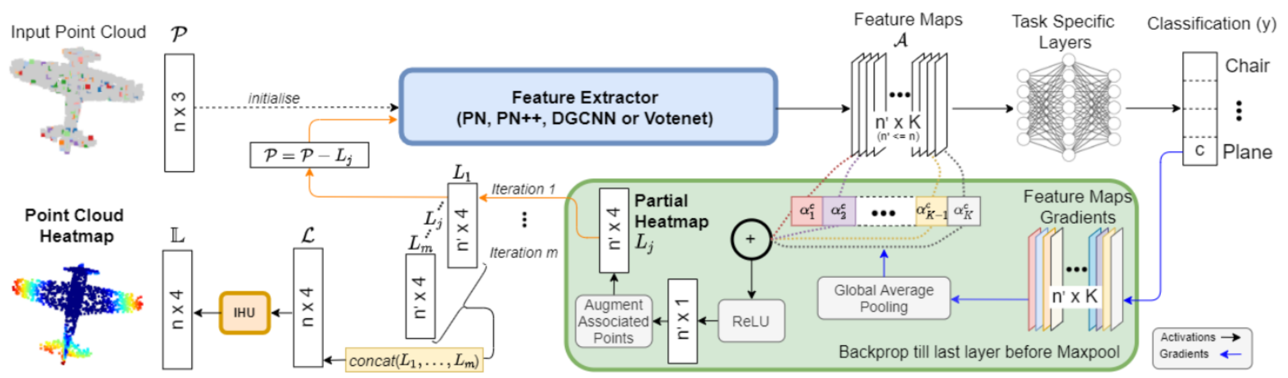


Table 5. Results different backbones from Open-Shape with Cosine similarities with prototypes

Model	SR 1 (easy)		SR 2 (hard)		Avg	
	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
pointbert-VITG14	52.3	93.2	55.3	91.2	53.8	92.2
pointbert-VITB32	**87.1**	**60.2**	63.0	91.3	**75.1**	**75.8**
pointbert-VITL14	83.4	72.1	**64.5**	**88.2**	74.0	80.2

Results on the Synthetic to Real Benchmark track. Each column title indicates the chosenknown class set, the other two sets serve as unknown.

PointBERT-VTB32 + Classifier Head						
Method	SR 1 (easy)		SR 2 (hard)		Avg	
	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
MSP	74.4	80.0	66.6	90.8	70.5	85.4
MLS	80.8	77.0	69.4	89.8	75.1	83.4
ODIN	80.8	79.9	66.0	90.8	73.4	85.4
Energy	80.9	78.2	62.7	86.6	71.8	82.4
GradNorm	81.8	76.6	64.4	89.4	73.1	83.0
ReAct	80.5	75.6	62.9	85.4	71.7	80.5

ScanObjectNN dataset, which is also used for testing. This practice can lead to overfitting, where our model performs well on the specific characteristics of the test dataset rather than properly generalizing to unseen data. Consequently, the reported accuracies may be inflated, potentially giving a misleading impression of the model's actual performance. This issue was unavoidable within the scope of our work, as fully re-training PointBERT on a better separated dataset was computationally unfeasible, and using a different test set would have invalidated the comparison with previous models. On the positive side, our results indicate the potential for further improvement through more intense training. With an extended training period—possibly increasing the number of epochs and considering the feasibility of retraining or fine-tuning the entire model on a better separated dataset—we could enhance the model's performance. This approach would address the limitations of the current study and provide a more comprehensive evaluation of the model's capabilities, going beyond the constraints imposed by the overlap between training and test datasets.

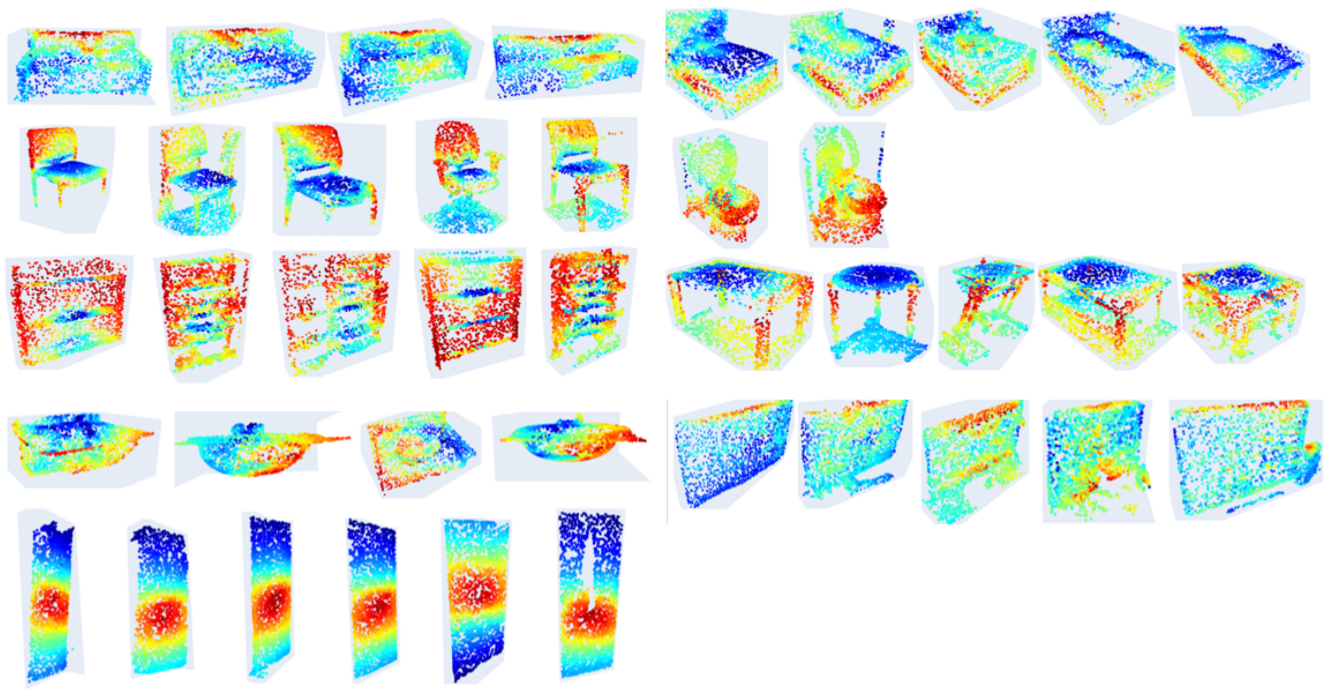
6. GradCAM Analysis

Gradient-weighted Class Activation Mapping (Grad-CAM) is a technique used to gain insights into the decision-making processes of convolutional neural networks (CNNs). Introduced by Selvaraju et al. in 2017, Grad-CAM provides visual explanations by highlighting the regions of an input image that are most influential in predicting a specific class. This is achieved by leveraging the gradients of the target class, flowing through the final convolutional layer of the CNN to produce a coarse localization map. This map, often referred to as a heatmap, overlays on the original image to illustrate which parts of the image the model focuses on when making its prediction. Although the original version of Grad-CAM was designed for 2D image-related tasks, Jawad Tayyub et al. published a paper in 2022 titled "Explaining Deep Neural Networks for Point Clouds using Gradient-based Visualisations." [10] This paper adapts the Grad-CAM architecture to work with 3D point cloud structures, so we decided this structure on top of DGCNN to produce our results.

6.1. GradCAM Architecture

The GradCAM architecture, shown in Figure 4, employs the Accumulated Piecewise Explanations (APE) approach to generate highly interpretable point cloud heatmaps. The process begins by classifying an input point cloud using a network like, in our case, DGCNN. The final feature maps, which capture high-level semantics, are extracted from the last convolutional layer of the network and used to generate coarse conceptual explanations by computing gradi-

Figure 5. Grad-CAM heatmaps for correctly classified point clouds across various object categories. Each point's relevance score ranges from low (blue) to high (red). The rows display heatmaps for: sofas/armchairs (top), chairs/toilets (second), libraries/tables (third), sinks/monitors (fourth), and doors (bottom).



ents with respect to these feature maps. These gradients are globally average pooled to create weights reflecting the contribution of neurons in each feature map. Partial heatmaps are constructed by taking a weighted sum of the feature maps, representing the contributions of corresponding input points. Since feature extraction layers generally have a lower resolution than the input point cloud, these partial heatmaps initially only reveal contributions of a subset of the point cloud. To cover the entire point cloud, already explained points are iteratively dropped and new partial heatmaps for different segments are computed, until all points have been explained. It should be noted that in our case the DGCNN architecture provides feature maps of shape [n_points, n_features], eliminating the need for this iterative loop that would be required in a more general implementation. The initial heatmap is refined through an iterative process called Iterative Heatmap Update (IHU). In each iteration, the lowest relevance points are dropped, and a new heatmap is computed for the remaining points. After several iterations, the final point cloud heatmap is created by combining the heatmaps from all iterations, ensuring that the most significant areas of the point cloud are highlighted.

6.2. Experiments

We divide our experiments into two sections. In the first section, we analyze the behavior of our network in situations of correct functioning for each calss. We aim to provide a qualitative explanation of the produced heatmaps and use these conclusions in the second section, where we focus on the most common misclassifications listed in Table 3, verifying the assumptions made earlier and offering more detailed explanations on the reasons behind the most frequent errors.

6.2.1 Correct Classification Analysis

As shown in Figure 5, Point cloud heatmaps seem here assign a number to each point in the range [0,1], indicating low relevance (blue) to high relevance (red). The proposed method identifies significant segments of objects which are critical for decision-making for the network. We can see samples where the network is able to isolate consistent and coherent regions and peculiar shapes of the point clouds to output its decision. For example, it gives great importance to the backrest and legs of chairs samples, which are sensible geometric features for robustly generalizing among different shapes of chairs (see how the rounded one in the image is still recognized).Tables are also recognized based on their legs, but the absence of the backrest is decisive for the correct decision. The same principle applies to toilets, where the distinguishing ovoidal shape of the seat becomes more important than the backrest. This type of analysis is crucial for understanding how the network discriminates between objects that are very similar from a point cloud perspective. For instance, doors are recognized based on points that are relatively far from the borders, while monitors show a different heatmap pattern, focusing on one or more of their narrow borders rather than central points: we will see how this point selection is crucial to understand misclassification between those two classes. Interestingly, we can already identify potential causes of future misclassifications by examining correctly classified cases. For example, classes where the network's regions of attention are less intuitive to a human observer might be associated with higher misclassification rates. Sofas, which we previously noted as a particularly challenging class, are discriminated by focusing on the narrow shape of the backrest border, whereas a human observer might give more importance to the shape of the armrests or the cushions. Also for this case we will see how this feature will be the cause of most misclassification cases related to this class.

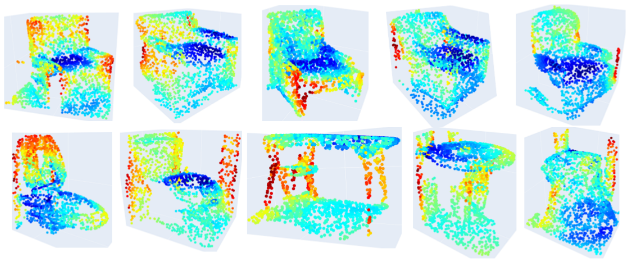
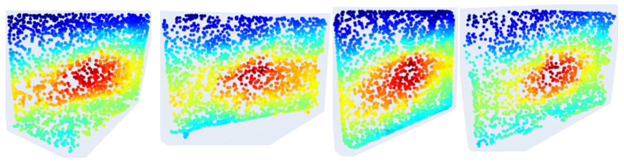


Figure 6. Significant examples of elements misclassified as chairs. The examples include from top left: two armchairs (belonging to the sofa class) classified as chairs due to their physical similarities with chairs; two additional armchairs misclassified as chairs due to lateral noise resembling armrests; two toilets misclassified for same reasons; a sample labeled "table" containing a chair in it; two noisy tables classified as chairs

Figure 7. Examples of monitors misclassified as doors: notice how the activation map is almost identical as correctly classified doors



6.2.2 Misclassification Instances

Figure 6 and Figure 7 show instances of common misclassification selected from Table 2. The gradients are computed with respect to the erroneously predicted class. It is clear how in most cases the network actually "saw" the same patterns that were correctly considered in true samples from said class, thus leading to overconfident predictions. Some of these categories can be explained with the weak embedding of some features of the true class (see chair-sofa), while it is more challenging to bring an intuitive explaination for some of the others. The sofa category is particularly interesting, since almost all instances of incorrect predictions are actually armchairs that in ScanObjectNN are still labeled as sofas, despite arguably have more features in common with chairs. The typical embedding of the sofa class, which we previously identified as a thin, long line along the backrest, fails in these instances, allowing features like the backrest and legs to become predominant. We suggest separating armchairs from sofas in the training set to achieve better differentiation. The door-monitor case shown in Figure 7 is more complex, as monitor samples are generally clean and quite different in shape from doors. However, the network often confidently selected the "door" region of interest rather than the monitor one, despite the samples being similar in shape and geometric features to correctly predicted monitors. We also selected samples where we think the error is actually dictated by excessively noisy or incomplete point clouds. Samples that include sections of background or surrounding environment, without further information of colour or texture, can be easily recognized as part of the object, providing the network with misleading information. This is particularly common when the network predictes a chair as shown in Fig 6. Many samples belonging to this category actually show that the network has interpreted thin strips of noisy points as actual legs of the chair or missing sections of the object as the backrest. Despite acknowledging that the ability of handling noisy real-world samples is a highly desirable feature for this model, we suggest that it is unfair to evaluate the network on such extreme cases (i.e. tables missing all four legs, or samples labeled "table" with a complete chair within) and is less indicative of the actual performance of the network.

7. Conclusions

In this study, we benchmarked several state-of-the-art 3D point cloud classifiers to assess their performance in cross-domain scenarios, particularly from synthetic to real-world data. By integrating Grad-CAM visualizations, we provided qualitative insights into the decision-making processes of one of these models. Our findings underscore the strengths and limitations of each approach in handling out-of-distribution (OOD) data, emphasizing the importance of robust feature extraction to avoid misclassification on similar samples or noisy datasets.

References

[1] Antonio Alliegro, Francesco Cappio Borlino, and Tatiana Tommasi. Towards open set 3d learning: Benchmarking and understanding semantic novelty detection on pointclouds. In

Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2022. 1

[2] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. ICLR, 2017. 2

[3] R. Huang, A. Geng, , and Y. Li. On the importance of gradients for detecting distributional shifts in the wild. NeurIPS, 2021. 2

[4] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural network. ICLR, 2018. 2

[5] Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding, 2023. 1

[6] W. Liu, X. Wang, J. Owens, , and Y. Li. Energy-based out-of-distribution detection. NeurIPS, 2020. 2

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. NeurIPS, 2017. 1

[8] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization, 2019. 1

[9] Y. Sun, C. Guo, and Y. Li. React: Out-of-distribution detection with rectified activations. NeurIPS, 2021. 2

[10] Jawad Tayyub, Muhammad Sarmad, and Nicolas Schönborn. Explaining deep neural networks for point clouds using gradient-based visualisations. ACCV, 2022. 6

[11] M. A. Uy, Q.-H. Pham, B.-S. Hua, D. T. Nguyen, and S.-K. Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. ICCV, 2019. 3

[12] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman. Open-set recognition: A good closed-set classifier is all you need. ICLR, 2022. 2

[13] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, , and J. M. Solomon. Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (TOG), 2019. 1

[14] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. CVPR, 2015. 3