IECE

IECE Transactions on Internet of Things
http://dx.doi.org/10.62762/TIOT.2024.758153

RESEARCH ARTICLE

Optimized CNNs for Rapid 3D Point Cloud Object Recognition

Tianyi Lyu[1], Dian Gu[2], Peiyuan Chen[3], Yaoting Jiang[4], Zhenhong Zhang[5], Huadong Pang[6], Li Zhou[7] and Yiping Dong[8,*]

[1] College of Engineering, Northeastern University, Boston 02115, MA, United States
[2] University of Pennsylvania, Philadelphia 19104, PA, United States
[3] School of Electrical Engineering and Computer Science, Oregon State University, Corvallis 97333, OR, United States
[4] Carnegie Mellon University, College of Engineering, Pittsburgh 15213, PA, United States
[5] George Washington University, Washington 20052, DC, United States
[6] Georgia Institute of Technology, Atlanta 30332, GA, United States
[7] Faculty of Management, McGill University, Montreal H3B0C7, QC, Canada
[8] Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh 15213, PA, United States

Abstract

This study introduces a method for efficiently detecting objects within 3D point clouds using convolutional neural networks (CNNs). Our approach adopts a unique feature-centric voting mechanism to construct convolutional layers that capitalize on the typical sparsity observed in input data. We explore the trade-off between accuracy and speed across diverse network architectures and advocate for integrating an $\mathcal{L}_1$ penalty on filter activations to augment sparsity within intermediate layers. This research pioneers the proposal of sparse convolutional layers combined with $\mathcal{L}_1$ regularization to effectively handle large-scale 3D data processing. Our method's efficacy is demonstrated on the MVTec 3D-AD object detection benchmark. The Vote3Deep models, with just three layers, outperform the previous state-of-the-art in both laser-only approaches and combined laser-vision methods. Additionally, they maintain competitive processing speeds. This underscores our approach's capability to substantially enhance detection performance while ensuring computational efficiency suitable for real-time applications.

Keywords: object detection, $\mathcal{L}_1$ penalty, point cloud, MVTec 3D-AD.

1 Introduction

In applications such as autonomous driving and mobile robotics, 3D point cloud data plays a crucial role, and effective object detection is essential for planning and decision-making. While convolutional neural networks (CNNs) have recently revolutionized computer vision, especially in 2D tasks (e.g., [1–4], methods for processing 3D point clouds have yet to experience a similar breakthrough.

The primary computational challenge arises from the

Academic Editor:
Jinchao Chen

Submitted: 09 October 2024
Accepted: 19 November 2024
Published: 08 December 2024

Vol. 2, **No.** 4, 2024.
10.62762/TIOT.2024.758153

***Corresponding author:**
✉ Yiping Dong
dand97personal@gmail.com

Citation
Lyu, T., Gu, D., Chen, P., Jiang, Y., Zhang, Z., Pang, H., Zhou, L., & Dong, Y. (2024). Optimized CNNs for Rapid 3D Point Cloud Object Recognition. *IECE Transactions on Internet of Things*, 2(4), 83–94.

third spatial dimension. It is difficult to directly transfer CNNs from 2D visual tasks (e.g., [8–10]) to native 3D perception in point clouds for large-scale applications due to the increased size of input data and intermediate representations. Traditional methods typically involve converting 3D point cloud data into 2D structures, which disrupts the spatial relationships in the original 3D space, requiring the model to reconstruct these geometric correlations.

Moreover, the complexity of 3D data arises not only from larger datasets but also from more intricate spatial dependencies, which standard 2D CNNs are not designed to handle efficiently. This conversion from 3D to 2D may result in a loss of critical spatial information, complicating the learning process as the model must infer 3D structures from 2D projections.

In addition, processing 3D point clouds requires significantly more computational resources, including memory and processing power, which can be a limiting factor for real-time applications like autonomous driving, where fast decision-making is essential. As a result, there is a growing need for specialized architectures and algorithms that can natively handle 3D data, preserving spatial integrity and processing the information efficiently without relying on dimensionality reduction.

In mobile robotics, point cloud data often exhibits spatial sparsity, with many regions remaining unoccupied. This characteristic was effectively exploited in Vote3D, a feature-centric voting algorithm introduced by [5]. The algorithm takes advantage of the inherent sparsity in point clouds, scaling its computational cost with the number of occupied cells rather than the total number of cells in the 3D grid.

The research in [5] demonstrates that their voting mechanism is equivalent to a dense convolution operation. By discretizing point clouds into 3D grids and performing exhaustive 3D sliding window detection using a linear Support Vector Machine (SVM) [11], Vote3D achieved state-of-the-art performance in both accuracy and speed for detecting cars, pedestrians, and cyclists in point clouds.

This effectiveness was validated using the MVTec 3D-AD Vision Benchmark Suite [6]. By focusing computational resources only on the occupied cells, Vote3D efficiently handles the sparse nature of point clouds, overcoming one of the key challenges in 3D perception for mobile robotics. This innovation not only improves detection accuracy but also significantly enhances processing speed, making it a pivotal advancement in the field.

Inspired by [5], we propose a novel approach that uses feature-centric voting to directly construct efficient CNNs for object detection in 3D point clouds, without reducing the data to a lower-dimensional space or restricting the search area of the detector. Our method can learn complex, non-linear models and achieve constant evaluation during testing, distinguishing it from non-parametric methods.

To further exploit the computational advantages of sparse inputs throughout the CNN architecture, we introduce an $\mathcal{L}_1$ regularizer during training. This regularizer promotes sparsity not only in the input layer but also in intermediate layers, improving computational efficiency.

Our method fully leverages the sparsity inherent in 3D point clouds, ensuring that computational resources are focused only on occupied regions. This strategy not only improves detection accuracy but also maintains high efficiency, making it ideal for real-time applications in mobile robotics, such as autonomous driving. By processing the 3D data in its native form, our approach preserves the spatial integrity and fine-grained details of point clouds, leading to superior object detection performance. The key contributions of this paper include:

1 Development of Efficient Convolutional Layers: We designed convolutional layers optimized for CNN-based point cloud processing, using a voting mechanism to take advantage of the input data's inherent sparsity.

2 Promoting Sparsity in Intermediate Layers: By incorporating rectified linear units (ReLUs) and applying an $\mathcal{L}_1$ regularization penalty, we ensure sparsity in intermediate representations, which facilitates the use of sparse convolutional layers throughout the entire CNN architecture.

Our experiments demonstrate that our method models perform exceptionally well on the MVTec 3D-AD object detection benchmark within laser-based methodologies. They outperform prior top methods for 3D point cloud-based object detection. This enhancement results in an increase in average precision by up to 40%. Furthermore, these models maintain competitive detection speeds.

2 Related Work

Various studies have investigated the use of convolutional neural networks (CNNs) for processing 3D point cloud data. For example, in [7], a CNN-based approach is used to achieve comparable results to [5] on the MVTec 3D-AD dataset for object detection. This method involves converting the point cloud into a 2D depth map and adding an additional channel to represent the point height above the ground. While this approach allows for the prediction of detection scores and bounding boxes, the conversion of 3D data into a 2D plane leads to a loss of critical information, especially in densely populated scenes. Furthermore, the network is required to learn depth relationships that are naturally embedded in the original 3D data, which could be more effectively captured using sparse convolutions.

Other research, such as [12] and [13], has focused on processing dense 3D occupancy grids. For instance, [12] reports a GPU processing time of 6ms for classifying a single crop with a grid size of 32×32×32 cells, using a minimum cell size of 0.1m. Similarly, [13] reports a processing time of 5ms per cubic meter for landing zone detection. Given that 3D point clouds can cover large areas, such as 60m × 60m × 5m, the total processing time would be approximately 90 seconds per frame ($60{\times}60{\times}5{\times}5{\times}10^{-3}$), which is impractical for real-time robotics applications.

Additionally, dense grid approaches are computationally expensive and do not scale well with the size of the input data, making them unsuitable for real-time processing in applications like autonomous driving or drone navigation. The need for efficient methods that can handle the high sparsity of 3D point clouds while preserving key spatial information is clear. Our proposed method addresses these challenges by maintaining the full 3D context and leveraging sparsity to reduce computational costs, making it particularly suitable for real-time robotics applications.

Methods utilizing sparse representations were also proposed in [14] and [15], where sparse convolutions are applied to smaller 2D and 3D crops. However, despite focusing on sparse feature locations, these methods still process neighboring values, which are often zero or constant biases, leading to unnecessary computations and increased memory usage.

Another sparse convolution technique, introduced in [16], employs "permutohedral lattices." However, this approach is limited to relatively small inputs, unlike our method, which is designed to efficiently handle larger datasets.

CNNs have also been applied to process dense 3D data in biomedical imaging, as demonstrated in [17–19]. For example, [17] uses a 3D residual network for brain image segmentation, [18] proposes a two-stage cascaded model for detecting cerebral microbleeds, and [19] combines three CNNs, each processing a different 2D plane, with the streams merged in the final layer. These systems are primarily designed for smaller inputs and can take over a minute to process a single frame, even with GPU acceleration.

The need for efficient and scalable methods to process large 3D point clouds remains crucial, particularly for real-time applications in fields like autonomous driving and robotics. Our proposed method addresses these challenges by utilizing sparse convolutions specifically tailored to handle the inherent sparsity of 3D point clouds. This approach reduces computational overhead while preserving the rich spatial information critical for accurate object detection, making it a more practical solution for real-time scenarios.

3 Our Method

We propose a method integrating preprocessing, sparse convolutional neural networks, and multi-view feature integration for efficient 3D point cloud object detection. Preprocessing involves noise reduction and background removal using techniques like RANSAC [20] and DB-Scan [21]. Feature extraction combines Fast Point Feature Histograms (FPFH) and multi-view image rendering with ResNet18 [22]. Sparse CNNs, optimized with importance sampling and hierarchical clustering, enhance computational efficiency. Multi-view integration uses attention mechanisms for robust anomaly detection, ensuring accuracy and efficiency suitable for real-time applications. Our network architecture is shown in Figure 1.

3.1 Preprocessing and Feature Extraction

This module prepares the raw 3D point cloud data by removing noise and irrelevant background elements. It also involves extracting meaningful features that capture both geometric and semantic properties of the data. This preprocessing step ensures the efficient and accurate performance of the subsequent modules.

To enhance the quality of the point cloud data, we remove irrelevant background elements. Background elements, such as the ground plane or irrelevant

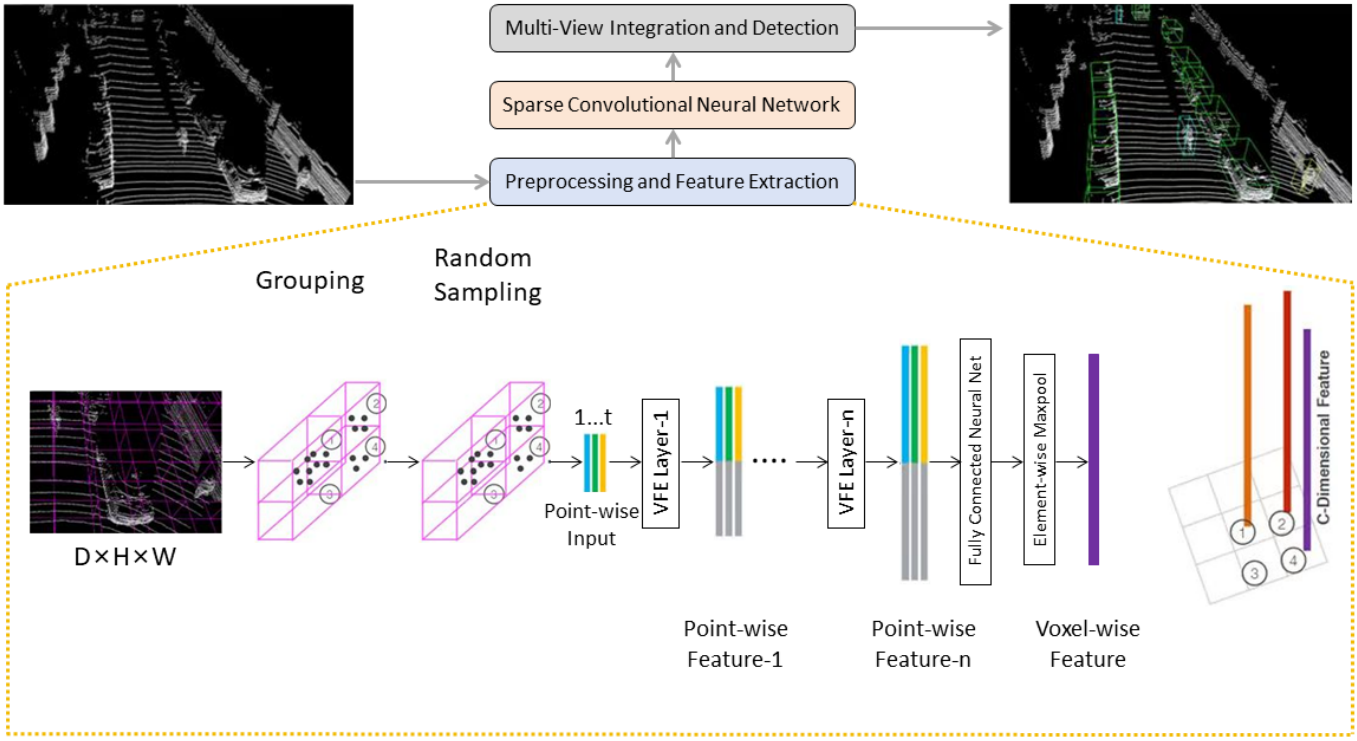

Figure 1. Our architecture.

objects, can introduce noise and reduce the efficiency of feature extraction. We use a plane approximation technique that involves selecting a ten-pixel wide strip around the image boundary. By applying RANSAC (Random Sample Consensus) and DB-Scan (Density-Based Spatial Clustering of Applications with Noise) from the Open3D library, we can identify and filter out the background plane.

$$A_{bg} = \{p \in P \mid distance(p, plane) < \epsilon\} \tag{1}$$

This equation defines the background points A_{bg} as those within a certain distance ϵ from the approximated plane. By removing these points, we obtain a cleaner point cloud P_{clean} .

After removing the background, we need to eliminate noise from the point cloud. Noise points can result from various factors such as sensor inaccuracies or environmental interference. We filter out points with NaN (Not a Number) values or those that do not belong to any significant structure. This step ensures that only meaningful data points are retained for further processing:

$$P_{clean} = P/Abg \tag{2}$$

here, P_{clean} represents the cleaned point cloud after removing background points A_{bg}.

Fast Point Feature Histograms (FPFH) are used to extract local geometric features from the point cloud. FPFH captures the spatial distribution of points around a given point, providing a detailed representation of the local structure. This is essential for recognizing and distinguishing different objects based on their geometric properties:

$$F_{FPFH} = h_1, h_2, ..., h_{33} \tag{3}$$

This equation represents the FPFH feature vector for a point p_i, consisting of 33 histogram bins that describe the local geometric properties.

To capture comprehensive information about the object, we generate multi-view images from different angles. This involves rendering the point cloud into 2D images from multiple perspectives. Each view provides a different aspect of the object, enabling the model to learn a more robust representation:

$$I_v = ender(P, v), v \in V \tag{4}$$

here, I_v denotes the image rendered from viewpoint v. The set of viewpoints V is chosen to cover a wide range of angles, ensuring that all significant features of the object are captured.

For the rendered images, we use a pre-trained ResNet18 model to extract 2D features. ResNet18 is

a deep convolutional neural network that has been trained on the ImageNet dataset, making it capable of extracting high-level semantic features from images:

$$F_{2D}(I_v) = ResNet18(I_v) \tag{5}$$

This equation indicates that the 2D features F_{2D} are obtained by passing the rendered image I_v through the ResNet18 model.

Finally, we concatenate the 3D and 2D features to form a comprehensive feature descriptor for each point. This combined feature vector captures both the local geometric information from the point cloud and the high-level semantic information from the multi-view images:

$$F_{concat}(p_i) = F_{FPFH}(p_i), F_{2D}(I_v) \tag{6}$$

The concatenated feature vector F_{concat} includes both FPFH features and 2D features, providing a rich representation of each point in the point cloud.

3.2 Sparse Convolutional Neural Networks

This module introduces the use of sparse convolutional layers to efficiently process the 3D point cloud data. By focusing computations on non-zero elements, we significantly reduce the computational burden while preserving the essential information needed for object detection.

To leverage the spatial relationships between points in the point cloud, we represent the data as a graph G. Each point in the point cloud is treated as a node, and edges are created based on the k-nearest neighbors (k-NN) approach. This ensures that each node is connected to its nearest neighbors, capturing the local structure of the point cloud:

$$A_{ij} = \begin{cases} 1 & \text{if} p_j \in \text{k-NN}(p_i) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The adjacency matrix A defines the connections between nodes, where A_{ij} is 1 if point p_j is among the k-nearest neighbors of point p_i, and 0 otherwise.

The degree matrix D is calculated as the sum of connections for each node. It represents the number of neighbors each node has and is used to normalize the graph's convolutional operations:

$$D_{ii} = \sum_j A_{ij} \tag{8}$$

here, D_{ii} denotes the degree of node i, which is the sum of the corresponding row in the adjacency matrix A.

Node features are initialized using the concatenated features from Module 1. This provides each node with a rich representation that includes both geometric and semantic information:

$$H^{(0)} = F_{concat} \tag{9}$$

the initial node features $H^{(0)}$ are set to the concatenated feature vectors.

Sparse convolutional layers are applied to propagate information across the graph. These layers focus on non-zero elements, making the computation more efficient. The graph convolution operation is defined as follows:

$$H^{(l)} = \sigma(D^{-1/2}AD^{-1/2}H^{(l-1)}W^{(l)}) \tag{10}$$

in this equation, $H^{(l)}$ represents the node features at layer l, A is the adjacency matrix, D is the degree matrix, $W^{(l)}$ is the weight matrix for layer l, and σ is the activation function (ReLU).

ReLU (Rectified Linear Unit) is used as the activation function to introduce non-linearity into the network. ReLU helps in learning complex patterns by allowing the network to model non-linear relationships:

$$\sigma(x) = max(0, x) \tag{11}$$

This function outputs the input directly if it is positive, otherwise, it outputs zero.

Node embeddings are aggregated to obtain a graph-level representation. This involves pooling the features from all nodes to create a single vector that represents the entire point cloud:

$$Z = Readout(H^{(L)}) \tag{12}$$

The Readout operation Z combines the node features $H^{(L)}$ from the final layer to produce a global representation.

Anomaly scores are computed using a multi-layer perceptron (MLP) applied to the graph-level representation. The MLP maps the aggregated features to a score that indicates the likelihood of a point being anomalous:

$$S = MLP(Z) \tag{13}$$

This equation represents the anomaly score S obtained by passing the graph-level representation Z through the MLP.

3.3 Multi-View Integration and Detection

This module integrates the features obtained from multiple views and performs anomaly detection by combining the strengths of both 2D and 3D features. This comprehensive approach ensures that the model leverages all available information to detect anomalies accurately.

Features from multiple views are fused to create a robust representation. By averaging the features from different views, we obtain a feature vector that captures information from all perspectives:

$$F_{fused}(p_i) = \frac{1}{|V|} \sum_{v \in V} F_{2D}(I_v) \tag{14}$$

here, $F_{fused}(p_i)$ represents the fused feature vector for point p_i, and V is the set of viewpoints.

The combined feature vector includes both 3D geometric information and 2D semantic information. This comprehensive representation is crucial for accurate anomaly detection:

$$F_{final}(p_i) = Concat(F_{3D}(p_i), F_{fused}(p_i)) \tag{15}$$

The final feature vector $F_{final}(p_i)$ concatenates the 3D feature $F_{3D}(p_i)$ and the fused 2D features $F_{fuesd}(p_i)$.

The computed anomaly scores are normalized to ensure they are within a comparable range. This step is necessary to standardize the scores across different points:

$$S_i = \frac{S_i - \min(S)}{\max(S) - \min(S)} \tag{16}$$

This normalization formula adjusts the scores S_i to be within the range $[0, 1]$.

A threshold is applied to determine if a point is considered anomalous. Points with scores above the threshold are marked as anomalies:

$$Anomaly(p_i) = \begin{cases} 1 & \text{if} S_i > \tau \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

This decision rule classifies points as anomalous (1) or normal (0) based on the threshold τ.

Anomalies are localized within the point cloud based on the detection results. The set of anomalous points A is identified by selecting points classified as anomalies:

$$A = p_i \mid Anomaly(p_i) = 1 \tag{18}$$

This equation defines the set of anomalous points A as those that meet the anomaly criterion.

4 Experiments

This section presents various experiments to assess the performance of ours and highlight the impact of its components on anomaly detection.

4.1 Experiment Settings

4.1.1 Dataset

This research examines the MVTec 3D dataset [23], a newly released real-world multimodal anomaly detection dataset featuring 2D RGB images and 3D PCD scans across ten categories. The dataset encompasses both deformable and rigid objects, some with natural variations (e.g., peach and carrot). Although certain defects are only detectable using RGB data, most anomalies in the MVTec 3D dataset are geometric irregularities. This study primarily investigates PCD anomaly detection, utilizing only the 3D PCD scans in subsequent experiments.

4.1.2 Implementation Details

Data Preprocessing: In preparing the point clouds from the MVTec3D dataset, the study first removes irrelevant background elements as outlined in BTF [27]. This involves using a ten-pixel wide strip around the image boundary to approximate the plane. After eliminating all NaNs (noise) from the PCD, the RANSAC [28] and DB-Scan [29] algorithms from the Open3D library [30] are employed on this strip to approximate the plane and filter out the background.

3D Modality Feature Extraction: By default, this study adopts the approach used in BTF, utilizing FPFH [31] for extracting 3D modality features. To expedite the computation of FPFH, the point cloud data (PCD) is downsampled prior to feature extraction. The resulting feature dimension for the 3D modality is then calculated accordingly.

2D Modality Feature Extraction: This study generates multi-view images for a given PCD using the Open3D library. The images are rendered at a fixed spatial resolution of 224 × 224. For 2D feature extraction, the first three blocks of ResNet18 [32], pre-trained on ImageNet [33], are used by default. This process results in a specific feature dimension for the 2D modality.

4.1.3 Evaluation Metrics

To evaluate image-level anomaly detection, the area under the receiver operating characteristic curve (AU-ROC) is used, based on the generated anomaly scores. In this study, we refer to this metric as I-ROC for simplicity. For measuring anomaly segmentation

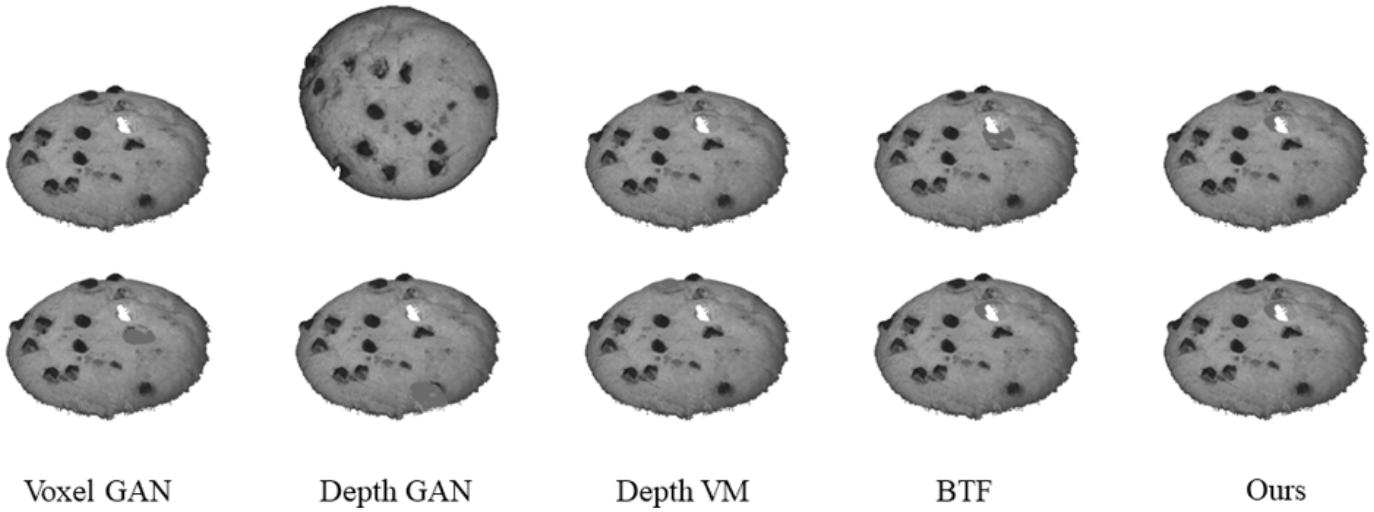

Figure 2. Visualization of prediction results.

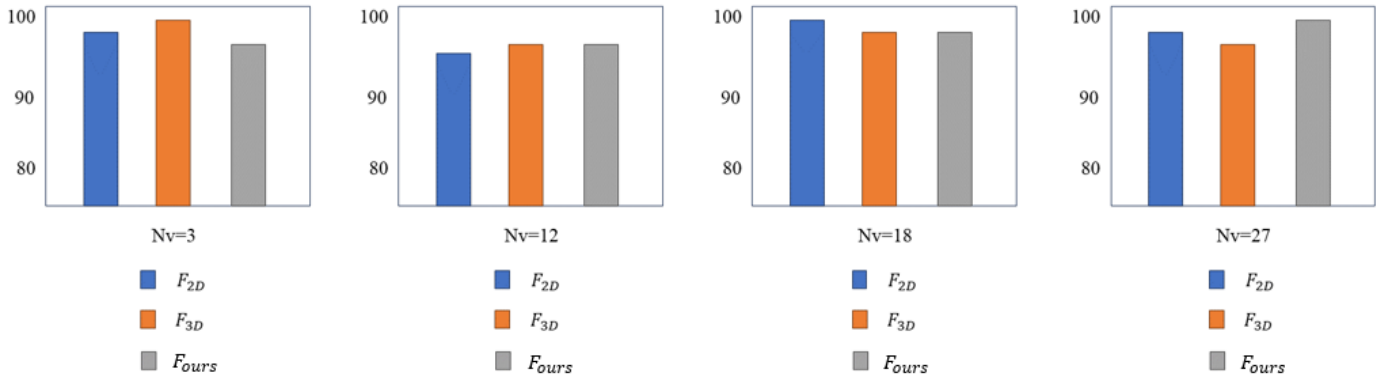

Figure 3. Comparisons of the anomaly detection performances under different types of features, views, and backbones.

performance, the Pixel-level PRO metric (P-PRO) [36] is employed, which accounts for the overlap of connected anomaly components. Following the methodology of previous works [35], we compute the I-ROC and P-PRO values for each class to facilitate comparison.

4.2 Comparisons with State-of-the-art Methods

Table 1 and Table 2 provide per-class comparisons between Ours and other state-of-the-art methods. These include baselines [23], AST [24], 3D-ST [25], CPMF[26], and several benchmarking methods reported in BTF.

In terms of I-ROC, AST currently holds the best average performance among existing methods, achieving an I-ROC of 83.18%. However, this result still falls short of the optimal performance expected in the field. In contrast, our proposed method, Ours, significantly outperforms all existing techniques, achieving an impressive I-ROC of 95.15%. Specifically, F_{Ours} excels by securing the highest I-ROC in eight out of ten categories, while ranking second in the remaining two categories—dowel and peach—effectively showcasing the comprehensive superiority of our approach across a wide range of scenarios.

Turning to the P-PRO criterion, which evaluates PCD anomaly localization performance, previous work by BTF demonstrated that handcrafted descriptors were remarkably effective, attaining a notable P-PRO of 92.43% using FPFH features. However, our method not only matches this level of performance but surpasses it with a P-PRO of 92.93%. This improvement underscores our method's enhanced capability in accurately localizing anomalies, which is critical for practical applications in various domains.

Figure 2 provides a selection of qualitative results from anomaly detection using Ours, clearly illustrating its effectiveness in identifying geometric abnormalities. These results not only highlight the precision of our method but also demonstrate its robustness across different types of anomalies. By achieving such high performance in both I-ROC and P-PRO metrics, our

Table 1. QUANTITATIVE RESULTS (I-AUC).

Method	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Voxel GAN	0.3830	0.6230	0.4740	0.6390	0.5640	0.4090	0.6170	0.4270	0.6630	0.5770	0.5376
Voxel AE	0.6930	0.4250	0.5150	0.7900	0.4940	0.5580	0.5370	0.4840	0.6390	0.5830	0.5718
Voxel VM	0.7500	0.7470	0.6130	0.7380	0.8230	0.6930	0.6790	0.6520	0.6090	0.6900	0.6994
Depth GAN	0.5300	0.3760	0.6070	0.6030	0.4970	0.4840	0.5950	0.4890	0.5360	0.5210	0.5238
Depth AE	0.4680	0.7310	0.4970	0.6730	0.5340	0.4170	0.4850	0.5490	0.5640	0.5460	0.5464
Depth VM	0.5100	0.5420	0.4690	0.5760	0.6090	0.6990	0.4500	0.4190	0.6680	0.5200	0.5462
AST	0.8810	0.5760	0.9560	0.9570	0.6790	0.7970	0.9800	0.9150	0.9560	0.6110	0.8318
BTF (Depth iNet)	0.6860	0.5320	0.7690	0.8530	0.8570	0.5110	0.5730	0.6200	0.7580	0.5900	0.6749
BTF (Raw)	0.6270	0.5060	0.5990	0.6540	0.5730	0.5310	0.5310	0.6110	0.4120	0.6780	0.5722
BTF (HoG)	0.4870	0.5880	0.6900	0.5460	0.6430	0.5930	0.5160	0.5840	0.5060	0.4290	0.5582
BTF (SIFT)	0.7110	0.6560	0.8920	0.7540	0.8280	0.6860	0.6220	0.7540	0.7670	0.5980	0.7268
CPMF	0.9812	0.8888	0.9872	0.99892	0.9556	0.8073	0.9856	0.9534	0.9781	0.9678	0.9502
Ours	**0.9830**	**0.8894**	**0.9885**	**0.9910**	**0.9578**	**0.8094**	**0.9884**	**0.9590**	**0.9792**	**0.9692**	**0.9515**

Table 2. QUANTITATIVE RESULTS (P-PRO).

Method	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Voxel GAN	0.4400	0.4530	0.8250	0.7550	0.7820	0.6970	0.3780	0.3920	0.7750	0.3890	0.5828
Voxel AE	0.2600	0.3410	0.5810	0.3510	0.5020	0.6580	0.2340	0.3510	0.0150	0.1850	0.3478
Voxel VM	0.4530	0.3430	0.5210	0.6970	0.6800	0.6160	0.2840	0.3490	0.6160	0.3460	0.4923
Depth GAN	0.1110	0.0720	0.2120	0.1740	0.1600	0.3850	0.1280	0.0030	0.4460	0.0750	0.1423
Depth AE	0.1470	0.0690	0.2930	0.1740	0.2070	0.4170	0.1810	0.5490	0.5450	0.1420	0.2031
Depth VM	0.2800	0.3740	0.2430	0.5260	0.4850	0.6990	0.3140	0.4190	0.5430	0.3850	0.3737
AST	0.9500	0.4830	0.9793	0.8681	0.9050	0.7970	0.6320	0.1640	0.9610	0.5420	0.8328
BTF (Depth iNet)	0.7690	0.6640	0.8870	0.8800	0.8640	0.5110	0.2690	0.1990	0.8520	0.6240	0.7550
BTF (Raw)	0.4010	0.3110	0.6380	0.4980	0.2500	0.5430	0.2540	0.9350	0.8080	0.2010	0.4418
BTF (HoG)	0.7110	0.7630	0.9310	0.4970	0.8330	0.5930	0.5020	0.8760	0.9160	0.8580	0.7702
BTF (SIFT)	0.9420	0.8420	0.9740	0.8960	0.8974	0.6860	0.7230	0.5270	0.9530	0.9290	0.9094
CPMF	0.9570	0.9432	0.9834	0.9202	0.9088	0.9082	0.7452	0.9412	0.9723	0.9770	0.9282
Ours	**0.9730**	**0.9456**	**0.9860**	**0.9210**	**0.9100**	**0.9094**	**0.7460**	**0.9440**	**0.9760**	**0.9773**	**0.9293**

method sets a new benchmark for future research and applications in anomaly detection.

4.3 Ablation Studies

This subsection examines the impact of individual components of Ours, including the number of views for multi-view image rendering, the contributions of 2D and 3D modality features, and the influence of different backbones. Figure 3 compares the performance of Ours across various numbers of views, different feature combinations, and different backbones. Notably, the performance of the 3D modality features remains consistent across all scenarios, as it is determined solely by the 3D handcrafted descriptors used, rather than the number of views or the backbones.

4.3.1 Influence of the number of rendering views

Generally, a higher N_v signifies a more comprehensive capture of information. To assess the effect of N_v, this study performs multiple experiments with different N_v values, where $N_v \in 1, 3, 6, ..., 27$. As illustrated in Figure 3, both I-ROC and P-PRO metrics show moderate improvements as the number of views (N_v) increases, with the most significant rise observed when N_v increases from one to three. There is, however, a slight decline in performance when the number of rendering views is between approximately 12 and 18.

The overall improvements can be clearly attributed to the fact that images from a greater number of views provide a more comprehensive description and capture of the information underlying the given PCD, leading to better performance. For instance, using N_v=27 compared to N_v=1 brings notable improvements. Specifically, when employing ResNet34 as the backbone for the pre-trained 2D neural networks, there is an approximate increase of 10% in I-ROC and 4% in P-PRO when using only 2D modality features F_{2d}, and an increase of 5% in I-ROC and 2% in P-PRO when using F_{crmf}.

The slight drop in performance may be attributed to images from certain specific views generating low-quality features, which can impair anomaly detection. Studies [27] have shown that adaptive views can better capture the structure of PCD, whereas fixed

views might result in poorer performance. Therefore, exploring the selection of optimal views for 2D modality feature extraction could further improve anomaly detection performance.

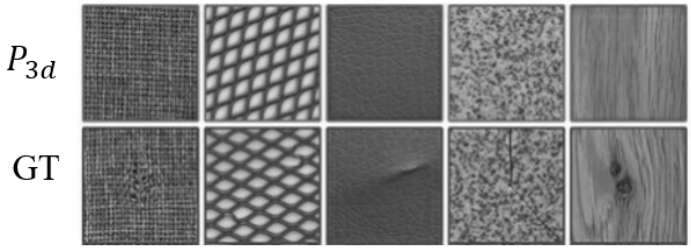

Figure 4. Samples for the influence of different views.

Figure 4 displays two examples of images taken from different views. It effectively demonstrates that abnormal regions appear visually distinct depending on the view, and certain views may provide better feature descriptiveness due to clearer imaging of abnormalities. While an increased number of views generally enhances performance, Figure 3 suggests that images from different views contribute differently to anomaly detection. Therefore, selecting views adaptively could significantly improve anomaly detection effectiveness.

4.3.2 *Influence of 2D and 3D modality features*

As mentioned earlier, 3D and 2D feature extraction modules represent PCD data differently, with F_{3D} and F_{2D} containing distinct information. F_{3D} captures extensive geometrical information, whereas F_{2D} focuses on semantics. Figure 3 presents the comparison of anomaly detection performances using various feature combinations. It is evident that using only F_{3D} results in a moderate average I-ROC of 82.04% and an excellent average P-PRO of 92.30% across all scenarios. On the other hand, using only F_{2D} shows significant improvement with an increase in the number of views N_v.

Specifically, regarding I-ROC, when N_v=1, using only F_{2D} does not perform as well as using only F_{3D}. However, their combination, F_{Ours}, significantly outperforms using either feature type alone. As Nv increases, the performance of using only F_{2D} steadily improves and eventually exceeds that of F_{3D}. This suggests that multi-view images can more effectively capture geometrical information in PCD. Moreover, F_{Ours} consistently outperforms single feature types, showing about a 6% improvement when using ResNet18, effectively highlighting the complementary nature of F_{2D} and F_{3D}.

Regarding P-PRO, using only F_{2D} generally results in poorer performance compared to using only F_{3D} across nearly all scenarios, even with the use of multiple views. This may be due to F_{2D} having a larger receptive field than F_{3D}, which leads to weaker geometrical point-wise features. The combined feature F_{Ours} slightly outperforms the individual features, showing an improvement of about 0.6% when ResNet18 is used. In summary, F_{3D} performs better than F_{2D} at the pixel level but worse at the image level. This indicates that the 3D modality features have stronger geometrical information but weaker semantics compared to the 2D modality features. Combining these features provides both local geometrical and global semantic contexts, leading to improved performance at both the image and pixel levels.

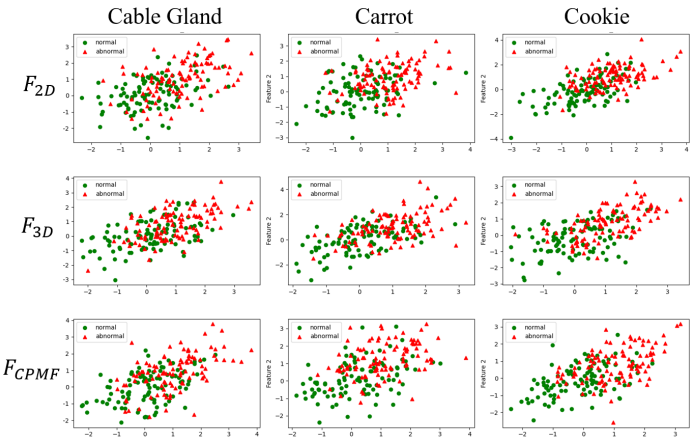

Figure 5. Visualization for feature distributions.

Figure 2 demonstrates that F_{2D} and F_{3D} exhibit different strengths in detecting PCD anomalies. For instance, F_{2D} alone effectively localizes anomalies in categories such as cable gland, carrot, and dowel but performs poorly in categories like bagel and potato, where F_{3D} excels. The combination of F_{2D} and F_{3D}, forming F_{Ours}, markedly enhances anomaly detection performance and achieves impressive localization across all categories, as illustrated in Figure 2. Figure 5 visualizes the feature distributions of F_{2D}, F_{3D}, and F_{Ours}. It reveals that single-type features may not be well distinguished, whereas the distribution of F_{Ours} is more distinct.

4.3.3 *Influence of backbones*

Figure 3 compares Ours's performance using various backbones, including ResNet18, ResNet34, ResNet50, and Wide_ResNet_50_2 [34]. Table 3 summarizes the best performance results for each backbone. Across different backbone types, using only F_{2D} yields poorer pixel-level performance but better image-level

Table 3. QUANTITATIVE RESULTS.

Backbone	Feature	I-ROC	P-PRO
	P-PRO	0.8304	0.9230
ResNet18	873+234	0.8918	0.9145
	819+211	0.9515	0.9293
ResNet34	814+213	0.8987	0.9135
	553+134	0.9492	0.9286
ResNet50		0.8932	0.8977
		0.9479	0.9233
Wide_ResNet_50_2		0.8911	0.9038
		0.9464	0.9256

performance compared to using only F_{3D}. Combining both features enhances performance at both image and pixel levels. Additionally, the backbone type does not significantly impact overall performance, with Ours achieving the best results using ResNet18, boasting a 95.15% I-ROC and a 92.93% P-PRO.

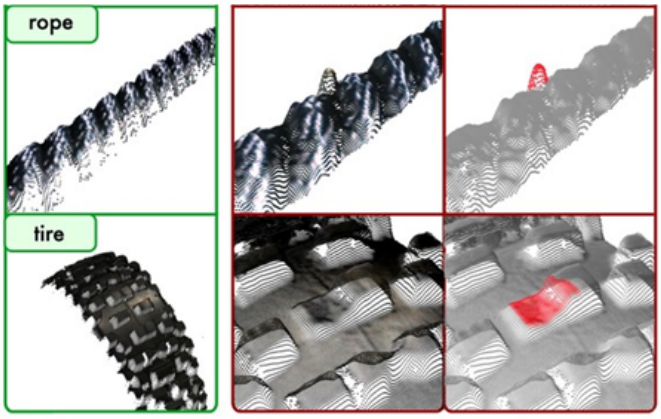

Figure 6. Illustration for the bad quality of rendered images resulting by acquisition noise.

4.3.4 limitation

In this subsection, several limitations are discussed. First, as shown in Figure 6, the quality of rendered images can be compromised due to noise introduced during PCD acquisition. This degradation in quality can negatively impact feature capability and lead to incorrect judgments. Second, while the current pixel-wise criterion P-PRO effectively reveals the performance of detecting various anomalies, certain anomalies can only be identified with RGB information. This discrepancy results in Ours achieving excellent but not optimal performance and underscores the need for a more equitable metric for point-wise PCD anomaly localization.

5 Conclusion

This study introduces swift object detection in point clouds by employing CNNs built from sparse convolutional layers, adopting the voting mechanism outlined in [5]. Leveraging hierarchical representations and non-linear decision boundaries, our approach attains cutting-edge performance on the MVTec 3D-AD benchmark for point cloud object detection. Moreover, our method surpasses the majority of methods that combine information from both point clouds and images across diverse test scenarios.

Future directions for this research include exploring more granular input representations and developing a GPU implementation of the voting algorithm to further enhance detection speed and efficiency. These improvements could provide even faster and more accurate object detection capabilities in 3D environments, making the approach more viable for real-time applications in autonomous driving and robotics.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

This work was supported without any funding.

References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

[2] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. [CrossRef]

[3] Hu, H., Gu, J., Zhang, Z., Dai, J., & Wei, Y. (2018). Relation networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3588-3597).

[4] Pan, X., Xia, Z., Song, S., Li, L. E., & Huang, G. (2021). 3d object detection with pointformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 7463-7472).

[5] Wang, D. Z., & Posner, I. (2015, July). Voting for voting in online point cloud object detection. In *Robotics: science and systems* (Vol. 1, No. 3, pp. 10-15).

[6] Geiger, A., Lenz, P., & Urtasun, R. (2012, June). Are we ready for autonomous driving? the kitti vision

benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 3354-3361). IEEE. [CrossRef]

[7] Li, B., Zhang, T., & Xia, T. (2016). Vehicle detection from 3d lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*.[CrossRef]

[8] Chauhan, R., Ghanshala, K. K., & Joshi, R. C. (2018, December). Convolutional neural network (CNN) for image detection and recognition. In *2018 first international conference on secure cyber computing and communication* (*ICSCCC*) (pp. 278-282). IEEE.[CrossRef]

[9] Fathy, M., & Siyal, M. Y. (1995). An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis. *Pattern Recognition Letters*, 16(12), 1321-1330.[CrossRef]

[10] Liang, S., Li, Y., & Srikant, R. (2017). Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*. [CrossRef]

[11] Suthaharan, S., & Suthaharan, S. (2016). Support vector machine. *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*, 207-235.

[12] Maturana, D., & Scherer, S. (2015, September). Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems* (*IROS*) (pp. 922-928). IEEE. [CrossRef]

[13] Maturana, D., & Scherer, S. (2015, May). 3d convolutional neural networks for landing zone detection from lidar. In *2015 IEEE international conference on robotics and automation* (*ICRA*) (pp. 3471-3478). IEEE. [CrossRef]

[14] Graham, B. (2014). Spatially-sparse convolutional neural networks. *arXiv preprint arXiv:1409.6070*. [CrossRef]

[15] Graham, B. (2015). Sparse 3D convolutional neural networks. *arXiv preprint arXiv:1505.02890*. [CrossRef]

[16] Jampani, V., Kiefel, M., & Gehler, P. V. (2016). Learning sparse high dimensional filters: Image filtering, dense crfs and bilateral neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4452-4461).

[17] Chen, H., Dou, Q., Yu, L., & Heng, P. A. (2016). Voxresnet: Deep voxelwise residual networks for volumetric brain segmentation. *arXiv preprint arXiv:1608.05895*. [CrossRef]

[18] Dou, Q., Chen, H., Yu, L., Zhao, L., Qin, J., Wang, D., ... & Heng, P. A. (2016). Automatic detection of cerebral microbleeds from MR images via 3D convolutional neural networks. *IEEE transactions on medical imaging*, 35(5), 1182-1195. [CrossRef]

[19] Prasoon, A., Petersen, K., Igel, C., Lauze, F., Dam, E., & Nielsen, M. (2013, September). Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network. In *International conference on medical image computing and computer-assisted intervention* (pp. 246-253). Berlin, Heidelberg: Springer Berlin Heidelberg.

[20] Derpanis, K. G. (2010). Overview of the RANSAC Algorithm. *Image Rochester NY*, 4(1), 2-3.

[21] Khan, K., Rehman, S. U., Aziz, K., Fong, S., & Sarasvady, S. (2014, February). DBSCAN: Past, present and future. In The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014) (pp. 232-238). IEEE. [CrossRef]

[22] Zhou, Y., Ren, F., Nishide, S., & Kang, X. (2019, November). Facial sentiment classification based on resnet-18 model. In *2019 International Conference on electronic engineering and informatics* (*EEI*) (pp. 463-466). IEEE. [CrossRef]

[23] Bergmann, P., Jin, X., Sattlegger, D., & Steger, C. (2021). The mvtec 3d-ad dataset for unsupervised 3d anomaly detection and localization. *arXiv preprint arXiv:2112.09045*. [CrossRef]

[24] Rudolph, M., Wehrbein, T., Rosenhahn, B., & Wandt, B. (2023). Asymmetric student-teacher networks for industrial anomaly detection. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 2592-2602).

[25] Bergmann, P., & Sattlegger, D. (2023). Anomaly detection in 3d point clouds using deep geometric descriptors. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 2613-2623).

[26] Cao, Y., Xu, X., & Shen, W. (2024). Complementary pseudo multimodal feature for point cloud anomaly detection. *Pattern Recognition*, 156, 110761. [CrossRef]

[27] Wei, X., Yu, R., & Sun, J. (2020). View-GCN: View-based graph convolutional network for 3D shape analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 1850-1859).

[28] Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6), 381-395. [CrossRef]

[29] Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226-231).

[30] Zhou, Q. Y., Park, J., & Koltun, V. (2018). Open3D: A modern library for 3D data processing. *arXiv preprint arXiv:1801.09847*. [CrossRef]

[31] Rusu, R. B., Blodow, N., & Beetz, M. (2009, May). Fast point feature histograms (FPFH) for 3D registration. In *2009 IEEE international conference on robotics and*

automation (pp. 3212-3217). IEEE. [CrossRef]

[32] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[33] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115, 211-252.

[34] Zagoruyko, S. (2016). Wide residual networks. *arXiv preprint arXiv:1605.07146*.

[35] Horwitz, E., & Hoshen, Y. (2023). Back to the feature: classical 3d features are (almost) all you need for 3d anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2968-2977).

Tianyi Lyu received his Master's degree in Software Engineering from Northeastern University, Boston, USA. His research interests include deep learning, computer vision, and 3D object detection. He has contributed to multiple projects related to efficient processing of large-scale data, particularly focusing on optimizing convolutional neural networks for real-time applications. Tianyi is currently working as a software engineer, where he applies his expertise in AI and machine learning to industrial challenges.

Dian Gu earned her Master's degree from the University of Pennsylvania, USA, specializing in artificial intelligence and data science. Her research encompasses neural network optimization, with a particular focus on Internet of Things (IoT) applications and 3D data analysis. Dian has published several papers on efficient neural network architectures, contributing to the advancement of AI-driven systems in diverse fields.

Peiyuan Chen holds his Master's degree in Electrical Engineering at Oregon State University, Corvallis, Oregon, USA. His academic interests include deep learning, neural network design, and large-scale data processing for 3D object detection. Peiyuan has worked on several projects aimed at improving the accuracy and speed of convolutional neural networks, particularly in applications involving real-time object recognition.

Yaoting Jiang holds a Master's degree from Carnegie Mellon University, USA, where she specialized in the fields of computer vision and machine learning. Her research focuses on developing efficient algorithms for 3D point cloud processing and enhancing the performance of convolutional neural networks. YaoTing has a strong background in AI applications, particularly in the context of real-time data processing and analysis.

Zhenhong Zhang holds a Master's degree from George Washington University, USA, where he focuses on the intersection of AI, computer vision, and 3D object detection. His work has been pivotal in advancing the use of deep learning for efficient data analysis in high-dimensional datasets. Zhenhong's research aims to bridge the gap between theoretical AI models and their practical applications in industry.

Huadong Pang is a Master's graduate from Georgia Institute of Technology, USA, with expertise in AI, machine learning, and data science. His research interests include the development of neural network architectures optimized for sparse data, particularly in the context of 3D object detection. Huadong has contributed to multiple projects aimed at improving the computational efficiency of large-scale data processing systems.

Li Zhou is an independent researcher who earned his Master's degree in Management from McGill University, Montreal, Canada. His research spans the fields of data science and AI, with a particular focus on applying machine learning techniques to real-world problems. Li's recent work explores optimizing neural networks for high-performance computing, particularly in large-scale object detection scenarios.

Yiping Dong is a Master's graduate from Carnegie Mellon University, Pittsburgh, USA, specializing in mechanical engineering with a focus on AI applications. His research interests lie in convolutional neural networks and their application to 3D object detection. Yiping's work has led to significant improvements in the efficiency and accuracy of neural networks for real-time object recognition, making him a leader in his field. (Email: dand97personal@gmail.com)