Special Section on 3DOR 2025

Denoising-While-Completing Network (DWCNet): Robust point cloud completion under corruption

Keneni W. Tesema [a],*, Lyndon Hill [b], Mark W. Jones [a], Gary K.L. Tam [a]

[a] *Swansea University, Fabian Way, Swansea, SA1 8EN, Wales, United Kingdom*
[b] *Beam, Bristol, England, United Kingdom*

ARTICLE INFO

ABSTRACT

Point cloud completion is crucial for 3D computer vision tasks in autonomous driving, augmented reality, and robotics. However, obtaining clean and complete point clouds from real-world environments is challenging due to noise and occlusions. Consequently, most existing completion networks – trained on synthetic data – struggle with real-world degradations. In this work, we tackle the problem of completing and denoising highly corrupted partial point clouds affected by multiple simultaneous degradations. To benchmark robustness, we introduce the Corrupted Point Cloud Completion Dataset (CPCCD), which highlights the limitations of current methods under diverse corruptions. Building on these insights, we propose DWCNet (Denoising-While-Completing Network), a completion framework enhanced with a Noise Management Module (NMM) that leverages contrastive learning and self-attention to suppress noise and model structural relationships. DWCNet achieves state-of-the-art performance on both clean and corrupted, synthetic and real-world datasets. The dataset and code will be publicly available at https://github.com/keneniwt/DWCNET-Robust-Point-Cloud-Completion-against-Corruptions.

1. Introduction

Point cloud completion is the task of completing a partial point cloud input so that it fully represents a 3D shape [1,2]. It is a critical component in various tasks such as object recognition, 3D reconstruction, and point cloud pre-training [1,2]. These tasks have practical applications in diverse fields including autonomous driving [3,4], robotics [5], and augmented reality [6], among others.

In recent years, research in point cloud completion has witnessed considerable advancements, with a primary focus on deep learning-based techniques [1,2]. Algorithms like ODGNet [7] and AdaPoinTr [8] have demonstrated promising results on datasets such as PCN [9], MVP [10] and KITTI [11]. These techniques heavily depend on comprehensive training data from datasets like ShapeNet [12].

Supervised point cloud completion differs from other point cloud learning tasks such as segmentation or object detection in the type of supervision it requires. While segmentation and detection rely on labeled annotations (e.g., class labels or bounding boxes), completion networks require clean and complete 3D point clouds as ground truth during training. However, obtaining such high-quality reference data from real-world scans is extremely challenging. As a result, most existing methods are trained and evaluated on synthetic datasets. Although

these datasets are diverse and consistent [9,10,13], they are typically generated by uniformly sampling 3D meshes [14], which makes them overly clean and unrepresentative of the noise and artifacts found in real-world data [15]. This leads to two key observations: (1) the lack of a benchmark dataset designed to systematically evaluate robustness against corrupted inputs, and (2) the inability of current completion methods to generalize well to real-world-like corruptions.

There have been limited attempts to incorporate noise and corruptions into partial point clouds for completion [8,16]. However, these efforts address only Gaussian noise, which represents just one of the many potential corruptions in real-world point clouds. Furthermore, the fidelity of synthetic partial point clouds used to train completion algorithms remains subpar compared to real-world scans. Even the real-world dataset most commonly used for evaluation, KITTI [11], lacks environmental corruptions [17]. Point clouds from the ScanNet dataset [18], often used for evaluation [19,20], are preprocessed before completion, and the results are typically qualitative, leading to subjective assessments. Thus, there is a clear need for a corrupted benchmark dataset to enable objective and qualitative robustness evaluation.

To address the first issue, we introduce the Corrupted Point Cloud Completion Dataset (CPCCD) (Fig. 1). We classify the corruptions observed in the real-world ScanObjectNN dataset [21] (Fig. 2), then

* Correspondence to: Computational Foundry, Bay Campus, Swansea University, SA18EN, United Kingdom.
  E-mail addresses: k.w.tesema@swansea.ac.uk (K.W. Tesema), l.hill@ieee.org (L. Hill), m.w.jones@swansea.ac.uk (M.W. Jones), k.l.tam@swansea.ac.uk (G.K.L. Tam).
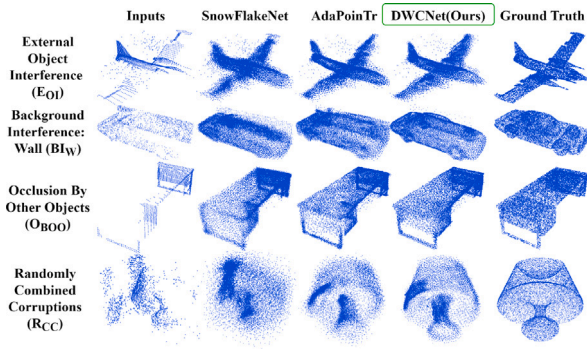
Fig. 1. We propose a corrupted point cloud completion benchmark dataset (CPCCD) and a robust point cloud completion network, DWCNet.

Fig. 2. Real world scans (like ScanObjectNN dataset) often contain corruptions (shown in the ellipses) while point clouds in point cloud completion datasets (like PCN) are often clean.

mimic and incorporate these corruptions into the clean partial point clouds from the PCN dataset. We categorize the corruptions into two types: External Corruptions, originating from points belonging to other objects or the background, and Internal Corruptions, which displace or distort points from the target object (shown in Fig. 3). Unlike previous works [15,17], which use fixed corruption levels, we introduce randomness by allowing a range of values for the defining parameters, within spatial constraints. This design aims to reflect the unpredictable nature of noise and corruption in real-world point cloud scans, which is lacking in current completion benchmark datasets.

To address the lack of robust completion algorithms, we introduce Denoising-While-Completing (DWCNet). DWCNet utilizes a novel Noise Management Module (NMM) to classify features from a noisy partial point cloud into clean and noisy categories, alleviating the issues of outliers. Clean features are filtered through contrastive learning in feature space. NMM employs Multi-Head Self-Attention (MHSA) to determine structural relations and multi-scale convolutions to capture noise at different scales. This integrated approach ensures that the point cloud is both denoised and completed in a single, cohesive step, enhancing output quality. Overall, our contributions are:

- We formulate and systematically approach an existing but underexplored problem in point cloud completion: the challenge of completing highly corrupted (noisy) partial point clouds.
- We introduce a novel corrupted point cloud completion dataset (CPCCD) as the first robustness benchmark in the field of point cloud completion.
- We offer the first systematic evaluation of the robustness of completion networks, examining how robustness relates to different types of corruptions and network architectures.
- We introduce DWCNet, a completion algorithm that integrates denoising and completion through a novel Noise Management Module, producing relatively clean, complete point clouds from noisy inputs. DWCNet achieves state-of-the-art results on the PCN, CPCCD, and ScanObjectNN datasets, demonstrating robustness on corrupted data.

2. Related work

This section surveys datasets and techniques used in point cloud completion. Point clouds, essential for 3D vision and research, can be acquired through active or passive methods [22]. Active acquisition directly captures point clouds using sensors like LiDAR, Structured Light, and Time of Flight (ToF) cameras [23], while passive acquisition derives point clouds from other data types such as meshes, depth images, and videos [22]. Point clouds from active acquisition are more susceptible to noise and corruption due to factors like the acquisition
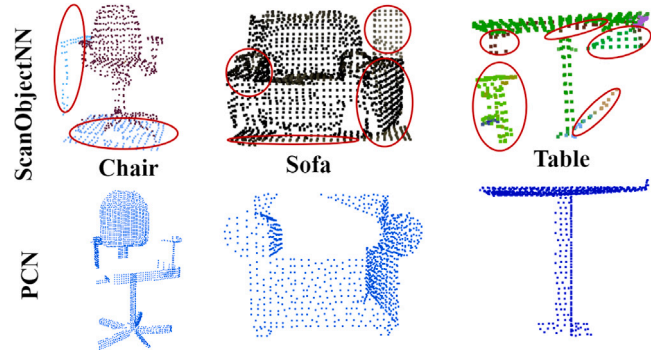
environment, measurement devices, or human error [24], in contrast to those acquired passively.

Datasets in Point Cloud Completion: Due to challenges in active acquisition, researchers often use passively acquired, clean, and synthetic pairs of partial and complete point clouds for training. Notable benchmarks include PCN [9], MVP [10], and Completion3D [13]. Techniques such as back projection [9], adding random Gaussian noise [16], and "noised back projection" from depth images [8] have been employed to make partial point clouds resemble real-world scans. However, these methods do not fully capture the noise and corruptions common in real-world data [15,17,24,25]. While recent work has explored corruptions in point clouds for other tasks [15,17,24,25], they remain absent in point cloud completion datasets. We address this gap by introducing a robustness benchmark with corrupted and noisy partial point clouds, offering a more challenging and practical evaluation environment.

Techniques in Point Cloud Completion: Point cloud completion techniques can be broadly categorized into traditional and deep learning-based approaches [2]. Traditional methods rely on parameter optimization to fill small missing regions [2], whereas deep learning methods use neural networks to extract features and complete larger missing areas [1]. Recent research emphasizes improving performance through architectural innovations, including point-based, convolutional, graph-based, generative, and transformer-based models. Among these, transformer-based networks like [8] have shown strong results on both synthetic and real-world datasets.

Despite these advancements, most methods assume clean point clouds, overlooking the effects of noise and corruption. A few notable efforts address this issue: Agrawal et al. [26] apply a denoising auto-encoder before using a network trained on clean data; Arora et al. [16] study noise tolerance with Gaussian noise; PDCNet [27] uses density-based clustering; Ma et al. [28] apply segmentation and Farthest Point Sampling (FPS) to exclude outliers; and Li et al. [29] use outlier suppression during patch training. However, only very few approaches consider multiple noise and corruption types simultaneously. Our work evaluates various completion models under these conditions and introduces DWCNet, a robust network enhanced with a specialized module for handling corruption.

Corruption in Point Cloud Learning tasks: Recently, there has been growing interest in studying dataset corruptions in 3D learning tasks. Notable efforts include ShapeNet-C [15], ModelNet-C [24], Robo3D [17], and 3D-Corruptions-AD [25]. Ren et al. [15] introduce ModelNet-C and ShapeNet-C by applying various noise types and augmentations for classification and part segmentation. Robo3D and 3D-Corruptions-AD focus on outdoor autonomous driving scenarios, providing corrupted LiDAR scans and benchmarking models on detection and segmentation. Robo3D also introduces corrupted versions of real-world
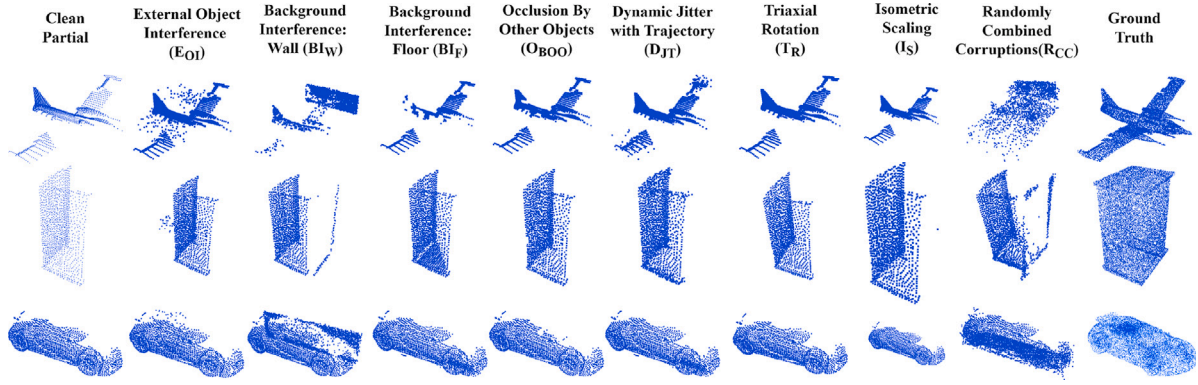
Fig. 3. CPCCD Dataset and its different types of corruptions.

Table 1
Comparison of different completion benchmark datasets. "Type" shows whether the datasets are synthetic or real-world and "Paired" shows if the datasets contain pairs of partial and complete point clouds.

Dataset	Type	Paired	Corruption
PCN [9]	Synthetic	Yes	Back-projection
Completion3D [13]	Synthetic	Yes	Cropped
MVP [10]	Synthetic	Yes	Back projection
ShapeNet33(54) [12]	Synthetic	Yes	Back projection, cropped
KITTI [11]	Real-world	No	Occlusion
ScanNet [18]	Real-world	No	Occlusion
Projected Shapenet [8]	Synthetic	Yes	Back projection with Gaussian noise
Ours(CPCCD)	Synthetic	Yes	Back projection, External object Interference (E_{OI}), Background Interference Wall, Floor (BI_w, BI_f), Occlusion by Other Objects(O_{BOO}), Dynamic Jitter with Trajectory (D_{JT}), Triaxial Rotation (T_R), Isometric Scaling (I_S), Random Combined Corruptions (R_{CC})

datasets: KITTI-C, Waymo-C, and nuScenes-C. In contrast, our proposed dataset, CPCCD, is designed specifically for the task of completion of point clouds of objects captured in indoor environments. Although it follows the broader trend of augmenting existing datasets, CPCCD introduces a diverse range of corruption types with randomized parameters to better reflect the unpredictable nature of real-world noise — rather than relying on rigidly defined severity levels or fixed noise patterns.

Point Cloud Denoising aims to remove noise and corruption to recover clean point clouds [30]. Methods include filter-based, optimization-based, and deep learning approaches. Most are designed for specific noise types and struggle with the complex, compound noise in real-world scans [30]. Current state-of-the-art techniques typically denoise complete point clouds and handle only limited corruption types [30]. Among the limited works that attempt to address multiple noise types are: Score-Denoise [31], which estimates the noise distribution score of noisy point clouds; PathNet [32], which introduces path-selective denoising for different noise types; NoiseTrans [33], which leverages self-attention mechanisms for point cloud denoising; and Total Denoising [34], which adapts unsupervised image denoising techniques to 3D point clouds. While these methods show progress on specific or similar noise types, none of them address denoising for partial point clouds or diverse corruptions simultaneously, which is the focus of our work.

Contrastive learning is a self-supervised or supervised technique that enables a model to distinguish between similar and dissimilar samples [35,36]. It helps learn meaningful 3D representations by leveraging local geometric consistency, and has been applied in point cloud pretraining [37], as well as improving performance in point cloud detection [38] and segmentation [35].

3. Corrupted Point Cloud Completion Dataset (CPCCD)

As identifying clean vs. noisy points at a fine level is infeasible [30], both completion and denoising methods typically rely on synthetic

datasets for training and evaluation [30]. However, current completion benchmark datasets [9,10,13] exhibit little to no corruption. Real-world datasets like KITTI [11] and ScanNet [39] are preprocessed before evaluation [17,19]. The domain gap between real and synthetic data complicates fair evaluation [19], and qualitative results are often subjective. Due to the lack of ground truth and the unpredictable nature of real-world corruptions, real data cannot serve as a reliable robustness benchmark. Our goal is to create a robustness benchmark for object-level point cloud completion by simulating corruptions likely to occur in indoor environments.

To create CPCCD (Fig. 3), we first visually inspect a real-world indoor point cloud dataset, ScanObject-NN [21] (shown in Fig. 2), to identify common corruptions in indoor environments. We selected ScanObject-NN because it covers all objects in the ScanNet dataset, rather than just those chosen by researchers in completion studies [19, 40]. After identifying these corruptions, we apply them to the partials in the PCN dataset [9].

We categorized the observed corruptions into two groups: External and internal corruptions. **External corruptions** add points from unrelated objects or backgrounds (e.g., floors, walls), while **Internal corruptions** disrupt or displace points within the target object, often due to sensor errors, occlusions, or scaling issues. We applied these corruptions to the partials in the PCN dataset [9], preserving the train, test, and validation splits. The PCN dataset contains partial and complete point clouds for 30,974 shapes from 8 categories (Plane, Cabinet, Car, Chair, Lamp, Couch, Table, Boat), derived from ShapeNet [12]. Complete point clouds are generated by uniformly sampling ShapeNet CAD models, and partials are created by back-projecting depth images from 8 viewpoints. We then applied the observed eight different corruptions (discussed below) to each of the partials. CPCCD consists of the equivalent of 8 PCN datasets, with 8 variations for each of the point clouds in the training, testing and validation dataset. A breakdown about the number of point clouds in CPCCD is given in Appendix C. We compare CPCCD to other completion datasets in Table 1.

Table 2
CPCCD dataset and its parameter definition. Possible Severity level (P) represent all the possible combinations of parameter values that can determine the severity level of the corruption.

Noise	Noise subclass	Purpose	Parameter domain	Possible severity levels
External	External object interference (E_{OI})	Represents corruption by points belonging to the other objects	$No\_of\_Objects(N_o) = [1, 2, 3],$ $No\_of\_points(N_p) = [1/16, 1/12, 1/8, 1/4],$ $No\_of\_Shapes(N_s) = [12],$ $Distance\_from\_obj(Nd) = [0.05, 0.2]$	$P_{E_{OI}} = N_o \times N_p \times N_s \times N_d$
External	Background interference: Wall (BI_w)	Represents points from walls or vertical flat surfaces captured in the point cloud scan of the object	Distance from the object, : $N_d = [0.01 - 0.05]$	$P_{BI_W} = N_d$
External	Background interference: Floor (BI_f)	Represents the floor points	–	$P_{BI_F} = 1$
Internal	Occlusion by other objects (O_{BOO})	Represents missing points due to occlusions by other objects in the scene	$No\_of\_Objects(N_o) = [1, 2, 3, 4],$ $No\_of\_points(N_p) = [1/8, 1/7, 1/6, 1/5, 1/4, 1/3],$ $No\_of\_Shapes(N_s) = [12],$ $Distance\_from\_obj(N_d) = [0.05, 0.2]$	$P_{O_{BOO}} = N_o \times N_s \times N_d \times N_s$
Internal	Dynamic jitter with trajectory (D_{JT})	Represents displacement of points in the object scan in the trajectory	Jitter amount(J_a) = $[0.01, 0.05]$, Trail Distance(T_d) = $[0.02, 0.04]$	$P_{D_{JT}} = J_a \times T_d$
Internal	Triaxial rotation (T_R)	Represents change in viewpoints due to environmental factors	$\theta_x, \theta_y, \theta_z$	$P_{T_R} = \theta x \times \theta y \times \theta z$
Internal	Isometric scaling (I_S)	Represents difference of scale between real world objects	$N_s = [0.25, 2]$	$P_{I_S} = N_s$
Random	Random combined corruption (R_{CC})	Represents the various combinations that corruptions appear in the real world	A random combination of any 2 to 7 of the corruptions above $\{S \subseteq C \mid 2 \le |S| \le 7\}$	$P_{R_{CC}} = P_{E_{OI}} \times P_{BI_W} \times P_{BI_F} \times$ $P_{O_{BOO}} \times P_{D_{JT}} \times P_{T_R} \times P_{I_S}$

External Object Interference (E_{OI}): To generate external object corruption, we randomly select one to three shapes from a set of 12 basic shapes, $N_s = 12$: Circle, Square, Rectangle, Triangle, Ellipse, Hexagon, Diamond, Parallelogram, Cylinder, Sphere, Cube, and Pyramid. These shapes were chosen as they can represent parts of more complex objects. Simulating the scenario where external objects are captured in the point cloud scan of the target object (like those in Fig. 2), we randomly apply $N_o = [1, 2, 3]$ shapes. We control their proximity to the target object using N_d, which ranges from $N_d = [0.05, 0.2]$ to ensure that the shapes do not occlude the object but remain within its bounding box. The added shapes contribute N_p points, where $N_p = [1/16, 1/12, 1/8, 1/4]$, ensuring that the corruption introduces a meaningful challenge rather than simple outliers that could be removed by basic outlier removal techniques.

Background Interference: Wall (BI_w): This simulates the presence of walls, doors, and other vertical, flat objects typically found in indoor scenes. The wall is positioned behind the target object, so points from the wall appear within the object's bounding box without occluding it. The proximity is controlled by N_d, where $N_d = [0.01, 0.05]$. Points on the wall that are occluded by the object are removed by projecting the object's points onto the wall and eliminating overlapping points.

Background Interference: Floor (BI_F): This corruption simulates the floor and appears only when the bottom quarter of the object is captured in the point cloud. Points from the floor that may be occluded by the object are removed by projection.

Occlusion by Other Objects (O_{BOO}): This corruption simulates occlusion by other objects in the scene. It removes points from the object of interest based on simple shapes $N_s = 12$ (as described in external object interference (E_{OI})). N_p represents the number of points dropped due to occlusion, with values $N_p = [1/8, 1/7, 1/6, 1/5, 1/4, 1/3] * N_t$, where N_t is the total number of points in the object. This range ensures a meaningful portion of the object remains while causing a significant change.

Dynamic Jitter with Trajectory (D_{JT}): This corruption simulates jitter caused by motion. Unlike other corruption datasets [15], we apply jitter only to a select few points in the object, with displacement J_a ranging

from $[0.01, 0.05]$. These points are then displaced along a trail distance $T_d = [0.02, 0.04]$ meters in a random direction. The small trail distance reflects an indoor environment scenario.

Isometric Scaling (I_s): This corruption simulates scale mismatches between the input and ground truth, as well as real-world size differences between objects (e.g., a chair vs. a car). To account for this, we vary the scale of the partials by a factor I_s within the range $[0.25, 2]$.

Triaxial Rotation (T_R): Rotation accounts for any misalignment during acquisition. We rotate the angles of the x, y, and z axes by $\theta_x, \theta_y, \theta_z$ which can range from zero to ten degrees each.

Randomly Combined Corruption (R_{CC}): This corruption involves a random combination of two or more of the seven corruptions described above. It simulates how real-world corruptions often occur in an unpredictable and overlapping manner. Let C be the set of all possible corruptions: $C = [E_{OI}, BI_W, BI_F, O_{BOO}, D_{JT}, I_S, T_R]$. The random corruption R_{CC} is any subset S of C such that: $R_{CC} \in \{S \subseteq C \mid 2 \le |S| \le 7\}$. All corruptions and their levels are described in Table 2. Examples of objects from the CPCCD dataset are shown in Fig. 3.

4. Denoising-While-Completing Network (DWCNet)

Our robustness analysis on the proposed CPCCD dataset (Section 5) shows that state-of-the-art completion models are highly vulnerable to noisy inputs. Although retraining on CPCCD improves their robustness, these models still struggle to effectively remove noise. This limitation motivates our proposed solution: the Denoising-While-Completing Network (DWCNet). DWCNet tackles the limitation by integrating a dedicated **Noise Management Module (NMM)** to a base completion algorithm to simultaneously denoise and complete corrupted partial point clouds. The key idea behind DWCNet is to identify and filter out noisy features in the pipeline, allowing the model to concentrate on clean, informative data during completion. This selective processing reduces noise propagation and leads to more accurate and robust reconstructions.

In our design, we choose AdaPoinTr [8] as the base completion algorithm due to its integrated Gaussian denoising mechanism and strong performance on CPCCD (see Section 5). We then introduce the NMM module in its core to separate clean and noisy features (Fig. 4).
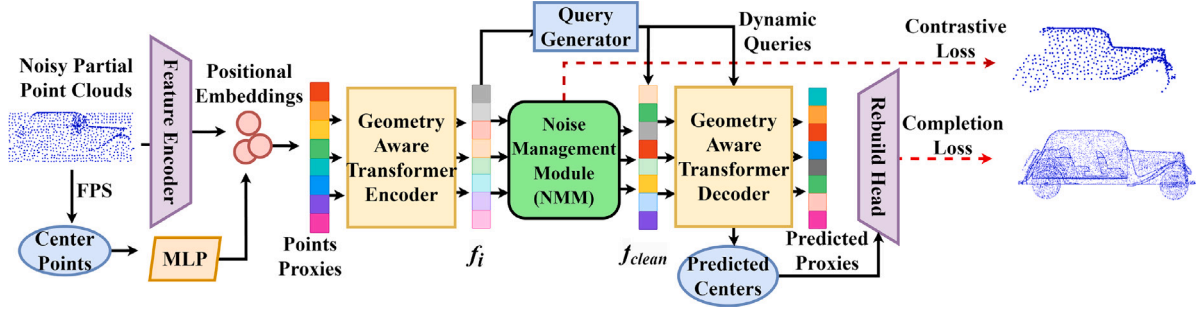
Fig. 4. Overall architecture of DWCNet. The input point cloud is first down-sampled and local features are extracted using DGCNN [41], following the backbone of AdaPoinTr [8]. The extracted features, combined with positional embeddings, are passed through a transformer encoder. The resulting encoded features are then processed by the NMM, which classifies them as either "clean" or "noisy". Only the clean features are passed to the transformer decoder, which predicts a coarse point cloud and point proxies. These are further refined and upsampled to a complete point cloud by using a folding decoder.
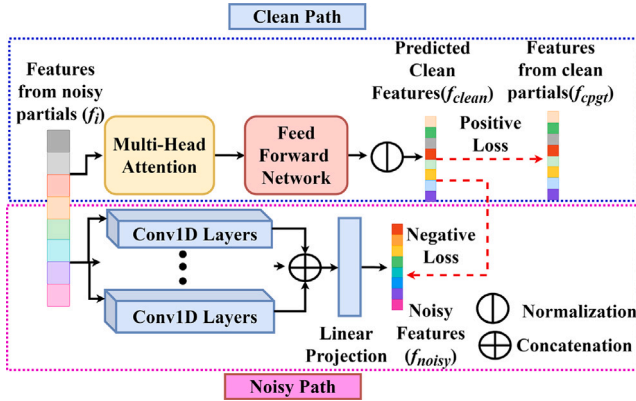

Fig. 5. Noise Management Module (NMM).

Specifically, the DWCNet pipeline begins with an input point cloud $P = \{P_x, P_y, P_z\}$ of size $N \times 3$, where $N < 16\,384$. This input is downsampled to 2048 points and grouped into local patches (e.g. 32 groups of 64 points), which are embedded into higher-dimensional representations – point proxies – using multilayer perceptrons (MLPs) and positional encodings, resulting in a feature tensor of shape 32×256. These proxies are processed by a geometry-aware Transformer encoder that preserves local structure. The resulting global feature f_i is passed to the Noise Management Module (NMM), which separates it into clean (f_{clean}) and noisy (f_{noisy}) components, padding as needed to maintain dimensional consistency. The clean features f_{clean} replace f_i for subsequent processing. An adaptive query generation module then produces 1024 learned queries of shape 1024×256, which are refined by a Transformer decoder. The decoder output is mapped to 3D coordinates via a point generation head, yielding a coarse completed point cloud of shape 1024×3. This is then upsampled to a final point cloud of shape $16\,384 \times 3$.

4.1. Noise Management Module (NMM)

Inspired by speech enhancement research [42–44], which jointly models and separates signal and noise, we propose the Noise Management Module (NMM). It is a dual-path neural network that classifies features encoded by the completion network as noisy or clean using self-attention and contrastive learning [45]. The clean features are then passed back into the completion pipeline to predict the final complete point clouds.

The architectural details of NMM are as follows. NMM processes input features through two parallel pathways. The first path is a *clean path*, where a transformer-based module captures long-range dependencies via multi-head self-attention [46], followed by a position-wise

feed-forward network (FFN). Pre-Layer normalization is applied to prevent feature magnitude divergence and residual connections are used to mitigate vanishing gradients. The self-attention employs eight attention heads to capture global dependencies while the FFN refines features per-position. Given an input feature f_i,

$$f_i \in \mathbb{R}^{B \times L \times D}$$

where B is the batch size, L is the sequence length D is the feature dimension, and $+$ represents addition.

$$f_{\text{MHSA}} = \text{LayerNorm}(f_i + \text{MHSA}(f_i)) \tag{1}$$

$$f_{clean} = \text{LayerNorm}(f_{\text{MHSA}} + \text{FFN}(f_{\text{MHSA}})) \tag{2}$$

where MHSA represents multi-head self-attention and LayerNorm represents normalization. This can be summarized as:

$$f_{clean} = f_i + \underbrace{\text{MHSA}(f_i)}_{\text{Global context}} + \underbrace{\text{FFN}(f_i + \text{MHSA}(f_i))}_{\text{Feature refinement}} \tag{3}$$

The second path is a *noisy path* that uses multi-scale convolutions, consisting of multiple parallel 1D convolutions, to capture noise at different scales. The process is summarized in the equation below.

$$f_{\text{noisy}} = W_m \cdot \text{Concat}\left[\{\text{Conv}_k(f_i^T)\}_{k \in \{1,3,5\}}\right] \tag{4}$$

where f_{noisy} is noisy features, $W_m \in \mathbb{R}^{D \times 3D}$ is learnable merge weights, Conv_k is 1D convolution with kernel size k, $f_i^T \in \mathbb{R}^{B \times D \times L}$ is transposed input feature. k represents convolution kernel sizes of $\{1, 3, 5\}$. B is the batch size, L is the sequence length and D is the feature dimension (1024 in our case).

The features f_{clean} and f_{noisy} from the clean and noisy paths are then linearly projected and merged. The dual-path architecture of the NMM enables explicit separation of processing pathways, allowing specialized feature learning for clean and noisy components. The full NMM architecture is shown in Fig. 5.

The NMM learns discriminative features through a contrastive learning approach, maximizing similarity between clean features and ground truth while minimizing similarity between clean and noisy features. In the NMM, contrastive learning is implemented as follows: First, all input features from the corrupted partial, already processed by the transformer encoder, f_i, clean partial ground truth f_{cpgt}, and complete ground truth point clouds f_{gt} are normalized along the feature dimension. After normalization, two losses are used to optimize feature separation. The first is the *Positive Loss*, which measures the cosine similarity between clean features f_{clean} and the clean partial ground truth f_{cpgt}. Minimizing this loss maximizes similarity between the clean features f_{clean} and the clean partial ground truth f_{cpgt}. The second is the *Negative Loss*, which computes all pairwise similarities between the clean f_{clean} and noisy features f_{noisy}. Diagonal masking is applied to avoid self-comparisons, and temperature scaling [47] is used for stable optimization. Temperature scaling helps balance the positive and

negative terms: a lower temperature $t = 0$ exaggerates the difference between clean and noisy features, while a higher t (e.g., 1.0) leads to a more uniform separation. For contrastive learning, we use a temperature scaling $t = 1$ following findings from our ablation study (see Section 7).

Mathematically, the losses are defined as follows. Given normalized features from the clean partial point cloud $f_{cpgt} \in \mathbb{R}^{B \times L \times D}$, the clean complete ground truth $f_{gt}, \in \mathbb{R}^{B \times L \times D}$, the noisy partial input $f_{input} \in \mathbb{R}^{B \times L \times D}$, the clean features from the clean path $f_{clean}, \in \mathbb{R}^{B \times L \times D}$ and the temperature parameter $t = 1$, the Positive Loss is defined as follows.

$$\mathcal{L}_{pos} = -\frac{1}{B \cdot L} \sum_{i=1}^{B} \sum_{j=1}^{L} f_{clean}^{(i,j)} \cdot f_{cpgt}^{(i,j)} \tag{5}$$

where $\cdot$ denotes the dot product. B is the batch size, L is the sequence length and D is the feature dimension(1024). The *Negative Loss* is defined as follows.

$$\mathcal{L}_{neg} = t \cdot \log \sum_{i=1, j=1, i \neq j}^{B \cdot L} \exp \left(\frac{f_{clean}^{(i)} \cdot f_{noisy}^{(j)}}{t} \right) \tag{6}$$

The total combined loss L_{NMM} is:

$$\mathcal{L}_{NMM} = \mathcal{L}_{pos} + \mathcal{L}_{neg} \tag{7}$$

This contrastive loss from NMM is then added to completion loss for an end-to-end training.

$$\mathcal{L}_{Total} = \mathcal{L}_{completion} + \mathcal{L}_{NMM} \tag{8}$$

5. Robustness evaluation

The following experiments simulate the scenario where existing techniques, trained on the clean PCN dataset, are tested on the corrupted CPCCD dataset. These models are then finetuned and retested on CPCCD. We now discuss the experiment setup, completion algorithms, and evaluation metrics.

5.1. Experimental setup

Our primary selection criterion for completion models is the network architecture. We aimed to test at least one algorithm from all architecture categories: Convolution-based, point-based, graph-based, transformer-based, and generative model-based (Generative Adversarial Network (GAN)-based, Variational AutoEncoder (VAE)-based, and generative-diffusion model-based). It is important to note that most completion networks are hybrids of various architectures. Although we tried to include models from all categories, the availability of source codes and pre-trained models influenced our final selection. We prioritized networks that use the PCN dataset for evaluation, as CPCCD is based on this dataset.

We selected the following completion networks: point-based encoder–decoder (PCN [9]), graph-based (FoldingNet [48]), convolution-based (GRNet [49]), and transformer-based (PoinTr [50], SnowflakeNet [51], and AdaPoinTr [8]). We evaluated the selected algorithms using their provided pretrained weights on the different corruptions of the CPCCD dataset, with results shown in Tables 3, 4, 5 and A.15. Then, we finetuned these models on the Rcc category of the CPCCD dataset and evaluated them again, as shown in Tables 6, 7, 8 and A.16. All models were trained and finetuned using an NVidia A100 GPU (40 GB VRAM), i9-12900K Intel(R) CPU, following the procedures outlined in the original papers.

5.2. Evaluation metrics

We employ Chamfer Distance, F-score and Fidelity, which are commonly used by state-of-the-art completion algorithms, as our performance metrics.

Chamfer Distance (CD) [52]: represents the average distance of closest point between two point clouds. Given two sets of points D_1 and D_2, the Chamfer Distance between the prediction set D_1 and the ground truth D_2 is defined as:

$$CD(D_1, D_2) = \frac{1}{|D_1|} \sum_{x \in D_1} min_{y \in D_2} \|x - y\|_2 + \frac{1}{|D_2|} \sum_{y \in D_2} min_{x \in D_1} \|y - x\|_2 \tag{9}$$

where $\|\cdot\|_2$ is the Euclidean distance. Chamfer Distance is used both as an evaluation measure and a loss function for optimizing learning-based algorithms. It is the most popular metric used in completion. However, it can be insensitive to local density distribution and outliers [53]. The lower the Chamfer Distance, the better the performance. For our experiment, we use CD-L1 which uses L1-norm to calculate difference between two points. L1-norm calculates the sum of the absolute values of the vector elements [54].

F-score [55]: measures completion accuracy through precision and recall. Given the predicted point cloud (D_1) and ground truth (D_2), the F-score is defined as:

$$F\text{-}score(\delta) = \frac{2G(\delta)H(\delta)}{G(\delta) + H(\delta)} \tag{10}$$

where $G(\delta)$ and $H(\delta)$ represent point-wise precision and recall for threshold δ. The higher the F-score the better.

Fidelity [9]: measures the preservation of input data. It computes the average distance from each point in the input to its nearest neighbor in the output. Lower fidelity indicates superior performance, as it means the output more closely matches the input.

6. Results and robustness analysis

The first experiment was conducted to evaluate the ability of completion networks to generalize to corrupted data. In this section, we make observations and analyze the results.

6.1. Robustness before finetuning

The CD_{L1}, $Fscore$ and $fidelity$ results before finetuning are shown in Tables 3–5 respectively.

All evaluated networks were highly vulnerable to external corruptions, particularly external object interference (E_{OI}) and background interference (BI_w and BI_F). While most algorithms showed relative robustness against internal corruptions like dynamic jitter with trajectory (D_{JT}) and occlusion by other objects (O_{BOO}), they performed poorly against isometric scaling (I_S) and triaxial rotation (T_R).

Among all corruptions, Randomly Combined Corruptions (R_{CC}) resulted in the highest Chamfer Distance losses, indicating the worst performance across all categories. Following R_{CC}, the most detrimental corruption was Background Interference: wall (BI_w), which covers a large portion of the bounding box, often causing significant misrepresentation (e.g., Fig. 6, second column, rows two to six shows a sofa instead of a chair). Similar issues were observed with background interference floor (BI_F), where the presence of the floor negatively affected the completion of the chair's legs (Fig. 6, third column, rows four to seven). Isometric scale corruption, reflecting natural scale differences in real-world objects and sensor types, did not affect local details but significantly impacted performance, as algorithms trained on Chamfer Distance struggled with out-of-distribution scales. Dynamic jitter with trajectory (D_{JT}) produced noisy outputs, while triaxial rotation (T_R)
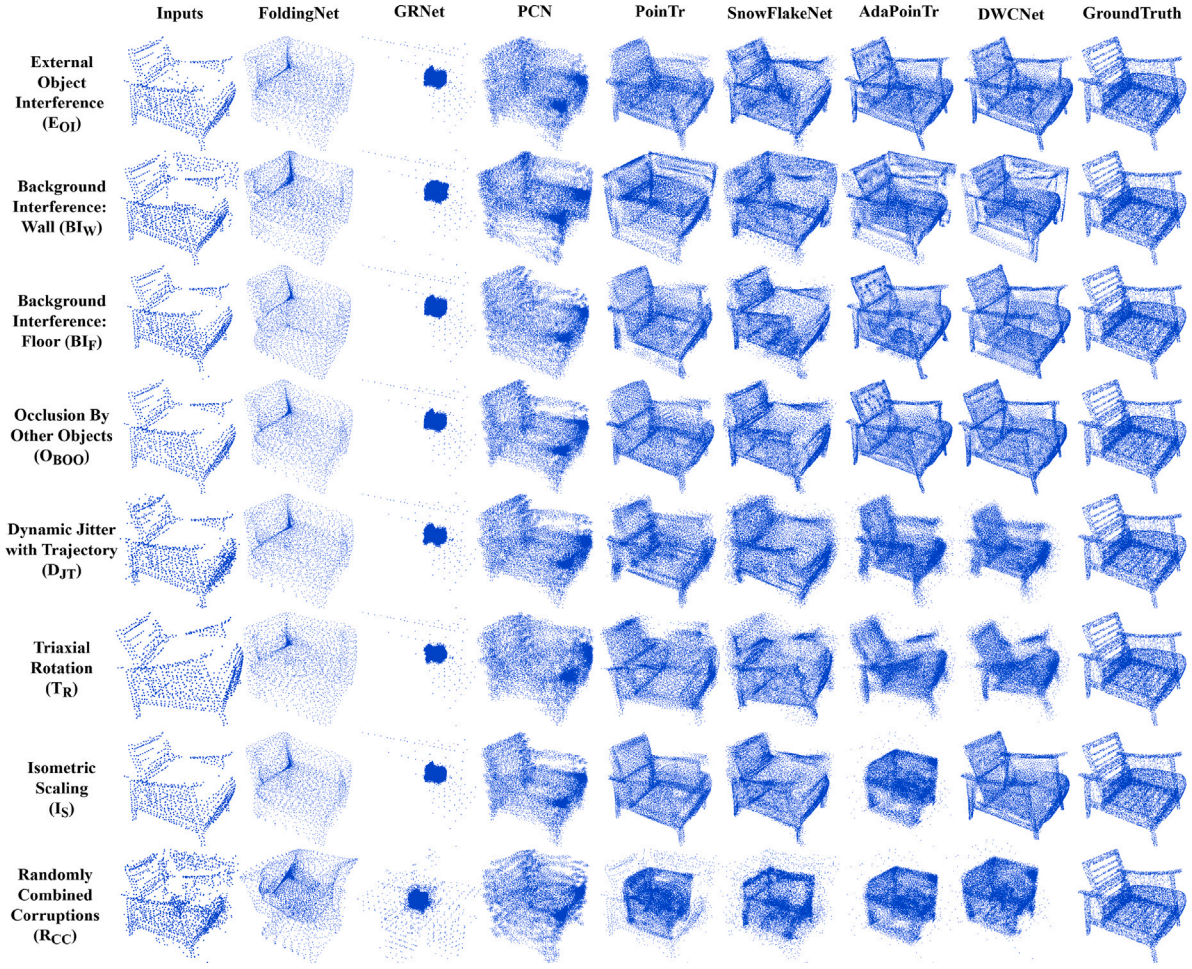
Fig. 6. Example of results on CPCCD dataset before-finetuning: chair.

Table 3

Completion results on CPCCD datasets. The result shows CD_{L1} results of the completion. The lower the CD_{L1} the better the result. The superscripts 'PB', 'GB', 'CB' and 'TB' stand for Point-Based, Graph-Based, Convolution-Based and Transformer-Based respectively.

Network (CDL1↓)	Clean	E_{OI}	BI_W	BI_F	O_{BOO}	D_{JT}	T_R	I_S	R_{CC}
FoldingNet[GB]	26.493	33.135	33.135	42.259	26.469	27.007	29.417	44.834	51.694
PCN[PB]	33.148	40.046	50.847	38.985	33.152	34.347	34.8	45.107	58.376
GRNet[TB]	24.686	25.794	48.249	34.961	25.248	24.731	28.295	44.795	63.516
PoinTr[TB]	8.388	**13.516**	25.481	**14.836**	8.483	**10.837**	**14.699**	24.302	35.63
SnowFlakeNet[TB]	8.18	15.181	26.731	18.728	8.256	10.85	14.854	**21.843**	**35.424**
AdaPoinTr[TB]	6.516	14.764	**24.994**	16.167	6.576	11.68	15.916	23.016	36.682
DWCNET[TB]	**6.493**	14.819	25.648	16.036	**6.531**	11.919	15.915	273.589	57.919

Table 4

Completion results on CPCCD datasets. The result shows F_{Score} results of the completion. The higher the F_{Score} the better the result. The superscripts 'PB', 'GB', 'CB' and 'TB' stand for Point-Based, Graph-Based, Convolution-Based and Transformer-Based respectively.

Network ($Fscore$ ↑)	Clean	E_{OI}	BI_W	BI_F	O_{BOO}	D_{JT}	T_R	I_S	R_{CC}
FoldingNet[GB]	0.213	0.109	0.109	0.101	0.214	0.176	0.131	0.083	0.052
PCN[PB]	0.244	0.136	0.115	0.178	0.242	0.200	0.184	0.133	0.072
GRNet[TB]	0.273	0.267	0.150	0.183	0.260	0.261	0.174	0.123	0.083
PoinTr[TB]	0.754	0.592	0.477	0.611	0.747	**0.634**	**0.471**	**0.398**	**0.280**
SnowFlakeNet[TB]	0.741	0.573	0.485	0.553	0.738	0.623	0.438	0.375	0.251
AdaPoinTr[TB]	0.845	**0.628**	**0.545**	0.654	0.842	0.620	0.438	**0.398**	0.253
DWCNET[TB]	**0.847**	0.625	0.532	**0.666**	**0.845**	0.609	0.439	0.385	0.254

showed relatively better qualitative results despite higher Chamfer Distance losses.

From the qualitative evaluation shown in Figs. 6 and 7, we see that GRNet has the worst results (see Figure D.31). We hypothesize that this is due to the gridding loss function used to train GRNet. Further

tests and analysis is needed to pinpoint how exactly the level and type of corruption relates with different training loss functions. We found that point-based networks, FoldingNet and PCN were able to adjust to the isometric scaling (I_S) and triaxial rotation (T_R), but failed at capturing the details, whereas transformer based networks were good
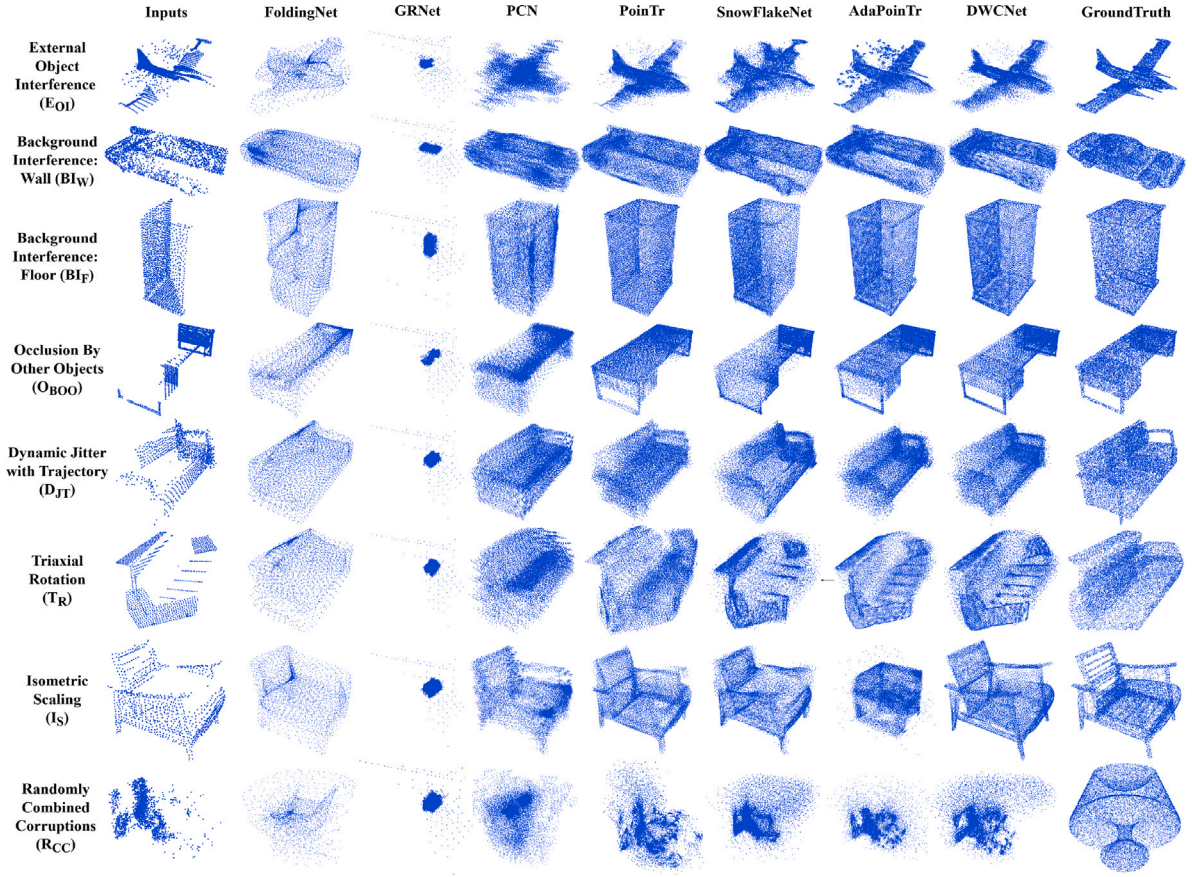
Fig. 7. Example of results on CPCCD dataset before-finetuning: multiple categories.

Table 5

Completion results on CPCCD datasets. The result shows *fidelity* results of the completion. The lower the *fidelity* the better the result. The superscripts 'PB', 'GB', 'CB' and 'TB' stand for Point-Based, Graph-Based, Convolution-Based and Transformer-Based respectively.

Network (*fidelity* ↓)	Clean	E_{OI}	BI_W	BI_F	O_{BOO}	D_{JT}	T_R	I_S	R_{CC}
FoldingNet[GB]	6.525	6.548	6.548	11.509	6.507	5.794	6.469	7.277	9.974
PCN[PB]	10.373	10.634	17.271	12.568	10.365	9.916	9.977	11.226	17.032
GRNet[TB]	4.945	5.252	15.174	8.357	5.074	4.785	5.216	10.233	22.962
PoinTr[TB]	0.179	**0.689**	**4.75**	**1.392**	0.183	**0.326**	**0.474**	1.868	6.419
SnowFlakeNet[TB]	0.176	1.241	5.77	2.619	0.181	0.353	0.535	1.715	6.687
AdaPoinTr[TB]	**0.093**	1.096	5.215	2.222	**0.096**	0.399	0.51	**1.344**	**5.859**
DWCNET[TB]	0.098	1.082	5.226	2.28	0.099	0.409	0.509	31 652.809	485.849

Table 6

Completion results on CPCCD datasets after fine-tuning. The result shows CD_{L1} results of the completion. The lower the CD_{L1} the better the result. The superscripts 'PB', 'GB', 'CB' and 'TB' stand for Point-Based, Graph-Based, Convolution-Based and Transformer-Based respectively.

Network (CD_{L1} ↓)	Clean	E_{OI}	BI_W	BI_F	O_{BOO}	D_{JT}	T_R	I_S	R_{CC}
FoldingNet[GB]	16.228	16.912	16.828	16.672	16.304	16.359	16.288	17.461	19.584
PCN[PB]	31.932	32.683	34.090	32.872	31.973	32.044	32.098	33.225	36.114
GRNet[TB]	32.957	32.877	39.338	36.745	33.311	32.556	33.458	34.464	40.478
PoinTr[TB]	8.760	9.002	14.181	11.192	8.857	9.024	9.766	11.217	17.692
SnowFlakeNet[TB]	10.115	10.658	14.291	12.408	10.194	10.628	12.129	13.570	18.508
AdaPoinTr[TB]	8.150	8.309	8.724	8.568	8.211	8.584	8.918	9.196	10.317
DWCNet[TB] (Ours)	**7.727**	**7.856**	**8.130**	**8.052**	**7.793**	**8.059**	**8.435**	**8.709**	**9.754**

at preserving detail but failed to adjust scale as required. DWCNET outperformed other models on both the clean and O_{OBOO} corruption categories, setting a new benchmark as the state-of-the-art on the PCN dataset.

6.2. Robustness after finetuning

The CD_{L1}, *Fscore* and *fidelity* results after finetuning are shown in Tables 6–8 respectively. The CD_{L2} results can be found in Table A.16.

After fine-tuning on CPCCD, all networks adapted to the corruptions (Table 6). Transformer-based networks outperformed point-based and convolution-based networks, adapting more quickly to the corruptions. Interestingly, after fine-tuning on randomly combined corruptions (R_{CC}), all algorithms exhibited similar performance across the CPCCD categories. Our method DWCNet with the noise management module performs really well on the majority of the corruptions. While the networks learned to tolerate the corruptions, they were unable to effectively remove them as can be seen in Figs. 8 and 9.

Table 7
Completion results on CPCCD datasets after fine-tuning. The result shows $Fscore$ results of the completion. The higher the $Fscore$ the better the result. The superscripts 'PB', 'GB', 'CB' and 'TB' stand for Point-Based, Graph-Based, Convolution-Based and Transformer-Based respectively.

Network ($fscore$ ↑)	Clean	E_{OI}	BI_W	BI_F	O_{BOO}	D_{JT}	T_R	I_S	R_{CC}
FoldingNet[GB]	0.279	0.256	0.273	0.268	0.276	0.276	0.276	0.257	0.219
PCN[PB]	0.182	0.170	0.174	0.169	0.182	0.178	0.175	0.163	0.142
GRNet[TB]	0.179	0.178	0.123	0.132	0.171	0.177	0.165	0.153	0.113
PoinTr[TB]	0.712	0.708	0.611	0.644	0.706	0.705	0.660	0.649	0.525
SnowFlakeNet[TB]	0.659	0.645	0.568	0.593	0.655	0.634	0.552	0.536	0.441
AdaPoinTr[TB]	0.756	0.750	0.733	0.738	0.751	0.734	0.710	0.694	0.646
DWCNet[TB] (Ours)	**0.777**	**0.774**	**0.762**	**0.765**	**0.774**	**0.760**	**0.740**	**0.724**	**0.676**

Table 8
Completion results on CPCCD datasets after fine-tuning. The result shows $fidelity$ results of the completion. The lower the $fidelity$ the better the result. The superscripts 'PB', 'GB', 'CB' and 'TB' stand for Point-Based, Graph-Based, Convolution-Based and Transformer-Based respectively.

Network ($fidelity$ ↓)	Clean	E_{OI}	BI_W	BI_F	O_{BOO}	D_{JT}	T_R	I_S	R_{CC}
FoldingNet[GB]	0.425	0.524	0.531	0.463	0.428	0.457	0.430	0.537	0.927
PCN[PB]	7.728	7.766	8.239	7.948	7.734	7.778	7.608	7.907	8.395
GRNet[TB]	9.208	9.078	10.212	9.562	9.310	8.951	9.071	8.936	10.128
PoinTr[TB]	0.190	0.237	1.549	0.547	0.193	0.214	0.235	0.799	2.924
SnowFlakeNet[TB]	0.308	0.393	1.130	0.604	0.308	0.331	0.401	0.804	1.971
AdaPoinTr[TB]	0.137	0.149	0.175	0.164	0.137	0.164	0.163	0.183	0.260
DWCNet[TB] (Ours)	**0.127**	**0.135**	**0.144**	**0.148**	**0.128**	**0.142**	**0.147**	**0.165**	**0.238**
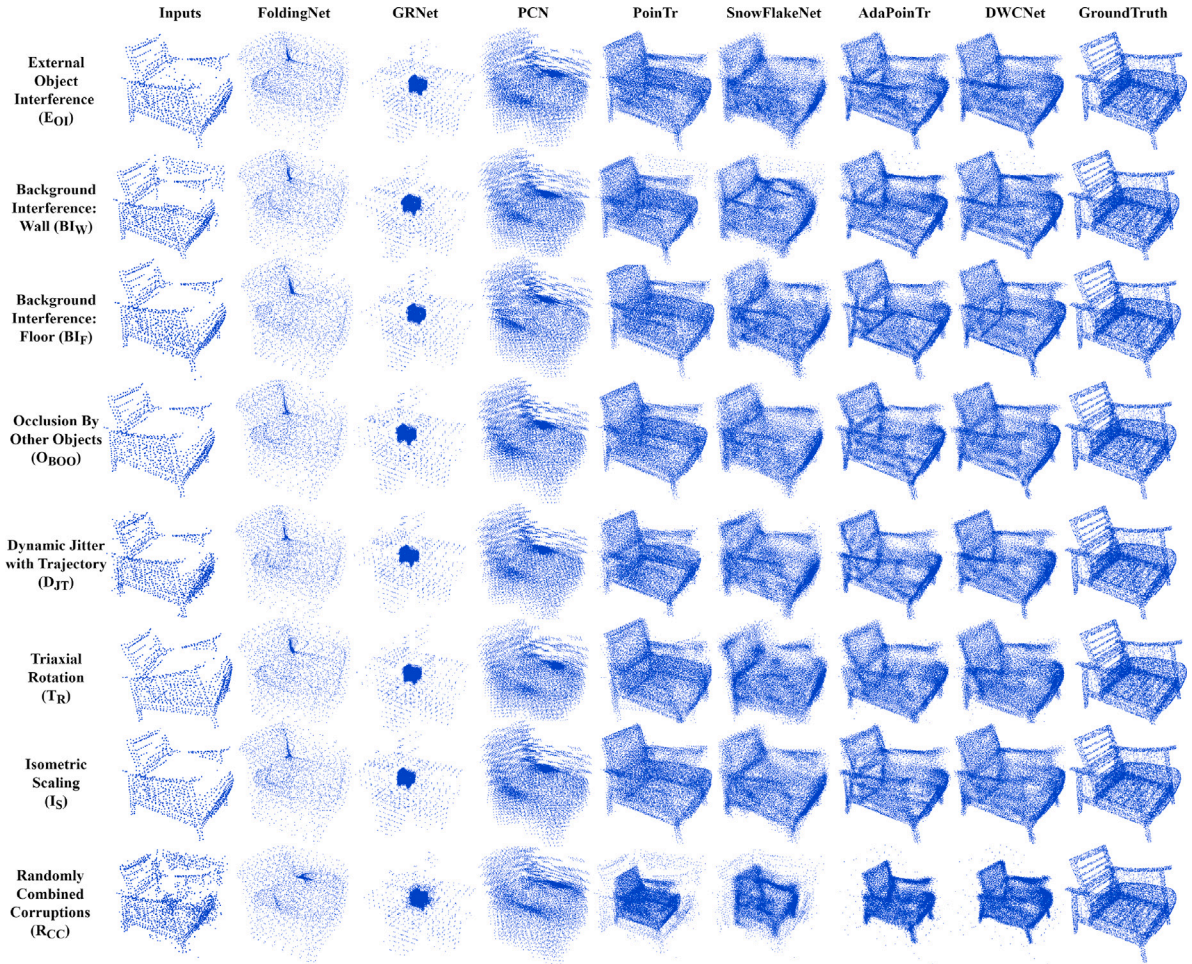
Fig. 8. Example of results on CPCCD dataset after-finetuning: chair.

6.3. Evaluation on ScanObjectNN

Our goal is to enhance the robustness of SOTA completion algorithms to better handle noise and corruption in real-world point clouds.

To evaluate this, we tested the performance on the real-world dataset ScanObjectNN [21].

We conducted experiments on 20 chairs, 20 tables, 20 desks, and 20 sofas from the ScanObjectNN dataset. Since real-world scans lack
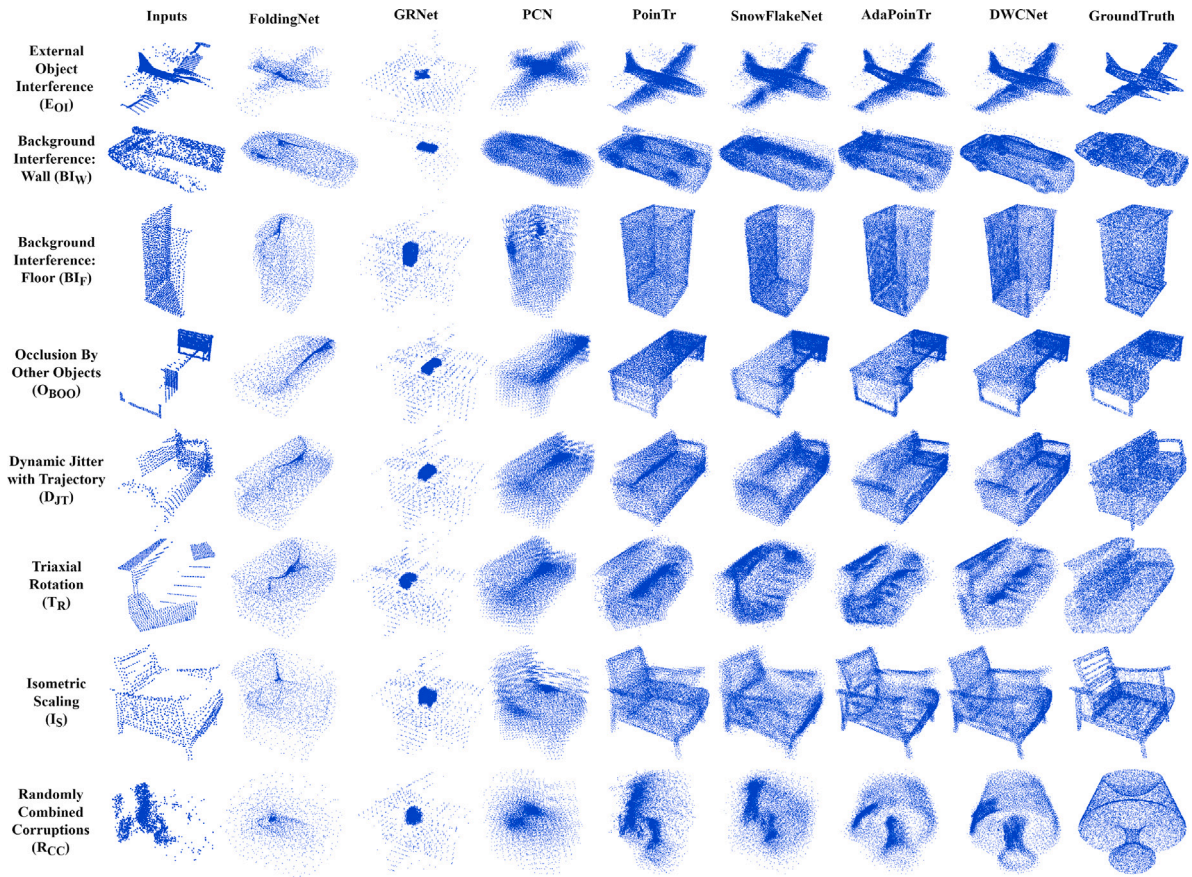
Fig. 9. Example of results on CPCCD dataset after-finetuning: multiple categories.
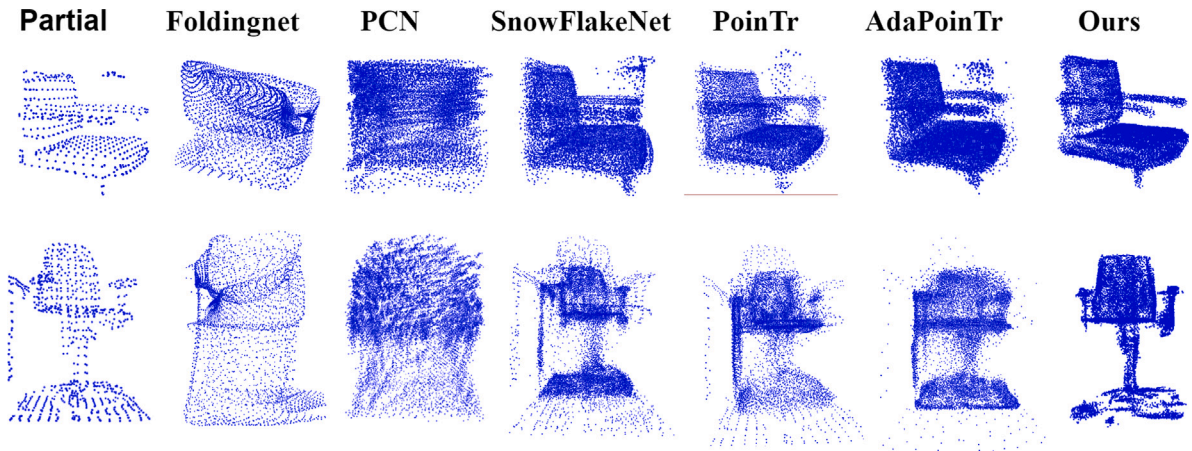

Fig. 10. Some examples of results on ScanObjectNN dataset.

Table 9
Ablation studies without(w/o) and with NMM.

Dataset(CD_{L1} ↓)	DWCNET w/o NMM	DWCNET
PCN	8.351	8.122
Rcc	13.403	10.263

Table 10
Ablation studies on the baseline model.

Path	CD_{L1} ↓	$Fscore$ ↑	$Fidelity$ ↓
No NMM	13.403	0.56	
Clean path only	10.255	0.646	0.259
Noisy path only	28.777	0.398	7.579
DWCNet	10.263	0.646	0.268

ground truth data, we present qualitative results in Fig. 10. The first row shows completion results for a real chair scan with external object interference (E_{OI}) and occlusion by other objects (O_{BOO}). Most models mistakenly treat the external corruption as part of the object, leading to noisy results and failing to complete the chair's legs. In

the second row, we show results for a highly corrupted chair scan with background interference from walls and floors (BI_W and BI_F) and isometric scale (I_S) corruption. Similar to the results in Fig. 6,

Table 11

Ablation Studies on the effect of the temperature scaling parameter t at 100 epochs.

t (100 epochs)	0.0000005	0.05	0.1	0.25	0.35	0.5	0.75	0.95	1.0	10	100	1000	10 000
Rcc (CD) ↓	31.608	11.773	11.503	11.610	11.502	11.432	11.670	11.592	11.438	11.279	11.2	11.528	11.712
Rcc (Fscore) ↑	0.334	0.570	0.586	0.584	0.586	0.582	0.581	0.583	0.592	0.594	0.598	0.584	0.573

PCN and FoldingNet handled scale corruption better than Transformer-based models (SnowFlakeNet, PoinTr, and AdaPoinTr). While PCN and FoldingNet improved in capturing global details after fine-tuning, they struggled with local details. Transformer-based models captured better local details but could not adapt to scale corruption and failed to ignore dense external object interference (E_{OI}).

7. Ablation study

In this section, we perform an ablation study on the baseline model, components, and loss functions of NMM. First, we present the results of the ablation study in Table 9, comparing DWCNet with and without the Noise Management Module (NMM). On the PCN dataset, DWCNet without NMM (equivalent to AdapoinTr) achieves a Chamfer Distance of 8.351, while integrating NMM reduces it to 8.122. Though the improvement is modest, it clearly demonstrates the added value of NMM. The impact is even more pronounced on the CPCCD dataset, where the Chamfer Distance drops significantly from 13.403 to 10.263 after 200 epochs of training with NMM. These results highlight NMM's effectiveness in handling data corruptions and improving point cloud completion, reinforcing its essential role in enhancing robustness and accuracy in real-world applications.

7.1. Path modeling

In this section, we discuss the effect of having only a clean path, only noisy path or both paths in the architecture. For the noisy path experiment, we had to calculate the clean features, f_{clean} that are normally produced by the clean path, by subtracting the noisy features f_{noisy} from the input features f_i produced by the geometric-aware transformer instead.

$$f_{\text{clean}} = f_{\text{i}} - f_{\text{noisy}} \tag{11}$$

All experiments were conducted under the same conditions, with a temperature scaling value of $t = 0.1$ and a training duration for 200 epochs. We observed that the clean path contributes the most to the training, and the worst results were observed from the noisy path. The difference between the clean path only and the using both paths was not too significant, so we performed the ablation study below on the architecture components and contrastive loss for further investigation into the effectiveness of each path (see Table 10).

7.2. Architecture components

To evaluate the contributions of attention mechanisms and multiscale convolutions in our model, we conducted a series of ablation experiments. In the "No Attention" configuration, we replaced the transformer module in the clean path with a simple MLP layer and trained the model for 100 epochs. In the "Single-Scale Convolution (SSC)" experiment, we substituted the multiscale convolution (MSC) in the noisy path with a standard single-scale convolution, also training for 100 epochs. Finally, in the "No Attention + SSC" setup, we combined both modifications—using the MLP in the clean path and the SSC in the noisy path. All experiments were conducted under the same conditions, with a temperature scaling value of $t = 0.1$ and a training duration of 100 epochs (see Table 12).

We found that while having no attention affected NMM's performance negatively, the use of Single Scale convolution or Multiscale convolution did not have any significant effect. This finding supports our observation that the clean path contributes the most in early training.

Table 12

Ablation studies on architecture components. SSC refers to Single Scale Convolution and MSC stands for Multi Scale Convolution.

Path	CD_{L1} ↓	$Fscore$ ↑
No attention	20.846	0.357
SSC	11.503	0.589
No attention and SSC	11.47	0.585
DWCNet	11.503	0.586

Table 13

Ablation studies on the effect of the temperature scaling parameter t at 350 epochs.

t (350 epochs)	0.1	0.5	1.0	100	Clean loss
Rcc (CD↓)	10.328	9.957	9.754	9.752	9.989
Rcc (Fscore↑)	0.642	0.663	0.676	0.676	0.661

7.3. Contrastive loss variants

From the ablation studies, we observed that the *clean path*, which contributes to the *Positive Loss*, plays a crucial role in completing highly corrupted partial point clouds. This effectiveness is likely due to the availability of clean partial inputs that provide strong supervision. The Positive Loss encourages alignment between clean features and their corresponding ground truth, reinforcing meaningful representations.

To complement this, we introduced the *Negative Loss* to penalize alignment between mismatched (clean-noisy) pairs and promote discriminative feature learning. Although the Positive Loss proved more impactful, removing the Negative Loss risks *representation collapse*, where features converge to trivial, non-informative embeddings due to the absence of contrastive pressure [56].

To better balance these losses, we investigated the effect of the *temperature scaling parameter t*, which modulates the sharpness of the Negative Loss (see Table 11). At extremely low values (e.g., $t = 0.0000005$), the Negative Loss became excessively large, leading to poor completion performance. While differences between $t = 0.05$ and $t = 1.0$ were subtle over 100 epochs, a trend toward improved performance with higher t values emerged. We extended training to 350 epochs and evaluated $t = 0.1$, $t = 0.5$, $t = 1.0$ and $t = 100.0$ (see Table 13). We found that $t = 1.0$ and $t = 100.0$ yielded the most favorable results on the Rcc category, with chamfer distances of 9.754 and 9.752, respectively. While $t = 100.0$ offered a marginal advantage on Rcc, the model trained at $t = 1.0$ outperformed across all other CPCCD categories and was therefore selected as the optimal configuration as shown in Table 14.

8. Conclusion and future work

In this paper, we introduce the Corrupted Point Cloud Completion Dataset (CPCCD) – the first dataset specifically curated to evaluate the robustness of point cloud completion algorithms under noise-induced corruption. Through comprehensive experiments, we demonstrate that fine-tuning existing completion models on CPCCD substantially improves their resilience to noise commonly encountered in real-world scenarios. To further advance robustness, we propose a novel Noise Management Module (NMM), which facilitates joint denoising and completion of severely corrupted partial point clouds within our robust completion framework, DWCNet. While our approach yields promising results, we recognize the need for more accurate simulation of real-world noise characteristics and structural degradation. As part of

Table 14
Completion results for DWCNet ($t = 1$) and DWCNet ($t = 100$) at 350 epochs.

Network (CDL1↓)	Clean	E_{OI}	BI_W	BI_F	O_{BOO}	D_{JT}	T_R	I_S	R_{CC}
DWCNet ($t = 100$)^{TB}	7.756	7.921	8.193	8.15	7.839	8.126	8.499	8.791	**9.752**
DWCNet ($t = 1$)^{TB}	**7.727**	**7.856**	**8.130**	**8.052**	**7.793**	**8.059**	**8.435**	**8.709**	9.754

future work, we plan to expand CPCCD by incorporating additional object categories from established clean completion benchmarks such as ShapeNet34 and ShapeNet55. Furthermore, we aim to enhance the noise-removal and structure-preservation capabilities of completion algorithms. By addressing these challenges, we seek to underscore the practical significance of completing highly corrupted partial point clouds — an area that remains underexplored in the current literature.

CRediT authorship contribution statement

Keneni W. Tesema: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Lyndon Hill:** Writing – review & editing, Supervision, Resources, Funding acquisition, Conceptualization. **Mark W. Jones:** Writing – review & editing, Supervision, Software, Funding acquisition, Conceptualization. **Gary K.L. Tam:** Writing – review & editing, Supervision, Software, Resources, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.cag.2025.104401.

Data availability

Data will be made available on request.

References

[1] Guo Yulan, Wang Hanyun, Hu Qingyong, Liu Hao, Liu Li, Bennamoun Mohammed. Deep learning for 3D point clouds: A survey. PAMI 2020;43(12):4338–64.

[2] Tesema Keneni W, Hill Lyndon, Jones Mark W, Ahmad Muneeb I, Tam Gary KL. Point cloud completion: A survey. IEEE Trans Vis Comput Graphics 2023;30(10):6880–99. http://dx.doi.org/10.1109/TVCG.2023.3344935.

[3] Qi Charles R, Litany Or, He Kaiming, Guibas Leonidas J. Deep hough voting for 3D object detection in point clouds. In: ICCV. 2019, p. 9277–86.

[4] Zheng Yuchao, Li Yujie, Yang Shuo, Lu Huimin. Global-PBNet: A novel point cloud registration for autonomous driving. T-its 2022;23(11):22312–9.

[5] Varley Jacob, DeChant Chad, Richardson Adam, Ruales Joaquín, Allen Peter. Shape completion enabled robotic grasping. In: IROS. 2017, p. 2442–7.

[6] Jaklič Aleš, Erič Miran, Mihajlović Igor, Stopinšek Žiga, Solina Franc. Volumetric models from 3D point clouds: The case study of sarcophagi cargo from a 2nd/3rd century AD roman shipwreck near sutivan on island brač, Croatia. J Archaeol Sci 2015;62:143–52.

[7] Cai Pingping, Scott Deja, Li Xiaoguang, Wang Song. Orthogonal dictionary guided shape completion network for point cloud. In: AAAI. vol. 38, 2024, p. 864–72.

[8] Yu Xumin, Rao Yongming, Wang Ziyi, Lu Jiwen, Zhou Jie. AdaPoinTr: Diverse point cloud completion with adaptive geometry-aware transformers. 2023, arXiv:2301.04545.

[9] Yuan Wentao, Khot Tejas, Held David, Mertz Christoph, Hebert Martial. PCN: Point completion network. In: 3DV. 2018, p. 728–37.

[10] Pan Liang, Chen Xinyi, Cai Zhongang, Zhang Junzhe, Zhao Haiyu, Yi Shuai, Liu Ziwei. Variational relational point completion network. In: CVPR. 2021, p. 8524–33.

[11] Geiger Andreas, Lenz Philip, Stiller Christoph, Urtasun Raquel. Vision meets robotics: The KITTI dataset. IJRR 2013.

[12] Chang Angel X, Funkhouser Thomas, Guibas Leonidas, Hanrahan Pat, Huang Qixing, Li Zimo, Savarese Silvio, Savva Manolis, Song Shuran, Su Hao, et al. Shapenet: An information-rich 3D model repository. 2015, arXiv:1512.03012.

[13] Tchapmi Lyne P, Kosaraju Vineet, Rezatofighi Hamid, Reid Ian, Savarese Silvio. TopNet: Structural point cloud decoder. In: CVPR. 2019, p. 383–92.

[14] Zhou Yuhan, Tu Fengjiao, Sha Kewei, Ding Junhua, Chen Haihua. A survey on data quality dimensions and tools for machine learning. 2024, arXiv preprint arXiv:2406.19614.

[15] Ren Jiawei, Pan Liang, Liu Ziwei. Benchmarking and analyzing point cloud classification under corruptions. In: Proceedings of the 39th international conference on machine learning. vol. 162, 2022, p. 18559–75.

[16] Arora Himanshu, Mishra Saurabh, Peng Shichong, Li Ke, Mahdavi-Amiri Ali. Multimodal shape completion via imle. 2021, arXiv:2106.16237.

[17] Kong Lingdong, Liu Youquan, Li Xin, Chen Runnan, Zhang Wenwei, Ren Jiawei, Pan Liang, Chen Kai, Liu Ziwei. Robo3D: Towards robust and reliable 3D perception against corruptions. In: Proceedings of the IEEE/CVF international conference on computer vision. 2023, p. 19994–20006.

[18] Dai Angela, Chang Angel X, Savva Manolis, Halber Maciej, Funkhouser Thomas, Nießner Matthias. Scannet: Richly-annotated 3D reconstructions of indoor scenes. In: CVPR. 2017, p. 5828–39.

[19] Cui Ruikai, Qiu Shi, Anwar Saeed, Liu Jiawei, Xing Chaoyue, Zhang Jing, Barnes Nick. P2C: Self-supervised point cloud completion from single partial clouds. 2023, arXiv:2307.14726.

[20] Wang Jun, Cui Ying, Guo Dongyan, Li Junxia, Liu Qingshan, Shen Chunhua. PointAttN: You only need attention for point cloud completion. 2022, arXiv:2203.08485.

[21] Uy Mikaela Angelina, Pham Quang-Hieu, Hua Binh-Son, Nguyen Duc Thanh, Yeung Sai-Kit. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: International conference on computer vision. ICCV, 2019.

[22] Wang Qian, Tan Yi, Mei Zhongya. Computational methods of acquisition and processing of 3D point cloud data for construction applications. Arch Comput Methods Eng 2019;27:479–99.

[23] Li Leihui, Wang Riwei, Zhang Xuping. A tutorial review on point cloud registrations: Principle, classification, comparison, and technology challenges. Math Probl Eng 2021;2021.

[24] Ren Jiawei, Pan Liang, Liu Ziwei. Benchmarking and analyzing point cloud classification under corruptions. Int Conf Mach Learn (ICML) 2022.

[25] Dong Yinpeng, Kang Caixin, Zhang Jinlai, Zhu Zijian, Wang Yikai, Yang Xiao, Su Hang, Wei Xingxing, Zhu Jun. Benchmarking robustness of 3D object detection to common corruptions. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023, p. 1022–32.

[26] Gurumurthy Swaminathan, Agrawal Shubham. High fidelity semantic shape completion for point clouds using latent optimization. In: WACV. 2019, p. 1099–108.

[27] Li Jianxin, Si Guannan, Liang Xinyu, An Zhaoliang, Tian Pengxin, Zhou Fengyu. Partition-based point cloud completion network with density refinement. Entropy 2023;25(7):1018.

[28] Ma Changfeng, Yang Yang, Guo Jie, Wang Chongjun, Guo Yanwen. Completing partial point clouds with outliers by collaborative completion and segmentation. 2022, arXiv:2203.09772.

[29] Li Ren-Wu, Wang Bo, Li Chun-Peng, Zhang Ling-Xiao, Gao Lin. High-fidelity point cloud completion with low-resolution recovery and noise-aware upsampling. 2021, arXiv:2112.11271.

[30] Zhou Lang, Sun Guoxing, Li Yong, Li Weiqing, Su Zhiyong. Point cloud denoising review: From classical to deep learning-based approaches. Graph Model 2022;121:101140.

[31] Luo Shitong, Hu Wei. Score-based point cloud denoising. In: Proceedings of the IEEE/CVF international conference on computer vision. 2021, p. 4583–92.

[32] Wei Zeyong, Chen Honghua, Nan Liangliang, Wang Jun, Qin Jing, Wei Mingqiang. PathNet: Path-selective point cloud denoising. IEEE Trans Pattern Anal Mach Intell 2024;46(6):4426–42.

[33] Hou Guangzhe, Qin Guihe, Sun Minghui, Liang Yanhua, Yan Jie, Zhang Zhong-han. NoiseTrans: Point cloud denoising with transformers. 2023, arXiv preprint arXiv:2304.11812.

[34] Hermosilla Pedro, Ritschel Tobias, Ropinski Timo. Total denoising: Unsuper-vised learning of 3D point cloud cleaning. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019, p. 52–60.

[35] Xie Saining, Gu Jiatao, Guo Demi, Qi Charles R, Guibas Leonidas, Litany Or. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In: Computer vision–ECCV 2020: 16th European conference, glasgow, UK, August 23–28, 2020, proceedings, part III 16. Springer; 2020, p. 574–91.

[36] Pan Zhiyi, Liu Guoqing, Gao Wei, Li Thomas H. Epcontrast: effective point-level contrastive learning for large-scale point cloud understanding. In: 2024 IEEE international conference on multimedia and expo. ICME, IEEE; 2024, p. 1–6.

[37] Du Anan, Zhou Tianfei, Pang Shuchao, Wu Qiang, Zhang Jian. Pcl: Point contrast and labeling for weakly supervised point cloud semantic segmentation. IEEE Trans Multimed 2024.

[38] Wang Zhoutao, Xie Qian, Wei Mingqiang, Long Kun, Wang Jun. Multi-feature fusion VoteNet for 3D object detection. ACM Trans Multimed Comput Commun Appl (TOMM) 2022;18(1):1–17.

[39] Avetisyan Armen, Dahnert Manuel, Dai Angela, Savva Manolis, Chang Angel X, Nießner Matthias. Scan2cad: Learning cad model alignment in rgb-d scans. In: CVPR. 2019, p. 2614–23.

[40] Öcal Başak Melis, Tatarchenko Maxim, Karaoglu Sezer, Gevers Theo. RealDiff: Real-world 3D shape completion using self-supervised diffusion models. 2024, arXiv preprint arXiv:2409.10180.

[41] Phan Anh Viet, Le Nguyen Minh, Nguyen Yen Lam Hoang, Bui Lam Thu. DGCNN: A convolutional neural network over large-scale labeled graphs. Neural Netw 2018;108:533–43.

[42] Hu Yuchen, Chen Chen, Zou Heqing, Zhong Xionghu, Chng Eng Siong. Unifying speech enhancement and separation with gradient modulation for end-to-end noise-robust speech separation. In: ICASSP 2023-2023 IEEE international con-ference on acoustics, speech and signal processing. ICASSP, IEEE; 2023, p. 1–5.

[43] Sivaraman Aswin, Kim Minje. Efficient personalized speech enhancement through self-supervised learning. IEEE J Sel Top Signal Process 2022;16(6):1342–56.

[44] Kharitonov Eugene, Rivière Morgane, Synnaeve Gabriel, Wolf Lior, Mazaré Pierre-Emmanuel, Douze Matthijs, Dupoux Emmanuel. Data augmenting contrastive learning of speech representations in the time domain. In: 2021 IEEE spoken language technology workshop. SLT, IEEE; 2021, p. 215–22.

[45] Chen Ting, Luo Calvin, Li Lala. Intriguing properties of contrastive losses. Adv Neural Inf Process Syst 2021;34:11834–45.

[46] Cordonnier Jean-Baptiste, Loukas Andreas, Jaggi Martin. Multi-head attention: Collaborate instead of concatenate. 2020, arXiv preprint arXiv:2006.16362.

[47] Zhang Oliver, Wu Mike, Bayrooti Jasmine, Goodman Noah. Temperature as uncertainty in contrastive learning. 2021, arXiv preprint arXiv:2110.04403.

[48] Yang Yaoqing, Feng Chen, Shen Yiru, Tian Dong. Foldingnet: Point cloud auto-encoder via deep grid deformation. In: CVPR. 2018, p. 206–15.

[49] Xie Haozhe, Yao Hongxun, Zhou Shangchen, Mao Jiageng, Zhang Shengping, Sun Wenxiu. Grnet: Gridding residual network for dense point cloud completion. In: EECV. 2020, p. 365–81.

[50] Yu Xumin, Rao Yongming, Wang Ziyi, Liu Zuyan, Lu Jiwen, Zhou Jie. Pointr: Diverse point cloud completion with geometry-aware transformers. In: ICCV. 2021, p. 12498–507.

[51] Xiang Peng, Wen Xin, Liu Yu-Shen, Cao Yan-Pei, Wan Pengfei, Zheng Wen, Han Zhizhong. Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In: ICCV. 2021, p. 5499–509.

[52] Nguyen Trung, Pham Quang-Hieu, Le Tam, Pham Tung, Ho Nhat, Hua Binh-Son. Point-set distances for learning representations of 3D point clouds. In: ICCV. 2021, p. 10478–87.

[53] Wu Tong, Pan Liang, Zhang Junzhe, Wang Tai, Liu Ziwei, Lin Dahua. Density-aware chamfer distance as a comprehensive metric for point cloud completion. 2021, arXiv:2111.12702.

[54] Moravec Jaroslav. A comparative study: L1-norm vs. l2-norm; point-to-point vs. point-to-line metric; evolutionary computation vs. gradient search. Appl Artif Intell 2015;29(2):164–210.

[55] Sokolova Marina, Japkowicz Nathalie, Szpakowicz Stan. Beyond accuracy, F-score and ROC: A family of discriminant measures for performance evaluation. In: Adv. in artif. intell.. 2006, p. 1015–21.

[56] Jing Li, Vincent Pascal, LeCun Yann, Tian Yuandong. Understanding dimensional collapse in contrastive self-supervised learning. 2021, arXiv preprint arXiv:2110.09348.