ScalaDAT: Scalable Density-Aware Tail-Drop for Point Cloud Coding

Zhe Luo, Stuart Perry, Wenjing Jia

***Abstract*—Three-dimensional (3D) point clouds are becoming increasingly vital in applications such as autonomous driving, augmented reality, and immersive communication, demanding real-time processing and low latency. However, their large data volumes and bandwidth constraints hinder the deployment of high-quality services in resource-limited environments. Scalable coding, which allows for decoding at varying levels of detail, provides an alternative by allowing initial partial decoding with subsequent refinement. Although recent learning-based point cloud geometry coding methods have achieved notable success, their fixed latent representation does not support scalable decoding.**

To bridge this gap, we propose ScalaDAT, a novel density-aware tail-drop mechanism for scalable point cloud coding. By leveraging density information as a guidance signal, latent features and coordinates are decoded adaptively based on their significance, therefore achieving scalable decoding at multiple bitrates using one single model.

Experimental results on benchmark datasets show that the proposed ScalaDAT not only enables scalable coding but also achieves superior coding efficiency compared to state-of-the-art learning-based coding techniques, with over 28.6% BD-rate improvement for PSNR-D2 on SemanticKITTI and over 18.15% for ShapeNet.

***Index Terms*—Scalable coding, Point cloud compression, Tail-drop, Density-aware.**

I. Introduction

Point clouds, generated by 3D capturing technologies such as Light Detection and Ranging (LiDAR) scanners, provide detailed 3D representations of environments via millions of spatial points with attributes like color, intensity, and reflectivity. Unlike 2D images with regular pixel grids, point clouds are inherently irregular and unstructured, presenting significant challenges for storage, transmission, and processing [1]. A single LiDAR scan can generate hundreds of millions of points, resulting in substantial storage and bandwidth demands [2], [3]. Consequently, point cloud coding (PCC) is essential for practical applications, including autonomous driving, virtual and augmented reality (VR/AR), and 3D mapping [4], [5], [6]. However, the irregular structure of point clouds makes conventional 2D and video coding methods unsuitable, necessitating specialized PCC techniques to preserve geometric fidelity efficiently.

Traditional PCC methods, such as Geometry-based point cloud compression (G-PCC) and Video-based Point Cloud Compression (V-PCC) [7], [8], encode geometry hierarchically or project 3D data onto 2D planes to exploit video codecs. These methods have achieved widespread adoption [1]. However, they falter on large-scale datasets, such as SemanticKITTI [9], exhibiting low coding efficiency, degraded reconstruction quality, and high computational overhead. In contrast, learning-based PCC leverages deep learning to better capture spatial and structural complexities [10], [11], [12], drawing from 2D innovations like hyperprior variational models [13] to outperform traditional codecs at low bitrates [10]. Nevertheless, they typically produce a single, monolithic bitstream that requires full decoding for reconstruction, which is inadequate for time-sensitive applications such as autonomous driving. Thus, scalable coding represents a pivotal advancement for large-scale point cloud processing, blending practical benefits with theoretical novelty.

In 2D image and video domains, scalable coding is a well-established technique for efficient data delivery and real-time adaptability [14]. For images, the JPEG standard [15] uses multi-scan encoding for incremental detail enhancement, aiding previews on bandwidth-limited networks via initial low-quality representations refined scalably. Similarly, JPEG 2000 [16] leverages wavelet transforms to provide scalable resolution and quality, supporting scalable decoding that adapts flexibly to user needs or network conditions. In the video domain, the H.264 Scalable Video Coding (SVC) extension [17] adopts a layered architecture: a base layer provides foundational quality, with enhancement layers incrementally improving resolution, frame rate, or fidelity to enable adaptive streaming amid varying network conditions or device capabilities. Together, these standards exemplify scalable coding's advantages—reduced latency and optimized resource utilization—as foundations of modern multimedia systems.

As conceptually illustrated in Fig. 1, Scalable Point Cloud Compression (SPCC) enables partial decoding for incremental refinement, allowing scalable reconstructions via a Scalable Ratio (SR) α, which dynamically trades bitrate for geometric quality. It is crucial to formally distinguish the specific type of scalability addressed here. Traditional standard codecs, such as G-PCC [7], inherently possess spatial scalable capabilities (often referred to as scalable decoding) through their hierarchical octree Level-of-Detail (LoD) subdivisions. Similarly, foundational learning-based multiscale frameworks like SparsePCGC [18] achieve scalability across different spatial resolutions via sequential refinement stages.

However, achieving fine-grained, *channel-based* scalable geometry coding utilizing a single-training paradigm, where scalability is achieved by dynamically pruning latent feature channels rather than relying on discrete spatial octrees or

Z. Luo, W. Jia, and S. Perry are with the School of Electrical and Data Engineering, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: zhe.luo-1@student.uts.edu.au).

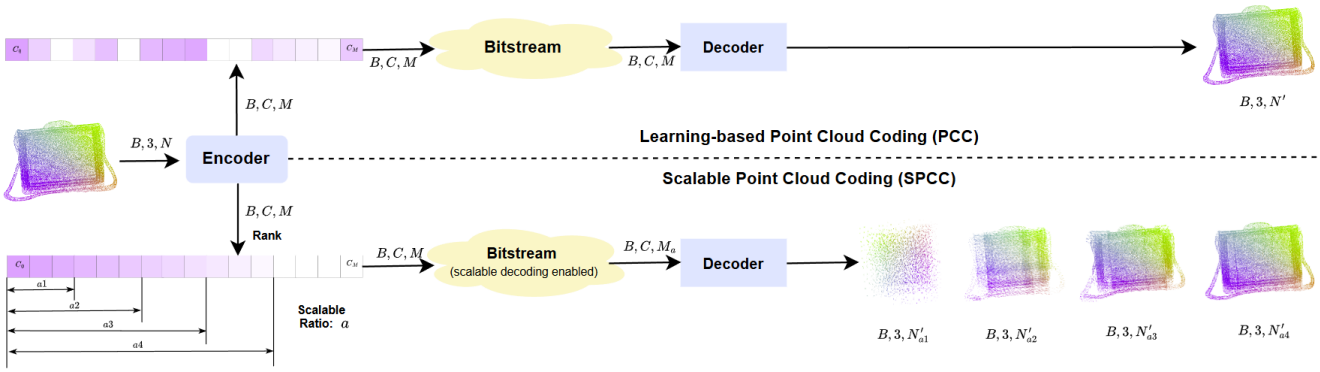


Fig. 1: Comparisons between Point Cloud Coding (PCC) and Scalable Point Cloud Coding (SPCC). SPCC generates a scalable bitstream post-encoding, enabling customized decoding for variable coding ratios a and reconstruction qualities.

multi-stage resolutions, remains an active area of research. Recent learning-based advancements focus primarily on attribute compression, as demonstrated by Rudolph *et al.* [19]. To bridge the gap in geometry coding, this work draws inspiration from established channel/layer scalable coding paradigms in 2D videos (e.g., H.264 SVC [17]), enabling efficient data delivery and dynamic bandwidth adaptability in constrained environments without claiming strict low-latency guarantees.

Inspired by prior work [20], [21], we introduce a **d**ensity-**a**ware **t**ail-drop scalable point cloud coding (ScalaDAT) method. Similar to Scalable JPEG [15] and H.264 Scalable Video Coding (SVC) [17], ScalaDAT enables incremental decoding by training the model to selectively discard less critical features, enabling scalable quality improvements at higher bitrates during decoding. Unlike 2D and video counterparts that rely on uniform grids or temporal continuity, our approach addresses the irregular, unstructured geometry of point clouds by explicitly leveraging density. High-density regions typically indicate intricate geometry (*e.g.*, object surfaces), whereas sparse areas usually correspond to simpler, less critical structures. By exploiting density variations, ScalaDAT prioritizes structurally significant details, improving rate-distortion efficiency through informed feature selection. In detail, our approach delivers the following key contributions:

- We propose ScalaDAT, a novel framework that achieves scalable point cloud coding with a single training stage.
- We develop a density-aware tail-drop approach that leverages point and structure density to prioritize structurally significant regions, enhancing efficiency by focusing on complex, high-density areas.
- We validate ScalaDAT on SemanticKITTI and ShapeNet, demonstrating superior BD-Rate performance compared to state-of-the-art methods.

The remainder of this paper is organized as follows: Section II provides a concise overview of existing point cloud coding and related image scalable coding strategies. Section III details the proposed ScalaDAT framework, including the Density-aware Tail-drop strategy. Experimental results with ablation studies are presented in Section IV, followed by a discussion and conclusion in Section V.

II. Related Work

A. Point Cloud Coding

Traditional point cloud coding techniques employ spatial data structures and quantization to reduce redundancy without neural networks. For example, MPEG's G-PCC [7] uses octree representations and triangular surface coding for geometric encoding in static scenes, while V-PCC [8] projects 3D data onto 2D planes to exploit video codecs for temporal redundancy in dynamic sequences. However, these methods incur high computational complexity, substantial storage needs, and limited scalability across datasets. To mitigate these limitations, learning-based methods leverage deep neural networks for compact, efficient representations, surpassing traditional approaches in scalability and performance.

Point-based methods process raw points directly, avoiding losses from intermediate transformations. Inspired by PointNet++ [22], Hao *et al.*'s KNN-based encoding [23] and D-PCC's density preservation [24] enhance reconstruction quality and adaptability across sparse or irregular distributions. Octree-based methods refine hierarchical tree coding with neural networks. OctSqueeze [25] introduces probabilistic occupancy modeling, followed by Tingyu *et al.*'s hierarchical latent variables [26]. Attention mechanisms in OctAttention [27] further optimize context modeling by exploiting large-scale octree-structured contexts, balancing efficiency and detail preservation. Sparse tensor-based methods represent another important direction. Wang *et al.* [18] proposed a multiscale point cloud geometry compression framework that encodes point clouds at multiple spatial resolutions using sparse tensors, achieving hierarchical representation through sequential refinement stages. This was further extended in SparsePCGC [12], which employs sparse convolutions for multiscale representation and achieves state-of-the-art compression performance. More recently, Wang *et al.* [28] introduced a versatile point cloud compressor using universal multiscale conditional coding, further advancing the sparse tensor paradigm. Voxel-based methods treat point clouds as occupancy grids , applying 3D CNNs. Quach *et al.*'s convolutional autoencoder [29], PCGC [11], and JPEG Pleno [10] pioneered this approach, while sparse convolutions [12] and block-based partitioning [30] address sparsity inefficiencies. These

learning-based strategies outperform traditional techniques, offering robust adaptability to diverse point cloud densities and structures with enhanced efficiency and reconstruction fidelity.

B. Scalable Image Coding

Scalable coding has been extensively explored in learning-based image coding, building on foundational works by Ballé *et al.* [31], [13] and Minnen *et al.* [32]. Early efforts partitioned latent representations into a base layer and enhancement layers, each decoded by separate networks to produce intermediate image versions [33]. Others employed Recurrent Neural Networks (RNNs) to scalably encode quantisation residuals [34], [35], [36], but these approaches suffer from high computational costs, memory usage, and low throughput [37].

Alternatively, Lu *et al.* [38] proposed nested quantisation, defining multiple levels with nested grids to scalably refine all latents from coarsest to finest. Building on this, Lee *et al.* [14] represented encoded features in ternary digits (trits) and transmitted them in decreasing significance order using rate-distortion priorities, sending critical information first. However, this results in degraded performance and degraded quality at low bitrates, often requiring a post-processing network for refinement. Similarly, Li *et al.* [39] replaced uniform quantisers with dead-zone quantisers to reduce redundant symbols, thereby improving the efficiency of scalable coding. Despite strong scalable coding performance in images, these methods [14], [39] rely on fixed-rate models, limiting their flexibility.

To overcome the limitations of RNN and nested quantisation, tail-drop techniques enable variable-rate coding, reducing computational complexity while improving coding efficiency. Koike *et al.* [20] showed that discarding the least important principal components yields variable rate dimensionality reduction with graceful degradation. Based on this, Hojjat *et al.* [21] introduced a double-tail-drop scalable training protocol that prioritises latent and hyper-latent channels by relevance [31], [13], enabling transmission in order of channel importance and eliminating the need for nested quantisation.

C. Scalable Point Cloud Coding

Although scalable coding is well established in 2D images, transferring such techniques to point cloud geometry is non-trivial, as the assumptions underlying image-domain tail-drop do not hold for point clouds. First, image codecs operate on a fixed, decoder-known regular grid, so double-tail-drop schemes such as ProgDTD [21] prune latent and hyper-latent channels while the spatial structure remains implicitly preserved. In point cloud coding, by contrast, the geometry constitutes the coding target itself; the coordinates and latent features thus form two coupled modalities whose channels must be dropped synchronously to preserve their correspondence. Second, the number of reconstructed points is not fixed by resolution but inferred by the decoder, so unstructured channel pruning readily introduces point-count deviation, spatial overlap, and noise. Third, the strongly non-uniform density of point clouds [24] renders a single global channel-importance ordering suboptimal, since it over-compresses detail-rich dense regions while over-allocating bits to sparse ones. As compressing an irregular, unevenly sampled point set is inherently harder than coding signals on regular grids [23], a direct transplantation of image-domain tail-drop is ineffective, motivating the density-aware and synchronized design of ScalaDAT.

Notable advancements include Huang *et al.*'s [40] octree-based subdivision, which estimates geometric centres of tree-front cells to support scalable coding. Recent work applies deep learning to do scalable point cloud coding. Rudolph *et al.* [19] proposed a progressive attribute coding method for point clouds that operates by encoding quantization residuals between successive quality layers. Specifically, their approach uses a learned lightweight affine transformation within the entropy bottleneck to progressively refine attribute (color) information, transmitting coarse-to-fine layers that can be independently decoded. While effective for attribute coding, their method does not address progressive geometry coding, which is the focus of this work. Similarly, Gokulnath *et al.* [41] projected the 3D model from multiple viewpoints to form a sequence of view-specific six-dimensional (RGB+XYZ) images on 2D grids, then used a symmetry-based convolutional neural pyramid to encode them scalably from coarse to fine, exploiting inter-projection redundancies for efficient transmission. Nonetheless, scalable geometry coding for point clouds remains underexplored, particularly on challenging benchmarks like SemanticKITTI [9] and ShapeNet [42]. The vast scale of LiDAR data and the geometric diversity of models pose significant challenges for PCC in both data volume and detail preservation. These challenges underscore the significance of scalable PCC for improving data efficiency in resource-intensive domains such as autonomous navigation and immersive media [19], [41].

To address this need, this work proposes ScalaDAT, which leverages density information to guide scalable coding and prioritise critical regions of the point cloud. Unlike the spatial-resolution-based scalability in SparsePCGC [12], which achieves progressive decoding by sequentially refining coarse-to-fine spatial representations across multiple model stages, ScalaDAT operates within a single resolution level and achieves progressiveness through density-guided channel pruning. This enables finer granularity of rate control and does not require the overhead of multi-stage encoding-decoding pipelines. Through tail-drop, it selectively preserves essential features, reducing decoder computational load while enabling efficient, controllable scalable decoding.

III. Methodology

A. Problem Definition

A point cloud can be represented as a matrix $\mathbf{X} \in \mathbb{R}^{3\times N}$, where each column $\mathbf{x}_i \in \mathbb{R}^3$ denotes the coordinates of the i-th point, for $i = 1, 2, \ldots, N$. Learning-based point cloud coding aims to learn an effective neural coding model $\mathcal{F}_\theta$ that maps the input to a compressed and subsequently reconstructed form:

$$\mathcal{F}_\theta : (\mathbf{X}) \mapsto \mathcal{B} \mapsto (\mathbf{X}'), \tag{1}$$

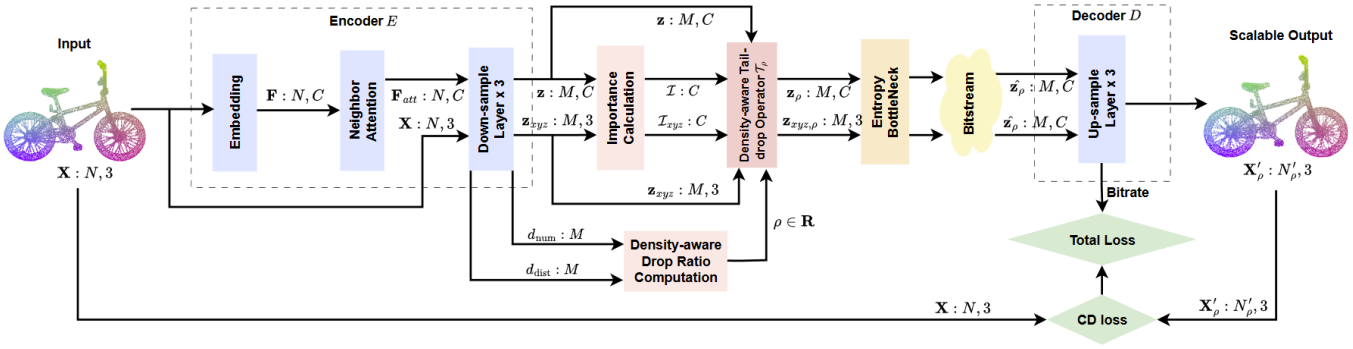


Fig. 2: The architecture of the proposed ScalaDAT. The scalable coding model of ScalaDAT leverages an end-to-end autoencoder which consists of an Encoder E, a Density-aware Tail-drop Operator, an Entropy Bottleneck, and a Decoder D. The total loss is defined as a trade-off among density loss, Chamfer Distance (CD) loss, and bitrate. See Sect. III-B for further details.

where $\mathbf{X}'$ is the reconstructed coordinates, and $\mathcal{B}$ represents the complete monolithic bitstream requiring full decoding for reconstruction in standard coding.

We consider two coding paradigms. Traditional (non-scalable) learning-based point cloud coding optimizes a fixed rate-distortion trade-off by minimizing the loss function:

$$\mathcal{L} = \mathcal{D}(\mathbf{X}, \mathbf{X}') + \lambda \mathcal{R}, \tag{2}$$

where $\mathcal{D}(\mathbf{X}, \mathbf{X}')$ is the distortion metric, commonly the Chamfer Distance [43], measuring the discrepancy between the original point cloud $\mathbf{X}$ and its reconstruction $\mathbf{X}'$; $\mathcal{R}$ represents the bit rate, typically derived from the entropy of quantized latent representations; and λ is a parameter that controls the trade-off between distortion and bitrate.

In contrast, scalable point cloud coding (SPCC) structures latents for incremental reconstruction from a single bitstream. In ScalaDAT, we achieve the SPCC by ranking latent channels by variance-based importance, enabling rate-scalable decoding through selective channel activation while preserving rate-distortion efficiency across different bitrates. To better demonstrate the scalable coding's capability and performance, we introduce a controllable Scalable Ratio (SR) $\alpha \in [0, 1]$ specifying the fraction of latent channels to activate in decompression, enabling rate-scalable reconstruction:

$$\mathcal{F}_\theta^{\text{prog}} : (\mathbf{X}) \mapsto \mathcal{B} \mapsto \mathcal{B}_\alpha \mapsto (\mathbf{X}'_\alpha), \tag{3}$$

where $\mathcal{B}_\alpha \subset \mathcal{B}$ represents a subset of the total bitstream containing approximately α proportion of the total bits. In decompression, the reconstruction quality monotonically improves with α, while computational cost and bitrate scale proportionally. This enables dynamic adaptation, allowing decoders to begin with minimal channels for coarse reconstruction and scalably refine quality by incorporating additional channels as bandwidth permits.

B. The ScalaDAT Framework

Our framework extends the density-preserving D-PCC architecture [24] by transforming a monolithic coding model into a progressive one with density-aware adaptation capabilities. As shown in Fig. 2, the overall design follows an autoencoder architecture specialized for point cloud processing and entropy-constrained coding. The encoder consists of three downsampling stages, configurable with factors of 1/2, 1/3, or 1/4, and a decoder with a maximum upsampling factor of 8. The progressive coding process is formalized as:

$$\begin{aligned}
\mathbf{F} &= F(\mathbf{X}), \\
\mathbf{z}, \mathbf{z}_{xyz}, \mathbf{d} &= E(\mathbf{X}_{xyz}, \mathbf{F}), \\
\mathbf{z}_\rho, \mathbf{z}_{xyz,\rho} &= \mathcal{T}_\rho(\mathbf{z}, \mathbf{z}_{xyz}, \mathbf{d}), \\
\hat{\mathbf{z}_\rho}, \hat{\mathbf{z}}_{xyz,\rho} &= \mathcal{B}_z(\mathbf{z}_\rho), \mathcal{B}_{xyz}(\mathbf{z}_{xyz,\rho}), \\
R_\rho &= \text{BPP}(\hat{\mathbf{z}_\rho}, \hat{\mathbf{z}}_{xyz,\rho}), \\
\mathbf{X}'_\rho &= D(\hat{\mathbf{z}_\rho}, \hat{\mathbf{z}}_{xyz,\rho}).
\end{aligned} \tag{4}$$

Here, F extracts initial features $\mathbf{F} \in \mathbb{R}^{C \times N}$, which are refined by the encoder E through a series of down-sampling layers equipped with a point transformer mechanism. This yields downsampled coordinates $\mathbf{z}_{xyz} \in \mathbb{R}^{3 \times M}$, latent features $\mathbf{z} \in \mathbb{R}^{C \times M}$, and density statistics $\mathbf{d} \in \mathbb{R}^M$. Here, M denotes the number of down-sampled points and C the feature channels. The density-aware tail-drop operator $\mathcal{T}\rho$ prunes latent representations according to a drop ratio $\rho \in [0, 1]$ (linked to the Scalable Ratio $\alpha = 1 - \rho$). Quantization is performed by Entropy bottlenecks $\mathcal{B}_\mathbf{z}$ and $\mathcal{B}_{\mathbf{z}_{xyz}}$, from which the bitrate R_ρ is computed. The decoder D reconstructs $\mathbf{X}'_\rho \approx \mathbf{X}$, with reconstruction quality directly controlled by ρ.

The density-aware tail-drop mechanism is pivotal for enabling scalable coding. Training employs a stochastic drop ratio ρ as in [20], [21], which is dynamically adjusted based on local density statistics [24]. This allows the model to prioritize dense regions while adaptively handling sparse areas, making it robust to incomplete feature representations. This stochastic training strategy allows the model to reconstruct point clouds from varying levels of latent feature completeness during scalable coding at inference based on customized $\rho \in [0, 1]$. At inference, varying ρ produces reconstructions at different bitrates without retraining, enabling efficient scalable coding with a single trained model.

C. Channel-Importance based Density-aware Tail Drop

1) Density-Aware Tail Drop: In contrast to uniform channel drop techniques in all regions, such as those implemented in

ProgDTD [21] and [20], our approach introduces a density-aware tail-drop mechanism with purpose of adapting to the local density characteristics of the point cloud.

Specifically, for point cloud $\mathbf{P} = \{p_1, ..., p_N\}$ and its downsampled set $\mathbf{X} = \{x_1, ..., x_M\}$, the density-aware drop ratio, denoted by ρ, is defined as:

$$\rho = \rho_{\max} - (\rho_{\max} - \rho_{\min}) \cdot \delta, \tag{5}$$

where $\rho_{\min}$ and $\rho_{\max}$ define the range of the drop ratio, empirically set to 0.15 and 0.4, and δ is the composite density score, governing channel preservation during coding, which is calculated by averaging normalized point concentration with inverted normalized distance as:

$$\delta = \frac{1}{2}\left(\frac{d_{\text{num}}}{d_{\max}} + \left(1 - \frac{d_{\text{dist}}}{m_{\max}}\right)\right). \tag{6}$$

Here, d_{num} quantifies local point concentration and d_{dist} captures spatial distribution; $d_{\max}$ and $m_{\max}$ are their expected upper bounds, dynamically updated during training to normalize density metrics to the range of $[0, 1]$ across different datasets. For each downsampled point x_i, d_{num} is defined by counting points near p_i that collapse to x_i as:

$$d_{\text{num}}(x_i) = |\{p_j : \text{NN}(p_j) = x_i\}|, \tag{7}$$

and d_{dist} measures the total distance between x_i and each original point in the neighborhood of p_i that collapses to x_i:

$$d_{\text{dist}}(x_i) = \frac{1}{d_{\text{num}}(x_i)} \sum_{p_j \in \text{NN}^{-1}(x_i)} \|p_j - x_i\|_2. \tag{8}$$

Here, $\text{NN}(p_j)$ returns the nearest downsampled point to p_j, and $\text{NN}^{-1}(x_i)$ is the set of original points mapped to x_i.

From Per-Point Density to Global Channel Drop. The per-point density score $\delta(x_i)$ and corresponding drop ratio $\rho(x_i)$ are computed locally for each downsampled point. To obtain a single effective drop ratio for the entire point cloud, we average across all M downsampled points within each training sample:

$$\bar{\rho} = \frac{1}{M} \sum_{i=1}^{M} \rho(x_i). \tag{9}$$

This averaged $\bar{\rho}$ determines how many channels are globally dropped from the latent representation via the binary mask $\mathbf{M}$ in Eq. (13). Point clouds dominated by high-density regions produce a lower $\bar{\rho}$ (more channels retained), while predominantly sparse point clouds yield a higher $\bar{\rho}$ (more channels dropped). During training, $\bar{\rho}$ is further perturbed stochastically within $[\rho_{\min}, \rho_{\max}]$ to ensure robustness across all scalable ratios. At inference, the user directly specifies the Scalable Ratio $\alpha = 1 - \rho$, enabling explicit rate-quality control without requiring density computation at the decoder.

In SemanticKITTI [9] and ShapeNet [42], point clouds exhibit considerable variability in density and distribution, making fixed normalization parameters inadequate. To address this, we adopt the Exponential Moving Average (EMA) approach [44], which dynamically updates the normalization parameters $d_{\max}$ and $m_{\max}$ to reflect the varying density and distance distributions across different datasets. Denote the normalization parameters for d_{num} and d_{dist} as $\theta \in \{d_{\max}, m_{\max}\}$, and the current training iteration as t. The normalization parameters $d_{\max}$ and $m_{\max}$ are dynamically updated as:

$$\theta^{(t)} = (1 - \gamma) \cdot \theta^{(t-1)} + \gamma \cdot P_{95}(m^{(t)}), \tag{10}$$

to incorporate both the previous parameters $\theta^{(t-1)}$, and the 95th percentile of current batch statistics $P_{95}(m^{(t)})$[1], enabling gradual adaptation to dataset characteristics while maintaining stability.

2) Global and Local Variance-based Channel Importance: Point clouds contain complex surface geometries with sharp edges, fine details, and intricate structures, leading to strong spatial variations across feature channels [22], [45]. We propose an enhanced channel-importance calculation method that augments variance-based metrics with gradient information while maintaining low computational cost.

Specifically, let the variance feature values in channel c be denoted as Var_c. The gradient metric for the channel, denoted as Grad_c, measures local detail variations by summarizing the differences between adjacent positions within the channel:

$$\text{Grad}_c = \frac{1}{N-1} \sum_{j=1}^{N-1} |\mathbf{z}_{c,j+1} - \mathbf{z}_{c,j}|, \tag{11}$$

where $\mathbf{z}_{c,j}$ is the feature value of channel c at position j in the encoded latent space.

The final channel importance, denoted as $\mathcal{I}_c$, is then computed as a weighted combination of normalized variance and gradient metric in order to capture both globally informative features with high variance and locally discriminative features with high gradient as:

$$\mathcal{I}_c = \beta \cdot \text{norm}(\text{Var}_c) + (1 - \beta) \cdot \text{norm}(\text{Grad}_c). \tag{12}$$

β is empirically set as 0.6 to balance the relative contribution between global and local components.

3) Density-aware Tail Drop: While ProgDTD [21] extends the tail-drop [20] from a latent feature-only drop strategy to a both latent and hyper-latent [13] drop strategy in 2D image coding, we propose tail-drop for both latent features and down-sampled coordinates to achieve better performance, as confirmed by our Ablation Study in Sect. IV-E.

Learning-based PCC normally relies on two equally critical data modalities: semantic features that encode surface properties and coordinate features that capture spatial geometry [24] [10] [46]. Unlike the hyper-latent in image coding, coordinate information constitutes the core geometric data that directly determines reconstruction quality. Meanwhile, the importance of coordinate features varies across spatial regions, with surfaces typically demonstrating high spatial correlation and redundancy. Thus, we propose a density-aware tail drop method that employs a combined drop strategy to achieve this by applying density-guided scalable coding to both features and coordinates based on their channel importance, to ensure geometric consistency between coordinate and semantic representations. Specifically, given a drop ratio ρ and channel importance of features and coordinates, denoted as $\mathcal{I}_z = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_{C_z}\}$ and $\mathcal{I}_{xyz} = \{\mathcal{I}_1^{xyz}, \mathcal{I}_2^{xyz}, \ldots, \mathcal{I}_{C_{xyz}}^{xyz}\}$,

[1]Experiments across various percentiles (90th-99th) and adaptation rates ($[0.01, 0.5]$) revealed that P_{95} and $\gamma = 0.1$ yield optimal performance.

respectively, our density-aware tail drop strategy retains the top $(1-\rho)$ fraction of channels in both feature and coordinate spaces, as:

$$\mathbf{z}_\rho = \mathbf{z} \odot \mathbf{M}(\mathcal{I}_z, \rho), \quad \mathbf{z}_{xyz,\rho} = \mathbf{z}_{xyz} \odot \mathbf{M}(\mathcal{I}_{xyz}, \rho). \quad (13)$$

Here, $\mathbf{M}(\mathcal{I}, \rho) \in \{0,1\}^C$ is a binary mask that preserves the top $(1-\rho) \times C$ channels according to importance ranking $\mathcal{I}$, C is the total channel count, and $\odot$ denotes element-wise multiplication. This synchronized drop with a common ρ maintains correspondence between feature and coordinate channels throughout the coding process.

D. Loss Function

Following D-PCC [24], we adopt an integrated loss $\mathcal{L}$, which is formulated as a weighted sum of geometry reconstruction quality terms and coding efficiency constraints:

$$\mathcal{L} = \mathcal{L}_{\text{CD}} + \sigma \cdot \mathcal{L}_{\text{Dens}} + \omega \cdot \mathcal{L}_{\text{Coord}} + \eta \cdot \mathcal{L}_{\text{Points}} + \lambda \cdot \mathcal{R}_{\text{BPP}}. \quad (14)$$

Here, $\mathcal{L}_{\text{CD}}$ measures the Chamfer Distance between the original and reconstructed point clouds, ensuring geometric fidelity, $\mathcal{L}_{\text{Dens}}$ preserves local density distributions, $\mathcal{L}_{\text{Coord}}$ regularizes the quantized spatial representation before and after the entropy bottleneck, $\mathcal{L}_{\text{Points}}$ constrains the total number of reconstructed points, and finally, $\mathcal{R}_{\text{BPP}}$ represents the bit rate loss. The weighting coefficients σ, ω, η, and λ control the trade-off between different reconstruction quality aspects and coding efficiency. Consistent with D-PCC [24], we use the same weighting coefficients, enabling fair comparison with baseline methods.

E. Scalable Coding Evaluation

1) Channel-based Scalable Coding Evaluation: To enable scalable rate–distortion control at inference, our framework adopts a channel-based retention scheme. Given the importance ranking $\mathcal{I}$ computed from latent features and coordinates, we simulate different coding ratios by retaining varying ratios of channels. Given a drop ratio ρ, the top $k = \lceil (1-\rho) \cdot C \rceil$ channels are reserved, and the scalable reconstruction becomes:

$$\hat{\mathbf{z}}_\rho = \hat{\mathbf{z}} \odot \mathbf{M}(\mathcal{I}_\ddagger, \rho), \quad \hat{\mathbf{z}}_{xyz,\rho} = \hat{\mathbf{z}}_{xyz} \odot \mathbf{M}(\mathcal{I}_{xyz}, \rho), \quad (15)$$

where $\hat{\mathbf{z}}$ and $\hat{\mathbf{z}}_{xyz}$ represent quantized latent features and quantized coordinates after entropy bottleneck compression, respectively (see Eq. 4). By construction, retaining the most informative channels first allows the reconstruction quality to improve monotonically as k increases, with the largest gains concentrated in the low-k regime.

2) Bitrate Calculation: Following the entropy-based measurement protocols from D-PCC [24] and H.264 SVC [17], we compute the effective bits-per-point (BPP) for each scalable level. Given a drop ratio ρ, the BPP is:

$$\mathcal{R}_{\text{BPP}_\rho} = \frac{1}{N} \left(\sum_{i \in \mathcal{S}_\rho} -\log_2 p(\hat{\mathbf{z}}_i) + \sum_{j \in \mathcal{S}_{xyz,\rho}} -\log_2 p(\hat{\mathbf{z}}_{xyz,j}) \right), \quad (16)$$

where $\mathcal{S}_\rho$ and $\mathcal{S}_{xyz,\rho}$ denote the sets of retained channel indices for features and coordinates, respectively, and $p(\cdot)$ represents the learned probability distributions from our entropy models $\mathcal{B}_{\mathbf{z}}$ and $\mathcal{B}_{\mathbf{z}_{xyz}}$.

The BPP in Eq. (16) is computed from the learned entropy model's probability estimates (i.e., cross-entropy) rather than from actual arithmetic coding output. This is standard practice in learning-based compression [13], [24], [12], as the cross-entropy of the learned distribution provides a tight upper bound on the actual coded bitstream length under optimal entropy coding. In our implementation, we verified that the actual bitstream sizes produced by arithmetic coding differ from the entropy-based estimates by less than 1%, confirming the accuracy and validity of this estimation approach for rate-distortion evaluation.

IV. Experiments

A. Datasets and Implementation

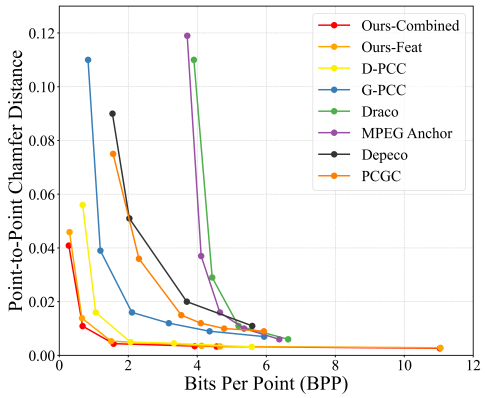
To evaluate the performance of the proposed ScalaDAT, we conducted experiments and ablation studies on two benchmark datasets, *i.e.*, SemanticKITTI [9] and ShapeNet [42]. SemanticKITTI consists of approximately 43,000 LiDAR scan frames from urban and suburban driving environments, each containing around 120,000 points with higher density near vehicles due to sensor proximity. ShapeNet [42] comprises 51,300 synthetic 3D models across 55 categories, with each model averaging 55,000 points. When preprocessing these datasets for training and testing, we strictly adhere to the requirements specified in D-PCC [24]. The training objective combines distortion loss, density loss (initialized at 10^{-4}), and coordinate loss (initialized at 5×10^{-5}). Optimization is performed over 50 epochs using the Adam optimizer, with an initial learning rate of 10^{-3}, decayed by 0.5 every 15 epochs. All experiments are conducted with an NVIDIA A40 GPU.

Based on experimental results, we increased the number of latent feature channels from the original 8 in [24] to 32 to enable effective scalable coding. This provides sufficient channel diversity, enhancing representation capacity and coding efficiency: 16 channels yielded inadequate diversity, while 64 introduced redundancy without commensurate gains, making 32 optimal for quality-efficiency balance. We thus adopted 32 channels across all experiments. Further, unlike D-PCC [24], which optimizes at batch size 1, we used 32 for ScalaDAT to reduce computational demands.

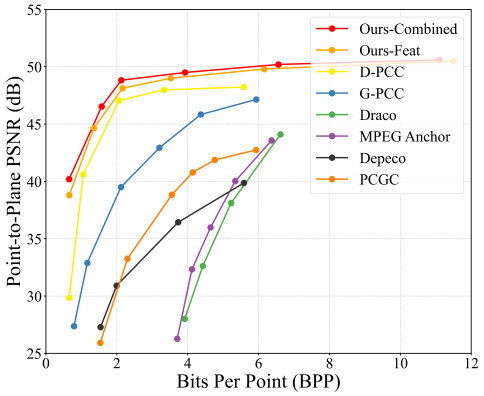
B. Evaluation Metrics

As there is currently no standardization or benchmark for scalable point cloud geometry coding, we compare our results against SOTA learning-based point cloud coding models, on the same datasets, including D-PCC [24], G-PCC [7], Google Draco [47], MPEG Anchor [48], PCGC [11], Depeco [49], and JPEG Pleno [10].

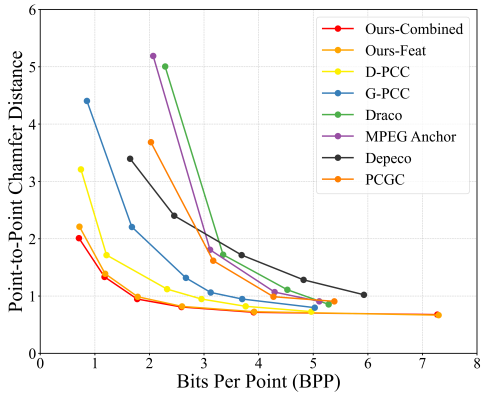
We evaluate ScalaDAT using a suite of metrics: PSNR-D1, PSNR-D2, Chamfer Distance (CD), and Bjontegaard Rate (BD-Rate) [50], [51], to collectively assess perceptual quality, geometric fidelity, and coding efficiency.



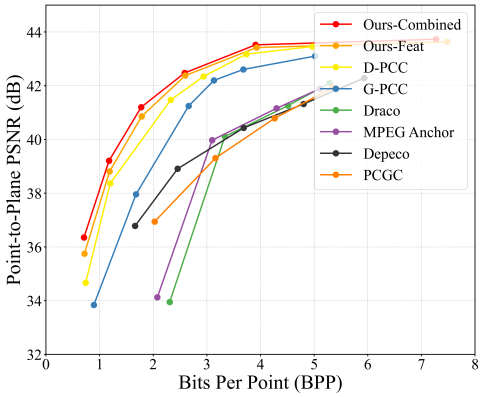
(a) SemanticKITTI: BPP vs Chamfer Distance



(b) SemanticKITTI: BPP vs PSNR

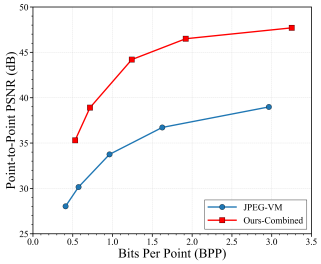


(c) ShapeNet: BPP vs Chamfer Distance

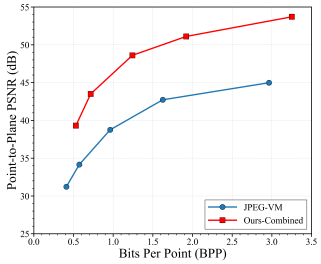


(d) ShapeNet: BPP vs PSNR

Fig. 3: Quantitative results of non-scalable ScalaDAT on SemanticKITTI and ShapeNet: BPP vs. CD (a, c) and BPP vs. PSNR-D2 (b, d). Models are trained using scalable coding. Results compare two tail-drop strategies: Combined Drop (red lines) and Feature-Only Drop (orange lines), with the model trained using scalable coding. The differences between these strategies will be described in terms of the Ablation Study below.



(a) BPP vs PSNR-D1



(b) BPP vs PSNR-D2

Fig. 4: Quantitative comparison of non-scalable ScalaDAT with JPEG-VM [10] on SemanticKITTI in terms of PSNR-D1 and PSNR-D2 for each point cloud model.

PSNR-D1 and **PSNR-D2** are variants of the peak signal-to-noise ratio (PSNR) measuring the quality of a reconstructed point cloud relative to the original. Specifically,

$$\text{PSNR-D} = 10 \cdot \log_{10}\left(\frac{3 \cdot \text{Peak}_D^2}{\text{MSE}_{\max}^{(D)}}\right), \tag{17}$$

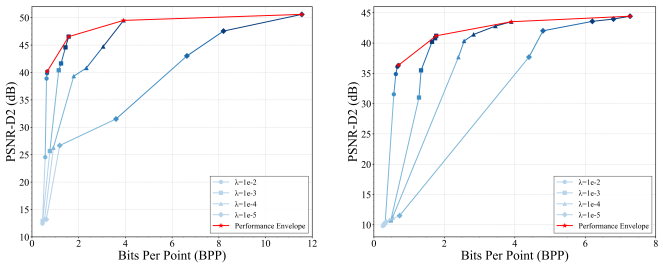
where D denotes the distance metric. The peak signal value, Peak_D, is defined as the squared length of the bounding-box diagonal, *i.e.*, $\|\mathbf{p}_{\max} - \mathbf{p}_{\min}\|_2^2$, with $\mathbf{p}_{\max}$ and $\mathbf{p}_{\min}$ being the maximum and minimum coordinates of the original point cloud, as in D-PCC [24]. The same Peak_D is used for both PSNR-D1 and PSNR-D2 to ensure comparability. $\text{MSE}_{\max}^{(D)}$ is the maximum of two directional mean squared errors (MSE) between the original and reconstructed point clouds:

$$\text{MSE}_{\max}^{(D)} = \max\left(\text{MSE}_{o\to r}^{(D)}, \text{MSE}_{r\to o}^{(D)}\right), \tag{18}$$

where $\text{MSE}_{o\to r}^{(D)}$ is computed from the original to the reconstructed, and $\text{MSE}_{r\to o}^{(D)}$ vice versa, both calculated with distance metric D. For PSNR-D1, D uses Euclidean nearest-neighbor distances to assess geometric accuracy; For PSNR-D2, D uses distances projected along surface normals to capture perceptual distortions and surface smoothness, critical for applications such as virtual reality.

CD evaluates geometric fidelity and density preservation via the average bidirectional nearest-neighbor distances, and is widely used for shape comparison and as a loss in 3D learning tasks [52], [53]. **BD-Rate** quantifies coding efficiency as the relative bitrate difference between two rates at matched quality (typically using PSNR or a similar metric); lower values indicate better performance.

The PSNR values reported in this work are computed following D-PCC [24], where the peak signal value is defined as the squared length of the bounding-box diagonal of the *original* point cloud, i.e., $\text{Peak} = \|\mathbf{p}_{\max} - \mathbf{p}_{\min}\|_2^2$. This differs from methods such as SparsePCGC [12], which uses the voxel



(a) SemanticKITTI (b) ShapeNet

Fig. 5: Scalable performance of ScalaDAT (PSNR-D2 vs BPP) on SemanticKITTI and ShapeNet across different λ values with simultaneous latent feature and coordinate drop.

grid resolution (e.g., $2^{18} - 1$ for 18-bit precision) as the peak value, resulting in higher absolute PSNR numbers. The difference in absolute PSNR values across papers is therefore attributable to the choice of peak normalization rather than actual quality differences. All methods in our experiments are evaluated under identical conditions with the same peak definition, ensuring fair and consistent comparison.

C. Evaluation Results

1) Rate Distortion Evaluation: Results demonstrate the superior RD performance on the benchmark datasets SemanticKITTI [9] and ShapeNet [42]. As shown in Fig. 3, our approach consistently outperforms SOTA methods in quantitative evaluations using CD, PSNR-D1, and PSNR-D2 metrics.

To comprehensively evaluate our method, we report non-scalable results from training and testing without channel drop. Notably, ScalaDAT consistently achieves significantly lower CD values across all bitrates, especially in the low-bitrate range of 0.5–2.0 BPP. For instance, on SemanticKITTI (Fig. 3b), it attains $\sim$ 43 dB PSNR-D2 at 1.0 BPP, a quality level that competing methods only reach at substantially higher bitrates. Similarly, on ShapeNet (Fig. 3d), it achieves $\sim$ 39.2 dB at 2.0 BPP, underscoring its efficiency in preserving geometric fidelity. Besides quality improvements, ScalaDAT also extends the supported larger bitrate range than D-PCC [24]: from 0–6 BPP to 0–11 BPP on SemanticKITTI and 0–5 BPP to 0–7 BPP on ShapeNet, enhancing flexibility for bandwidth-limited applications. Moreover, the BD-rate metric shows that our combined Drop method achieves 28.6% bitrate savings on SemanticKITTI and 18.15% on ShapeNet relative to D-PCC [24], at equivalent quality levels. This consistent performance across structured outdoor environments and object-centric data highlights the versatility and robustness of the proposed method for scalable point cloud coding.

To benchmark our ScalaDAT model against the JPEG-VM standard [10] for learning-based PCC, we utilized the SemanticKITTI dataset [9]. Point clouds were voxelized at a 10-bit precision to balance geometric fidelity and coding efficiency. Each LiDAR point cloud was normalized to a unit cube via scene-adaptive bounding boxes, with outliers removed and 5% padding to reduce boundary artifacts, and then quantized to integers in [0, 1023]. The preprocessed dataset was evaluated using checkpoints from both ScalaDAT and JPEG-VM [10]. As shown in Fig. 4, ScalaDAT achieves substantial gains over JPEG-VM, with BD-rate reductions exceeding 65% for PSNR-D1 and 55% for PSNR-D2 on SemanticKITTI. These BD-rate savings are largely attributable to evaluating JPEG-VM with publicly released pre-trained models on SemanticKITTI without retraining.

2) Scalable Performance on PSNR-D2: We evaluate scalable coding using the **PSNR-D2** metric, following the de facto benchmark for point-cloud compression [24]. Figs. 5a and 5b show the rate–distortion (RD) trade-off for SemanticKITTI and ShapeNet, respectively. Each plot includes multiple curves for different values of the trade-off parameter λ that controls the balance between coding efficiency and reconstruction quality.

In detail, on SemanticKITTI (Fig. 5a), PSNR-D2 starts relatively high even at low bits (0–1 BPP) and increases gradually; Once BPP exceeds 1, it rises more steeply and tends to saturate around 4–6 BPP. PSNR-D2 starts relatively high even at low bit budgets (0–1 BPP) and increases gradually; once BPP exceeds $\sim$ 1, it rises more steeply and tends to saturate around 4–6 BPP. Conversely, on ShapeNet (Fig. 5b), PSNR-D2 begins lower at small BPP and improves more slowly. In both datasets, we observe a critical threshold near 1 BPP beyond which additional bits yield markedly better reconstruction quality, consistent with higher scalable gains. These differences reflect intrinsic point-cloud characteristics: SemanticKITTI's LiDAR-derived regular structure and lower complexity enable efficient coding and superior quality at low bitrates (*e.g.*, 0–1 BPP), whereas ShapeNet's diverse, intricate objects require more bits to achieve comparable quality, particularly evident in the low-BPP regime, where SemanticKITTI achieves robust reconstruction with fewer bits.

3) Computational Complexity Analysis: To address the operational feasibility of the proposed SPCC framework, a computational complexity analysis was conducted compared to the single-rate baseline D-PCC [24]. A critical advantage of ScalaDAT lies in its training efficiency; unlike prior scalable methods that require complex multi-stage recursive training for different quality layers, ScalaDAT achieves full scalable capabilities through a single training iteration.

Regarding inference complexity tested on an NVIDIA A40 GPU, to support fine-grained scalable channel dropping, ScalaDAT utilises an expanded 32-channel configuration, which naturally results in a moderately larger model size ($\sim$45 MB) compared to the 8-channel baseline. While the full-channel reconstruction (SR = 1.0) requires comparable encoding and decoding latency to the baseline, operating at lower Scalable Ratios dynamically reduces the computational load on the decoder side. Specifically, as the SR decreases, the density-aware tail-drop mechanism prunes unselected latent channels before the decoder phase. Consequently, the decoder physically bypasses the inverse convolution operations for the dropped channels, leading to a measurable reduction in decoding latency compared to processing a full monolithic bitstream. This dynamically scaling computational load perfectly aligns with the requirements of edge devices in bandwidth-constrained and power-limited environments.

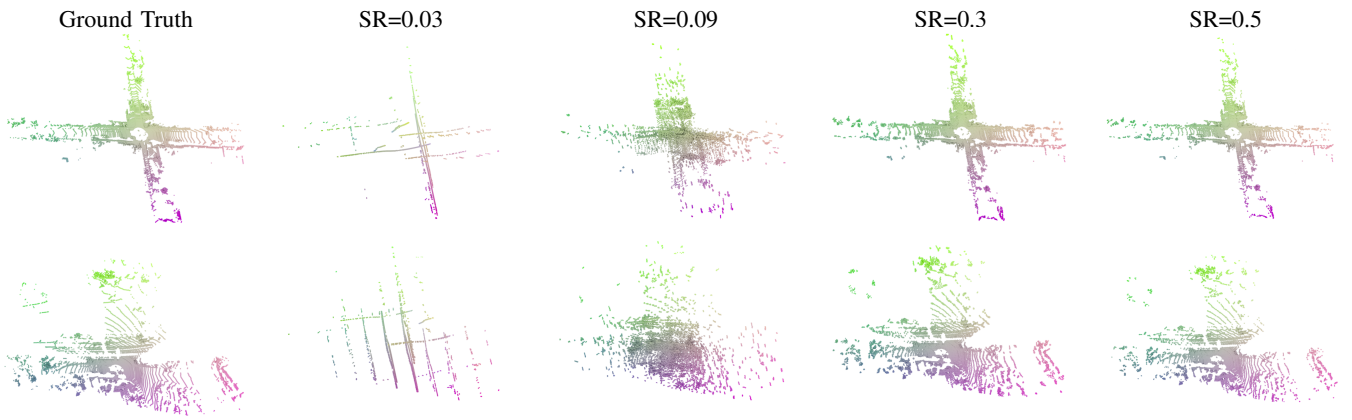


Fig. 6: Visualization of the scalable coding applied to two SemanticKITTI models [9] with various SR values, illustrating the impact of Combined-Drop.

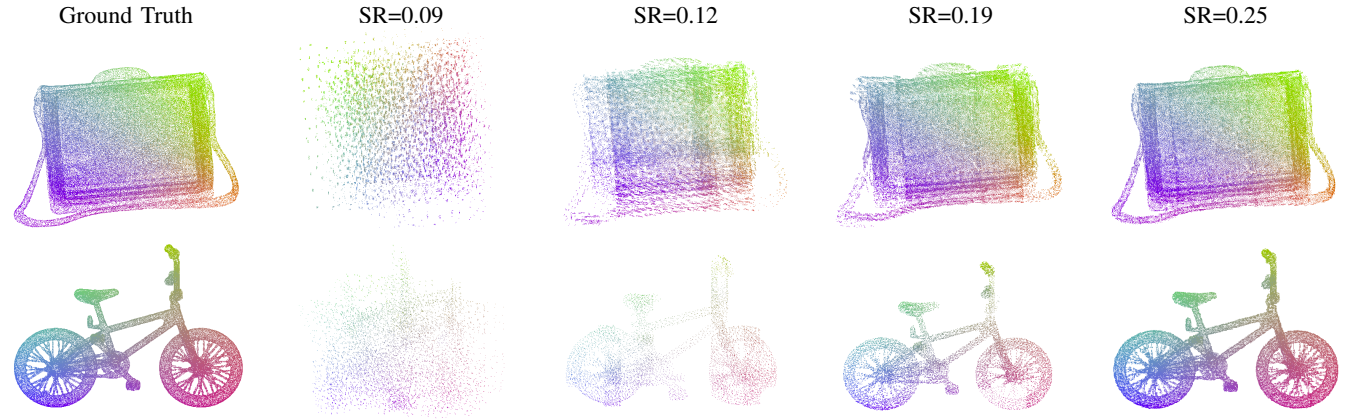


Fig. 7: Visualization of the scalable coding applied to two ShapeNet models [42] with lambda as 0.001, illustrating the impact of combined feature and coordinate drop. From left to right: Ground Truth, Scalable Ratio (SR) = 0.09, 0.12, 0.19, and 0.25, corresponding to drop rates (1-SR) of 0.91, 0.88, 0.81, and 0.75, respectively.

D. Visualization

Figs. 6 and 7 depict scalable reconstruction of point clouds from SemanticKITTI and ShapeNet datasets when $\lambda = 0.001$. For instance, in Fig. 6, at a scalable ratio (SR) of 0.03, the essential scene structure remains discernible, preserving core geometry with a minimal bitstream. As SR increases, the decoder incrementally activates additional latent channels according to our density-aware importance ranking (from highest to lowest geometric significance). This prioritization yields rapid gains in reconstruction quality, highlighting the effectiveness of our density-aware channel importance assessment in favoring geometrically critical coordinates and features for scalable transmission. Notably, Fig. 7 features two objects selected for their sparse-to-dense point distributions. At very low PR, reconstructions may exhibit minor spatial overlap and noise; we hypothesize that as SR increases and additional components of the bitstream are decoded, the enriched latent features improve point localization while suppressing spurious and overlapping points, thereby improving overall reconstruction performance.

E. Ablation Studies

We compare two strategies: combined coordinate-feature drop and feature-only drop, where models were trained with $\lambda \in [10^{-2}, 10^{-5}]$. Each model was trained once per λ and subsequently evaluated through scalable testing across multiple scalable ratios. Results are displayed in Fig. 3, Fig. 5a, and Fig. 9b, which indicate the higher PSNR-D2 performance for the combined-drop method, underscoring its effectiveness through BD-Rate improvements exceeding 12.3% on the SemanticKITTI dataset and and 9.6% on ShapeNet dataset compared to the feature-only strategy statistically.

Besides, we set $\lambda = 0.001$ here to show the visualizations of different drop strategies' results and get Fig. 8, which is consistent with the setting of Sect. IV-D. Compared with Fig. 6, even though feature-only drop shows strong central and structural information when SR is low (*e.g.*, 0.03), its BPP starts from a much higher value, which further proves the benefit of combined-drop that it can achieve the scalable coding starting from a much smaller BPP value. In details, the feature-only method shows an elevated initial BPP of 0.77 because of direct coordinate encoding, in contrast to the combined-drop approach starting at 0.008 BPP, which can be observed from BPP-SR relationships Fig. 9a.

V. Conclusion

In this work, we presented ScalaDAT, a novel framework for scalable point cloud coding that introduces Tail-drop, an

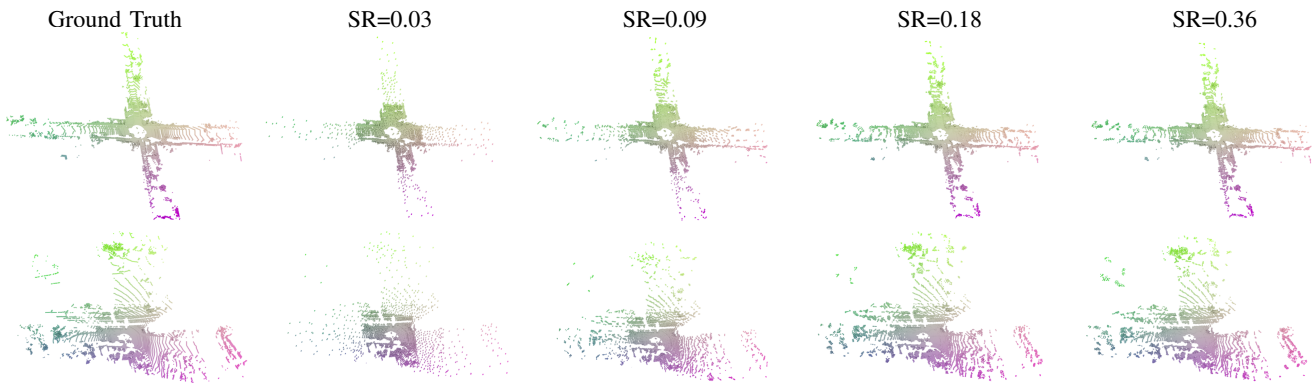
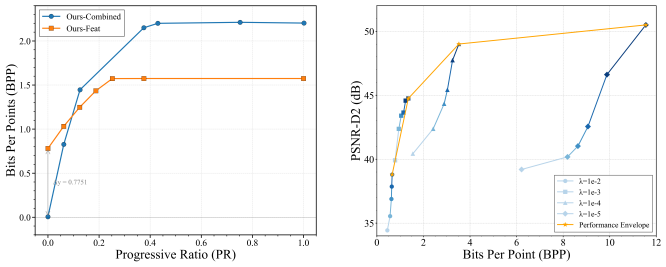


Fig. 8: Visualization of progressive coding with feature-only drop applied to two SemanticKITTI models. From left to right: Ground Truth, SR = 0.03, 0.09, 0.18, and 0.36, yielding the corresponding drop rates (1-SR) of 0.97, 0.91, 0.82, and 0.64, respectively.



(a) (b)

Fig. 9: Ablation study of scalable coding for different drop strategies and λ values on SemanticKITTI: (a) BPP vs SR for different drop strategies, and (b) PSNR-D2 vs BPP for feature-only drop with varying λ values.

innovative data selection method that strategically discards less critical information while maintaining reconstruction quality. Unlike conventional methods, ScalaDAT achieves scalable coding with just a single training iteration, significantly reducing computational overhead while maintaining competitive performance. Comprehensive experiments conducted on SemanticKITTI and ShapeNet datasets demonstrate superior BD-Rate performance compared to state-of-the-art (SOTA) methods. Our exploration of density-aware Tail-drop and latent space drop strategies provides new insights into coding dynamics, revealing opportunities for further optimization.

REFERENCES

[1] M. Quach, J. Pang, D. Tian, G. Valenzise, and F. Dufaux, "Survey on deep learning-based point cloud compression," *Frontiers in Signal Processing*, vol. 2, p. 846972, 2022.

[2] Y. Cao, Y. Wang, and H. Chen, "Real-time lidar point cloud compression and transmission for resource-constrained robots," *arXiv preprint arXiv:2502.06123*, 2025.

[3] T. Golla and R. Klein, "Real-time point cloud compression," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 5087–5092.

[4] K. Liu, K. You, P. Gao, and M. Paul, "Att 2 cpc: Attention-guided lossy attribute compression of point clouds," *IEEE Transactions on Artificial Intelligence*, 2024.

[5] X. Yue, B. Wu, S. A. Seshia, K. Keutzer, and A. L. Sangiovanni-Vincentelli, "A lidar point cloud generator: from a virtual world to autonomous driving," in *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, 2018, pp. 458–464.

[6] T. Fukuda, Y. Zhu, and N. Yabuki, "Point cloud stream on spatial mixed reality-toward telepresence in architectural field," in *Proceedings of the 36th International Conference on Education and Research in Computer Aided Architectural Design in Europe (eCAADe) [Volume 2]*, 2018.

[7] W. MDGC, "7, "G-PCC codec description v9"," *MPEG-3DG. G-PCC Codec Description v9. ISO/IEC JTC1/SC29/WG7 N0011*, 2020.

[8] ——, "7, "V-PCC codec description v12"," *MPEG-3DG. V-PCC Codec Description v12. ISO/IEC JTC1/SC29/WG7 N0012*, 2020.

[9] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9297–9307.

[10] A. F. Guarda, N. M. Rodrigues, M. Ruivo, L. Coelho, A. Seleem, and F. Pereira, "IT/IST/Ipleiria response to the call for proposals on JPEG pleno point cloud coding," *arXiv preprint arXiv:2208.02716*, 2022.

[11] J. Wang, H. Zhu, H. Liu, and Z. Ma, "Lossy point cloud geometry compression via end-to-end learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 12, pp. 4909–4923, 2021.

[12] J. Wang, D. Ding, Z. Li, X. Feng, C. Cao, and Z. Ma, "Sparse tensor-based multiscale representation for point cloud geometry compression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[13] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," *arXiv preprint arXiv:1802.01436*, 2018.

[14] J.-H. Lee, S. Jeon, K. P. Choi, Y. Park, and C.-S. Kim, "Dpict: Deep progressive image compression using trit-planes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2022, pp. 16 113–16 122.

[15] G. K. Wallace, "The JPEG still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[16] D. S. Taubman, M. W. Marcellin, and M. Rabbani, "JPEG2000: Image compression fundamentals, standards and practice," *Journal of Electronic Imaging*, vol. 11, no. 2, pp. 286–287, 2002.

[17] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h. 264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[18] J. Wang, D. Ding, Z. Li, and Z. Ma, "Multiscale point cloud geometry compression," in *2021 Data Compression Conference (DCC)*. IEEE, 2021, pp. 73–82.

[19] M. Rudolph, A. Riemenschneider, and A. Rizk, "Progressive coding for deep learning based point cloud attribute compression," in *Proceedings of the 16th International Workshop on Immersive Mixed and Virtual Environment Systems*, 2024, pp. 78–84.

[20] T. Koike-Akino and Y. Wang, "Stochastic bottleneck: Rateless auto-encoder for flexible dimensionality reduction," in *2020 IEEE International Symposium on Information Theory (ISIT)*, 2020, pp. 2735–2740.

[21] A. Hojjat, J. Haberer, and O. Landsiedel, "ProgDTD: Progressive learned image compression with double-tail-drop training," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2023, pp. 1130–1139.

[22] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[23] H. Xu, X. Zhang, and X. Wu, "Fast point cloud geometry compression with context-based residual coding and inr-based refinement," in *Proceedings of the European conference on computer vision (ECCV)*. Springer, 2025, pp. 270–288.

[24] Y. He, X. Ren, D. Tang, Y. Zhang, X. Xue, and Y. Fu, "Density-preserving deep point cloud compression," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2022, pp. 2333–2342.

[25] L. Huang, S. Wang, K. Wong, J. Liu, and R. Urtasun, "Octsqueeze: Octree-structured entropy model for lidar compression," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2020, pp. 1313–1323.

[26] T. Fan, L. Gao, Y. Xu, D. Wang, and Z. Li, "Multiscale latent-guided entropy model for lidar point cloud compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 12, pp. 7857–7869, 2023.

[27] C. Fu, G. Li, R. Song, W. Gao, and S. Liu, "Octattention: Octree-based large-scale contexts model for point cloud compression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 1, 2022, pp. 625–633.

[28] J. Wang, R. Xue, J. Li, D. Ding, Y. Lin, and Z. Ma, "A versatile point cloud compressor using universal multiscale conditional coding – part I: Geometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[29] M. Quach, G. Valenzise, and F. Dufaux, "Learning convolutional transforms for lossy point cloud geometry compression," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 4320–4324.

[30] A. F. Guarda, N. M. Rodrigues, and F. Pereira, "Point cloud coding: Adopting a deep learning-based approach," in *2019 Picture Coding Symposium (PCS)*. IEEE, 2019, pp. 1–5.

[31] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimization of nonlinear transform codes for perceptual quality," in *2016 Picture Coding Symposium (PCS)*. IEEE, 2016, pp. 1–5.

[32] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[33] C. Cai, L. Chen, X. Zhang, G. Lu, and Z. Gao, "A novel deep progressive image compression framework," in *2019 Picture Coding Symposium (PCS)*. IEEE, 2019, pp. 1–5.

[34] E. Diao, J. Ding, and V. Tarokh, "Drasic: Distributed recurrent autoencoder for scalable image compression," in *2020 Data Compression Conference (DCC)*. IEEE, 2020, pp. 3–12.

[35] K. Islam, L. M. Dang, S. Lee, and H. Moon, "Image compression with recurrent neural network and generalized divisive normalization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2021, pp. 1875–1879.

[36] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. J. Hwang, J. Shor, and G. Toderici, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4385–4393.

[37] C.-H. Huang and J.-L. Wu, "Unveiling the future of human and machine coding: A survey of end-to-end learned image compression," *Entropy*, vol. 26, no. 5, p. 357, 2024.

[38] Y. Lu, Y. Zhu, Y. Yang, A. Said, and T. S. Cohen, "Progressive neural image compression with nested quantization and latent ordering," in *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2021, pp. 539–543.

[39] S. Li, H. Li, W. Dai, C. Li, J. Zou, and H. Xiong, "Learned progressive image compression with dead-zone quantizers," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 6, pp. 2962–2978, 2022.

[40] Y. Huang, J. Peng, C.-C. J. Kuo, and M. Gopi, "Octree-based progressive geometry coding of point clouds." in *PBG@ SIGGRAPH*, 2006, pp. 103–110.

[41] G. Vadivel, S.-C. Cheng, C.-C. Huang, and J.-Y. Su, "Progressive point cloud compression with the fusion of symmetry based convolutional neural pyramid and vector quantization," in *2021 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*. IEEE, 2021, pp. 1–2.

[42] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[43] S. Perry, H. P. Cong, L. A. da Silva Cruz, J. Prazeres, M. Pereira, A. Pinheiro, E. Dumic, E. Alexiou, and T. Ebrahimi, "Quality evaluation of static point clouds encoded using mpeg codecs," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 3428–3432.

[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. pmlr, 2015, pp. 448–456.

[45] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[46] Z. Luo, W. Jia, and S. Perry, "Transformer-based geometric point cloud compression with local neighbor aggregation," in *2023 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE, 2023, pp. 223–228.

[47] F. Galligan, M. Hemmer, O. Stava, F. Zhang, and J. Brettle, "Google/draco: a library for compressing and decompressing 3d geometric meshes and point clouds," *Draco: a library for compressing and decompressing 3D geometric meshes and point clouds*, 2018.

[48] R. Mekuria, K. Blom, and P. Cesar, "Design, implementation, and evaluation of a point cloud codec for tele-immersive video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 4, pp. 828–842, 2016.

[49] L. Wiesmann, A. Milioto, X. Chen, C. Stachniss, and J. Behley, "Deep compression for dense point cloud maps," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2060–2067, 2021.

[50] S. Perry, "JPEG point cloud coding common test conditions v3.6," ISO/IEC JTC1/SC29/WG1, Tech. Rep. N91058, 2021.

[51] ——, "JPEG pleno point cloud coding common test conditions v3.2," ISO/IEC JTC1/SC29/WG1, Tech. Rep. N86044, 2020.

[52] F. Lin, Y. Yue, Z. Zhang, S. Hou, K. Yamada, V. Kolachalama, and V. Saligrama, "Infocd: a contrastive chamfer distance loss for point cloud completion," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[53] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 605–613.