APSIPA Transactions on Signal and Information Processing, 2025, 14, e101

Original Paper

OSC-Net: Object Semantic-aware Compression Network for 3D Point Cloud

Kangrui Luo[1+], Donghan Bu[1+], Anhong Wang[1*], Junhui Hou[2] and Yakun Yang[1*]

[1] *School of Electronic and Information Engineering, Taiyuan University of Science and Technology, China*
[2] *Department of Computer Science, City University of Hong Kong, Hong Kong*

ABSTRACT

Point cloud compression can effectively save the amount of data required for transmission and storage of point clouds. However, the commonly used methods of point cloud compression have serious impacts on the performance of downstream visual tasks due to the ignorance of the semantic information represented by point cloud. Towards this end, this paper proposes an *object semantic-aware compression network* for 3D point cloud, namely OSC-Net. Firstly, a ground points removal module based on the elevation difference is designed, enabling the network to pay more attention to the semantic information of objects. Secondly, a 3D voxel attention module is proposed to extract multiple priors in deep entropy model that can predict the probability distribution of occupied symbols in voxel space. Finally, experimental results show that our proposed network gains a notable bitrate saving of 16.71% compared to the baseline on the KITTI 3D object detection dataset, while maintaining a comparable detection accuracy.

Keywords: Point Cloud Compression, 3D Object Detection, Deep learning

*Corresponding Authors: Anhong Wang, ahwang@tyust.edu.cn and Yakun Yang, yyk@tyust.edu.cn.
+These authors contributed equally to this work and should be considered co-first authors.

Received 31 August 2024; revised 11 November 2024; accepted 09 January 2025
ISSN 2048-7703; DOI 10.1561/116.20240065

1 Introduction

In autonomous driving, point cloud is a crucial 3D data format that comprises a series of dispersed, unordered, and topologically unstructured points. Each point cloud contains a wealth of information, including not only geometric details in 3D coordinate form, but also various attribute information such as color, normal vector, and refractive index [15, 33]. However, the large amount of point cloud data poses significant challenges in terms of transmission and storage, thereby hindering its widespread application within the field of autonomous driving [31, 34].

Point cloud compression can reduce the storage requirements and computational costs of various visual tasks [21], thereby improving the speed of downstream applications and saving storage space and transmission bandwidth in autonomous driving.Therefore, it becomes especially popular to study point cloud compression techniques [12, 24, 30].

In contrast to image and video compression, the compression of point cloud data poses an enormous challenge primarily attributed to its inherent sparsity. Early endeavors in this domain saw researchers adopting diverse data structures, including octrees [17] and KD-trees [4] as means to organize the unstructured point clouds. However, these approaches often overlooked the sparsity of point cloud, thereby limiting their compression ratios. Due to the effective training on the large scale dataset, the deep learning-based point cloud compression methods outperform the Moving Picture Experts Group (MPEG) international point cloud geometric compression standard G-PCC which uses traditional octree-based method [8].

Typical learning-based point cloud compression methods include VoxelDNN [18], VoxelContext-Net [22], MSVoxelDNN [19], PCC-S [2], OctAttention [6] and EHEM [23]. VoxelDNN adaptively divides the point cloud into multiple voxel blocks and predicts the occupancy probability of the voxel blocks sequentially using a 3D convolutional network, which incorporates the idea of autoregressive modeling. Though this approach improves the lossless compression rate of the point cloud, the contextual information of the point cloud data is neglected. MSVoxelDNN innovatively achieves parallel processing by decoupling certain dependencies among voxels within the same group, resulting in a substantial enhancement in coding speeds when compared to its predecessor, VoxelDNN. Nevertheless, this decoupling of dependencies leads to a compromise in the precision of the generated context information. PCC-S trained a deep entropy model on the KITTI dataset [7], pioneering the integration of sibling nodes context to minimize voxel redundancy. Furthermore, the introduction of a two-step reconstruction strategy significantly increased the performance. OctAttention applied an entropy model based on self attention mechanism and achieved good results. But the global self attention mechanism will bring huge computational complexity. EHEM proposed a hierarchical

attention structure and a grouped context structure, achieving better rate distortion performance and significant decoding delay reduction.

Recently, research on point cloud compression has shifted towards combining specific visual tasks, such as classification [29] and object detection [16] [13]. However, there are still many redundant point cloud data in Liu *et al.* [16], and no special attention has been paid to the semantic information that adapts to object detection.

Based on the above analysis, in order to focus more on the object to be detected and reduce the redundant in point cloud data, we propose an object semantic-aware compression network for 3D point cloud (OSC-Net). This method preserves the semantic information in object detection while reducing the number of points.

Our contributions in this work can be summarized as follows:

- We propose an object semantic-aware compression network for 3D point cloud. This network is designed to save bitrate while ensuring high accuracy of the object detection task.
- In order to save bitrate and ensure high accuracy of the object detection, we propose a ground points removal module based on the elevation difference, enabling the network to focus on object semantic information within the point cloud.
- We propose a 3D voxel attention module in the deep entropy model. 3D voxel attention module enhances semantic learning from point cloud data, thereby improving the accuracy of probability distribution of occupancy symbols.
- On the KITTI 3D object detection dataset, the reconstructed point clouds of our method demonstrate similar object detection performance, but the encoding bitrate is significantly reduced. Experimental results show that our proposed method gains a notable bitrate saving of 16.71% compared to the baseline method.

The remainder of this paper is organized as follows. Section 2 gives a brief review of related work. Section 3 describes the architecture of our proposed object semantic-aware compression network. Section 4 shows the detailed experimental results and analysis. Section 5 concludes this paper.

2 Related Work

2.1 Traditional Point Cloud Compression Methods

The most representative of the traditional point cloud compression methods are the three compression platforms proposed by the MPEG: static 3D point cloud

compression test model class 1 (TMC1), dynamic 3D point cloud compression test model class 2 (TMC2), and dynamically acquired 3D point cloud test model class 3 (TMC3). Due to the similarity of the 3D geometric compression methods of the TMC1 and the TMC3, the two are formed into a new platform called G-PCC. The TMC2 is called the video-based point cloud compression method V-PCC. The performance of the context model significantly affects the coding efficiency [24].

2.2 Deep learning based point cloud compression methods

Traditional point cloud coding methods have certain limitations when dealing with large-scale point cloud scenes. In recent years, deep learning based point cloud compression methods have received much attention and research.

Some researchers [1, 26–28] used a distance image based point cloud compression method to convert the point cloud into a depth map or distance map, and then compress it using the image compression method. However, these methods ignore the spatial information of the point cloud to some extent. Huang et al. [10] proposed a method that directly compresses the raw point cloud data for feature coding, but the point-based coding method leads to its inefficiency in processing large-scale point clouds. Yan et al. [32] proposed a method that employed a voxel-based approach to compress point cloud data, involving the quantization of the point cloud into voxels. However, this method is characterized by high computational complexity and overlooks the sparsity of the original point cloud. After that, Huang et al. [9] proposed a method that encoded the point cloud as an octree, used neural networks to train an entropy model of the octree structure, predicted the node occupancy symbols in conjunction with the context information, and compressed the encoding. However, it still ignored the dependency of neighboring nodes of the octree.

Que et al. [22] achieves better coding performance by embedding the octree information into voxels to get context information. The use of voxel context-based 3D coordinate refinement after decoding reduces the loss caused by quantizing the point cloud coordinates to integer precision. The method proposed by PCC-S fuses ancestor nodes, neighbor nodes, and sibling nodes into the voxel context based on combining octree and voxel, and adds surface information as a strong prior, which gives better coding performance than the VoxelContext-Net. Octattention [6] and EHEM [23] models use self attention mechanisms to explore dependency relationships in large-scale environments, achieving better coding performance and demonstrating the advantages of attention mechanisms in entropy models.

2.3 Point cloud compression method combined with tasks

The methods in Section 2.2 are all optimized for feature extraction as well as decoding and reconstruction loss of point cloud data but fail to adequately

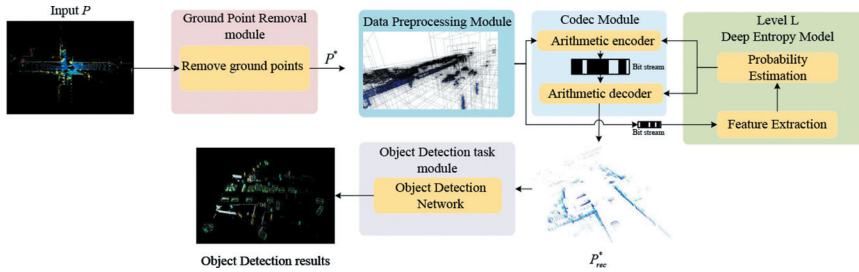

Figure 1: OSC-Net overview framework.

match the visual task. When utilizing point cloud data, the reconstructed point cloud should not only preserve its original information but also be adapted to specific tasks and various applications. As a result, point cloud coding methodologies that seamlessly integrate specific tasks have progressively emerged as a focal point of research.

In 2019, Dovrat et al. [5] pioneeringly introduced S-Net, the inaugural deep learning-based downsampling network designed specifically for both point cloud classification and reconstruction tasks. S-Net crafts subsets of point clouds that closely resemble the original shapes through sampling loss constraints, while also generating tailored subsets optimized for downstream machine vision tasks via task-specific loss constraints. In 2021, Lin et al. [14] proposed DA-Net, which extends S-Net using a density-adaptive sampling strategy, thus reducing the effect of noise points and improving the performance of downstream classification tasks. In 2023, Ulhaq et al. [29] proposed the first point cloud coding compression network specifically for classification tasks. This network was built based on PointNet and achieves a better trade-off between code rate and classification accuracy, as compared to non-task-specific compression networks. Liu et al. [16] proposed a method for jointly optimizing point cloud compression and object detection. By designing a gradient bridge function, this method enables gradient back-propagation from the detector to the codec. In 2024, Li et al. [13] proposed a 3D multi-scale feature compression method for object detection called 3D-MSFC. 3D-MSFC uses sparse convolution [3] to extract, compress, and reconstruct 3D multi-scale sparse features, which are then fed into a 3D detection network to obtain detection results. The importance of each scale in object detection accuracy is analyzed.

In this paper, we propose an OSC-Net which combines point cloud compression with object detection. Firstly, filter out ground points from LiDAR data and focus on the region of interest for object detection. Subsequently, the proposed 3D voxel attention module extracts features from the voxel space, improving compression performance while maintaining object detection accuracy in the reconstructed point cloud.

3 Methodology

3.1 Overview

The overview of OSC-Net is shown in Figure 1. Given a set of point clouds containing $\boldsymbol{N}$ points, each point containing 3 dimensions, denoted as $\boldsymbol{P}$. Firstly, $\boldsymbol{P}$ is inputted into the ground points removal module, and the ground points is removed from the original point cloud data based on the elevation difference information to generate $\boldsymbol{P}^*$. Secondly, $\boldsymbol{P}^*$ is inputted into the data preprocessing module to organize the structure of the octree, and the occupied information are filled into the voxel space to generate a hierarchical voxel context. The Level L deep entropy model is then utilized to extract both global and local features from the hierarchical voxel context. Subsequently, the probability distribution of the occupancy symbol for each non-leaf node, spanning 256 classes, is predicted based on varying levels of features. These occupancy symbols consist of a sequence of 8-bit binary symbols, each representing the occupancy status of one of the 256 voxel spaces. Finally, the occupancy symbols are encoded into a more compact bit stream using the arithmetic encoder in the codec module, and the point cloud is reconstructed using the arithmetic decoder.

The reconstructed point cloud is denoted as $\boldsymbol{P}^*_{rec}$. In the object detection task module, the reflectivity information, r, is added to $\boldsymbol{P}^*_{rec}$, which is then inputted into the object detection network to perform the detection task and output the object detection result.

3.2 Ground Points Removal Module

In LiDAR point cloud data, ground points exhibit distinct ripple-like features. In terms of point cloud encoding, the ground points in the point cloud dataset consume significantly computational resources during the coding processes. For the task of object detection, the ground points belong to background information, but the attention given by the object detector to background information is not apparent.

Therefore, we propose an algorithm for removing ground points based on the elevation difference in the ground points removal module. This method can reduce the input data volume of point cloud encoding networks while making the network more focused on detecting objects. Specifically, the fundamental principle underlying the ground points classification, leveraging the elevation difference algorithm, involves classifying the point cloud by calculating the distances between each current point and its neighboring points. This process aims to accurately extract the ground points, as illustrated in Figure 2.

Firstly, for calculating the elevation difference between the points, it is necessary to construct a neighborhood N_i with a radius of $\boldsymbol{R}$,taking the current

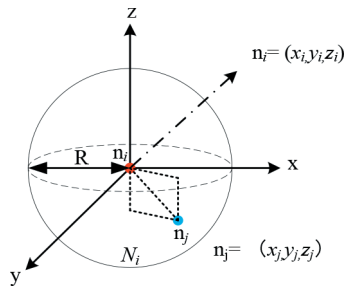

Figure 2: The schematic diagram of remove ground points based on the elevation difference.

point $n_i = (x_i, y_i, z_i)$ as the center, as shown in Eq. (1):

$$N_i = \{\mathrm{n}_j | \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq R\} \tag{1}$$

where $i \in [1, N]$, N is the number of input point cloud points, $n_j = (x_j, y_j, z_j)$ is a non-n_i point in the current neighborhood N_i.After constructing the neighborhood, the elevation difference is obtained by calculating the difference in z-values between n_i and n_j, where the maximum elevation difference is D_i, as shown in Eq. (2):

$$D_i = \max_{\mathrm{n}_j \in N_i \setminus \{i\}} |\mathrm{z}_i - \mathrm{z}_j| \tag{2}$$

Next, we set the neighborhood threshold, $h_{threshold}$. By comparing D_i with $h_{threshold}$, we can know whether it belongs to a ground point. If $D_i < h_{threshold}$, point n_i is confirmed as a ground point and removed, otherwise it is regarded as a non-ground point. Finally, by applying our method to all points, the ground points can be removed from the original point cloud $\boldsymbol{P}$. Removing ground points can reduce the amount of data and enable the network to focus on the semantic information of objects.

3.3 Data Preprocessing Module

The primary objective of the data preprocessing module is to organize the point cloud data through an octree approach. This methodology aims to mitigate the unstructured complexity inherent in the point cloud and to populate the voxel space with precise occupancy information derived from the octree structure. The use of data preprocessing method based on octree and voxel combination can enable deep entropy models to better learn the dependency relationships between adjacent nodes at the same depth, accurately predict the occupancy information of non empty voxels, and further improve compression performance. Our method for constructing the octree is shown in Figure 3.

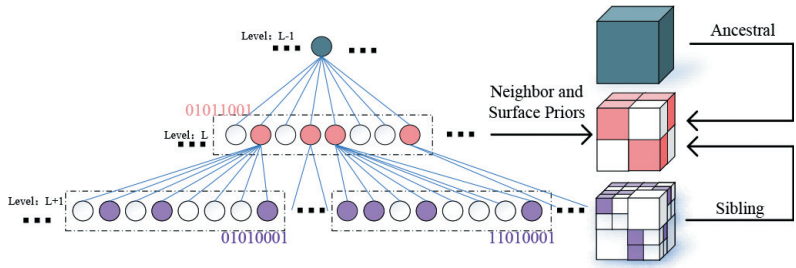

Figure 3: An example of the process of building a two-level octree.

Firstly, the three-dimensional space containing $\boldsymbol{P^*}$ is uniformly divided along the X-axis, Y-axis, and Z-axis into eight voxel spaces, ensuring that each voxel occupies the same volume proportion.For the voxel space occupied by a point, the occupancy information is set to "1" and continues to be divided into 8 sub-voxel spaces, and for the unoccupied space, the occupancy information is set to "0" and the division is stopped.

Afterward, iteratively divide each voxel space in the same way until the maximum depth level L_{max}. By adopting this approach, the point cloud can be sequentially represented as a stream of 8-bit binary occupancy symbols, such as 01011001. This process ultimately generates a series of binary voxel spaces and the corresponding stream of occupancy symbols, denoted as $\mathrm{S}, \mathrm{S} = [s_1, s_2, ..., s_i, ..., s_u]$, where u is the total number of voxel spaces, and s_i denotes the occupancy symbols corresponding to the i-th voxel.

Subsequently, for the i-th non-empty voxel space, the coordinates of its central point are adopted as the localization coordinates for that specific voxel space. The 8-bit occupancy symbol of the i-th voxel space, the octree depth L_i, the sibling index, the voxel size, the occupancy code of the parent node, and the coordinate information (x_i,y_i,z_i) are populated into the corresponding voxel space to generate the local voxel context V_i, and at the same time, generate its sibling nodes voxel context V_{sib}.The V_i's size is $9 \times 9 \times 9$ and the V_{sib}'s size is $4 \times 4 \times 4$ in our method.

3.4 Deep Entropy Model

Inputting hierarchical contexts V_i and V_{sib} into the deep entropy model allows learning the spatial characteristics of the point cloud data and predicting the probability of occurrence of occupancy symbols. The deep entropy model consists of sibling dependence prior branch, neighbor dependence prior branch, and surface prior branch. The structure of our deep entropy model network is shown in Figure 4.

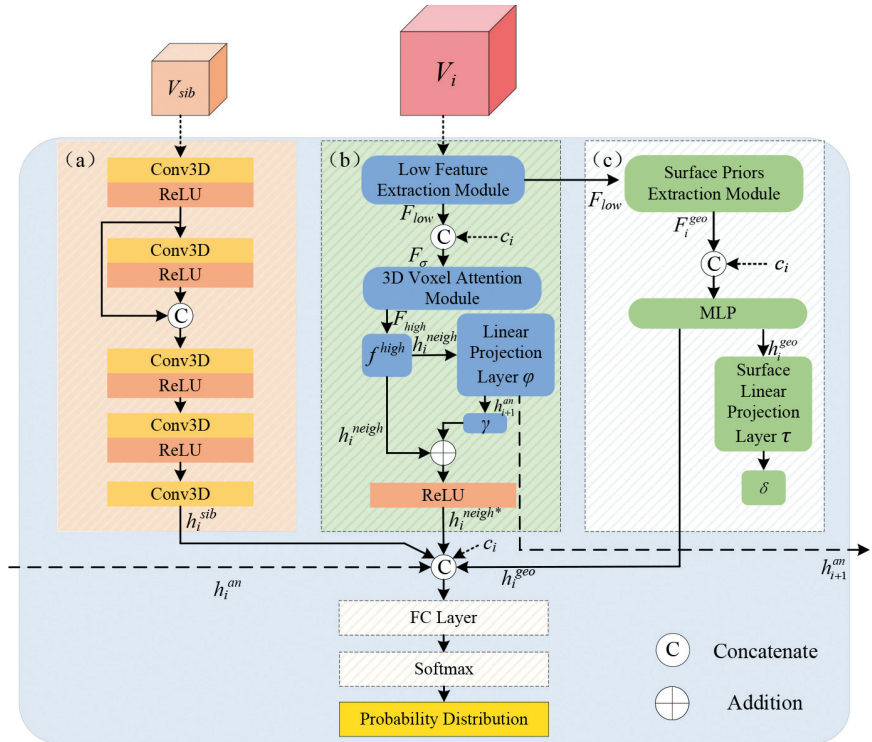

Figure 4: Our deep entropy model network. The deep entropy model consists of three feature extraction branches, namely (a)the sibling dependence prior branch, (b)the neighbor dependence prior branch and (c)the surface prior branch. Our proposed 3D Voxel Attention is in (b) neighbor dependence prior branch.

3.4.1 The neighbor dependence prior branch

In the neighbor dependence prior branch, low feature extraction is first performed on local voxel context V_i of size $9 \times 9 \times 9$ to obtain the latent features F_{low}. The Low Feature Extraction Module consists of two convolutional layers followed by ReLU activation functions. To further enhance the expression of features, F_{low} is concatenated with node information c_i to obtain the feature F_σ before passing it to the 3D Voxel Attention Module. This can supplement the node information during feature extraction and strengthen the network's perception of the voxel's spatial location, enabling the 3D Voxel Attention Module to accurately capture positional features in space.

Our proposed 3D Voxel Attention Module is shown in Figure 5. Using the average pooling function, the feature F_σ is decomposed into direction-aware feature mappings z_{c_W}, z_{c_H}, and z_{c_D} along the three spatial directions,

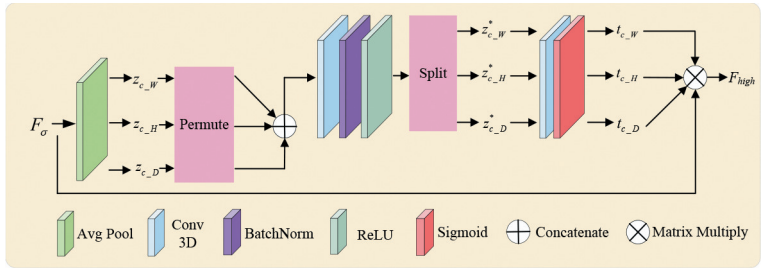

Figure 5: Network structure of 3D Voxel Attention.

respectively, as shown in Eq. (3):

$$\begin{cases} z_{c\_W} = \frac{1}{D\times H}\sum_{i=1}^{D}\sum_{j=1}^{H} F_\sigma(i,j) \\ z_{c\_H} = \frac{1}{D\times W}\sum_{i=1}^{D}\sum_{k=1}^{W} F_\sigma(i,k) \\ z_{c\_D} = \frac{1}{W\times H}\sum_{j=1}^{W}\sum_{k=1}^{H} F_\sigma(j,k) \end{cases} \tag{3}$$

where z_{c_W} refers to compressing the F_σ scale to (DH1) on the c-th channel. Similarly, z_{c_H} compresses the scale to (D × 1 × W) and z_{c_D} compresses the scale to (1 × H × W).

The transformation of (3) aggregates features along each of the three directions in 3D space, enabling the 3D Voxel Attention Module to capture long-range dependencies along one of the spatial directions while retaining positional information in the other two spatial directions, which helps the network to more accurately localize objects of interest.The resulting feature mappings z_{c_W}, z_{c_H} and z_{c_D} in different directions are then transposed and concatenated in the last dimension, and intermediate feature mappings are obtained through BatchNormal and Non-linear layers. This enables the interaction of contextual features in the three spatial directions, making the network more focused on the hierarchical context structure in the channel and space. After that, the intermediate feature mapping is split into three independent tensors $z^*_{c\_W}$, $z^*_{c\_H}$ and $z^*_{c\_D}$ along the spatial dimension. The split in Figure 5 represents the above process. And these three tensors are adjusted to feature maps t_{c_W}, t_{c_H} and t_{c_D} respectively with the same number of channels as the input F_σ using the convolutional transform as shown in Eq. (4):

$$\begin{cases} t_{c\_W} = Sigmoid(Conv3D(z^*_{c\_W})) \\ t_{c\_H} = Sigmoid(Conv3D(z^*_{c\_H})) \\ t_{c\_D} = Sigmoid(Conv3D(z^*_{c\_D})) \end{cases} \tag{4}$$

The three feature maps obtained reflect well the relationship between the object of interest and the three spatial directions, enabling the network to accurately localize the position of the non-empty voxels. Ultimately, the high-level semantic feature F_{high} obtained after the 3D Voxel Attention Module can be expressed as the input F_σ multiplied by three feature maps as shown in Eq. (5):

$$F_{high} = F_\sigma \times t_{c\_W} \times t_{c\_H} \times t_{c\_D} \tag{5}$$

Afterward, the neighbor dependence h_i^{niegh} is obtained by the feature extraction function f^{high}, as shown in Figure 4. The feature extraction function f^{high} is composed of two FC layers and a ReLu activation function. To ensure the robustness of the deep entropy model, we adopt the approach of processing the ancestor context information as a priori information as well. Specifically, since the ancestor context information h_{i+1}^{an} of the current processing node contains occupancy symbols at a shallow depth, we use a level-by-level training approach for nodes at each level of octree depth in the deep entropy model, where h_i^{niegh} is processed as h_{i+1}^{an} and passed to the next level. Based on this, a linear layer $[\gamma]$ is applied to reduce the features to their original dimensions, and then the features are fused with h_i^{niegh} through a skip connection to recover the original neighbor context feature h_i^{niegh*} of the current node, denoted as Eq. (6):

$$h_i^{neigh*} = \mathrm{ReLU}[\gamma(h_{i+1}^{an}) + h_i^{neigh}] \tag{6}$$

where h_i^{niegh*} is the original neighbor dependence of the current node, h_{i+1}^{an} is the ancestor dependence passed from the current depth node to the next depth node, γ is a linear layer that reduces the feature dimensions, and ReLU is the activation function, which ultimately yields the outputs h_i^{niegh*} and h_{i+1}^{an} of the neighbor dependence prior branch.

3.4.2 The surface prior branch

During the actual acquisition of point cloud data utilizing LiDAR technology, a vast quantity of discrete point data pertaining to the surface of the target object can be captured by transmitting a laser beam towards the object and subsequently receiving the reflected signal. Given that the surface of the object typically exhibits a complex curved structure, the point cloud data inherently contains a substantial amount of curved surface information. Consequently, the incorporation of surface priors can significantly enhance the learning of point cloud data features.

Specifically, the input to the surface prior branch is the latent feature F_{low}. The F_{low} is fed into the Surface Prior Extraction Module to extract the surface prior feature F_i^{geo}. Next, the surface prior dependence h_i^{geo} is extracted using

MLP after concatenating F_i^{geo} with the node information c_i, as shown in Eq. (7):

$$h_i^{geo} = MLP([F_i^{geo}, c_i]) \tag{7}$$

The Surface Linear Projection Layer τ is also utilized to extract the quadratic surface parameter $\delta = [x^2, y^2, xy, x, y, 1] \in R^6$ from the surface prior dependence h_i^{geo}, as shown in Eq. (8):

$$\delta = \tau(h_i^{geo}) \tag{8}$$

3.4.3 The sibling dependence prior branch

To fully learn the specific local structure and features of the octree subspace, we refer to PCC-S which introduces sibling nodes as a complement to a priori information when constructing context information. Assuming that an occupied leaf node n is located at the octree $\mathrm{L}-1$ level depth, and continuing to octree it until $\mathrm{L}+1$ depth, the three-dimensional space represented by this node n is then divided into $4 \times 4 \times 4$ subspaces, which is the input V_{sib} to obtain the sibling dependence h^{sib}, which is further provided to the deep entropy model as prior information.

3.5 Loss function

The total loss function we use contains the cross-entropy loss and the surface loss, as shown in Eq. (9):

$$L = L_{CE} + \lambda L_{sf} \tag{9}$$

where L_{CE} is the cross-entropy loss function, L_{sf} is the surface loss function, and λ is the weight of the surface loss.

Since the output of the deep entropy model is a probability distribution, we preprocess the input point cloud to compute the probability distribution of the symbols occupied by the original point cloud as the ground truth. Minimizing the difference between the model's predictions and the ground truth by minimizing the cross-entropy loss can make the model's predictions as close as possible to the original point cloud. Defining $\mathrm{q}(s_i)$ as the probability distribution of the occupied symbol s_i and $\mathrm{p}(s_i)$ as the true probability distribution of the ground truth, the cross-entropy loss function L_{CE} and the predicted probability of the occupied symbol s_i are shown in Eq. (10) and Eq. (11):

$$L_{CE} = -\sum_i p(s_i) \log q(s_i) \tag{10}$$

$$q(s) = \prod_i q(s_i | h_i^{an}, h_i^{neigh*}, h_i^{sib}, h_i^{geo}, c_i; \theta) \tag{11}$$

where h_i^{an}, h_i^{neigh*}, h_i^{sib}, h_i^{geo}, c_i and θ respectively refer to ancestor dependence, neighbor dependence, siblings's dependence, surface prior dependence, node information, and entropy model parameters. After that, a quadratic surface Z is fitted using the original point cloud, which constitutes the surface loss function L_{sf} by minimizing the minimum distance from each point to the surface, as shown in Eq. (12):

$$L_{sf} = ||Z - \delta^T||_2^2 \tag{12}$$

where δ is the quadratic surface parameter learned in the surface prior branch, see Eq. (8).

4 Experiments

4.1 Experimental Setup

Both our point cloud encoding network and object detection network are trained using the KITTI 3D object detection dataset. We use Pointpillar [11] as an object detection network. In the training of the object detection network, the samples are initially divided into 7481 training samples and 7518 test samples, and 80 rounds of training are performed using the Adam optimizer with a learning rate of 2e-4, saving the last model as the one used in the experiments.In the training of the deep entropy model, we randomly selected 2000 groups of point clouds in the training samples and performed 20 rounds of training using the Adam optimizer with a learning rate of 1e-4. In the testing phase, 550 sets of point clouds from the object detection test set are extracted, and the coding network is used to compress and reconstruct these 550 sets of point clouds, and the final object detection test is performed on these 550 sets of reconstructed point clouds. In addition, we conducted experiments on the Waymo Open Dataset [25] to further determine the advantages of OSC-Net. We use PyTorch [20] to implement all our models and train them on Nvidia Quadro RTX 8000 GPU.

4.2 Evaluation Metrics

Since we have adapted the raw point cloud data to fit the point cloud 3D object detection task, we no longer use the peak signal-to-noise ratio (D1-PSNR, D2-PSNR) and Chamfer Distance to evaluate the reconstructed point cloud, and instead use the coding code rate to evaluate the rate accuracy performance compared to the accuracy of the object detection task. Bpp stands for Bits per point and is the most commonly used metric for evaluating the compression performance of point clouds. Since we only consider the geometric compression of the point cloud in this section, the size of the original point cloud data is

calculated as $96 \times N$, where N is the number of points and 96 is the size of the coordinates x, y, and z, where each coordinate is represented as a 32-bit floating-point number. Bpp is defined as Eq. (13):

$$BPP = |bit| / N \tag{13}$$

where $|bit|$ is the total bits.In point cloud object detection, we use average precision AP as an evaluation metric. The average precision AP is defined as the area under the curve of precision and recall (P-R curve) as shown in Eq. (14):

$$AP = \sum_{r=0}^{1} (r_{n+1} - r_n) \rho_{interp}(r_{n+1}) \tag{14}$$

where the precision on each recall r is interpolated by finding the maximum value of the precision on that recall with the precision on its right-hand side recall, as shown in Eq. (15):

$$\rho_{interp}(r_{n+1}) = \max_{\tilde{r}:\tilde{r} \geq r_{n+1}} \rho(\tilde{r}) \tag{15}$$

where $\rho(\tilde{r})$ is the object detection precision at recall.

4.3 Performance Evaluation

To evaluate the effectiveness of the proposed methods, we report the performance of the proposed methods in an object detection task. The comparison methods for quan-titative evaluation are G-PCC, Draco, Voxelcontext-Net, and PCC-S. For qualitative evaluation, the ground points classified by the ground points removal module and the point cloud after removing the ground points are first visualized. Secondly, the object detection results of reconstructed point clouds for PCC-S and OSC-Net are visualized. Finally, the results of object detection for reconstructed point clouds with different octree depths are visualized.

4.3.1 Quantitative Assessment

To address the issue of how $h_{threshold}$ affects object detection performance in different fields, we conducted quantitative analysis experiments on the KITTI dataset by setting the octree depth to 12. By adjusting the $h_{threshold}$ and observing its impact on detection performance and Bpp, the results are shown in Table 1. According to the Table 1, an excessively large threshold results in a notable decline in object detection performance. The reason behind this is that a highly elevated domain $h_{threshold}$ prompts the ground point removal module to incorrectly classify points within objects as ground points,

Table 1: The impact of different $h_{threshold}$ on detection performance and Bpp.

$h_{threshold}$	0.5	0.3	0.21	0.2	0.19	0.1
Car bbox AP	56.86%	74.31%	88.19%	89.65%	89.67%	89.71%
Bpp	2.135	2.730	3.187	3.230	3.319	3.811

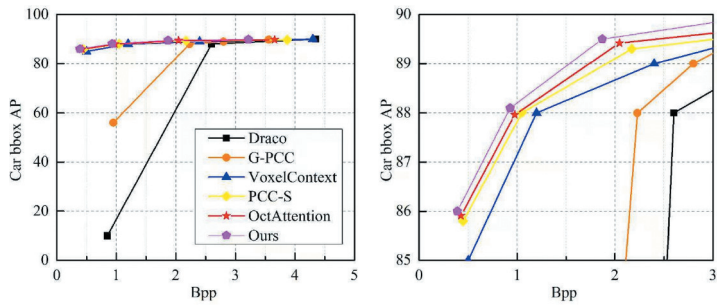

Figure 6: Comparison chart of rate accuracy performance between OSC-Net and comparison methods on KITTI. Fig.6 (b) is an enlarged view of the local details in Fig.6 (a).

ultimately undermining detection accuracy. Conversely, when a relatively low $h_{threshold}$ value is chosen, there is no marked enhancement in object detection performance, yet a substantial number of ground points still remain to be removed, causing an increase in bitrate. After conducting numerous experiments, we ultimately settled on a domain $h_{threshold}$ of 0.2. This choice enables the network to prioritize object semantics while minimizing the bitrate to the fullest extent feasible.

To quantitatively evaluate the performance of our method, we report a comparison of our method on KITTI with other methods at the same octree depth, as shown in Figure 6. We also compare OSC-Net with other methods on the Waymo Open dataset. We report 3D mean Average Precision (mAP)for vehicle detection on the LEVEL 1, as shown in Figure 7. On these two datasets, OSC-Net achieved lower encoding Bpp compared to other methods while maintaining higher detection accuracy. With the same octree depth, OSC-Net reconstructs the point cloud with lower Bpp and superior object detection performance.

4.3.2 Qualitative Assessment

The work of the ground points removal module is visualized in Figure 8 to visually demonstrate the effectiveness of the proposed algorithm.

Figure 9 visualizes the results of different methods for object detection in reconstructed point clouds. We randomly selected two groups of point clouds,

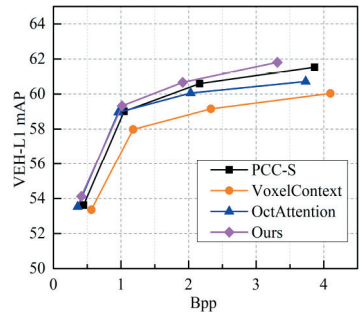

Figure 7: Comparison chart of rate accuracy performance between OSC-Net and comparison methods on the Waymo Open dataset.

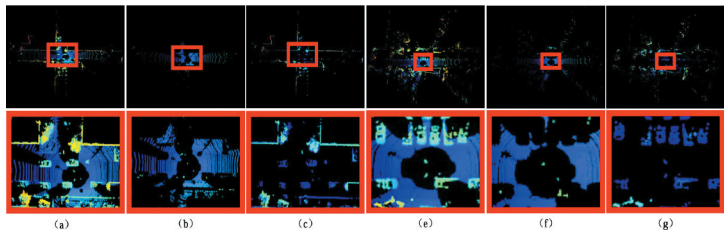

Figure 8: Visualization of ground truth, ground points data, and point cloud removed ground points data. (a) and (e) is ground truth data, (b) and (f) are ground points classified based on difference of elevation, and (c) and (g) are point clouds after removing ground points.

'000031.bin' and '000006.bin', and used different methods to codify and decode them before performing the object detection task. The reconstructed point cloud of OSC-Net achieves close performance to the original point cloud, and the detection results of the proposed method are more accurate compared with PCC-S.

To verify the detection effect of the proposed method at different Bpp, Figure 10 visualizes the reconstructed point clouds of '000006.bin' (left panel) and '000981.bin' (right panel) for two sets of point clouds at different octree depths object detection results.

4.3.3 Ablation Experiment

As shown in Table 2, we conducted three octree-depth ablation studies, thus confirming the effectiveness of the use or omission of our 3D voxel attention module and de-ground points module in terms of coding effectiveness and detection performance. As can be observed from the table, the Bpp is highest

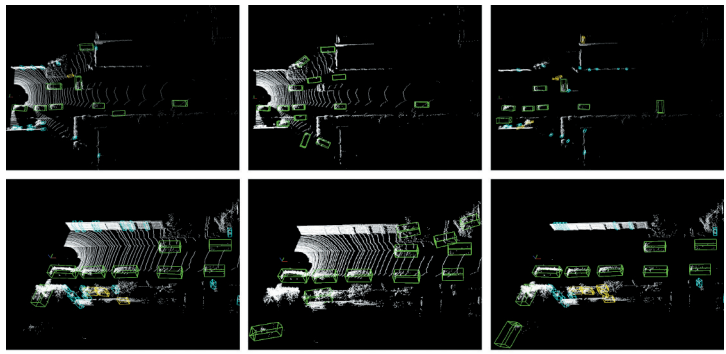

Figure 9: Visualization of ground truth and reconstruction point cloud object detection results. Column (a) represents the result of object detection using ground truth data, column (b) represents the result of object detection using point cloud data decoded and reconstructed by PCC-S network at octree depth of 12, and column (c) represents the result of object detection using point cloud data decoded and reconstructed by OSC-Net at octree depth of 12.

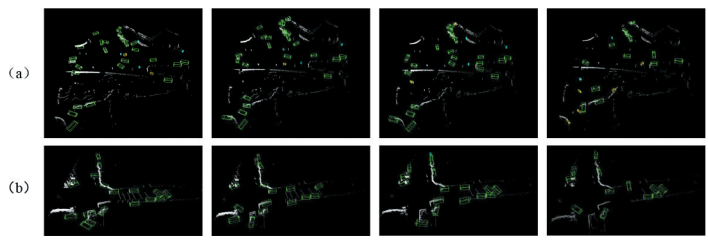

Figure 10: Visualization of reconstructed point cloud object detection results, displayed from left to right, at various octree levels corresponding to octree depths of 12 to 9, with average bits per point (Bpp) of 3.230, 1.875, 0.930, and 0.393, respectively. (a) is "000006" in the dataset and (b) is "000981".

when both modules are not used, and the simultaneous use of both modules can save more code rate while maintaining the accuracy of the object detection task.

5 Conclusion

We propose OSC-Net, an object semantic-aware compression network for 3D point cloud. Firstly, by leveraging the elevation difference, we identify and eliminate ground points within the original point cloud that fall outside the region of interest for the object detection. Simultaneously, we reduce data redundancy so that the network can focus on object semantic information.

Table 2: Results of ablation experiments (Bpp/bbox AP).

3D Voxel Attention	Ground Points Removal	Bpp/AP		
		Level 10	Level 11	Level 12
✗	✗	1.129/87.86%	2.143/89.19%	3.864/89.51%
✓	✗	1.006/88.03%	2.034/89.37%	3.804/89.71%
✗	✓	0.983/87.02%	1.963/88.52%	3.415/88.21%
✓	✓	0.930/87.97%	1.875/88.90%	3.230/89.36%

Secondly, benefiting from utilizing the 3D voxel attention module, hierarchical voxel context is extracted from raw point cloud data, capturing both channel and spatial features. Our proposed approach empowers the deep entropy model to anticipate a significantly more precise probability distribution of occupancy symbols, enhancing its predictive capabilities. Experimental results show that our method outperforms previous methods in terms of compression performance and object detection accuracy. It is hoped that our work will inspire researchers to further combine the coding with the object detection in future work to promote the application of point cloud coding technology in autonomous driving.

Acknowledgements

This work has been supported in part by the National Natural Science Foundation of China (62072325, U23A20314), Industrial Vision Application of Shanxi Provincial Technology Innovation Center (IVA-SXTIC2022), Shanxi Key Core Technology & Common Technology Research and Development Project (20201102011), Shanxi S&T Major Project (20191102010), Shanxi University S&T Achievements Transformation Cultivation Project (20191042), Shanxi S&T Achievements Transformation Project (201804D131035).

References

[1] J.-K. Ahn, K.-Y. Lee, J.-Y. Sim, and C.-S. Kim, "Large-scale 3D point cloud compression using adaptive radial distance prediction in hybrid coordinate domains", *IEEE Journal of Selected Topics in Signal Processing*, 9(3), 2014, 422–34.

[2] Z. Chen, Z. Qian, S. Wang, and Q. Chen, "Point cloud compression with sibling context and surface priors", in *European Conference on Computer Vision*, Springer, 2022, 744–59.

[3] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, 3075–84.

[4] O. Devillers and P.-M. Gandoin, "Geometric compression for interactive transmission", in *Proceedings Visualization 2000. VIS 2000 (Cat. No. 00CH37145)*, IEEE, 2000, 319–26.

[5] O. Dovrat, I. Lang, and S. Avidan, "Learning to sample", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, 2760–9.

[6] C. Fu, G. Li, R. Song, W. Gao, and S. Liu, "OctAttention: Octree-Based Large-Scale Contexts Model for Point Cloud Compression", 2022.

[7] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite", in *2012 IEEE conference on computer vision and pattern recognition*, IEEE, 2012, 3354–61.

[8] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai, "An overview of ongoing point cloud compression standardization activities: Video-based (V-PCC) and geometry-based (G-PCC)", *APSIPA Transactions on Signal and Information Processing*, 9, 2020, e13.

[9] L. Huang, S. Wang, K. Wong, J. Liu, and R. Urtasun, "Octsqueeze: Octree-structured entropy model for lidar compression", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, 1313–23.

[10] T. Huang and Y. Liu, "3d point cloud geometry compression on deep learning", in *Proceedings of the 27th ACM international conference on multimedia*, 2019, 890–8.

[11] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, 12697–705.

[12] M. Li, Z. Jinming, M. Dongchao, and F. Yingxun, "An Improved RANSAC Surface Reconstruction Study", in *Journal of Physics: Conference Series*, Vol. 1284, No. 1, IOP Publishing, 2019, 012020.

[13] Z. Li, C. Tian, H. Yuan, X. Lu, and H. Malekmohamadi, "3D-MSFC: A 3D multi-scale features compression method for object detection", *Displays*, 85, 2024, 102880.

[14] Y. Lin, Y. Huang, S. Zhou, M. Jiang, T. Wang, and Y. Lei, "DA-Net: density-adaptive downsampling network for point cloud classification via end-to-end learning", in *2021 4th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)*, IEEE, 2021, 13–8.

[15] B. Liu, N. Shangguan, K. Jiang, and J. Lin, "Triangular mesh model reconstruction from scan point clouds based on template", *Tsinghua Science and Technology*, 14(S1), 2009, 56–61.

[16] B. Liu, S. Li, X. Sheng, L. Li, and D. Liu, "Joint Optimized Point Cloud Compression for 3d Object Detection", in *2023 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2023, 1185–9.

[17] D. Meagher, "Geometric modeling using octree encoding", *Computer graphics and image processing*, 19(2), 1982, 129–47.

[18] D. T. Nguyen, M. Quach, G. Valenzise, and P. Duhamel, "Learning-based lossless compression of 3d point cloud geometry", in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, 4220–4.

[19] D. T. Nguyen, M. Quach, G. Valenzise, and P. Duhamel, "Multiscale deep context modeling for lossless point cloud geometry compression", in *2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, IEEE, 2021, 1–6.

[20] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library", *Advances in neural information processing systems*, 32, 2019.

[21] Y. Qian, J. Hou, Q. Zhang, Y. Zeng, S. Kwong, and Y. He, "Task-Oriented Compact Representation of 3D Point Clouds via A Matrix Optimization-Driven Network", *IEEE Transactions on Circuits and Systems for Video Technology*, 33(11), 2023, 6981–95.

[22] Z. Que, G. Lu, and D. Xu, "Voxelcontext-net: An octree based framework for point cloud compression", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, 6042–51.

[23] R. Song, C. Fu, S. Liu, and G. Li, "Efficient Hierarchical Entropy Model for Learned Point Cloud Compression", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, 14368–77.

[24] C. Sun, H. Yuan, X. Mao, X. Lu, and R. Hamzaoui, "Enhancing octree-based context models for point cloud geometry compression with attention-based child node number prediction", *IEEE Signal Processing Letters*, 2024.

[25] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, and B. Caine, "Scalability in Perception for Autonomous Driving: Waymo Open Dataset", 2019.

[26] X. Sun, H. Ma, Y. Sun, and M. Liu, "A novel point cloud compression algorithm based on clustering", *IEEE Robotics and Automation Letters*, 4(2), 2019, 2132–9.

[27] X. Sun, S. Wang, and M. Liu, "A novel coding architecture for multi-line LiDAR point clouds based on clustering and convolutional LSTM network", *IEEE Transactions on Intelligent Transportation Systems*, 23(3), 2020, 2190–201.

[28] C. Tu, E. Takeuchi, A. Carballo, and K. Takeda, "Point cloud compression for 3d lidar sensor using recurrent neural network with residual blocks", in *2019 international conference on robotics and automation (ICRA)*, IEEE, 2019, 3274–80.

[29] M. Ulhaq and I. V. Bajić, "Learned point cloud compression for classification", in *2023 IEEE 25th International Workshop on Multimedia Signal Processing (MMSP)*, IEEE, 2023, 1–6.

[30] Y. Wang, Y. Wang, T. Cui, and Z. Fang, "Occupancy map-based low complexity motion prediction for video-based point cloud compression", *Journal of Visual Communication and Image Representation*, 100, 2024, 104110.

[31] Z. Wang, C. Li, J. Ma, Z. Feng, and L. Xiao, "PVI-Net: Point–Voxel–Image Fusion for Semantic Segmentation of Point Clouds in Large-Scale Autonomous Driving Scenarios", *Information*, 15(3), 2024, 148.

[32] W. Yan, S. Liu, T. H. Li, Z. Li, G. Li, *et al.*, "Deep autoencoder-based lossy geometry compression for point clouds", *arXiv preprint arXiv:1905.03691*, 2019.

[33] H. Yuan, D. Zhang, W. Wang, and Y. Li, "A sampling-based 3D point cloud compression algorithm for immersive communication", *Mobile Networks and Applications*, 25(5), 2020, 1863–72.

[34] Q. Zhang, J. Hou, and Y. Qian, "Pointmcd: Boosting deep point cloud encoders via multi-view cross-modal distillation for 3d shape recognition", *IEEE Transactions on Multimedia*, 2023.