**LighTN: Light-Weight Transformer Network for Performance-Overhead Tradeoff in Point Cloud Downsampling**

Xu Wang, Yi Jin, Yigang Cen, Member, IEEE, Tao Wang, Bowen Tang, and Yidong Li, Senior Member, IEEE

**Abstract**—Downsampling is a crucial task for processing large scale and/or dense point clouds with limited resources. Owing to the development of deep learning, approaches of task-oriented point cloud downsampling have significant performance gains in preserving geometric information. However, most downsampling methods are limited by the disordered and unstructured point cloud data, making it difficult to continually improve the performance. To address this issue, we propose a light-weight Transformer network (LighTN) for the task-oriented point cloud downsampling as an end-to-end solution. In LighTN, we design an energy-efficient and permutation invariant single-head self-correlation module to extract refined global geometric features. Moreover, we present a novel sampling loss function to guide LighTN to focus on critical point cloud regions with more uniform distributions and prominent point coverage. Extensive experiments on classification, registration, and reconstruction tasks demonstrate that LighTN can achieve the state-of-the-art performance-overhead tradeoff and high-quality qualitative results.

**Index Terms**—Point cloud, deep learning, downsampling, transformer, energy-efficient.

I. INTRODUCTION

POINT clouds are the most fundamental and popular representation of geometric objects in the perception of 3D visual scenes [1], [2], [3], [4], [5]. When scanning a 3D scene with rich details in the real world, it is common to acquire dense points on the outer surface of the objects. However, it is challenging to process large-scale and/or dense point clouds due to the restrictions of low power devices or terminals with limited resource overhead. Meanwhile, the capacity of a point set is not always proportional to data quality and recognition effect. Therefore, point cloud downsampling techniques are proposed to output a robust and concise point set from the raw data.

For common shape analysis and geometry processing tasks, the traditional two-stage downsampling approaches, such as the farthest point sampling (FPS) [6] and the random sampling (RS) [7], have been widely adopted. These approaches selected critical points based on low-level information without considering deep semantic features and downstream tasks. Subsequently, the downsampling process is usually repeated to obtain the desired output precision, which may outweigh the resource savings brought by the simplified point set, as shown in the top part of Fig. 1.

Recently, many deep learning-based methods have been employed to identify critical points by embedding the point cloud into high-dimensional feature space. Existing deep downsampling approaches can be classified into two categories: 1) specific downsampling layer [8], [9], [10], and 2) general downsampling module [11], [12], [13]. All these methods can simplify the raw point cloud, but each has its drawbacks. In detail, the specific downsampling layer technologies do not guarantee generality to other tasks. Besides, the embedded requirement is unfriendly...
to priori defined network architectures and pre-trained parameters. There are two reasons: 1) any slight variation in pre-defined architectures may result in reduced performance, and 2) the networks with a complex framework and a high accuracy will cause expensive resources to retrain. To tackle aforementioned issues, some researchers proposed to build a general downsampling module independent of downstream task networks. However, almost existing general downsampling modules adopt the low resource consumption PointNet-like architectures as the backbone to individually process each point, neglecting the correlations and relationships among the points.

Motivated by the success of the Transformer architectures, it is hoped to leverage the Transformer-based approaches to improve the learning ability of the downsampling backbone. Our insight is that the Transformer shows greater strength in dealing with disordered and unstructured point cloud data and modeling the relationship among the points than PointNet-like frameworks. The potential Transformer-based downsampling frameworks are outlined in Fig. 1. To improve the performance, existing Transformers often stack more transformer blocks and introduce Convolutional Neural Network for local feature extraction. However, such scaling comes at the cost of more resources which may counteract the resource-saving brought by downsampling. Therefore, this article presents a novel general and light-weight Transformer network, named LighTN, for task-oriented point cloud downsampling. Compared with the existing complex Transformer family of networks, LighTN is more concise and low-overhead, and shows powerful task-oriented downsampling processing capabilities.

Specifically, the backbone of the LighTN is based on the Transformer network, mainly consisting of a single Transformer block and single-head self-correlation attention. For the self-correlation mechanism, we remove matrices projection of Query (Q), Key (K) and Value (V). Potentially, this removal is more suitable for extracting refined global geometric features of points because its internal symmetry matrix satisfies the permutation invariance. Then, to alleviate the decline of learnable parameters caused by the light-weight framework, we introduce the expand-reduce strategy to adjust the depth and width of the FFN layer (called scalable FFN). In this way, LighTN can learn sufficient geometric information and the increase in downsampling overhead is much lower than the resource savings brought by the simplified point set. Finally, we design a novel sampling loss to increase the distance between generated points and to make them cover more prominent regions. Consequently, LighTN can be developed to achieve optimal critical points extraction and accurate visualization with an efficient performance-overhead tradeoff.

The key contributions of this work are summarized as follows:

- We propose a light-weight Transformer framework, named LighTN, as the task-oriented and end-to-end solution to simplify the large-scale and/or dense point cloud.
- We design a novel sampling loss function to promote the more uniform distribution and prominent points coverage of sampled point clouds.
- Extensive experiments on classification, registration, and reconstruction tasks demonstrate that LighTN achieves state-of-the-art performance-overhead tradeoff and high-quality qualitative results.

The structure of this article is as follows. In Section II, we introduce overviews of deep learning on point clouds and downsampling methods. Then, Section III describes the architecture of LighTN both in outline and in detail. In Section IV, we validate the performance of LighTN in classification, registration, and reconstruction tasks using standard benchmark datasets and compare the results with state-of-the-art models. Finally, the conclusion is presented in Section V.

II. RELATED WORK

A. Deep Learning for 3D Point Clouds

Unlike structured 2D images, the point cloud comprises three-dimensional coordinates of unordered and irregular points, directly rendering the powerful 2D convolutional network unusable. Several types of research mainly focus on transforming 3D point clouds into regular representation in terms of multi-views [14], [15], [16] and voxel grids [17], [18], [19] to extract high-dimensional features. Zhou et al. [14] proposed a multi-view descriptor, MVDesc, to learn local features from each patch of view. They collect images from 3 fixed views for each object to enhance the efficiency. In order to discard task-independent fixed viewpoints, Li et al. [15] present a 2D convolution framework based on variable views to extract local view-based features. Their experiments showed that the saturated performance could be obtained with eight viewpoints. In contrast with the multi-view method based on 2D convolution, voxel grid approaches use 3D convolution to achieve specific tasks. Maturana et al. [20] voxelized the point cloud objects by occupancy grids, and then leveraged the standard 3D convolutional neural network to extract features from the raw volumetric data. Furthermore, Zhou et al. [17] utilize vertical column voxelization to improve the computational efficiency. Although regular representation methods have been widely adopted, they destroy the spatial structure of point clouds with inherent permutation invariants.

Opposite to the regular representation of point clouds, the PointNet [21] is the pioneering work that directly operates on points by using a symmetric function to ensure permutation invariant characteristics. However, limited by the learning capacity of PointNet, there is still room to improve the performance of feature extraction. For example, some methods [22], [23], [24], [25] have been proposed to extend PointNet by combining local-global geometric information for higher performance. Recently, the Transformer family of networks has shown more powerful learning capacity in visual tasks, especially the features, which will contain refined global context information after going through the self-attention mechanism [26], [27]. The work of three early research [28], [29], [30] demonstrated that the Transformer-based approaches possess the inherently permutation invariant for point cloud data. In the released work [28], Guo et al. propose a local-global Transformer, named PCT, with 2 layer local neighbor embedding [31] and 4 stacked offset-attention blocks. It should be pointed out that compared with PointNet, the classification accuracy of PCT on...
the ModelNet40 dataset is increased by 4%, but the computational cost is increased by more than 5 times. Besides, the work of Point Transformer [29] exposed that the network size of Point Transformer (51 MB) is almost 5.4 times than that of PointNet (9.4 MB) after an improvement of 3.6% in the classification accuracy. Unfortunately, the above Transformer-based architectures potentially consume too many resources, usually rendering the Transformer worthless for low overhead task networks in the downsampling range, such as PointNet. The recent work in [32] addresses the problem of model size for Transformer by designing singe-head attention and light-weight FFN in machine translation and language modeling tasks. Inspired by this work, we present a light-weight Transformer that captures the refined global geometric information with limited computation load and storage space. The details can be seen in Section III-C.

B. Task-Irrelevant Downsampling Methods

In the early works, non-learned predetermined point cloud downsampling approaches have been widely adopted. For example, FPS is an important downsampling technology frequently used in many point-based networks, e.g., PointNet++ [22], PointCNN [33], 41-spec-cp [34] and RS-CNN [35]. RS is also a crucial algorithm with excellent computational efficiency to process large-scale point clouds in deep learning scope, including the works of VoxelNet [36], MeteorNet [37], RandLA-Net [38] and P2B [39]. Besides, voxel-based simplification [40] has received attention. In such a simplification strategy, the point cloud is first transformed into a voxel box-based organization and then the traditional sampling algorithms (i.e., FPS and RS) are used to sample the points in each box. However, non-learned downsampling methods are task-irrelevant, which cannot optimize the data distribution of sampled points during model training. Therefore, the output precision of the following task network rapidly decreases when the downsampling ratio continually increases.

Besides non-learned methods, deep learning-based downsampling was also applied to several works. For example, Li et al. [41] propose a distinctive-guided downsampling network to enhance the sampling density of points in distinctive regions. In this work, they train the deep neural network to learn and detect distinctive regions in an unsupervised manner. To exhibit the visual recognizability similar to the originals, Li et al. [42] encode the structural information into the downsampled points. Some other works [8], [9], [10] develop downsampling layer that can be used along with specific neural networks to form end-to-end framework. Nevertheless, these approaches have poor compatibility with other task models.

C. Task-Oriented Deep Downsampling Methods

The concept of task-oriented downsampling is to design a task-driven sampler, which can be integrated with pre-trained task models to search sparse points while maintaining the task performance as much as possible. Noticeably, the task-oriented downsampling is an emerging field with only a few works as exemplary. The pioneering research of Dovrat et al. [11] presented S-NET, the first task-oriented point cloud downsampling network. Following this work, Lang et al. [12] propose a soft projection strategy to alleviate the bias that the generated points can not be guaranteed to be a proper subset of the input point cloud. Qian et al. [13] proposed a matrix optimization-driven network, MOPS-Net, to downsample the point cloud. Different from the above two works, MOPS-Net leverages the shared MLP to supplement the lack of point-wise local information. Lin et al. [43] combine the K-nearest neighboring algorithm with Local Adjustment (LA), allowing the sampled points to have noise immunity characteristics. However, these existing methods individually process each point, neglecting the correlations and relationships among the points. As a result, the performance gain is restricted. Recently, Wang et al. [44] pioneer a Transformer-based downsampling network PST-NET combined with local-global context information. Although the PST-NET acquired competitive accuracy, the complex structure reduces the expectation of resource saving. In this article, we design a light-weight Transformer network named LighTN, with favorable FLOPs and Parameters budgets.

III. METHOD

In point cloud downsampling scenarios, the goal is to preserve geometric information as much as possible for low performance degradation and accurate visualization. This article proposes a new method named LighTN as an end-to-end and task-oriented solution to achieve this goal. In this section, we firstly formulate the problem settings and then present our approach in detail. The overall framework of LighTN is presented in Fig. 2.

A. Problem Settings

Given an original point cloud \( P = \{ p_i \in \mathbb{R}^3, i = 1, 2, \ldots, N \} \) with \( N \) points, the target is to obtain a downsampled point cloud \( B = \{ b_j \in \mathbb{R}^3, j = 1, 2, \ldots, M \} \) with \( M \) points \( (M \ll N) \) that contains the optimal subset distribution with rich geometric information, where \( f \) represents features except 3D Cartesian coordinates. The mathematical expression of the above objective function can be expressed as follows:

\[
\arg \min L_{task}(F(B)), \quad B \subset P, \quad M \ll N,
\]

where \( F(\cdot) \) represents the downstream task network and \( L_{task} \) indicates the task loss function. In this definition, the evaluating indicator of the presented LighTN module becomes explicit, i.e. the minimal performance loss of a specific downstream task. Unfortunately, although many deep-based methods have been proposed to reduce the geometric information loss after downsampling, three critical issues still simultaneously hinder the performance improvement: 1) differentiable sampling: task-oriented downsampling networks require the end-to-end optimization during the training and testing phase; 2) learnable capacity: light-weight deep models must guarantee sufficient output accuracies with a limited number of nodes and layers; and 3) resource-efficiency: the resource overhead of downsampling networks must be less than the resource saved by the downsampled points through task models.

Differentiable sampling: The set of critical points generated by the task-oriented downsampling modules is not guaranteed
to be a proper subset of the original point cloud. Under this observation, the generated point sets will inevitably lose geometric information and cause visualization bias. In some studies, the additional matching operations, for example, the nearest neighbor search, are introduced to map each generated point to its nearest neighbor in the original point cloud. Nevertheless, the matching step restricts further performance improvement in critical point extraction since it is non-differentiable. Therefore, it is necessary to evolve the matching algorithm to achieve the differentiable relaxation. Ideally, the critical points generated by LighTN are the proper subset of point cloud \( P \), that is, each point \( b_j \) in \( B \) can find the corresponding point \( p_i \) in \( P \) to make the distance function between them infinitely close to 0. This purpose can be denoted as:

$$\forall b_j \in B, \exists p_i \in P \text{ s.t. } \text{dist}(b_j, p_i) \to 0. \tag{2}$$

where \( \text{dist}(b_j, p_i) = \| b_j - p_i \|_2 \).

Learnable capacity: Note that the widespread network frameworks such as MLP, CNN and Transformer have distinctive behaviors and complementary properties [45]. Usually, an effective combination of different frameworks can improve the learning capacity, but the resource overhead is also superimposed. Moreover, when the overhead of the task network is small, the downsampling modules are more limited by the network scale. On the other hand, the lightweight deep framework can save the resource overhead, but it remains challenging for the framework to learn sufficient relationships between points with the limited network scale.

To tackle aforementioned issues, this article will explore a novel network framework with performance-overhead tradeoffs.

Mathematically, we transform the learnable capacity into an output performance problem to meet the following formula:

$$\left\{ \min |\text{PERF}(P) - \text{PERF}(B)| \right. \over \max(N - M) \tag{3}$$

where \( \text{PERF}(\cdot) \) represents the output performance of downstream task networks, such as classification accuracy and mean rotation error. See ablation experiments for detailed results.

Resource-efficiency: The traditional Transformers typically stack multiple Transformer blocks containing multi-head attention mechanism to improve the model performance. In detail, the multi-head attention consists of multiple single-head attentions running in parallel, and each head contains a large number of matrix multiplication operations and weight parameters. For example, the computation of single-head Scaled Dot-Product self-attention can be defined as:

$$\text{SA}(P) = \text{FC}_{out}(\text{Atten}(\text{FC}_Q(P), \text{FC}_K(P), \text{FC}_V(P))), \tag{4}$$

$$\text{Atten}(Q, K, V) = \text{softmax} \left( \frac{Q \cdot K^T}{\sqrt{D/a}} \right) \cdot V \tag{5}$$

where \( \text{FC}(\cdot) \) represents the linear transformation through projection matrices, \( \text{softmax}(\cdot) \) is the activation function, scale \( Q \cdot K^T \) by \( 1/\sqrt{D/a} \) to improve network stability and \( D \) is the dimension of \( Q \) and \( K \) vectors. Note that \( a \) is the scaled factor for maintaining the computational cost of multi-head attention, similar to that of single-head mechanism with full dimensionality \( D \). Formally, the computation cost of single \( SA \) is \( O(4ND^2 + 2N^2D) \), where \( a \) is not considered here. Therefore,
the computation cost will be $m \cdot O(4ND^2 + 2N^2D)$ stands for stacking $m$ Transformer blocks. And the storage overhead increases with the number of heads.

Ideally, a task network with the downsampling design can effectively save resource overhead. Theoretically, we divide the resource overhead into computation load $C$ and storage space $S$ to meet the following formula:

$$\begin{align*}
C_{\text{task}}(B) + C_{\text{task}}(P) < C_{\text{task}}(P) \\
S_{\text{task}}(P) + S_{\text{task}}(P) < S_{\text{task}}(P)
\end{align*}$$

where $C_{\text{task}}(B)$ and $C_{\text{task}}(P)$ denote the computation load of the task network operating on downsampled point cloud $B$ and original point cloud $P$, and $C_{\text{sampling}}(P)$ means the computation cost of downsampling. Similarly, the $S$ has consistent representations as $C$.

### B. Differentiable Soft Projection

Inspired by the soft projection for differentiable relaxation [12], we propose a nonlinear soft projection approach to achieve differentiable sampling. First, we leverage the average weight of the $k$ nearest neighbors of $b_j$ in the $P$ as soft projected point $z$ to represent $b_j$. Hence, the soft projected point $z$ can be denoted as:

$$z = \sum_{i \in N_{\mathcal{P}}(b_j)} w_i \cdot p_i.$$  

Next, the weight $w_i$ is calculated by the distance between $b_j$ and its $k$ nearest neighbors, and the softmax function is applied:

$$w_i = \frac{e^{-\text{dist}^2_i/t}}{\sum_{l \in \mathcal{P}_k(b_j)} e^{-\text{dist}^2_l/t}},$$

where $t$ is a learnable temperature coefficient that controls the distribution shape of the weight $w_i$. Analytically, $w_i$ can be regarded as the probability distribution of each neighbor point $p_i$. Assuming that $z$ is the expectation value, only the nearest neighbor point can be approximated as a proper subset of the input point cloud when $t \rightarrow 0^+$. Lastly, we add the project loss in the sampling loss to optimize the convergence of soft projection:

$$L_{\text{soft}} = T(t), t \in [0, +\infty),$$

where $T(\cdot)$, as a function of $t$, introduces nonlinear relationships. After experiment, exponential function $\exp(\cdot)$ used in this research is more conducive to model convergence. The experimental details are presented in the ablation study.

### C. Light-Weight Transformer

The Transformer has shown superior performance in capturing relationship features between points, such as point cloud shapes and geometric dependencies. Typically, a standard Transformer framework contains six major components, 1) positional encoding; 2) input embedding layer; 3) multi-head self-attention block; 4) Feed Forward Networks; 5) layer normalization; and 6) skip connection. In this research, we present a light-weight LighTN for point cloud downsampling by redesigning the major components of the traditional Transformer networks. The lower left corner of Fig. 2 shows the overall framework of LighTN.

**Vanishing position embedding**: In 2D image recognition, position encoding is an essential mechanism for preserving the local relative position of patches, conducive to improving network performance [46], [47]. In contrast, the arrangement of point clouds has no fixed order with the irregular and unordered characteristics. Besides, the 3D coordinates can substitute the position encoding in real-scanned point clouds since they contain information of natural spatial location. Therefore, we remove the position encoding block to reduce the storage and computing overhead of LighTN.

**Light-weight input embedding block**: Given a sequence of $N$ points with $3 + f$ dimensional features, the first step is to embed the original point cloud into a high-dimensional feature space. After this operation, we can obtain $d_v$-dimensional embedded features $F_0 \in \mathbb{R}^{N \times d_v}$, which are more efficient to facilitates subsequent work. In LighTN, we utilize a shared linear layer as the input embedding block and empirically set $d_v = 64$. Compared to the computationally-saving input embedding setting in [48], our module has fewer layers and halved the feature dimension. The computational costs for the input embedding in the PCT [48] and LighTN are $O(2Nd_v^2)$ and $O(Nd_v^2)$, where $d_m = 2d_v$.

**Single head self-correlation layer**: We propose a light-weight self-correlation block to model the geometric relationships between $N$ input points with $d_v$ features. The central view is that the point cloud has the property of natural fault tolerance because there are many approximate solutions with the increase of point distribution density. Following this, we hypothesize that the learning ability of a single head self-attention layer allows LighTN to extract enough relationships between points. Experimental results in ablation studies support this view.

It is worth mentioning that the self-attention mechanisms were firstly proposed in natural language processing tasks [41], which can compute the correlation of a particular token with other input sequences. In these tasks, the order of words is important for text meaning. Accordingly, the input sequences are needed to perform linear transformation to obtain query $Q$ and key $K$ matrices before calculating the attention scores. In contrast, the order between two points is interchangeable with the disorder characteristic. Hence, we simultaneously remove the project matrix $W^Q$ and $W^K$. Mathematically speaking, the new output of attention scores is a symmetric matrix $A$ that satisfies $A^T = A$, where $A^T$ denotes the transpose. Therefore, the symmetric form of attention score matrix is more suitable for point cloud data. The experimental proof will be given in ablation experiments. Moreover, inspired by the work of Mobile-Former [49], we eliminate the project matrix $W^V$ for energy and storage saving as much as possible.

Due to the fact that calculating process of attention score matrix only involves the self-correlation operations on input data, we name it as the self-correlation layer. The lower right corner of Fig. 2 shows the specific architecture. Formally, (4) and (5)
are modified as follows:

\[ SA(X) = FC_{out}(C(X)), \] (10)  
\[ C(X) = \text{softmax} \left( \frac{X \cdot X^T}{\sqrt{D}} \right) \cdot X, \] (11)

where \( X \) represents the output features of the input embedding block. Lastly, the computational cost of self-correlation is only \( O(Nd_f^2 + 2N^2d_o) \).

**Scalable FFN:** Replacing the multi-head attention with a single head self-correlation layer greatly reduces the resource overhead of LightTN, but the learnable capacity is also decreased. Therefore, the core idea of the scalable FFN is to design a parameter-efficient architecture that can be easily scaled in depth and width. To achieve this design, we utilize the expand-reduce strategy [32] to scale the FFN. Verified by experiments, the optimal scalable FFN consists of three linear layers and uses the reduced strategy in the middle layer (\( r = 2 \)). Thus, the input feature dimensionality is reduced from \( d_f \) to \( d_f / r \), where \( r \) represents the reduction ratio. In theory, the computational cost of the scalable FFN only increases by \( O(Nd_f^2 / 4) \) compared to that of the standard FFN with two linear layers.

**Discussion:** Our light-weight framework is designed with consideration of the superior ability of the Transformer to capture the relationship features between points, and the fault-tolerant capacity from LighTN maintains the downsampling performance and improves efficiency by shrinking the main components of the standard Transformer framework.

**D. Loss Function**

To make LighTN focus on critical point cloud regions with more uniform distributions and prominent points coverage, we present a novel loss function consisting of two components, the task loss \( L_{task} \) and the repulsion loss \( L_{repl} \), as shown in Fig. 2.

Ideally, LighTN is aimed at generating a proper subset of the original point cloud. To achieve this goal, we first introduce a Chamfer Distance (CD) function to promote \( B \) to be the nearest point in \( P \). The CD distance between input point \( P \) and simplified point \( B \) is defined as:

\[ L_{CD}(B, P) = \frac{1}{M} \sum_{b_j \in B} \min_{p_i \in P} ||b_j - p_i||_2^2 + \frac{1}{N} \sum_{p_i \in P} \min_{b_j \in B} ||p_i - b_j||_2^2, \] (12)

Second, the main limitation of \( L_{CD} \) is its ignorance of the uniform distribution of points, making simplified point sets more challenging for global surface representation. To alleviate this problem, we adopt a repulsion loss [50] to encourage the uniformity of the generated points and improve the visualization quality. We define the repulsion loss as follows:

\[ L_{repl}(B) = \frac{1}{M} \sum_{1 \leq j \leq M} \sum_{j \in N_k(j)} \eta(||b'_j - b_j||_2), \] (13)

where \( \eta(r) = \max(0, h^2 - r^2) \) is a function to guarantee that the \( b_i \) is not too close to others in \( B \), \( h \) is the mean separation distance between generated points, and \( N_k(j) \) is the set of indices for the K-nearest neighbors of \( b_j \) (we set \( h = 0.001 \) and \( k = 15 \)). According to the discussion above, the total sampling loss is:

\[ L_{total} = L_{sampling}(P, B) + \delta L_{task}(B), \] (15)

where \( \delta \) is the balancing weight.

**IV. Experiments**

This section evaluates the performance and resource overhead of LightTN for the task-oriented point cloud downsampling task. In this research, we explore the effectiveness of LightTN on three machine learning tasks: classification, registration, and reconstruction. In our experiments, the proposed LightTN is compared with a series of state-of-the-art downsampling methods, including 1) commonly used traditional methods: FPS, random sampling and voxel-based simplification; and 2) task-oriented methods: simplified methods with non-differentiable matching operation (SANet [11] and PST-NET [44]); simplified methods with differentiable relaxation matching operation (SampleNet [12] (commonly used as a baseline), DA-Net [43] and MOPS-Net [13]).

**A. Dataset and Metric**

The classification and registration tasks are evaluated on ModelNet40 [51] dataset. ModelNet40 contains 12,311 manufactured 3D CAD models in 40 common object categories, i.e., airplane, bed, and door. For a fair comparison, we leverage the official train-test split strategy with 9,840 CAD models for the training stage and 2,648 CAD models for the testing stage. In order to obtain the 3D Cartesian coordinates of each CAD model, a uniform sample approach [21] is used to extract 1,024 points on mesh faces. More specifically, we use the XYZ-coordinate as the point cloud input without other attributes. For evaluation metrics, the performance of LightTN is taken as the accuracy \( ACC \) and the mean rotation error \( MRE \) is used for classification and registration tasks, respectively. The computation load and the storage space are used to represent the resource overhead. Besides, the downsampling ratio is defined as \( N/M \), where \( M \) represents the number of downsampled points.
The reconstruction task is tested on ShapeNet [52], a large-scale dataset of 3D shapes, which contains 16881 3D CAD models from 16 categories, i.e., table, car, and chair. In this experiment, we followed the settings of SampleNet to prepare the train and test sets, in which 2048 points are extracted from each model. Meanwhile, only XYZ-coordinates are used as input. For evaluation metric, we adopt the average normalized reconstruction error \( NRE \) to assess the performance.

### B. Experiments on Registration

#### Implementation details: Following the experiments of SampleNet and MOPS-Net, we adopt the PCRNet [53] as the task network of point cloud registration for fair comparisons. We implement the LighTN and PCRNet in PyTorch [54]. We set the initial learning rate to 0.001 for LighTN. In this section, we test the performance of the Car category (each car consists of 1024 points).

#### Performance and efficiency: Registration is a mapping process of finding the spatial transformation to align two or more point clouds into the same coordinate system. In this article, the evaluation metric used is the mean rotation error (MRE), which assesses how imprecisely the predicted rotation is aligned to the ground truth. Note that, a small number of generated points may be mapped to the same point on the original point cloud in the test stage. To ensure the number of downsampled points is equal to the \( M \), we follow SampleNet [12] and MOPS-Net [13] to use FPS to supplement missing points. The quantitative results are reported in Table I.

After the downsampled points are reduced to 64 or lower, it is obvious that the results of the traditional task-irrelevant methods suffer severe performance degradation. This is because fewer points are challenging to represent the global spatial characteristics of the original point cloud. In comparison, LighTN achieves the best alignment results in all tests, showing a better learning capacity than other task-orientated downsampling methods. Moreover, comparison results with and without FPS (abbreviated to FPS and non-FPS) show that the FPS-based completion operation can improve task performance. If not specified, the FPS-based completion strategy is adopted in all experiments in this article.

Fig. 4 shows the parameters increase versus the computation reduction. We introduce the floating-point operation (FLOPs) metric to measure the computation load. The memory space is used to represent learnable parameters. Significantly, the reported resource overhead is for the whole execution process, including the point cloud input going through the LighTN and task network (PointNet) to obtain the classification output. It can be observed that our LighTN achieves an optimal performance-overhead tradeoff compared to SampleNet that uses the low-overhead PointNet as a downsampling network. For example, setting the downsampling ratio of LighTN to 32, the FLOPs decrease 69.73% with only a 17.97% increase in parameters, while the mean rotation error remains at 5.60°. Table II reported the average running time from the original point cloud to the sampling points. We test the time consumption of all methods on a single TITAN Xp GPU. We can see that traditional non-learning based methods have relative fast execution time, but lower performance than task-oriented approaches. In addition, although the resource consumption of LighTN is higher than SampleNet, the actual running speed is slightly improved. The reason may be that LighTN has a shallower model structure.

#### Visualization results: Fig. 3 visualizes the point cloud registration results of LighTN and other methods, where the downsampling ratio is set to 16. We observe that FPS achieves a competitive registration accuracy, but it is not sensitive to prominent regions, for instance, the lower contour of the wheel. Besides, learn-based approaches perform better than traditional methods. These results demonstrate that the task-orientated downsampling is more beneficial to improve performance. Moreover, LighTN successfully focuses on both the edge information (vehicle contour) and prominent regions, proving that our work is more favorable for performance and visualization.

### C. Experiments on Classification

#### Implementation details: Following the experiments of SampleNet [11], PST-Net [44], SampleNet [12] and MOPS-Net [13], we use the PointNet [21] as our task network to perform point cloud classification. We implement the LighTN with PointNet in Tensorflow [55]. For LighTN, we employ the Adam optimizer with a mini-batch size of 32 and an initial learning rate of 0.01.
Fig. 3. Visualization of the sampled points for the registration task with $M = 64$. Upper: Template (in blue) and source (in black) point clouds from the side view. Sampled 64 points of a template and a source point cloud are marked with enlarged green and red dots, and then the transformed sampled source points are marked with magenta. Middle: Registration sampled results from the top view of template point cloud. Bottom: The alignment between source point cloud with 1024 points and ground truth. The transformation matrix between two original point clouds is obtained by sampled prediction. Representative regions with good registration effect are highlighted in red circles.

Table III: Classification Accuracy with Different Downsampling Methods on ModelNet40

| $M$ | Voxel [13] | RS [11] | FPS [11] | S-NET [11] | PST-NET [44] | SampleNet [12] | MOPS-Net [13] | DA-Net [43] | LighTN (Ours) |
|-----|------------|---------|---------|------------|--------------|----------------|----------------|-------------|---------------|
| 512 | 73.82      | 87.52   | 88.34   | 87.80      | 87.94        | 88.16          | 86.67          | 89.01       | 89.91         |
| 256 | 73.50      | 77.09   | 83.64   | 82.38      | 83.15        | 84.27          | 86.63          | 86.24       | 88.21         |
| 128 | 68.15      | 56.44   | 70.34   | 77.53      | 80.11        | 80.75          | 86.06          | 85.67       | 86.26         |
| 64  | 58.31      | 31.69   | 46.42   | 70.45      | 76.06        | 79.86          | 85.25          | 85.55       | 86.51         |
| 32  | 20.02      | 16.35   | 26.58   | 60.70      | 63.92        | 77.31          | 84.28          | 85.11       | 86.18         |
| 16  | 13.94      | 7.15    | 13.29   | 36.16      | 42.29        | 51.09          | 81.40          | -           | -             |
| 8   | 3.85       | 3.27    | 3.47    | 20.81      | 19.32        | 23.94          | 52.39          | -           | -             |
| 4   | -          | 1.43    | 1.63    | 5.47       | 5.40         | 5.55           | -              | -           | -             |
| 2   | -          | 1.22    | 1.02    | 2.80       | 3.57         | 1.45           | -              | -           | 7.78          |

$M$ represents the number of down-sampled points.

To ensure that the performance of PointNet is not disturbed during the training and test phases, we adopt the original network configuration without changes.

Performance and efficiency: The classification results on ModelNet40 are reported in Table III. We can see that the output accuracy of PointNet changes slightly when the number of sampled points is reduced to 512 for most methods. This demonstrates that the point clouds contain an overwhelming amount of redundant points, and removing them has almost no effect on the output accuracy. Thus, it is practical to explore downsampling methods to minimize the geometric information loss and facilitate the accurate visualization. In addition, task-oriented methods achieve a higher classification accuracy around all downsampling rates than traditional methods. For example, after the input points are down to 256 (reducing points by 75%), S-NET and PST-NET can keep the output accuracy of the PointNet above 82%. However, as the number of points decreases to 128 and below, the accuracy drops rapidly. The reason might be that the nearest neighbor search process from the generated points to the original point cloud introduces bias. Especially, a slight bias will cause a large performance drop when the number of sampled points is small. In comparison,
the more advanced end-to-end downsampling models, including SampleNet, MOPS-Net and DA-Net, perform better by introducing the soft projection. Moreover, our LighTN yields superior results under most downsampling ratios, demonstrating that the light-weight Transformer framework is more efficient than other state-of-the-art approaches.

Fig. 6 presents the parameter increase versus computation reduction. The evaluations show that LighTN achieves competitive results. For instance, the whole execution process of LighTN consumes only 223.2 M FLOPs and 4.24 MB parameters after setting the downsampling ratio to 32 (32 points simplified by LighTN passing through the PointNet). Compared to PointNet that learns on 1024 input points (the resource cost is 927.2 M FLOPs and 3.48 MB parameters), LighTN reduces 75.93% FLOPs with only a 21.91% increase in parameters, and the output accuracy remains at 86.18% (accuracy reduction of approximately 4% on average). Note that in S-NET, the whole execution process consumes more FLOPs than the task network with 1024 input points, which is seriously inconsistent with the original intention of downsampling.

Visualization results: We visualize the downsampled point sets ($M = 64$) over three categories: guitar, airplane and chair. As shown in Fig. 5, the non-differential-based S-NET pays more attention to the main structure of the object and fails to cover the prominent regions. In contrast, PST-NET and SampleNet have achieved better visualization, but there is still insufficient attention to some prominent areas. In the meantime, these three approaches expose the problem that the point cloud distribution is relatively concentrated and cannot fully cover the entire outer surface of the object. These results in mediocre visualization. By contrast, our LighTN shows the best downsampling ability to focus on critical point cloud regions with more uniform distributions and prominent points coverage. These observations also demonstrate that LighTN is more favorable for classification accuracy.

D. Experiments on Reconstruction

Implementation details: Following the experiments of SampleNet and S-NET, the performance and efficiency of LighTN were tested on the pre-trained task network Autoencoder [56]. In this condition, we implement the LighTN with Autoencoder
in Tensorflow. We apply the official hyper-parameters of the task network for a fair evaluation. For LighTN, the learning rate is set to 0.0005, and the epoch is 400. In this section, we test the performance and efficiency of four categories, including chair, car, table and airplane.

**Performance and efficiency:** Reconstruction from point cloud aims to solve the sparsity and irregularity problems. In this section, LighTN is used for the reconstruction of point clouds from sparsity points. We utilize the Chamfer distance to estimate the reconstruction error of the task network. The reconstruction results on ShapeNet are reported in Table IV. The smaller the NRE, the better. The results show that when enough points are reserved, the performance of FPS is even better than that of S-NET and SampleNet. The reason may be that uniform distribution of sampled points may increase the overall perception, so as to avoid extracting points with similar semantics. This also proves that it is meaningful to introduce repulsion loss. In addition, when the number of points drops to 256 or less, FPS and S-NET methods will suffer more performance degradation.

Correspondingly, the soft projection approach makes the performance of SampleNet more stable. Finally, LighTN produced the lowest NRE score under all downsampling ratios $\{2, 4, 8, 16, 32, 64\}$. All data are obtained by evaluating the whole model, including downsampling and task networks.

**Visualization results:** Fig. 7 visualizes the point cloud reconstruction results of LighTN versus competitive baselines, where the number of downsampling points is set to 32. Obviously, the task-oriented downsampling model based on LighTN achieves the best visualization results and is also the closest to the ground truth.

**TABLE IV**

| $M$ | FPS | S-NET | SampleNet | LighTN (QKV) | LighTN |
|-----|-----|-------|-----------|--------------|--------|
| 1024| 1.004 | 1.009 | 1.003 | 1.002 | 1.002 |
| 512 | 1.025 | 1.052 | 1.030 | 1.024 | 1.023 |
| 256 | 1.107 | 1.133 | 1.116 | 1.076 | 1.078 |
| 128 | 1.370 | 1.288 | 1.245 | 1.145 | 1.129 |
| 64  | 2.084 | 1.712 | 1.470 | 1.446 | 1.433 |
| 32  | 3.740 | 3.141 | 2.301 | 2.189 | 2.29  |

$M$ represents the number of downsampled points.
introduced a local-global attention mechanism to capture
fine-grained local features, which is crucial for vision models.

Inspired by their work, we present a local-global single-head
attention block, as shown in Fig. 13(a). Second, Wang et al.
[44] introduced a local-global attention mechanism to capture
fine-grained local features, which is crucial for vision models.

Classification results on data with a large point number: To
verify the potential of processing point clouds with a large
point number, we conduct a downsampling test on LighTN
on the ModelNet40 dataset, in which each point cloud has
10,000 points. The architecture of LighTN, task model, and
hyper-parameters are consistent with Section IV-C. The clas-
sification accuracy of the pre-trained PointNet on ModelNet40
with 10,000 points per point cloud is 89.1%, which proves that
the capacity of a point set is not always proportional to data
quality and recognition effect for shape analysis and geometry
processing tasks. Table V shows the classification accuracy of
LighTN and SampleNet after downsampling 10,000 points to
4096, 2048, and 1024 points. The high classification accuracies
prove that LighTN is capable of downsampling point clouds with
a large point number.

Classification results on real-scanned dataset: ScanOb-
jectNN is challenging for existing deep learning-powered frame-
works because real-scanned data are often cluttered with the
background and incomplete due to occlusions [57]. In this test,
the task model is set to PointNet, and hyper-parameter settings
in Section IV-C are used. We evaluate S-NET, SampleNet, and
LighTN on the ScanObjectNN dataset. Besides, RS and FPS are
employed as non-learning downsampling methods for compar-
ison. The classification accuracies on test set in Table VI show
that LighTN outperforms baseline approaches by a large mar-
gin. We visualize the task-oriented downsampling points ($M = 32$) in Fig. 9. We can see that LighTN still focuses on more
prominent regions.

**TABLE V**

| $M$  | SampleNet | LighTN |
|------|-----------|--------|
| 4096 | 88.89     | 89.02  |
| 2048 | 88.83     | 88.51  |
| 1024 | 87.90     | 88.17  |

$M$ represents the number of downsampled points.

**TABLE VI**

| $M$  | RS   | FPS  | S-NET | SampleNet | LighTN |
|------|------|------|-------|-----------|--------|
| 512  | 66.35| 67.24| 68.44 | 68.37     | 68.61  |
| 256  | 63.02| 65.80| 66.52 | 66.44     | 66.95  |
| 128  | 55.01| 60.27| 60.30 | 59.99     | 61.04  |
| 64   | 48.63| 52.51| 51.68 | 49.38     | 52.55  |
| 32   | 36.16| 46.05| 44.44 | 37.29     | 46.20  |
| 16   | 31.22| 38.19| 38.02 | 31.66     | 39.16  |
| 8    | 19.13| 29.88| 24.48 | 29.25     | 30.86  |
| 4    | 17.41| 19.33| 18.48 | 21.90     | 27.37  |
| 2    | 8.69 | 12.16| 11.33 | 9.82      | 19.07  |

$M$ represents the number of downsampled points.

Classification results on noise data: In this noise injection ex-
periment, we introduce Gaussian noise into the test data of Mod-
elNet40 to verify the robustness of LighTN on the classification
task. We add 5 levels of Gaussian noise to each dimensional of
original point clouds that are standard deviations of 1%, 2%, 3%, 4%, and 5% respectively. Here, the pre-trained LighTN in
Section IV-C is used for testing without retraining or fine-tuning.
Experimental results are illustrated in Fig. 10. LighTN main-
tains a high accuracy of classification even with 5% Gaussian
noise than S-NET and SampleNet run on clean test data. In ad-
dition, we visually demonstrate several downsampled examples
by LighTN over noisy data. As shown in Fig. 11, the attention of
LighTN to prominent regions is consistent under different noise
levels. These results indicate that LighTN is robust to noise.

**E. Robustness Test**

Classification results on data with a large point number: To
verify the potential of processing point clouds with a large
point number, we conduct a downsampling test on LighTN
on the ModelNet40 dataset, in which each point cloud has
10,000 points. The architecture of LighTN, task model, and
hyper-parameters are consistent with Section IV-C. The clas-
sification accuracy of the pre-trained PointNet on ModelNet40
with 10,000 points per point cloud is 89.1%, which proves that
the capacity of a point set is not always proportional to data
quality and recognition effect for shape analysis and geometry
processing tasks. Table V shows the classification accuracy of
LighTN and SampleNet after downsampling 10,000 points to
4096, 2048, and 1024 points. The high classification accuracies
prove that LighTN is capable of downsampling point clouds with
a large point number.

Classification results on real-scanned dataset: ScanOb-
jectNN is challenging for existing deep learning-powered frame-
works because real-scanned data are often cluttered with the
background and incomplete due to occlusions [57]. In this test,
the task model is set to PointNet, and hyper-parameter settings
in Section IV-C are used. We evaluate S-NET, SampleNet, and
LighTN on the ScanObjectNN dataset. Besides, RS and FPS are
employed as non-learning downsampling methods for compar-
ison. The classification accuracies on test set in Table VI show
that LighTN outperforms baseline approaches by a large mar-
gin. We visualize the task-oriented downsampling points ($M = 32$) in Fig. 9. We can see that LighTN still focuses on more
prominent regions.

**F. Analysis on Task-Oriented Downsampling**

Now, we investigate whether the downsampling points learned
through LighTN are task-oriented. Ideally, LighTN can adap-
atively adjust the distribution of the downsampled point cloud in
the same input according to different downstream tasks. Hence,
we run the LighTNs trained on the registration task (in Sec-
donaratively properties of self-attention and convolution, we design
four potential architectures for comparison on the classification
Task. First, extensive works prove that the framework based
on the multi-head self-attention mechanism can improve the
capacity of a point set is not always proportional to data
quality and recognition effect for shape analysis and geometry
processing tasks. Table V shows the classification accuracy of
LighTN and SampleNet after downsampling 10,000 points to
4096, 2048, and 1024 points. The high classification accuracies
prove that LighTN is capable of downsampling point clouds with
a large point number.

**G. Ablation Study**

To probe the validity of the specific designs in LighTN, we
conducted several controlled experiments.

Self-correlation configuration: In this ablation study, we aim
to show the advantages of the self-correlation block designed
in Section III-C. Considering the learning capacity and com-
plementary properties of self-attention and convolution, we design
four potential architectures for comparison on the classification
task. First, extensive works prove that the framework based
on the multi-head self-attention mechanism can improve the
global feature extraction ability. Thus, we choose a multi-head
dot-production attention block as our baseline approach for
the evaluation, As depicted in Fig. 13(a). Second, Wang et al.
[44] introduced a local-global attention mechanism to capture
fine-grained local features, which is crucial for vision models.
Inspired by their work, we present a local-global single-head
attention block, as shown in Fig. 13(b). Note that the resource
overhead of the local block is positively correlated with the num-
ber of layers; thus, we only use two Edge Convolutional layers
to prevent the excessive overhead. Finally, the projection ma-
trices can be removed for further computational savings [49],
two light-weight dot-production attention blocks (see Fig. 13(c)
and (d)) are thus designed.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 9. Visualization of sampled points for classification task by task-oriented downsampling approaches on ScanObjectNN dataset. Original point clouds contain 1024 points (in gray), and downsampled 32 points are marked with blue dots. The categories are shown: Chair, sofa and toilet. Representative regions with good sampling effect are highlighted in red circles.

Fig. 10. Comparisons of the classification performances of different task-oriented downsampling approaches applied to input point clouds with 5 noise levels and clean data.

Explicitly, all experiments were performed with the basic loss function $L_{CD} + L_{soft}(t^2)$. Fig. 14 shows the comparative results in terms of accuracy (ACC, $\%$), computation load (FLOPs, G) and storage space (Params, MB). From these results, we find that the downsampling network based on the local-global single-head attention mechanism almost obtains the best accuracy in all downsampling ratios. This suggests that the convolution with local connectivity and self-attention in the global receptive field have complementary properties. However, the resource overhead of local-global attention is not the ideal solution. In addition, the output accuracy of LighTN with a two-head ($m = 2$) attention model is just slightly higher than the single-head attention model and the three-head attention model. For example, setting the downsampling ratio to 16, the output accuracy of $QKV(m = 1)$, $QKV(m = 2)$ and $QKV(m = 3)$ are 85.41\%, 85.49\% and 85.37\%, respectively. The reason might be that the increased model complexity improves the difficulty of LighTN training and leads to learning redundant parameters. Also, the complex model structure brings a higher average computation load and storage space costs. For instance, comparing $QKV(m = 2)$ and $QKV(m = 3)$ with $QKV(m = 1)$, FLOPs increase by 65.6\% and 131.7\%, and Params increases by 1.8\% and 3.6\%. In summary, the single-head attention architecture achieves a competitive balance between the performance and the resource overhead in the downsampling range.

The other experiment of Non-QKV shows that LighTN with a self-correlation block has a better discrimination ability and a minimal resource overhead than $QKV(m = 1)$ after removing all projection matrices. The reason is that a single-head self-correlation mechanism without projection matrices is more suitable to calculate the attention score because its internal symmetry matrix satisfies the permutation invariance.

Scalable FFN configuration: Next, we conduct an ablation study on the scalable FFN module with an expand-reduce strategy. We take two linear layers, $L(512, M \times 3)$, without expand-reduction ($L = 2, r = 0$) as the baseline where the values in $L(\cdot)$ denote the number of nodes in the linear layer and $M$ denotes the number of downsampled points. Naturally, the number of nodes in the last linear layer cannot be changed. Otherwise, the output of generated points will deviate from the downsampling ratio. Experimental results are depicted in Fig. 15. We can see that three layers FFN and $r_{l2} = 2$ enable LighTN to obtain the best output accuracy while the FLOPs and Params increase only 0.0087 G FLOPs and 0.0488 MB on average. Considering all factors, we set $L = 3$ and $r_{l2} = 2$ as our default for all models. Besides, if the network needs to further reduce resource costs further, the setting of $L = 3$ and $r = 2$ tends to be a good choice.

Loss function configuration: As discussed in Section III-D, the proposed loss function of the $L_{sampling}$ are consisted of three components: $L_{CD}$, $L_{repl}$ and $L_{soft}$. We set $L_{CD} + L_{soft}(t^2)$ as the baseline. Then, we test the impact of loss functions on the
Fig. 11. Visualization of downsampled examples by LighTN against varying levels of Gaussian noise. (a) Noiseless; (b) 1% Gaussian noise; (c) 2% Gaussian noise; (d) 3% Gaussian noise; (e) 4% Gaussian noise; (f) 5% Gaussian noise. Original point clouds contain 1024 points (in gray), and downsampled 64 points are marked with blue dots. Representative regions with sampling consistency are highlighted with red circles.

Fig. 12. Visualization of downsampled examples by LighTN against varying downstream tasks. Original point clouds contain 1024 points (in gray), and downsampled 64 points are marked with blue dots.

Fig. 13. Different self-attention blocks. (a) Multi-head self-attention with projection matrices $W^Q$, $W^K$ and $W^V$ where $m$ is the number of heads, $D$ is the dimension of $Q$ and $K$ vector and $a$ is the scaled factor, as described in (5); (b) Single-head self-attention block with local convolution layer (Conv); (c) Single-head self-attention block where $W^Q$ is removed to save computations; (d) Single-head self-attention block where $W^K$ and $W^V$ are removed.

Classification

Registration

performance by conducting experiments on ModelNet40. For the project loss $L_{soft}$ in classification task, we test two nonlinear functions: $T(t) = t^2$ used in SampleNet [12] and $T(t) = e^t$. Table VIII reports the comparison results. For $L_{soft}$, exponential function $exp(\cdot)$ is more conducive to the convergence of model accuracy. In particular, these experiments indicate that our proposed sampling loss function $L_{sampling}$ enables the task network to obtain a higher accuracy. Moreover, more function $T$ of temperature coefficient $t$ are examined in the registration task. The experiment was repeated three times for reasonable testing and the average value was used as the output result, shown in Table VII. In summary, the results in Tables VII and VIII indicate that the combination of $L_{CD}$, $L_{soft}(e^t)$ and $L_{repl}$ contributes to the best performance of LighTN.

H. Discussion

We make the following two observations from the above experiments:

1) Task-oriented methods perform better than traditional two-stage methods in different downsampling applications. As a new task-oriented method, the proposed LighTN improves the performance-overhead tradeoff by learning the discriminant and correlative features for unordered and irregular point clouds. This is probably a result
TABLE VII
ABLATION STUDY: A CONTROLLED COMPARISON OF TEMPERATURE FUNCTIONS $T$ FOR REGISTRATION TASK

| $M$ | $L_{\text{CD}} + \lambda_{\text{res}}(x^2)[12]$ | $L_{\text{CD}} + \lambda_{\text{res}}(x)$ | $L_{\text{CD}} + \lambda_{\text{res}}(x^2) + \lambda_{\text{repl}}$ | $L_{\text{CD}} + \lambda_{\text{res}}(x^2) + \lambda_{\text{repl}}$ | $L_{\text{CD}} + \lambda_{\text{res}}(x^2) + \lambda_{\text{repl}}$ | $L_{\text{CD}} + \lambda_{\text{res}}(x^2) + \lambda_{\text{repl}}$ |
|-----|----------------------------------|----------------------------------|----------------------------------|----------------------------------|----------------------------------|----------------------------------|
| 512 | 4.22                            | 4.22                            | 4.18                            | 4.16                            | 4.25                            | 4.14                            |
| 256 | 4.43                            | 4.35                            | 4.36                            | 4.42                            | 4.44                            | 4.33                            |
| 128 | 4.82                            | 4.76                            | 4.85                            | 4.79                            | 4.73                            | 4.65                            |
| 64  | 5.41                            | 5.32                            | 5.39                            | 5.43                            | 5.26                            | 5.26                            |
| 32  | 5.82                            | 5.80                            | 5.75                            | 5.89                            | 5.99                            | 5.77                            |
| 16  | 6.93                            | 7.01                            | 7.13                            | 6.78                            | 6.58                            | 6.60                            |

$M$ represents the number of downsampled points.

Fig. 14. Comparison of self-attention blocks on ModelNet40 classification. The QKV indicates the attention mechanism with complete projection matrices. $\alpha$ is the scaled factor, $m$ represents the number of heads and Conv is convolutional operation. KV and Q represent modules of Fig. 13(c) and (d), respectively. Non-QKV is our designed self-correlation block. Note that the 9 mark points from left to right are correspond to downsampling ratios {2, 4, 8, 16, 32, 64, 128, 256, 512} respectively.

TABLE VIII
ABLATION STUDY: THE ACCURACY (%) COMPARISON OF DIFFERENT LOSS FUNCTIONS FOR CLASSIFICATION TASK ON MODELNET40

| $M$ | $L_{\text{CD}} + \lambda_{\text{res}}(x^2)[12]$ | $L_{\text{CD}} + \lambda_{\text{res}}(x')$ | $L_{\text{CD}} + \lambda_{\text{res}}(x') + \lambda_{\text{repl}}$ |
|-----|----------------------------------|----------------------------------|----------------------------------|
| 512 | 89.74                            | 89.14                            | 89.91                            |
| 256 | 87.88                            | 87.93                            | 88.21                            |
| 128 | 86.14                            | 86.62                            | 86.26                            |
| 64  | 85.45                            | 86.26                            | 86.51                            |
| 32  | 85.45                            | 85.41                            | 86.18                            |
| 16  | 79.61                            | 78.16                            | 79.34                            |
| 8   | 74.69                            | 51.17                            | 52.92                            |
| 4   | 22.08                            | 22.89                            | 22.08                            |
| 16  | 7.13                             | 7.33                             | 7.78                             |

$M$ represents the number of downsampled points.

of the fact that the light-weight self-correlation mechanism of LighTN produces more correlation and discriminant features of the point cloud input and has a minimal resource overhead.

2) The proposed LighTN demonstrates its superiority through experiments conducted in three different downsampling scenarios. By learning refined global geometric features and introducing a novel sampling loss function, LighTN works effectively on critical points extraction and visualization. Moreover, this article demonstrates that the Transformer and Convolutional Neural Network have complementary properties, which provide possible directions for future developments in this sector.

V. CONCLUSION

This article proposes a light-weight Transformer network LighTN to simplify the original point clouds. With the help of the single-head self-correlation mechanism and the scalable FFN architecture, LighTN shows an excellent learnable capacity with limited resource overheads. Then, we design a novel sampling loss function that guides LighTN to edge and prominent point cloud regions while ensuring that the downsampled point cloud has uniform distributions and adequate coverage. Extensive experiments on registration, classification and reconstruction tasks demonstrate that LighTN achieves the state-of-the-art task-oriented point cloud downscaling.
In the future, our main concern will be the minimum resource overhead and the efficiency of the method in existing real life scenarios. Firstly, the self-correlation mechanism involves massive matrix multiplication operations, which causes LightTNT to consume more computation resources than the PointNet-based downsampling approaches. Secondly, the integration of self-attention and convolution exhibits improved performance in ablation experiments. Nevertheless, it is not used in this article due to the resource overhead constraints. Thirdly, unlike common shape analysis and geometry processing tasks, point cloud data from real life scenarios contains richer semantic information, which brings greater challenges to the downsampling work. Therefore, a lower-energy addition replaces the matrix multiplication operation in the self-correlation module, and a light-weight Transformer-Convolutional network for the performance improvement are the directions of interest.

REFERENCES

[1] D. Zhang, X. Lu, H. Qin, and Y. He, “PointFilter: Point cloud filtering via encoder-decoder modeling,” IEEE Trans. Vis. Comput. Graph., vol. 27, no. 3, pp. 2015–2027, Mar. 2021.
[2] X. Zhao, B. Zhang, J. Wu, R. Hu, and T. Komura, “Relationship-based point cloud completion,” IEEE Trans. Vis. Comput. Graph., vol. 28, no. 12, pp. 4900–4902, Dec. 2022.
[3] X. Li et al., “A rotation-invariant framework for deep point cloud analysis,” IEEE Trans. Vis. Comput. Graph., vol. 28, no. 12, pp. 4503–4514, Dec. 2022.
[4] L. Li, Z. Li, S. Liu, and H. Li, “Planoptic point cloud compression using multiview extension of high efficiency video coding,” IEEE Trans. Multimedia, vol. 25, pp. 2007–2021, 2023.
[5] Z. Li, G. Li, T. H. Li, S. Liu, and W. Gao, “Semantic point cloud upsampling,” IEEE Trans. Multimedia, vol. 25, pp. 3432–3442, 2023.
[6] L. Li and M. Yang, “Point cloud registration based on direct deep features with applications in intelligent vehicles,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 8, pp. 13346–13357, Aug. 2022.
[7] Q. Hu et al., “RandLA-Net: Efficient semantic segmentation of large-scale point clouds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11108–11117.
[8] E. Nezhadzary, E. Taghavi, R. Razani, B. Liu, and J. Luo, “Adaptive hierarchical down-sampling for point cloud classification,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 12956–12966.
[9] F. Wu et al., “3D-transformer: Molecular representation with transformer in 3D space,” 2019.
[10] W. Wu, Y. Zhang, D. Wang, and Y. Lei, “SK-Net: Deep learning on point cloud via end-to-end discovery of spatial keypoints,” in Proc. 34th AAAI Conf. Artif. Intell., 2020, pp. 642–6429.
[11] O. Devrat, I. Lang, and S. Avidan, “SampleNet: Differentiable point cloud sampling,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 7578–7588.
[12] Y. Qian et al., “Task-oriented compact representation of 3D point clouds via a matrix optimization-driven network,” IEEE Trans. Circuits Syst. Video Technol., 2023.
[13] Y. Zhou et al., “Learning and matching multi-view descriptors for registration of point clouds,” in Proc. 15th Eur. Conf. Comput. Vis., 2018, pp. 505–522.
[14] L. Li, S. Zhu, H. Fu, P. Tan, and C. L. Tai, “End-to-end learning local multi-view descriptors for 3D point clouds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 1919–1928.
[15] Q. Yang et al., “Predicting the perceptual quality of point cloud: A 3D-to-2D projection-based exploration,” IEEE Trans. Multimedia, vol. 23, pp. 3877–3891, 2021.
[16] Y. Zhou et al., “End-to-end multi-view fusion for 3D object detection in LiDAR point clouds,” in Proc. 3rd Annu. Conf. Robot Learn., 2020, pp. 923–924.
[17] Z. Gojic, C. Zhou, J. D. Wegner, and A. Wieser, “The perfect match: 3D point cloud matching with smoothed densities,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 5545–5554.
[46] K. Han et al., “Transformer in transformer,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 15908–15919.

[47] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” in Proc. Int. Conf. Learn. Representations, 2020.

[48] M.-H. Guo et al., “PCT: Point cloud transformer,” Comput. Vis. Media, vol. 7, no. 2, pp. 187–199, 2021.

[49] Y. Chen et al., “Mobileformer: Bridging mobilenet and transformer,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 5260–5269.

[50] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, “EC-Net: An edge-aware point set consolidation network,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 386–402.

[51] Z. Wu et al., “3D ShapeNets: A deep representation for volumetric shapes,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 1912–1920.

[52] A. X. Chang et al., “ShapeNet: An information-rich 3D model repository,” 2015, arXiv:1512.03012.

[53] V. Sarode et al., “PCRNets: Point cloud registration network using pointnet encoding,” 2019, arXiv:1908.07906.

[54] A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” in Proc. Ann. Conf. Neural Inf. Process. Syst., 2019, pp. 8026–8037.

[55] M. Abadi et al., “TensorFlow: A system for large-scale machine learning,” in Proc. 12th USENIX Symp. Operating Syst. Des. Implementation, 2016, pp. 265–283.

[56] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, “Learning representations and generative models for 3D point clouds,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 40–49.

[57] M. A. Uy, Q. H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, “Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 1588–1597.

Xu Wang is currently working toward the Ph.D. degree with the School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China. His main research interests include machine learning and computer vision.

Yi Jin received the Ph.D. degree in signal and information processing from the Institute of Information Science, Beijing Jiaotong University, Beijing, China, in 2010. She is currently a Professor with the School of Computer and Information Technology, Beijing Jiaotong University. Her research interests include image processing and signal processing. From 2013 to 2014, she was a Visiting Scholar with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

Yidong Li (Senior Member, IEEE) received the B.Eng. degree in electrical and electronic engineering from Beijing Jiaotong University, Beijing, China, in 2003, and the M.Sc. and Ph.D. degrees in computer science from The University of Adelaide, Adelaide, SA, USA, in 2006 and 2010, respectively. He is currently the Vice-Dean and a Professor with the School of Computer and Information Technology, Beijing Jiaotong University. He has authored or coauthored more than 100 research papers in various journals, such as IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON KNOWLEDGE & DATA ENGINEERING, and IEEE TRANSACTIONS ON CYBERNETICS, and refereed conferences, such as SIGKDD, Conference on Computer Vision and Pattern Recognition, and AAAI. He has also coauthored/co-edited five books (including proceedings) and contributed several book chapters. His research interests include Big Data analysis, data privacy and security, advanced computing, and intelligent transportation. He has organized several international conferences and workshops and was also a program committee member for several major international conferences, such as ICML/PKDD, Pacific-Asia Conference on Knowledge Discovery and Data Mining, NFOSCALE, Western Athletic Conference, South Atlantic Conference, IEEE International Conference on Parallel and Distributed Computing, Applications and Technologies, DANTH, and IEEE International Symposium on Parallel Architectures, Algorithms and Programming.

Yigang Cen (Member, IEEE) received the Ph.D. degree in control theory and control engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2006. He is currently a Professor with the Institute of Information Science, Beijing Key Laboratory of Advanced Information Science & Technology, Beijing Jiaotong University, Beijing, China. In 2006, he joined the Signal Processing Centre, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, as a Research Fellow. From 2014 to 2015, he was a Visiting Scholar with the Department of Computer Science, University of Missouri, Columbia, MO, USA. His research interests include computer vision, image understanding/processing, and IOT.

Tao Wang received the Ph.D. degree from the School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China, in 2013. From 2014 to 2015, he was a Visiting Professor with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA. He is currently a Professor with the School of Computer and Information Technology, Beijing Jiaotong University. His research interests include computer vision and machine learning.

Bowen Tang is currently working toward the Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His research interests include computer architecture, system virtualization, and software security.