Flattening-Net 研究报告

摘要

本报告详细介绍了 Flattening-Net [9] 点云参数化方法，旨在将三维点云高效地转换为二维表示，以提升其分析效果。该方法通过表面展平模块将点云映射到二维平面，再通过网格重采样模块进行均匀排列，最终生成点云几何图像（PGI）。为了充分挖掘 PGI 的特征，论文提出了 CSConv 操作，并结合 PointNet [1] 提取结构化编码和位置编码。实验部分成功复现了 Flattening-Net [9] 的点云参数化算法，并在此基础上实现了点云分类任务，分类效果虽与原论文略有差异，但整体表现良好。此外，尝试了直接从 PGI 重建点云的方法，但由于 PGI 缺乏图像的像素连续性，重建效果未达预期，反映出二维卷积在处理 PGI 特殊结构时存在局限性。未来可考虑采用 Transformer [4] 架构来突破 PGI 的局限性，但此策略在本质上与传统的基于“采样 + 查询”的方法相似。

关键词：点云学习；参数化

1 引言

点云数据一般是由众多的三维坐标点集所组成，属于一种较为轻量的显式三维表达手段。这些点往往通过激光扫描、摄影测量或者其他的三维扫描技术从物体表面获取。点云数据在计算机视觉、机器人、地理信息系统等领域被广泛运用，主要用于三维模型重建、地形测绘、物体识别等工作。大多数的任务都需要开展基础性的点云分析，例如分类和分割。

传统的基于深度学习的点云分析方法 [1, 2, 6, 8, 10]，由于点云数据的无序性，通常采用“采样 + 查询”的策略来构建局部邻域，并通过卷积、注意力机制等手段提取局部特征。然后，使用对称函数聚合全局特征，以有效应对点云的无序性。然而，传统方法的计算开销较大，主要源于查询操作。近期，一些研究者提出了基于序列化的方法 [5, 7]，将点云按照特定规则排序后，利用注意力机制提取特征。这种方法在提升效果的同时，避免了查询操作带来的巨大开销，使得网络能够应用于大规模点云。

从现有方法的发展趋势来看，越来越多的研究人员开始尝试新的思路。除了上述的序列化方法，本研究报告关注的 Flattening-Net [9] 尝试将点云参数化为图像。尽管将点云参数化为图像会增加表示场景或物体所需的存储空间，但其显著优势在于能够充分利用图像领域中的先进方法，从而进一步提高点云分析效果。此外，随着点云规模的持续扩大，参数化方法所需的额外存储开销增长并不显著。

2 相关工作

关于点云分析的大致可以划分为基于“采样 + 查询”策略的传统方法和以序列化为基础的方法。

2.1 基于“采样 + 查询”的传统点云分析方法

PointNet [1] 是一种直接以原生点云数据为输入的特征提取网络。该网络采用简单的一维卷积作为基本构建块，在起始阶段利用可学习的网络对点云数据进行对齐（预处理），最后通过采用对称函数来应对点云的无序性。PointNet++ [2] 在 PointNet [1] 的基础上，为了更好地感知点云数据的局部结构，提出了新颖的 SA（Set Abstraction）结构。一个 SA 结构包含三个步骤：首先，利用最远点采样得到均匀分布的下采样点；其次，在每个下采样点附近通过球查询的方式采样特定数量的点，从而构成若干个组；最后，利用简单的 PointNet [1] 网络处理每个组中的点。这种“采样 + 查询”的点云处理方式对后续的点云相关工作产生了深远影响。DGCNN [6] 将图的概念应用于点云数据。在每个基础网络块中，首先利用 KNN 算法将点云数据转换为图结构（邻接矩阵），然后在图上应用二维卷积，最后通过聚合操作（如求和、取最值）将图转换回点云数据。DGCNN [6] 的最大优势在于在保留 PointNet++ [2] 特性（捕获局部特征）的同时，可以非常容易地嵌入到其他网络架构中。此外，在转换为图的过程中，可以自由地在特征维度上构造出不同形式的特征组合，从某种程度上来说，它是 PointNet++ [2] 的一种上层抽象。Point Transformer [10] 是 Transformer [4] 在点云领域的典型应用。与传统 Transformer [4] 所采用的标量注意力（Scalar Attention）不同，Point Transformer [10] 采用了能力更强的向量注意力（Vector Attention）作为基础，并结合相对位置编码构建了核心的 Transformer 块。为了避免计算量过大，Point Transformer [10] 采用了类似于 PointNet++ [2] 的方法，即先进行“采样 + 查询”的处理，然后在此基础上应用注意力机制。Point Transformer v2（PTv2）[8] 在 Point Transformer [10] 的基础上提出了 GVA（Grouped Vector Attention），GVA 本质上是在 Vector Attention 的基础上添加了类似于多头注意力机制的分组概念，有效减少了网络的参数。此外，论文中还提到了权重形式的相对位置编码以及基于分割的池化方法，以替代原来的最远点采样和 KNN 算法。

2.2 基于序列化的点云分析方法

OctFormer [5] 首次引入了点云序列化的概念，通过将点云数据组织成八叉树结构，在此基础上，利用点之间的相对位置信息，以类似于哈夫曼编码的方式为点云中的每个点进行编码。对所有编码进行排序后，得到的点云序列将按照 Z-Order 的顺序在空间中排列。之后，采用类似于自然语言处理的方法，将序列化后的点云进行分组，并在分组后的点云上应用注意力机制。这种处理方式在很大程度上降低了 Transformer [4] 的计算和存储开销，同时保持了良好的效果。Point Transformer v3（PTv3）[7] 在 OctFormer [5] 的基础上进一步强化了点云序列化——采用了四种序列化方法，以乱序的方式依次应用在每个 TransformerTransformer [4] 块中的四个 Attention 块中。此外，论文中还采用了各种技巧来提升网络速度，例如 xCEP 位置编码等，最终将点云的感受野从常见的 16 个点扩展到了 1024 个点，极大地减少了计算和存储开销。

3 本文方法

3.1 本文方法概述

Flattening-Net [9] 提出了一种将三维点云转换为二维表示的参数化方法，分为两个主要阶段。首先，通过将点云参数化为点云几何图像（Point Geometry Image，PGI），然后在此基础上进行分类、分割和重建等下游任务。整个过程可以通过示意图 1进行参考。

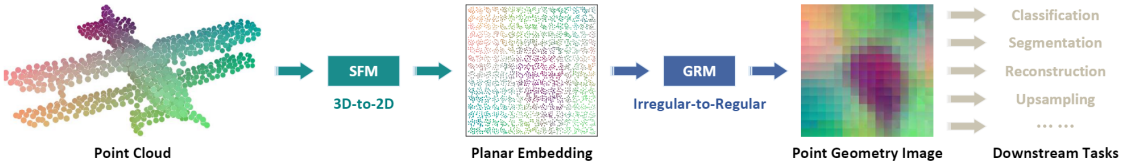

图 1. Flattening-Net 示意图 [9]

对于包含 N 个点的三维点集 $\mathcal{P} \in \mathbb{R}^{N\times 3}$，Flattening-Net [9] 的目标是将其转换为具有规则表示的二维点云几何图像（Point Geometry Image，PGI）。具体来说，该架构首先通过表面展平模块（Surface Flattening Module，SFM）将点云中的所有点结构化地映射到二维平面上，然后利用网格重采样模块（Grid Resampling Module，GRM）在均匀格点上重新分配这些映射点，从而构建出规则的网格结构。在得到点云对应的 PGI 之后，Flattening-Net [9] 采用为该数据结构专门设计的同心正方形卷积（Concentric-Square Convolution，CSConv）进行学习，以完成各种常见的下游任务。

3.2 表面展平模块

SFM 由两个核心组件构成：G2SD（Grid-to-Surface Deformation）和 S2PF（Surface-to-Plane Flattening）。这两个组件协同工作，共同实现将点云结构化地映射到二维平面的目标。

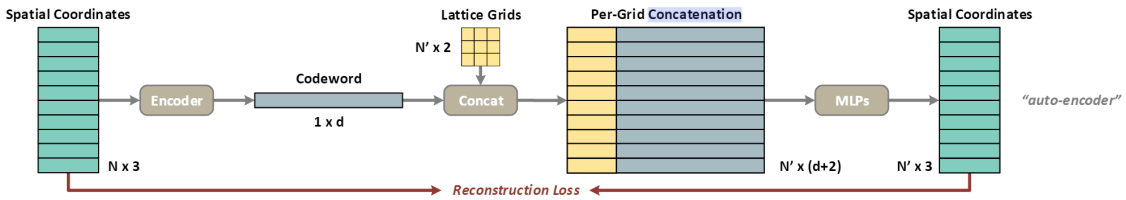

图 2. G2SD 网络架构示意图 [9]

3.2.1 G2SD

组件 G2SD 的核心任务是通过重建点云的过程，将点云按照二维晶格的顺序重新排列。其网络架构如图 2所示。具体来说，该组件首先利用编码器将点云的空间坐标编码为隐向量，然后执行图中所示的逐晶格拼接操作，即将隐向量拼接在预先准备好的二维晶格的每个格子的末尾，最后通过解码器重建原始点云。在训练网络的过程中，通常采用常见的点云重建损失函数，例如 Chamfer Distance（CD）。

3.2.2 S2PF

组件 S2PF 的核心任务是将三维点云映射到平面上，以确定每个点在平面上的对应位置。其网络架构如图 3所示。具体而言，该组件首先通过编码器将点云的空间坐标转换为隐向量，然后执行图中所示的逐点拼接操作，即将隐向量附加到每个点的坐标之后，最终通过解码器计算出每个点在平面上的位置。在训练网络的过程中，采用了斥力损失（Repulsion Loss）

$$\mathcal{L}_{\text{repulsion}}(\mathcal{F}) = \sum_{i=1}^{N} \max\left(0, \epsilon - \|\mathbf{f}_i - \mathbf{f}_j\|_2\right),$$

其中，$\mathbf{f}$ 表示每个点在平面上的映射位置，而 ϵ 是一个设定的阈值。该损失函数的目的是防止转换后的坐标过于集中，从而避免点云在平面上过度重叠。

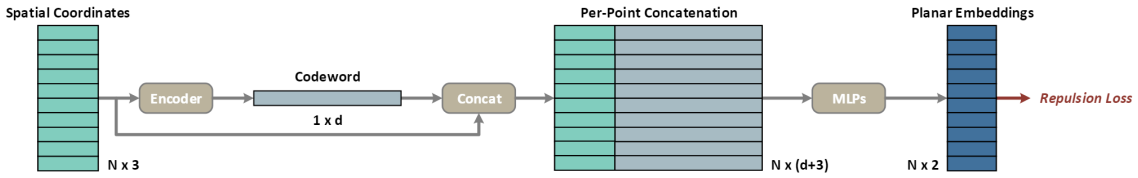

图 3. S2PF 网络架构示意图 [9]

3.3 网络重采样模块

GRM 的主要任务是将不规则分布的点均匀地排列在晶格的每个顶点上，如图 4所示。当点的数量与晶格顶点的数量完全匹配时，每个晶格顶点会选取距离最近的点坐标进行填充；而当点的数量少于晶格顶点数量时，在前一种情况的基础上，将剩余未填充的晶格顶点选取附近的点坐标进行填充。这一过程可以通过 Auction 算法高效求解。

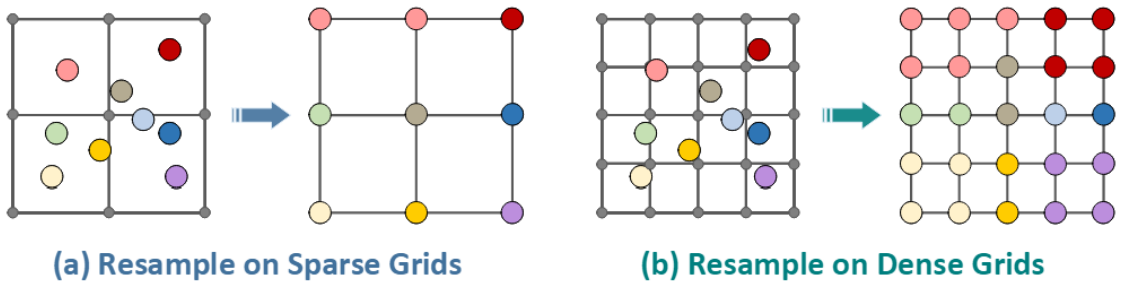

图 4. GRM 算法示意图

3.4 参数化方法

Flattening-Net [9] 的参数化方法综合运用了之前章节中介绍的所有模块和组件，其算法流程如图 5所示。首先，算法使用最远点采样（Farthest Point Sampling，FPS）方法将点云下采样为稀疏点云。然后，将预训练的 G2SD 组件应用于该稀疏点云，以获得排序后的稀疏点云。接着，利用 KNN 算法查询有序稀疏点云中每个点的邻近点，从而得到若干个块。之后，将预训练的 S2PF 组件应用于每个块，以获得相应的平面映射。在此基础上，应用 GRM 将不规则分布的点均匀排列，以形成密集网格。最后，将每个块对应的密集网格依序收集起来，即可得到点云所对应的点云几何图像。

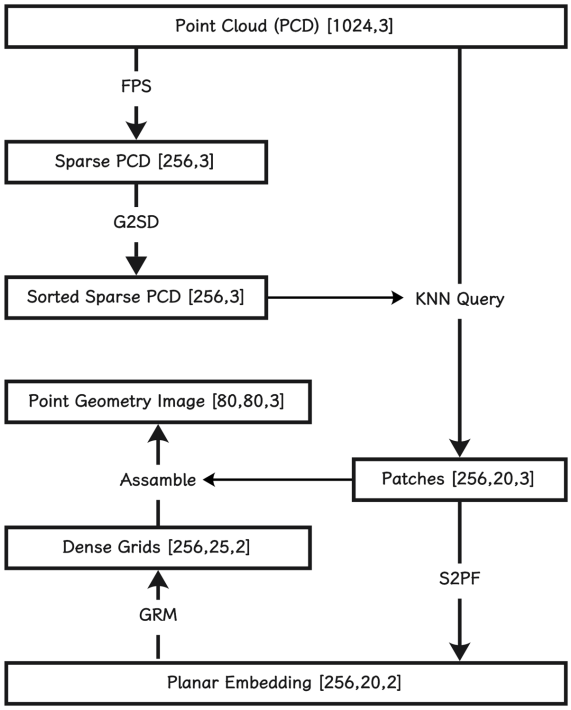

图 5. 参数化算法流程图 [9]

3.5 点云几何图像学习

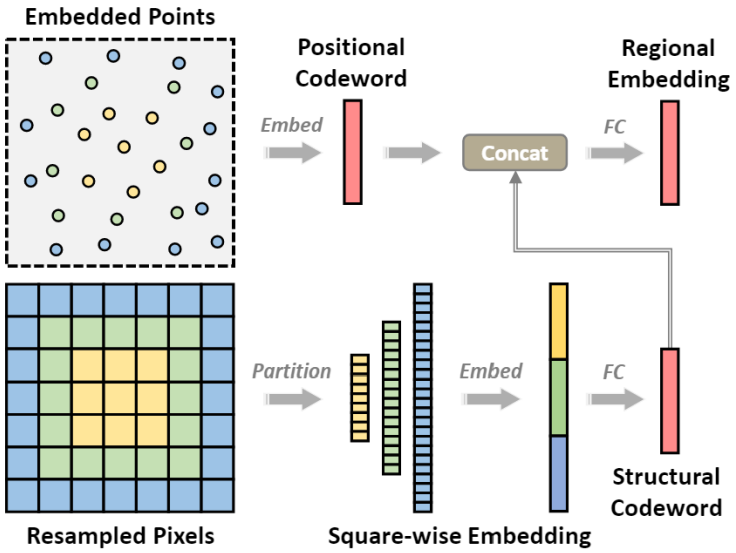

图 6. CSConv 架构示意图 [9]

Flattening-Net [9] 针对 PGI 提出了 CSConv 操作，旨在高效地提取 PGI 的特征，其结构如图 6所示。对于某个 PGI，首先将其以同心正方形的方式划分为内、中、外三个部分。然后将这些部分视为一个点集，利用 PointNet [1] 提取它们各自的全局特征，并将这些特征按照内、中、外的顺序依次拼接起来。最后，通过一个单独的全连接层将它们融合，形成结构化编码。同时，以类似的方式处理所有像素对应的三维点坐标，从而得到位置编码。最终，将结构化编码与位置编码逐特征拼接起来，并利用全连接层融合特征，得到最终的区域编码。

4 复现细节

4.1 与已有开源代码对比

本次复现主要参考了 Flattening-Net [9] 的公开源码，并在此基础上，结合论文中关于点云重建任务的描述，增加了基于 PGI 重建点云的代码实现。然而，重建效果并未达到论文中所展示的理想状态。尽管采用某些方法可以得到与论文相似的较好效果，但这却偏离了重建任务的初衷。具体分析将在第 5 节中详细阐述。

4.2 实验环境搭建

本次实验依托于 Docker 容器技术和 VSCode 编辑器的 Dev Containers 扩展。所有实验所需的依赖项均已预先配置在 Dockerfile 中。用户只需下载并启动 Docker，同时安装 Dev Containers 扩展，VSCode 将自动完成实验环境的构建。

5 实验结果分析

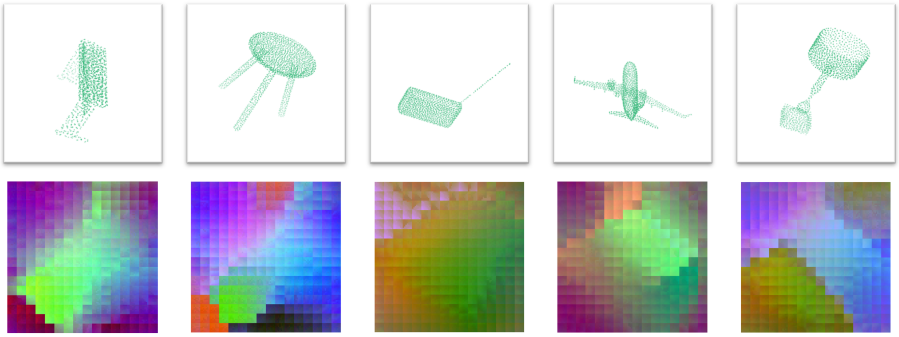

图 7. 点云参数化结果

本次实验首先实现了 Flattening-Net [9] 中最核心的部分，即点云的参数化方法，部分点云的参数化结果如图 7所示。此外还实现了基于 PGI 的点云分类任务，复现效果与原论文相比的结果如表 1所示。

表 1. 不同规模下点云分类结果对比 [9]

Point Counts	Mine OA(%)	Papar OA(%)
1024	91.69	93.40
2048	92.38	-

此外，本次实验还实现了点云重建的相关任务。若采用论文中所描述的重建方式，即仅重建稀疏点云，然后根据稀疏点云对原 PGI 进行采样以得到重建的完整点云，其结果如图所示。可以看到，确实可以得到较为理想的效果，但由于重建过程中过分依赖原始点云，因此这样的重建过程本身并没有太大意义。考虑到既然通过采样原始 PGI 就可以得到很好的效果，那么为什么不尝试直接重建 PGI，然后通过采样得到完整点云呢？基于此想法，本次实验尝

试了利用微调后的 SD-VAE 来重建 PGI，得到的结果与原始 PGI 的对比如图所示。而基于重建 PGI 采样得到的完整点云效果也如图所示，可以看到效果并不如预期那么理想。

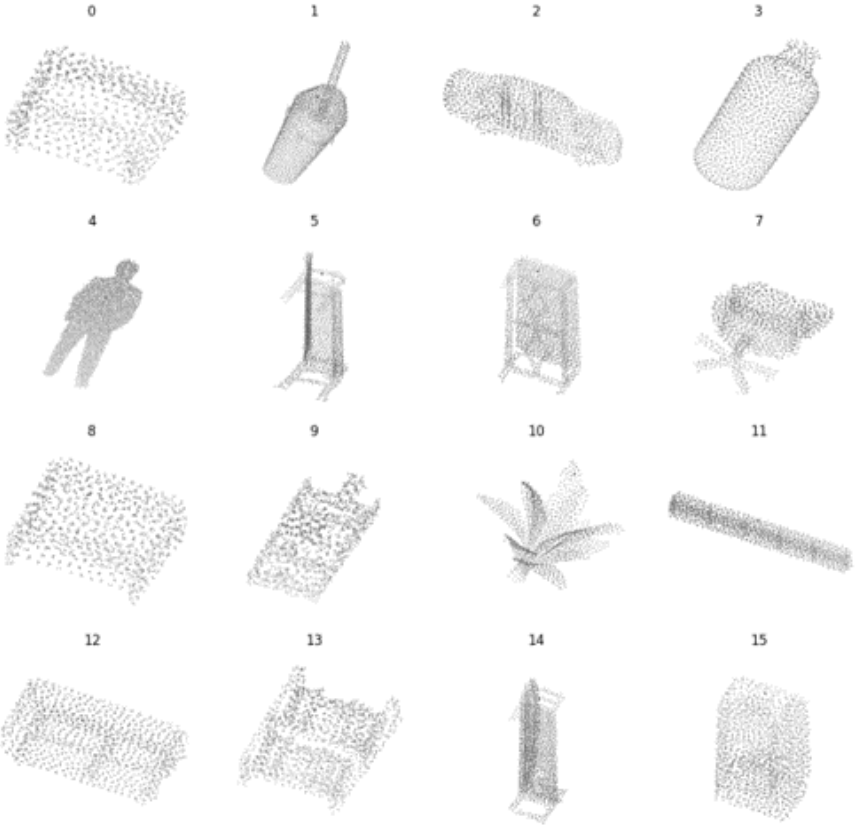

图 8. 基于原始 PGI 的采样结果

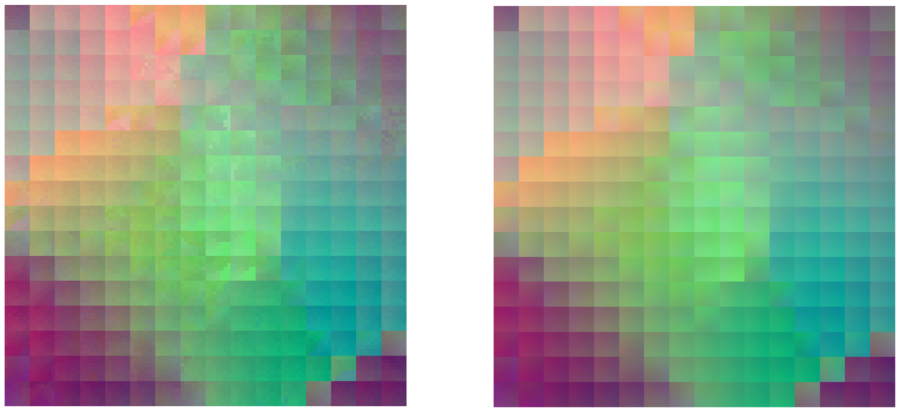

图 9. 原始 PGI（左） [9] 与重建 PGI（右）的结果对比

实际上，产生此结果的原因在于，尽管 PGI 在形式上与二维图像相似，但它本质上并不具备图像最关键的性质，即像素之间的连续性。虽然 PGI 从外观上看在块与块之间存在连续性，但仔细观察会发现，每个块内部本身是非常凌乱的。因此，适用于图像的二维卷积无法有效地提取相应的特征。这一点从重建结果中也可以得到印证，即 SD-VAE [3] 重建出的 PGI 中，块内部的像素之间呈现出连续变化的趋势。

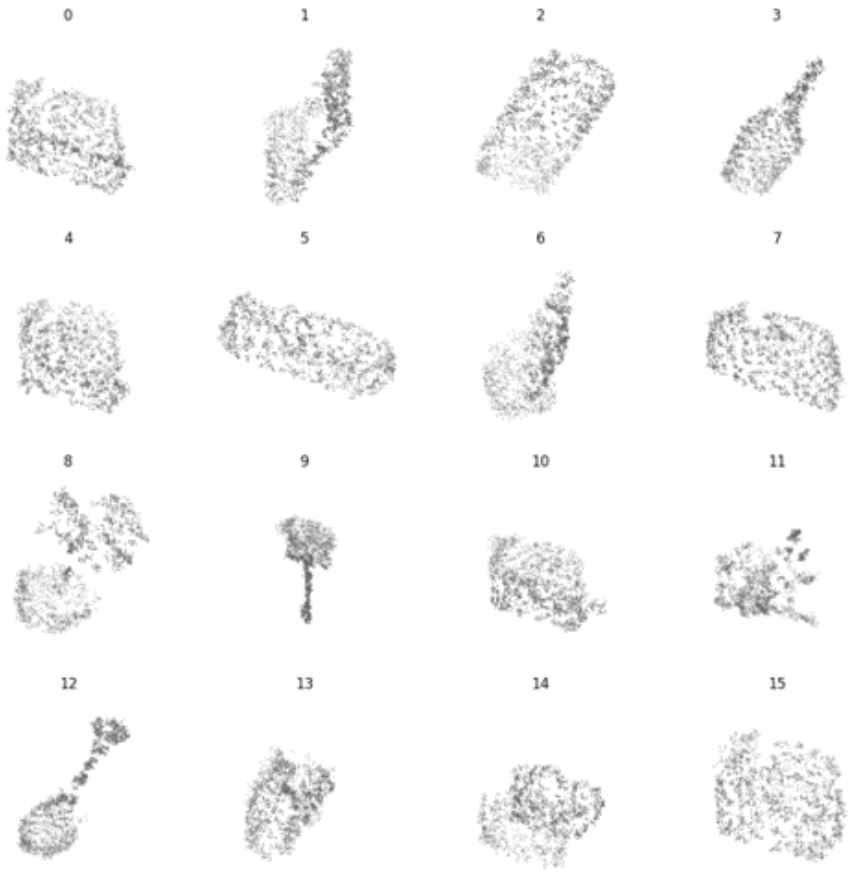

图 10. 基于重建 PGI 的采样结果

6 总结与展望

本报告深入阐述了 Flattening-Net [9] 点云参数化方法，旨在将三维点云高效地转换为二维表示，从而提升其分析效果。该方法首先利用表面展平模块将点云映射到二维平面，然后通过网格重采样模块进行均匀排列，最终生成点云几何图像（PGI）。为了充分挖掘 PGI 的特征，论文提出了 CSConv 操作，并巧妙地结合 PointNet [1] 来提取结构化编码和位置编码。在实验部分，我们成功复现了 Flattening-Net [9] 的点云参数化算法，将三维点云转换为 PGI，并在此基础上实现了点云分类任务。虽然分类效果与原论文存在一些差异，但整体表现仍然令人满意。此外，我们尝试了直接从 PGI 重建点云的方法，但由于 PGI 缺乏图像的像素连续性，重建效果未能达到预期，这反映出二维卷积在处理 PGI 特殊结构时存在一定的局限性。未来可以考虑采用 Transformer [4] 架构来突破 PGI 的局限性，但此策略在本质上与传统的基于“采样 + 查询”的方法几乎没有任何差别。

参考文献

[1] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016.

[2] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, AidanN. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Neural Information Processing Systems,Neural Information Processing Systems*, 2017.

[5] Peng-Shuai Wang. Octformer: Octree-based transformers for 3D point clouds. *ACM Transactions on Graphics (SIGGRAPH)*, 42(4), 2023.

[6] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 2019.

[7] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. In *CVPR*, 2024.

[8] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. In *NeurIPS*, 2022.

[9] Qijian Zhang, Junhui Hou, Yue Qian, Yiming Zeng, Juyong Zhang, and Ying He. Flattening-net: Deep regular 2d representation for 3d point cloud analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[10] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.