PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation

PointNet：用于 3D 分类和分割的点集深度学习

Charles R. Qi* Hao Su* Kaichun Mo Leonidas J. Guibas

Charles R. Qi* Hao Su* Kaichun Mo Leonidas J. Guibas

Stanford University 斯坦福大学

Abstract 抽象

Point cloud is an important type of geometric data structure. Due to its irregular format, most researchers transform such data to regular 3D voxel grids or collections of images. This, however, renders data unnecessarily voluminous and causes issues. In this paper, we design a novel type of neural network that directly consumes point clouds, which well respects the permutation invariance of points in the input. Our network, named PointNet, provides a unified architecture for applications ranging from object classification, part segmentation, to scene semantic parsing. Though simple, PointNet is highly efficient and effective. Empirically, it shows strong performance on par or even better than state of the art. Theoretically, we provide analysis towards understanding of what the network has learnt and why the network is robust with respect to input perturbation and corruption.

点云是一种重要的几何数据结构。由于其格式不规则，大多数研究人员将此类数据转换为常规的 3D 体素网格或图像集合。但是，这会使数据变得不必要地庞大并导致问题。在本文中，我们设计了一种直接使用点云的新型神经网络，它很好地尊重了输入中点的排列不变性。我们的网络名为 PointNet，为从对象分类、零件分割到场景语义解析的应用程序提供统一的架构。虽然简单，但 PointNet 非常高效。从经验上讲，它显示出与最先进的技术相当甚至更好的强劲性能。从理论上讲，我们提供分析以了解网络学到了什么，以及为什么网络在输入扰动和损坏方面是稳健的。

1 Introduction 1 介绍

[†]

[†]* indicates equal contributions.

In this paper we explore deep learning architectures capable of reasoning about 3D geometric data such as point clouds or meshes. Typical convolutional architectures require highly regular input data formats, like those of image grids or 3D voxels, in order to perform weight sharing and other kernel optimizations. Since point clouds or meshes are not in a regular format, most researchers typically transform such data to regular 3D voxel grids or collections of images (e.g, views) before feeding them to a deep net architecture. This data representation transformation, however, renders the resulting data unnecessarily voluminous — while also introducing quantization artifacts that can obscure natural invariances of the data.

在本文中，我们探讨了能够推理 3D 几何数据（如点云或网格）的深度学习架构。典型的卷积架构需要高度规则的输入数据格式，例如图像网格或 3D 体素的格式，以便执行权重共享和其他内核优化。由于点云或网格不是常规格式，因此大多数研究人员通常会将此类数据转换为常规的 3D 体素网格或图像集合（例如视图），然后再将其馈送到深度网络架构。然而，这种数据表示转换使生成的数据变得不必要地庞大，同时还引入了量化伪影，可能会掩盖数据的自然不变性。

For this reason we focus on a different input representation for 3D geometry using simply point clouds – and name our resulting deep nets *PointNets*. Point clouds are simple and unified structures that avoid the combinatorial irregularities and complexities of meshes, and thus are easier to learn from. The PointNet, however, still has to respect the fact that a point cloud is just a set of points and therefore invariant to permutations of its members, necessitating certain symmetrizations in the net computation. Further invariances to rigid motions also need to be considered.

出于这个原因，我们专注于使用简单的点云为 3D 几何图形提供不同的输入表示，并将我们得到的深网命名为 *PointNets*。点云是简单而统一的结构，可避免网格的组合不规则性和复杂性，因此更易于学习。然而，PointNet 仍然必须遵守这样一个事实，即点云只是一组点，因此不受其成员排列的影响，因此在网络计算中需要一定的对称性。还需要考虑刚性运动的进一步不变性。

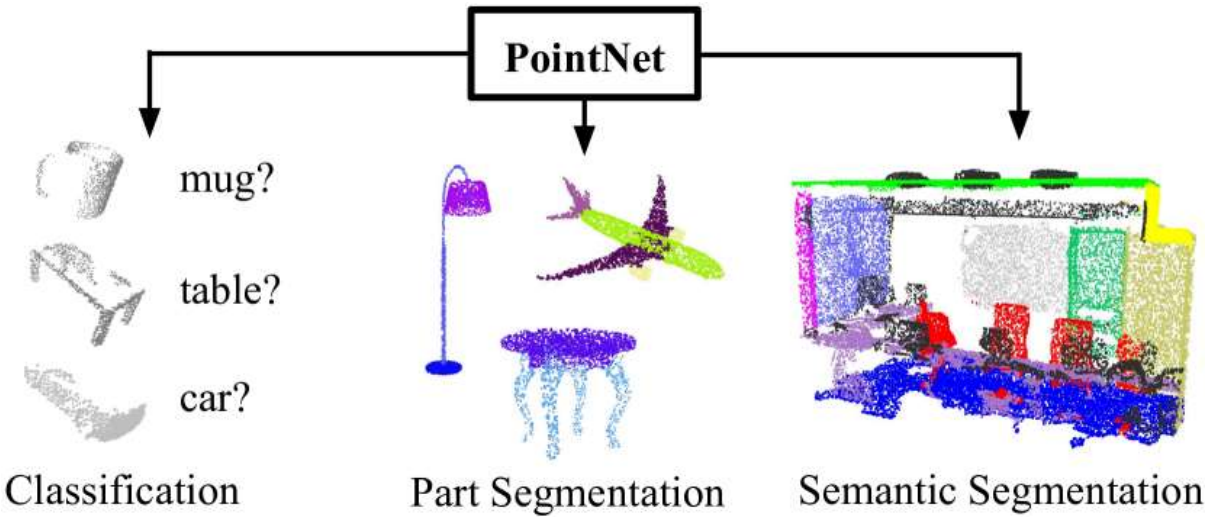


Figure 1: **Applications of PointNet.** We propose a novel deep net architecture that consumes raw point cloud (set of points) without voxelization or rendering. It is a unified architecture that learns both global and local point features, providing a simple, efficient and effective approach for a number of 3D recognition tasks.

图 1： **PointNet** 的应用。我们提出了一种新颖的深度网络架构，它使用原始点云（一组点）而不进行体素化或渲染。它是一个统一的架构，可以学习全局和局部点特征，为许多 3D 识别任务提供了一种简单、高效和有效的方法。

Our PointNet is a unified architecture that directly takes point clouds as input and outputs either class labels for the entire input or per point segment/part labels for each point of the input. The basic architecture of our network is surprisingly simple as in the initial stages each point is processed identically and independently. In the basic setting each point is represented by just its three coordinates (x, y, z). Additional dimensions may be added by computing normals and other local or global features.

我们的 PointNet 是一个统一的架构，它直接将点云作为输入，并输出整个输入的类标签或输入的每个点的点段/部分标签。我们网络的基本架构非常简单，因为在初始阶段，每个点都以相同且独立的方式进行处理。在基本设置中，每个点仅由其 三个坐标 (x, y, z) 表示。可以通过计算法线和其他局部或全局特征来添加其他维度。

Key to our approach is the use of a single symmetric function, max pooling. Effectively the network learns a set of optimization functions/criteria that select interesting or informative points of the point cloud and encode the reason for their selection. The final fully connected layers of the network aggregate these learnt optimal values into the global descriptor for the entire shape as mentioned above (shape classification) or are

used to predict per point labels (shape segmentation).

我们方法的关键是使用单个对称函数，即最大池化。实际上，网络学习了一组优化函数/标准，这些函数/标准选择点云中有趣或信息丰富的点，并对选择它们的原因进行编码。如上所述，网络的最终全连接层将这些学习到的最优值聚合到整个形状的全局描述符中（形状分类），或用于预测每个点的标签（形状分割）。

Our input format is easy to apply rigid or affine transformations to, as each point transforms independently. Thus we can add a data-dependent spatial transformer network that attempts to canonicalize the data before the PointNet processes them, so as to further improve the results.

我们的输入格式很容易应用刚性或仿射变换，因为每个点都是独立变换的。因此，我们可以添加一个数据依赖型空间转换器网络，该网络尝试在 PointNet 处理数据之前将数据规范化，从而进一步改进结果。

We provide both a theoretical analysis and an experimental evaluation of our approach. We show that our network can approximate any set function that is continuous. More interestingly, it turns out that our network learns to summarize an input point cloud by a sparse set of key points, which roughly corresponds to the skeleton of objects according to visualization. The theoretical analysis provides an understanding why our PointNet is highly robust to small perturbation of input points as well as to corruption through point insertion (outliers) or deletion (missing data).

我们提供了对我们方法的理论分析和实验评估。我们表明，我们的网络可以近似任何连续的集合函数。更有趣的是，事实证明，我们的网络学会了通过一组稀疏的关键点来总结输入点云，根据可视化，这大致对应于对象的骨架。理论分析有助于理解为什么我们的 PointNet 对输入点的小扰动以及通过点插入（异常值）或删除（缺失数据）造成的损坏具有高度鲁棒性。

On a number of benchmark datasets ranging from shape classification, part segmentation to scene segmentation, we experimentally compare our PointNet with state-of-the-art approaches based upon multi-view and volumetric representations. Under a unified architecture, not only is our PointNet much faster in speed, but it also exhibits strong performance on par or even better than state of the art.

在从形状分类、零件分割到场景分割的许多基准数据集上，我们实验性地将我们的 PointNet 与基于多视图和体积表示的最先进方法进行比较。在统一的架构下，我们的 PointNet 不仅速度更快，而且性能相当，甚至优于最先进的技术。

The key contributions of our work are as follows:

我们工作的主要贡献如下：

- We design a novel deep net architecture suitable for consuming unordered point sets in 3D;

  我们设计了一种新颖的深度网络架构，适用于在 3D 中使用无序点集;

- We show how such a net can be trained to perform 3D shape classification, shape part segmentation and scene semantic parsing tasks;

  我们展示了如何训练这样的网络来执行 3D 形状分类、形状部分分割和场景语义解析任务;

- We provide thorough empirical and theoretical analysis on the stability and efficiency of our method;

  我们对方法的稳定性和效率进行了全面的实证和理论分析;

- We illustrate the 3D features computed by the selected neurons in the net and develop intuitive explanations for its performance.

我们说明了网络中所选神经元计算的 3D 特征，并对其性能进行了直观的解释。

The problem of processing unordered sets by neural nets is a very general and fundamental problem – we expect that our ideas can be transferred to other domains as well.

神经网络处理无序集合的问题是一个非常普遍和基本的问题——我们希望我们的想法也可以转移到其他领域。

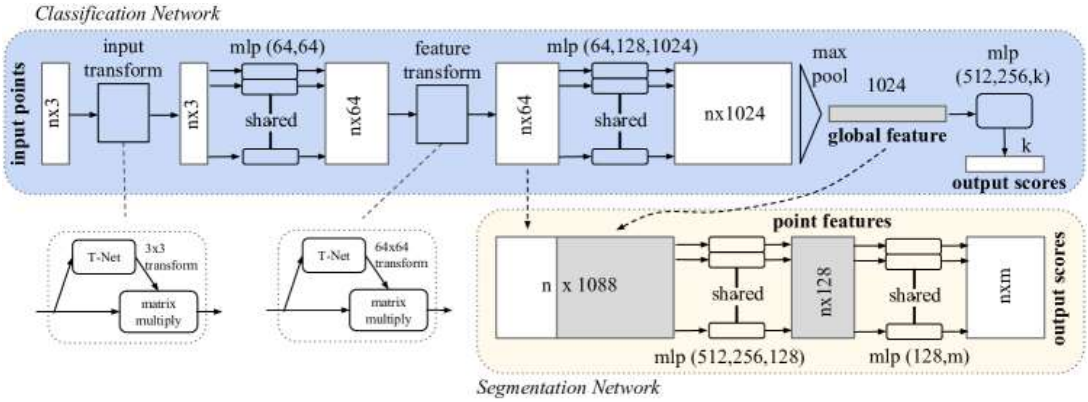


Figure 2: **PointNet Architecture.** The classification network takes n points as input, applies input and feature transformations, and then aggregates point features by max pooling. The output is classification scores for k classes. The segmentation network is an extension to the classification net. It concatenates global and local features and outputs per point scores. "mlp" stands for multi-layer perceptron, numbers in bracket are layer sizes. Batchnorm is used for all layers with ReLU. Dropout layers are used for the last mlp in classification net.

图 2： **PointNet** 架构。分类网络将点作为 n 输入，应用输入和特征转换，然后通过最大池化聚合点特征。输出是类的 k 分类分数。分段网络是分类网络的扩展。它连接全局和局部特征，并输出每个分数。"mlp" 代表多层感知器，括号中的数字是层大小。Batchnorm 用于具有 ReLU 的所有层。Dropout 层用于分类 net 中的最后一个 mlp。

2 Related Work 阿拉伯数字 相关工作

Point Cloud Features 点云功能

Most existing features for point cloud are handcrafted towards specific tasks. Point features often encode certain statistical properties of points and are designed to be invariant to certain transformations, which are typically classified as intrinsic [2, 24, 3] or extrinsic [20, 19, 14, 10, 5]. They can also be categorized as local features and global features. For a specific task, it is not trivial to find the optimal feature combination.

点云的大多数现有功能都是针对特定任务手工制作的。点特征通常对点的某些统计属性进行编码，并被设计为对某些变换不变，这些变换通常分为内在 [2， 24， 3] 或外内 [20， 19， 14， 10， 5]。它们也可以分为局部特征和全局特征。对于特定任务，找到最佳特征组合并非易事。

Deep Learning on 3D Data

3D 数据深度学习

3D data has multiple popular representations, leading to various approaches for learning. *Volumetric CNNs:* [28, 17, 18] are the pioneers applying 3D convolutional neural networks on voxelized shapes. However, volumetric representation is constrained by its resolution due to data sparsity and computation cost of 3D convo-

lution. FPNN [13] and Vote3D [26] proposed special methods to deal with the sparsity problem; however, their operations are still on sparse volumes, it's challenging for them to process very large point clouds. *Multiview CNNs:* [23, 18] have tried to render 3D point cloud or shapes into 2D images and then apply 2D conv nets to classify them. With well engineered image CNNs, this line of methods have achieved dominating performance on shape classification and retrieval tasks [21]. However, it's nontrivial to extend them to scene understanding or other 3D tasks such as point classification and shape completion. *Spectral CNNs:* Some latest works [4, 16] use spectral CNNs on meshes. However, these methods are currently constrained on manifold meshes such as organic objects and it's not obvious how to extend them to non-isometric shapes such as furniture. *Feature-based DNNs:* [6, 8] firstly convert the 3D data into a vector, by extracting traditional shape features and then use a fully connected net to classify the shape. We think they are constrained by the representation power of the features extracted.

3D 数据具有多种流行的表示形式，从而产生了多种学习方法。 体积 *CNN*：[28， 17， 18] 是将 3D 卷积神经网络应用于体素化形状的先驱。然而，由于数据稀疏性和 3D 卷积的计算成本，体积表示受到其分辨率的限制。FPNN [13] 和 Vote3D [26] 提出了处理稀疏性问题的特殊方法;但是，他们的作仍然在稀疏的卷上，处理非常大的点云对他们来说是一个挑战。 多视图 *CNN*：[23， 18] 尝试将 3D 点云或形状渲染成 2D 图像，然后应用 2D 卷积网络对它们进行分类。凭借精心设计的图像 CNN，这一系列方法在形状分类和检索任务中取得了主导性能 [21]。但是，将它们扩展到场景理解或其他 3D 任务（例如点分类和形状完成）并非易事。 光谱 *CNN*：一些最新的工作 [4， 16] 在网格上使用光谱 CNN。但是，这些方法目前受限于流形网格（如有机对象），并且不清楚如何将它们扩展到非等距形状（如家具）。 基于特征的 *DNN*：[6， 8] 首先通过提取传统的形状特征，将 3D 数据转换为向量，然后使用全连接网络对形状进行分类。我们认为它们受到提取特征的表示能力的限制。

Deep Learning on Unordered Sets

无序集上的深度学习

From a data structure point of view, a point cloud is an unordered set of vectors. While most works in deep learning focus on regular input representations like sequences (in speech and language processing), images and volumes (video or 3D data), not much work has been done in deep learning on point sets.

从数据结构的角度来看，点云是一组无序的向量。虽然深度学习中的大多数工作都集中在常规输入表示上，如序列（在语音和语言处理中）、图像和体积（视频或 3D 数据），但在点集的深度学习方面所做的工作并不多。

One recent work from Oriol Vinyals et al [25] looks into this problem. They use a read-process-write network with attention mechanism to consume unordered input sets and show that their network has the ability to sort numbers. However, since their work focuses on generic sets and NLP applications, there lacks the role of geometry in the sets.

Oriol Vinyals 等人 [25] 最近的一项工作研究了这个问题。他们使用带有注意力机制的读-处理-写网络来消费无序的输入集，并表明他们的网络具有对数字进行排序的能力。但是，由于他们的工作侧重于泛型集和 NLP 应用程序，因此在集中缺少几何图形的作用。

3 Problem Statement

3 问题陈述

We design a deep learning framework that directly consumes unordered point sets as inputs. A point cloud is represented as a set of 3D points $\{P_i| \ i = 1, ..., n\}$, where each point P_i is a vector of its (x, y, z) coordinate plus extra feature channels such as color, normal etc. For simplicity and clarity, unless otherwise noted, we only

use the (x, y, z) coordinate as our point's channels.

我们设计了一个深度学习框架，直接使用无序的点集作为输入。点云表示为一组 3D 点 $\{P_i | \ i = 1, ..., n\}$，其中每个点 P_i 是其 (x, y, z) 坐标加上额外特征通道（如颜色、法线等）的矢量。为简单明了，除非另有说明，否则我们只使用 (x, y, z) 坐标作为点的通道。

For the object classification task, the input point cloud is either directly sampled from a shape or pre-segmented from a scene point cloud. Our proposed deep network outputs k scores for all the k candidate classes. For semantic segmentation, the input can be a single object for part region segmentation, or a sub-volume from a 3D scene for object region segmentation. Our model will output $n \times m$ scores for each of the n points and each of the m semantic sub-categories.

对于对象分类任务，输入点云要么直接从形状中采样，要么从场景点云中预先分割。我们提议的深度网络输出 k 所有候选类的 k 分数。对于语义分割，输入可以是用于部分区域分割的单个对象，也可以是用于对象区域分割的 3D 场景中的子体积。我们的模型将输出 $n \times m$ 每个 n 点和每个 m 语义子类别的分数。

4 Deep Learning on Point Sets

4 基于点集的深度学习

The architecture of our network (Sec 4.2) is inspired by the properties of point sets in $\mathbb{R}^n$ (Sec 4.1).

我们网络的架构（第 4.2 节）的灵感来自（第 4.1 节）中 $\mathbb{R}^n$ 点集的属性。

4.1 Properties of Point Sets in $\mathbb{R}^n$

4.1 点集的属性 $\mathbb{R}^n$

Our input is a subset of points from an Euclidean space. It has three main properties:

我们的输入是来自欧几里得空间的点的子集。它有三个主要属性：

- Unordered. Unlike pixel arrays in images or voxel arrays in volumetric grids, point cloud is a set of points without specific order. In other words, a network that consumes N 3D point sets needs to be invariant to $N!$ permutations of the input set in data feeding order.

  无序。 与图像中的像素阵列或体积网格中的体素阵列不同，点云是一组没有特定顺序的点。换句话说，使用 N 3D 点集的网络需要对 $N!$ 数据馈送顺序中输入集的排列保持不变。

- Interaction among points. The points are from a space with a distance metric. It means that points are not isolated, and neighboring points form a meaningful subset. Therefore, the model needs to be able to capture local structures from nearby points, and the combinatorial interactions among local structures.

  点之间的交互。这些点来自具有距离度量的空间。这意味着点不是孤立的，相邻的点会形成一个有意义的子集。因此，该模型需要能够从附近的点捕获局部结构，以及局部结构之间的组合交互。

- Invariance under transformations. As a geometric object, the learned representation of the point set should be invariant to certain transformations. For example, rotating and translating points all together

should not modify the global point cloud category nor the segmentation of the points.

变换下的不变性。作为几何对象，点集的学习表示应该对某些转换不变。例如，一起旋转和平移点不应修改全局点云类别或点的分割。

4.2 PointNet Architecture

4.2 PointNet 架构

Our full network architecture is visualized in Fig 2, where the classification network and the segmentation network share a great portion of structures. Please read the caption of Fig 2 for the pipeline.

我们完整的网络架构如图 2 所示，其中分类网络和分割网络共享很大一部分结构。请阅读图 2 的标题，了解管道。

Our network has three key modules: the max pooling layer as a symmetric function to aggregate information from all the points, a local and global information combination structure, and two joint alignment networks that align both input points and point features.

我们的网络有三个关键模块：作为对称函数的最大池化层，用于聚合来自所有点的信息，局部和全局信息组合结构，以及两个对齐输入点和点特征的联合对齐网络。

We will discuss our reason behind these design choices in separate paragraphs below.

我们将在下面的单独段落中讨论这些设计选择背后的原因。

Symmetry Function for Unordered Input

无序输入的对称函数

In order to make a model invariant to input permutation, three strategies exist: 1) sort input into a canonical order; 2) treat the input as a sequence to train an RNN, but augment the training data by all kinds of permutations; 3) use a simple symmetric function to aggregate the information from each point. Here, a symmetric function takes n vectors as input and outputs a new vector that is invariant to the input order. For example, $+$ and $*$ operators are symmetric binary functions.

为了使模型不受输入排列的影响，存在三种策略：1） 将输入排序为规范顺序;2） 将输入视为训练 RNN 的序列，但通过各种排列来增强训练数据;3） 使用简单的对称函数来聚合来自每个点的信息。在这里，对称函数将 n 向量作为输入，并输出一个对输入阶数不变的新向量。例如， $+$ and $*$ 运算符是对称二进制函数。

While sorting sounds like a simple solution, in high dimensional space there in fact does not exist an ordering that is stable w.r.t. point perturbations in the general sense. This can be easily shown by contradiction. If such an ordering strategy exists, it defines a bijection map between a high-dimensional space and a $1d$ real line. It is not hard to see, to require an ordering to be stable w.r.t point perturbations is equivalent to requiring that this map preserves spatial proximity as the dimension reduces, a task that cannot be achieved in the general case. Therefore, sorting does not fully resolve the ordering issue, and it's hard for a network to learn a consistent mapping from input to output as the ordering issue persists. As shown in experiments (Fig 5), we find that applying a MLP directly on the sorted point set performs poorly, though slightly better than directly

processing an unsorted input.

虽然排序听起来像是一个简单的解决方案，但在高维空间中，实际上不存在一般意义上的稳定点扰动的排序。这很容易通过自相矛盾来证明。如果存在这样的排序策略，则它定义了高维空间和 $1d$ 实线之间的双射映射。不难看出，要求 Ordering 对于 Point Persharbions 保持稳定，相当于要求该映射在维度减小时保持空间接近性，这在一般情况下是无法完成的。因此，排序并不能完全解决排序问题，并且由于排序问题仍然存在，网络很难学习从输入到输出的一致映射。如实验所示（图 5），我们发现直接在排序的点集上应用 MLP 的性能很差，但比直接处理未排序的输入略好。

The idea to use RNN considers the point set as a sequential signal and hopes that by training the RNN with randomly permuted sequences, the RNN will become invariant to input order. However in "OrderMatters" [25] the authors have shown that order does matter and cannot be totally omitted. While RNN has relatively good robustness to input ordering for sequences with small length (dozens), it's hard to scale to thousands of input elements, which is the common size for point sets. Empirically, we have also shown that model based on RNN does not perform as well as our proposed method (Fig 5).

使用 RNN 的想法将点集视为顺序信号，并希望通过使用随机排列序列训练 RNN，RNN 将变得不受输入顺序的影响。然而，在"OrderMatters"[25]中，作者已经表明顺序确实很重要，不能完全忽略。虽然 RNN 对小长度（几十个）序列的输入排序具有相对较好的鲁棒性，但很难扩展到数千个输入元素，这是点集的常见大小。从实证上讲，我们还表明，基于 RNN 的模型的性能不如我们提出的方法（图 5）。

Our idea is to approximate a general function defined on a point set by applying a symmetric function on transformed elements in the set:

我们的想法是通过对集合中变换的元素应用对称函数来近似在点集上定义的一般函数：

$$f(\{x_1, ..., x_n\}) \approx g(h(x_1), ..., h(x_n)), \tag{1}$$

where $f : 2^{\mathbb{R}^N} \to \mathbb{R}$, $h : \mathbb{R}^N \to \mathbb{R}^K$ and $g : \underbrace{\mathbb{R}^K \times \cdots \times \mathbb{R}^K}_{n} \to \mathbb{R}$ is a symmetric function.

其中 $f : 2^{\mathbb{R}^N} \to \mathbb{R}$ ，和 $h : \mathbb{R}^N \to \mathbb{R}^K$ $g : \underbrace{\mathbb{R}^K \times \cdots \times \mathbb{R}^K}_{n} \to \mathbb{R}$ 是对称函数。

Empirically, our basic module is very simple: we approximate h by a multi-layer perceptron network and g by a composition of a single variable function and a max pooling function. This is found to work well by experiments. Through a collection of h, we can learn a number of f's to capture different properties of the set.

从经验上讲，我们的基本模块非常简单：我们 h 通过多层感知器网络以及 g 单个变量函数和最大池化函数的组合进行近似。通过实验发现这效果很好。通过 h 的集合 ，我们可以学习许多 f 来捕获集合的不同属性。

While our key module seems simple, it has interesting properties (see Sec 5.3) and can achieve strong performace (see Sec 5.1) in a few different applications. Due to the simplicity of our module, we are also able to provide theoretical analysis as in Sec 4.3.

虽然我们的 key 模块看起来很简单，但它具有有趣的特性（参见 5.3 节），并且可以在一些不同的应用程序中实现强大的性能（参见 5.1 节）。由于我们模块的简单性，我们还能够提供 Sec 4.3 中的理论分析。

Local and Global Information Aggregation

本地和全球信息聚合

The output from the above section forms a vector $[f_1, ..., f_K]$, which is a global signature of the input set. We can easily train a SVM or multi-layer perceptron classifier on the shape global features for classification. However, point segmentation requires a combination of local and global knowledge. We can achieve this by a

simple yet highly effective manner.

上一节的输出形成一个 vector $[f_1, ..., f_K]$，它是 input set 的全局签名。我们可以轻松地在形状全局特征上训练 SVM 或多层感知器分类器进行分类。但是，点分割需要本地和全球知识的结合。我们可以通过一种简单而高效的方式来实现这一目标。

Our solution can be seen in Fig 2 (*Segmentation Network*). After computing the global point cloud feature vector, we feed it back to per point features by concatenating the global feature with each of the point features. Then we extract new per point features based on the combined point features - this time the per point feature is aware of both the local and global information.

我们的解决方案如图 2 （*Segmentation Network*） 所示。在计算出全局点云特征向量后，我们通过将全局特征与每个点特征连接起来，将其反馈给每个点特征。然后，我们根据组合的点特征提取新的每点特征 - 这一次，每点特征同时了解局部和全局信息。

With this modification our network is able to predict per point quantities that rely on both local geometry and global semantics. For example we can accurately predict per-point normals (fig in supplementary), validating that the network is able to summarize information from the point's local neighborhood. In experiment session, we also show that our model can achieve state-of-the-art performance on shape part segmentation and scene segmentation.

通过这种修改，我们的网络能够预测依赖于局部几何和全局语义的每点数量。例如，我们可以准确地预测每个点的法线（补充中的图），验证网络是否能够从该点的局部邻域汇总信息。在实验过程中，我们还表明我们的模型可以在形状零件分割和场景分割方面实现最先进的性能。

Joint Alignment Network 联合对齐网络

The semantic labeling of a point cloud has to be invariant if the point cloud undergoes certain geometric transformations, such as rigid transformation. We therefore expect that the learnt representation by our point set is invariant to these transformations.

如果点云经历某些几何变换（例如刚性变换），则点云的语义标注必须是不变的。因此，我们期望我们的点集的学习表示对这些转换是不变的。

A natural solution is to align all input set to a canonical space before feature extraction. Jaderberg et al. [9] introduces the idea of spatial transformer to align 2D images through sampling and interpolation, achieved by a specifically tailored layer implemented on GPU.

一个自然的解决方案是在特征提取之前将所有 input set 对齐到一个规范空间。Jaderberg 等人 [9] 引入了空间转换器的概念，通过采样和插值来对齐 2D 图像，这是通过在 GPU 上实现的专门定制的层来实现的。

Our input form of point clouds allows us to achieve this goal in a much simpler way compared with [9]. We do not need to invent any new layers and no alias is introduced as in the image case. We predict an affine transformation matrix by a mini-network (T-net in Fig 2) and directly apply this transformation to the coordinates of input points. The mini-network itself resembles the big network and is composed by basic modules of point independent feature extraction, max pooling and fully connected layers. More details about the T-net are in the supplementary.

与 [9] 相比，我们的点云输入形式使我们能够以更简单的方式实现这个目标。我们不需要发明任何新图层，也没有像图像那样引入别名。我们通过微型网络（图 2 中的 T-net）预测仿射变换矩阵，并将此变换直接应用于输入点的坐标。迷你网络本身类似于大网络，由点独立特征提取、最大池化和全连接层等基本模块组成。有关 T-net 的更多详细信息，请参阅补充内容。

This idea can be further extended to the alignment of feature space, as well. We can insert another alignment network on point features and predict a feature transformation matrix to align features from different input

point clouds. However, transformation matrix in the feature space has much higher dimension than the spatial transform matrix, which greatly increases the difficulty of optimization. We therefore add a regularization term to our softmax training loss. We constrain the feature transformation matrix to be close to orthogonal matrix:

这个想法也可以进一步扩展到特征空间的对齐。我们可以在点特征上插入另一个对齐网络，并预测特征变换矩阵以对齐来自不同输入点云的特征。但是，特征空间中的变换矩阵比空间变换矩阵的维数高得多，这大大增加了优化的难度。因此，我们在 softmax 训练损失中添加了一个正则化项。我们将特征变换矩阵约束为接近正交矩阵：

$$L_{reg} = \|I - AA^T\|_F^2, \tag{2}$$

where A is the feature alignment matrix predicted by a mini-network. An orthogonal transformation will not lose information in the input, thus is desired. We find that by adding the regularization term, the optimization becomes more stable and our model achieves better performance.

其中 A 是 mini-network 预测的特征对齐矩阵。正交变换不会丢失输入中的信息，因此是需要的。我们发现，通过添加正则化项，优化变得更加稳定，我们的模型获得了更好的性能。

4.3 Theoretical Analysis

4.3 理论分析

Universal approximation 通用近似

We first show the universal approximation ability of our neural network to continuous set functions. By the continuity of set functions, intuitively, a small perturbation to the input point set should not greatly change the function values, such as classification or segmentation scores.

我们首先展示了神经网络对连续集合函数的通用逼近能力。通过集合函数的连续性，直观地说，对输入点集的微小扰动应该不会大大改变函数值，例如分类或分割分数。

Formally, let $\mathcal{X} = \{S : S \subseteq [0,1]^m \text{ and } |S| = n\}$, $f : \mathcal{X} \to \mathbb{R}$ is a continuous set function on $\mathcal{X}$ w.r.t to Hausdorff distance $d_H(\cdot,\cdot)$, i.e., $\forall \epsilon > 0, \exists \delta > 0$, for any $S, S^{'} \in \mathcal{X}$, if $d_H(S, S^{'}) < \delta$, then $|f(S) - f(S^{'})| < \epsilon$. Our theorem says that f can be arbitrarily approximated by our network given enough neurons at the max pooling layer, i.e., K in (1) is sufficiently large.

形式上，设 ，是 $\mathcal{X} = \{S : S \subseteq [0,1]^m \text{ and } |S| = n\}$ w.r.t 到 Hausdorff 距离 $d_H(\cdot,\cdot)$ 的连续集合函数，即 $\forall \epsilon > 0, \exists \delta > 0$ ，对于任何 $S, S^{'} \in \mathcal{X}$ ，如果 $d_H(S, S^{'}) < \delta$ ，则 $|f(S) - f(S^{'})| < \epsilon$ 。 $f : \mathcal{X} \to \mathbb{R}$ $\mathcal{X}$ 我们的定理说， f 只要在最大池化层（即 K （1） 中足够大）有足够的神经元，就可以用我们的网络任意近似。

Theorem 1. *Suppose $f : \mathcal{X} \to \mathbb{R}$ is a continuous set function w.r.t Hausdorff distance $d_H(\cdot,\cdot)$. $\forall \epsilon > 0$, $\exists$ a continuous function h and a symmetric function $g(x_1, ..., x_n) = \gamma \circ MAX$, such that for any $S \in \mathcal{X}$,*

$$\left| f(S) - \gamma\left(\underset{x_i \in S}{MAX} \{ h(x_i) \} \right) \right| < \epsilon$$

where $x_1, ..., x_n$ is the full list of elements in S ordered arbitrarily, γ is a continuous function, and MAX is a vector max operator that takes n vectors as input and returns a new vector of the element-wise maximum.

其中 $x_1, ..., x_n$ 是任意 S 排序的元素的完整列表， γ 是一个连续函数，*MAX* 是一个向量 *max* 运算符，它将 n 向量作为输入并返回元素最大值的新向量。

定理 1. 假设 f：X→R：f→XRf：\mathcal{X}\rightarrow\mathbb{R} 是一个连续集函数，w.r.t Hausdorff 距离 dH（·，·）下标dH··d_{H}（\cdot，\cdot）。∀ε>0for-allitalic-ε0\forall\epsilon>0，∃\存在一个连续函数 hhh 和一个对称函数 g（x1,...,xn）=γ∘MAXgsubscriptx1...下标xnγMAXg（x_{1}，\dots，x_{n}）=\gamma\circ\mbox{MAX}，使得对于任何 S∈XSXS\in\mathcal{X}，

The proof to this theorem can be found in our supplementary material. The key idea is that in the worst case the network can learn to convert a point cloud into a volumetric representation, by partitioning the space into equal-sized voxels. In practice, however, the network learns a much smarter strategy to probe the space, as we shall see in point function visualizations.

这个定理的证明可以在我们的补充材料中找到。关键思想是，在最坏的情况下，网络可以通过将空间划分为大小相等的体素来学习将点云转换为体积表示。然而，在实践中，网络学习了一种更智能的策略来探测空间，正如我们将在点函数可视化中看到的那样。

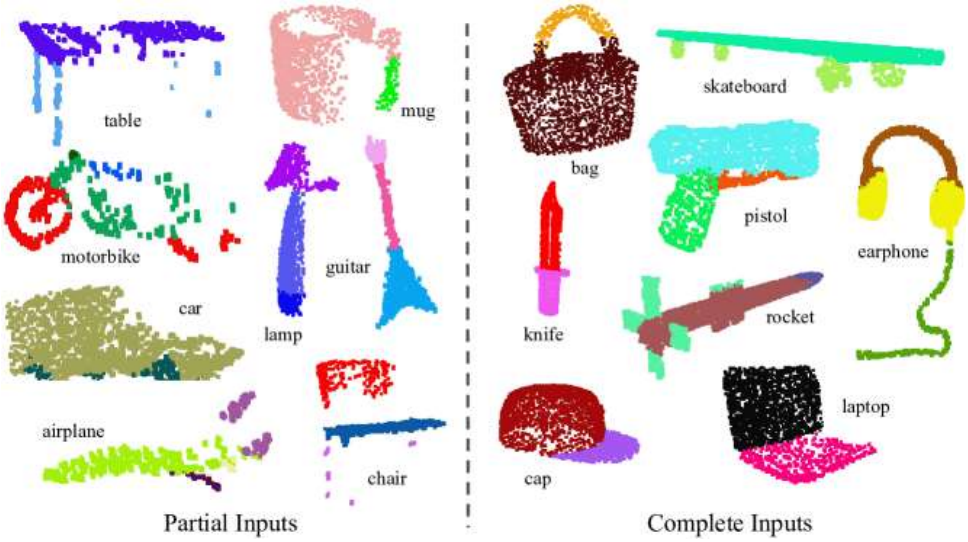


Figure 3: **Qualitative results for part segmentation.** We visualize the CAD part segmentation results across all 16 object categories. We show both results for partial simulated Kinect scans (left block) and complete ShapeNet CAD models (right block).

图 3：零件分割的定性结果。我们可视化了所有 16 个对象类别的 CAD 零件分割结果。我们显示了部分模拟 Kinect 扫描（左块）和完整 ShapeNet CAD 模型（右块）的结果。

Bottleneck dimension and stability

瓶颈尺寸和稳定性

Theoretically and experimentally we find that the expressiveness of our network is strongly affected by the dimension of the max pooling layer, i.e., K in (1). Here we provide an analysis, which also reveals properties related to the stability of our model.

从理论和实验上讲，我们发现我们网络的表达性受到最大池化层维度的强烈影响，即 （1） K 中。在这里，我们提供了一个分析，它还揭示了与模型稳定性相关的属性。

We define $\mathbf{u} = \underset{x_i \in S}{\mathrm{MAX}}\{h(x_i)\}$ to be the sub-network of f which maps a point set in $[0,1]^m$ to a K-dimensional vector. The following theorem tells us that small corruptions or extra noise points in the input set are not

likely to change the output of our network:

我们定义为 $\mathbf{u} = \underset{x_i \in S}{\mathrm{MAX}}\{h(x_i)\}$ 子网络 f，其子网络将一个点集 $[0,1]^m$ 映射到一个 K-维向量。以下定理告诉我们，输入集中的小损坏或额外的噪声点不太可能改变我们网络的输出：

Theorem 2. *Suppose* $\mathbf{u} : \mathcal{X} \to \mathbb{R}^K$ *such that* $\mathbf{u} = \underset{x_i \in S}{MAX}\{h(x_i)\}$ *and* $f = \gamma \circ \mathbf{u}$. *Then,*

(a) $\forall S, \exists\ \mathcal{C}_S, \mathcal{N}_S \subseteq \mathcal{X}$, $f(T) = f(S)$ *if* $\mathcal{C}_S \subseteq T \subseteq \mathcal{N}_S$;

$\forall S, \exists\ \mathcal{C}_S, \mathcal{N}_S \subseteq \mathcal{X}$ ， $f(T) = f(S)$ 如果 $\mathcal{C}_S \subseteq T \subseteq \mathcal{N}_S$;

(b) $|\mathcal{C}_S| \leq K$

定理 2. 假设 u：X→RK：u→XsuperscriptRK\mathbf{u}：\mathcal{X}\rightarrow\mathbb{R}^{K}，使得 u=MAXxi∈S{h（习）}usubscriptxiSMAXhsubscriptxi\mathbf{u}=\underset{x_{i}\in S}{\mbox{MAX}}\{h（x_{i}）\} 和 f=γ∘ufγuf=\gamma\circ\mathbf{u}。然后

We explain the implications of the theorem. (a) says that $f(S)$ is unchanged up to the input corruption if all points in $\mathcal{C}_S$ are preserved; it is also unchanged with extra noise points up to $\mathcal{N}_S$. (b) says that $\mathcal{C}_S$ only contains a bounded number of points, determined by K in (1). In other words, $f(S)$ is in fact totally determined by a finite subset $\mathcal{C}_S \subseteq S$ of less or equal to K elements. We therefore call $\mathcal{C}_S$ the *critical point set* of S and K the *bottleneck dimension* of f.

我们解释了该定理的含义。（a） 表示如果保留 中的所有 $\mathcal{C}_S$ 点，则在 input 损坏之前 $f(S)$ 保持不变;它也没有变化，额外的噪声点高达 $\mathcal{N}_S$ 。（b） 表示 $\mathcal{C}_S$ 仅包含由 K IN （1） 确定的有限点数。换句话说， $f(S)$ 实际上完全由小于或等于 K 元素的有限子集 $\mathcal{C}_S \subseteq S$ 决定。因此，我们将 的临界点集 S K 和 的瓶颈维度称为 f $\mathcal{C}_S$ 。

Combined with the continuity of h, this explains the robustness of our model w.r.t point perturbation, corruption and extra noise points. The robustness is gained in analogy to the sparsity principle in machine learning models. **Intuitively, our network learns to summarize a shape by a sparse set of key points.** In experiment section we see that the key points form the skeleton of an object.

结合 的 h 连续性，这解释了我们的模型在点扰动、损坏和额外噪声点方面的鲁棒性。鲁棒性是通过类比机器学习模型中的稀疏性原则获得的。直观地，我们的网络学会了通过一组稀疏的关键点来总结形状。在实验部分，我们看到关键点构成了物体的骨架。

5 Experiment 5 实验

	input 输入	#views	accuracy 准确性 avg. class 平均等级	accuracy 准确性 overall 整体
SPH [11]	mesh 网孔	-	68.2	-
3DShapeNets [28]	volume 卷	1	77.3	84.7
3D 快照网 [28]				
VoxNet [17]	volume 卷	12	83.0	85.9

Subvolume [18] 子卷 [18]	volume 卷	20	86.0	**89.2**
LFD [28]	image 图像	10	75.5	-
MVCNN [23]	image 图像	80	**90.1**	-
Ours baseline 我们的基线	point 点	-	72.6	77.4
Ours PointNet 我们的 PointNet	point 点	1	86.2	**89.2**

Table 1: **Classification results on ModelNet40.** Our net achieves state-of-the-art among deep nets on 3D input.

表 1： **ModelNet40** 上的分类结果。我们的网络在 3D 输入方面达到了最先进的深度网络。

Experiments are divided into four parts. First, we show PointNets can be applied to multiple 3D recognition tasks (Sec 5.1). Second, we provide detailed experiments to validate our network design (Sec 5.2). At last we visualize what the network learns (Sec 5.3) and analyze time and space complexity (Sec 5.4).

实验分为四个部分。首先，我们展示了 PointNet 可以应用于多个 3D 识别任务（第 5.1 节）。其次，我们提供详细的实验来验证我们的网络设计（第 5.2 节）。最后，我们将网络学到的内容可视化（第 5.3 节）并分析时间和空间复杂性（第 5.4 节）。

	mean 意味着	aero 航空	bag 袋	cap 头	car 汽车	chair 椅子	ear 耳朵 phone 电话	guitar 吉他	knife 刀	lamp 灯	laptop 笔记本	motor 发动机	mug 杯	pistol 手枪	rocket 火箭	skate 滑冰 board 板
# shapes # 形状		2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152
Wu [27] 吴 [27]	-	63.2	-	-	-	73.5	-	-	-	74.4	-	-	-	-	-	-
Yi [29] 彝 [29]	**81.4**	81.0	78.4	77.7	**75.7**	87.6	61.9	**92.0**	85.4	**82.5**	**95.7**	**70.6**	91.9	**85.9**	53.1	69.8
3DCNN	**79.4**	75.1	72.8	73.3	70.0	87.2	63.5	88.4	79.6	74.4	93.9	58.7	91.8	76.4	51.2	65.3
Ours 我们	**83.7**	**83.4**	**78.7**	**82.5**	74.9	**89.6**	**73.0**	91.5	**85.9**	80.8	95.3	65.2	**93.0**	81.2	**57.9**	**72.8**

Table 2: **Segmentation results on ShapeNet part dataset.** Metric is mIoU(%) on points. We compare with two traditional methods [27] and [29] and a 3D fully convolutional network baseline proposed by us. Our PointNet method

achieved the state-of-the-art in mIoU.

表 2： **ShapeNet** 零件数据集上的分割结果。度量是 mIoU（%） 的分数。我们与两种传统方法 [27] 和 [29] 以及我们提出的 3D 全卷积网络基线进行了比较。我们的 PointNet 方法实现了 mIoU 中最先进的技术。

5.1 Applications

5,1 应用

In this section we show how our network can be trained to perform 3D object classification, object part segmentation and semantic scene segmentation [1]

[1]More application examples such as correspondence and point cloud based CAD model retrieval are included in supplementary material.

. Even though we are working on a brand new data representation (point sets), we are able to achieve comparable or even better performance on benchmarks for several tasks.

在本节中，我们将展示如何训练我们的网络来执行 3D 对象分类、对象部分分割和语义场景分割 [1]

[1]More application examples such as correspondence and point cloud based CAD model retrieval are included in supplementary material.

。尽管我们正在开发一种全新的数据表示形式（点集），但我们能够在多项任务的基准上实现相当甚至更好的性能。

3D Object Classification 3D 对象分类

Our network learns global point cloud feature that can be used for object classification. We evaluate our model on the ModelNet40 [28] shape classification benchmark. There are 12,311 CAD models from 40 man-made object categories, split into 9,843 for training and 2,468 for testing. While previous methods focus on volumetric and mult-view image representations, we are the first to directly work on raw point cloud.

我们的网络学习可用于对象分类的全局点云功能。我们在 ModelNet40 [28] 形状分类基准上评估我们的模型。有来自 40 个人造对象类别的 12,311 个 CAD 模型，分为 9,843 个用于训练，2,468 个用于测试。虽然以前的方法侧重于体积和多视图图像表示，但我们是第一个直接处理原始点云的方法。

We uniformly sample 1024 points on mesh faces according to face area and normalize them into a unit sphere. During training we augment the point cloud on-the-fly by randomly rotating the object along the up-axis and jitter the position of each points by a Gaussian noise with zero mean and 0.02 standard deviation.

我们根据面面积在网格面上对 1024 个点进行统一采样，并将它们归一化为一个单位球体。在训练过程中，我们通过沿上轴随机旋转对象来动态增强点云，并通过平均值为零、标准差为 0.02 的高斯噪声使每个点的位置抖动。

In Table 1, we compare our model with previous works as well as our baseline using MLP on traditional features extracted from point cloud (point density, D2, shape contour etc.). Our model achieved state-of-the-art performance among methods based on 3D input (volumetric and point cloud). With only fully connected layers and max pooling, our net gains a strong lead in inference speed and can be easily parallelized in CPU as well. There is still a small gap between our method and multi-view based method (MVCNN [23]), which we

think is due to the loss of fine geometry details that can be captured by rendered images.

在表 1 中，我们将我们的模型与以前的工作以及使用 MLP 对从点云中提取的传统特征（点密度、D2、形状等值线等）的基线进行了比较。我们的模型在基于 3D 输入（体积和点云）的方法中取得了最先进的性能。由于只有完全连接的层和最大池化，我们的网络在推理速度方面取得了很大的领先优势，并且也可以在 CPU 中轻松并行化。我们的方法与基于多视图的方法 （MVCNN [23]）之间仍然存在很小的差距，我们认为这是由于渲染图像可以捕获的精细几何细节的损失。

3D Object Part Segmentation

3D 对象零件分割

Part segmentation is a challenging fine-grained 3D recognition task. Given a 3D scan or a mesh model, the task is to assign part category label (e.g. chair leg, cup handle) to each point or face.

零件分割是一项具有挑战性的精细 3D 识别任务。给定 3D 扫描或网格模型，任务是为每个点或面分配零件类别标签（例如椅子腿、杯柄）。

We evaluate on ShapeNet part data set from [29], which contains 16,881 shapes from 16 categories, annotated with 50 parts in total. Most object categories are labeled with two to five parts. Ground truth annotations are labeled on sampled points on the shapes.

我们评估了 [29] 的 ShapeNet 零件数据集，其中包含来自 16 个类别的 16,881 个形状，总共有 50 个零件进行注释。大多数对象类别都标有 2 到 5 个部分。真实值注释在形状上的采样点上进行标记。

We formulate part segmentation as a per-point classification problem. Evaluation metric is mIoU on points. For each shape S of category C, to calculate the shape's mIoU: For each part type in category C, compute IoU between groundtruth and prediction. If the union of groundtruth and prediction points is empty, then count part IoU as 1. Then we average IoUs for all part types in category C to get mIoU for that shape. To calculate mIoU for the category, we take average of mIoUs for all shapes in that category.

我们将零件分割表述为每点分类问题。评估指标是 mIoU on points。对于类别 C 的每个形状 S，要计算形状的 mIoU：对于类别 C 中的每个零件类型，计算真实值和预测之间的 IoU。如果 groundtruth 点和预测点的并集为空，则将部分 IoU 计为 1。然后，我们对类别 C 中所有零件类型的 IoU 进行平均，以获得该形状的 mIoU。为了计算该类别的 mIoU，我们取该类别中所有形状的 mIoU 的平均值。

In this section, we compare our segmentation version PointNet (a modified version of Fig 2, *Segmentation Network*) with two traditional methods [27] and [29] that both take advantage of point-wise geometry features and correspondences between shapes, as well as our own 3D CNN baseline. See supplementary for the detailed modifications and network architecture for the 3D CNN.

在本节中，我们将分割版本 PointNet（图 2，分割网络的修改版本）与两种传统方法 [27] 和 [29] 进行了比较，这两种方法都利用了逐点几何特征和形状之间的对应关系，以及我们自己的 3D CNN 基线。有关 3D CNN 的详细修改和网络架构，请参阅补充。

In Table 2, we report per-category and mean IoU(%) scores. We observe a 2.3% mean IoU improvement and our net beats the baseline methods in most categories.

在表 2 中，我们报告了每个类别的平均 IoU（%） 分数。我们观察到 2.3% 的平均借记奥位改善，我们的净收入在大多数类别中都超过了基线方法。

	mean IoU 均值 **IoU**	**overall accuracy** 整体精度

Ours baseline 我们的基线	20.12	53.19
Ours PointNet 我们的 **PointNet**	**47.71**	**78.62**

Table 3: **Results on semantic segmentation in scenes.** Metric is average IoU over 13 classes (structural and furniture elements plus clutter) and classification accuracy calculated on points.

表 3：场景中语义分割的结果。公制是 13 个类别（结构和家具元素加上地物）的平均 IoU，分类精度按点计算。

	table 桌子	**chair** 椅子	**sofa** 沙发	**board** 板	**mean** 意味着
# instance # 实例	**455**	**1363**	**55**	**137**	
Armeni et al. [1] **Armeni** 等人 **[1]**	46.02	16.15	**6.78**	3.91	18.22
Ours 我们	**46.67**	**33.80**	4.76	**11.72**	**24.24**

Table 4: **Results on 3D object detection in scenes.** Metric is average precision with threshold IoU 0.5 computed in 3D volumes.

表 4：场景中的 **3D** 对象检测结果。Metric 是平均精度，阈值 IoU 为 0.5，以 3D 体积计算。

We also perform experiments on simulated Kinect scans to test the robustness of these methods. For every CAD model in the ShapeNet part data set, we use Blensor Kinect Simulator [7] to generate incomplete point clouds from six random viewpoints. We train our PointNet on the complete shapes and partial scans with the same network architecture and training setting. Results show that we lose only 5.3% mean IoU. In Fig 3, we present qualitative results on both complete and partial data. One can see that though partial data is fairly challenging, our predictions are reasonable.

我们还对模拟的 Kinect 扫描进行了实验，以测试这些方法的稳健性。对于 ShapeNet 零件数据集中的每个 CAD 模型，我们使用 Blensor Kinect Simulator [7] 从六个随机视点生成不完整的点云。我们使用相同的网络架构和训练设置，在完整形状和部分扫描上训练我们的 PointNet。结果显示，我们只损失了 5.3% 的平均 IoU。在图 3 中，我们展示了完整和部分数据的定性结果。可以看出，尽管部分数据相当具有挑战性，但我们的预测是合理的。

Semantic Segmentation in Scenes

场景中的语义分割

Our network on part segmentation can be easily extended to semantic scene segmentation, where point labels become semantic object classes instead of object part labels.

我们的零件分割网络可以很容易地扩展到语义场景分割，其中点标签成为语义对象类，而不是对象部分标签。

We experiment on the Stanford 3D semantic parsing data set [1]. The dataset contains 3D scans from Matterport scanners in 6 areas including 271 rooms. Each point in the scan is annotated with one of the se-

mantic labels from 13 categories (chair, table, floor, wall etc. plus clutter).

我们在斯坦福 3D 语义解析数据集 [1] 上进行了实验。该数据集包含 6 个区域（包括 271 个房间）的 Matterport 扫描仪的 3D 扫描。扫描中的每个点都标有来自 13 个类别（椅子、桌子、地板、墙壁等以及杂物）的语义标签之一。

To prepare training data, we firstly split points by room, and then sample rooms into blocks with area 1m by 1m. We train our segmentation version of PointNet to predict per point class in each block. Each point is represented by a 9-dim vector of XYZ, RGB and normalized location as to the room (from 0 to 1). At training time, we randomly sample 4096 points in each block on-the-fly. At test time, we test on all the points. We follow the same protocol as [1] to use k-fold strategy for train and test.

为了准备训练数据，我们首先按 Room 划分点，然后将 Room 采样为面积为 1m x 1m 的块。我们训练 PointNet 的分割版本来预测每个块中的每个点类。每个点都由 XYZ、RGB 和房间的标准化位置（从 0 到 1）的 9 维向量表示。在训练时，我们动态地在每个块中随机采样 4096 个点。在测试时，我们会测试所有点。我们遵循与 [1] 相同的协议，使用 k-fold 策略进行训练和测试。

We compare our method with a baseline using handcrafted point features. The baseline extracts the same 9-dim local features and three additional ones: local point density, local curvature and normal. We use standard MLP as the classifier. Results are shown in Table 3, where our PointNet method significantly outperforms the baseline method. In Fig 4, we show qualitative segmentation results. Our network is able to output smooth predictions and is robust to missing points and occlusions.

我们将我们的方法与使用手工制作的点特征的基线进行比较。基线提取相同的 9 维局部特征和三个附加特征：局部点密度、局部曲率和法线。我们使用标准 MLP 作为分类器。结果如表 3 所示，其中我们的 PointNet 方法明显优于基线方法。在图 4 中，我们显示了定性分割结果。我们的网络能够输出平滑的预测，并且对缺失点和遮挡具有鲁棒性。

Based on the semantic segmentation output from our network, we further build a 3D object detection system using connected component for object proposal (see supplementary for details). We compare with previous state-of-the-art method in Table 4. The previous method is based on a sliding shape method (with CRF post processing) with SVMs trained on local geometric features and global room context feature in voxel grids. Our method outperforms it by a large margin on the furniture categories reported.

根据我们网络的语义分割输出，我们进一步构建了一个 3D 对象检测系统，使用连接组件进行对象建议（详见补充）。我们与表 4 中以前最先进的方法进行了比较。前面的方法基于滑动形状方法（使用 CRF 后处理），其中 SVM 在体素网格中的局部几何特征和全局房间上下文特征上进行训练。我们的方法在报告的家具类别中大大优于它。

Figure 4: **Qualitative results for semantic segmentation.** Top row is input point cloud with color. Bottom row is output semantic segmentation result (on points) displayed in the same camera viewpoint as input.

图 4：语义分割的定性结果。顶行是带有颜色的输入点云。底行是与输入相同的摄像机视点中显示的输出语义分割结果（在点上）。

5.2 Architecture Design Analysis

5,2 架构设计分析

In this section we validate our design choices by control experiments. We also show the effects of our network's hyperparameters.

在本节中，我们通过对照实验来验证我们的设计选择。我们还展示了网络超参数的效果。

Comparison with Alternative Order-invariant Methods

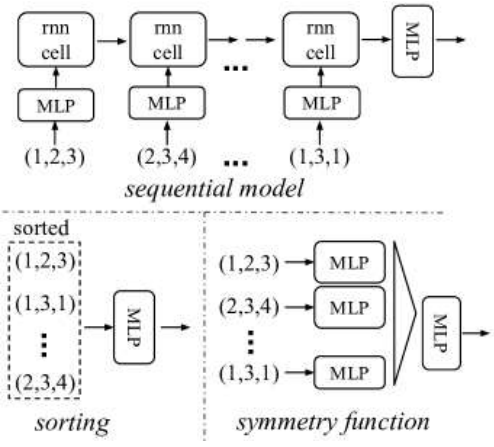
与其他 Order-invariant 方法的比较

As mentioned in Sec 4.2, there are at least three options for consuming unordered set inputs. We use the ModelNet40 shape classification problem as a test bed for comparisons of those options, the following two control experiment will also use this task.

如第 4.2 节所述，至少有三个选项可用于使用无序的 set inputs。我们使用 ModelNet40 形状分类问题作为比较这些选项的试验台，以下两个对照实验也将使用这个任务。

The baselines (illustrated in Fig 5) we compared with include multi-layer perceptron on unsorted and sorted points as $n \times 3$ arrays, RNN model that considers input point as a sequence, and a model based on symmetry functions. The symmetry operation we experimented include max pooling, average pooling and an attention based weighted sum. The attention method is similar to that in [25], where a scalar score is predicted from each point feature, then the score is normalized across points by computing a softmax. The weighted sum is then computed on the normalized scores and the point features. As shown in Fig 5, max-pooling operation achieves the best performance by a large winning margin, which validates our choice.

我们比较的基线（如图 5 所示）包括未排序和排序点上的多层感知器作为 $n \times 3$ 数组，将输入点视为序列的 RNN 模型，以及基于对称函数的模型。我们试验的对称作包括最大池化、平均池化和基于注意力的加权和。注意力方法类似于 [25] 中的方法，其中从每个点特征预测标量分数，然后通过计算 softmax 对分数进行跨点标准化。然后，根据归一化分数和点特征计算加权和。如图 5 所示，max-pooling作以较大的获胜边际实现了最佳性能，这验证了我们的选择。



	accuracy
MLP (unsorted input)	24.2
MLP (sorted input)	45.0
LSTM	78.5
Attention sum	83.0
Average pooling	83.8
Max pooling	**87.1**

Figure 5: **Three approaches to achieve order invariance.** Multi-layer perceptron (MLP) applied on points consists of 5 hidden layers with neuron sizes 64,64,64,128,1024, all points share a single copy of MLP. The MLP close to the output consists of two layers with sizes 512,256.

图 5：实现阶不变性的三种方法。应用于点的多层感知器 （MLP） 由 5 个隐藏层组成，神经元大小为 64,64,64,128,1024，所有点共享一个 MLP 副本。靠近输出的 MLP 由两层组成，大小为 512,256。

Effectiveness of Input and Feature Transformations

输入和特征转换的有效性

In Table 5 we demonstrate the positive effects of our input and feature transformations (for alignment). It's interesting to see that the most basic architecture already achieves quite reasonable results. Using input transformation gives a 0.8% performance boost. The regularization loss is necessary for the higher dimension transform to work. By combining both transformations and the regularization term, we achieve the best performance.

在表 5 中，我们展示了 input 和 feature 转换（用于对齐）的积极影响。有趣的是，最基本的架构已经取得了相当合理的结果。使用输入转换可以 0.8% 提高性能。正则化损失是更高维度转换工作所必需的。通过结合变换和正则化项，我们可以实现最佳性能。

Transform 变换	**accuracy** 准确性
none 没有	87.1
input (3x3) 输入 （**3x3**）	87.9
feature (64x64) 功能 （**64x64**）	86.9
feature (64x64) + reg. 功能 （**64x64**） + 注册表。	87.4
both 双	**89.2**

Table 5: **Effects of input feature transforms.** Metric is overall classification accuracy on ModelNet40 test set.

表 5：输入特征转换的效果。度量是 ModelNet40 测试集的总体分类准确率。

Robustness Test 稳健性测试

We show our PointNet, while simple and effective, is robust to various kinds of input corruptions. We use the same architecture as in Fig 5's max pooling network. Input points are normalized into a unit sphere. Results are in Fig 6.

我们展示了我们的 PointNet 虽然简单有效，但对各种输入损坏是健壮的。我们使用与 Figure 5 的最大池化网络相同的架构。输入点被规格化为一个单位球体。结果如图 6 所示。

As to missing points, when there are 50% points missing, the accuracy only drops by 2.4% and 3.8% w.r.t. furthest and random input sampling. Our net is also robust to outlier points, if it has seen those during training. We evaluate two models: one trained on points with (x, y, z) coordinates; the other on (x, y, z) plus point density. The net has more than 80% accuracy even when 20% of the points are outliers. Fig 6 right shows the net

is robust to point perturbations.

对于缺失点，当有 50% 缺失点时，精度只会下降 2.4%， 3.8% 并且最远和随机的输入采样。我们的网络对异常值点也很稳健，如果它在训练期间看到了这些点。我们评估了两个模型：一个在带有 (x, y, z) 坐标的点上训练;另一个是 ON (x, y, z) 正点密度。即使点中的点是异常值， 20% 网络也具有更高的 80% 准确性。图 6 右图显示了该网络对点扰动是鲁棒的。

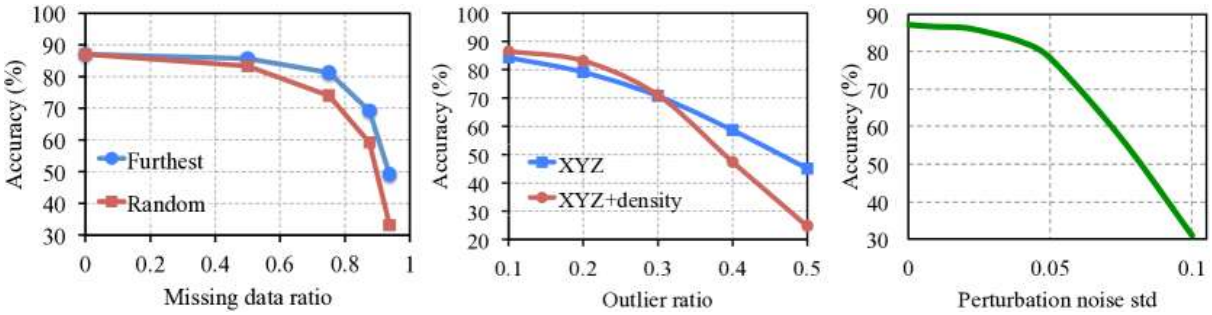


Figure 6: **PointNet robustness test.** The metric is overall classification accuracy on ModelNet40 test set. Left: Delete points. Furthest means the original 1024 points are sampled with furthest sampling. Middle: Insertion. Outliers uniformly scattered in the unit sphere. Right: Perturbation. Add Gaussian noise to each point independently.

图 6： **PointNet** 稳健性测试。该指标是 ModelNet40 测试集的总体分类准确率。左：删除点。Farthest 表示使用最远采样对原始 1024 个点进行采样。中图：插入。异常值均匀分布在单位球体中。右：扰动。将高斯噪声单独添加到每个点。

5.3 Visualizing PointNet

5,3 可视化 PointNet

In Fig 7, we visualize *critical point sets* $\mathcal{C}_S$ and *upper-bound shapes* $\mathcal{N}_S$ (as discussed in Thm 2) for some sample shapes S. The point sets between the two shapes will give exactly the same global shape feature $f(S)$.

在图 7 中，我们可视化了一些样本形状 S 的临界点集 $\mathcal{C}_S$ 和上限形状 $\mathcal{N}_S$ （如 Thm 2 中讨论的那样）。两个形状之间的点集将产生完全相同的全局形状特征 $f(S)$ 。

We can see clearly from Fig 7 that the *critical point sets* $\mathcal{C}_S$, those contributed to the max pooled feature, summarizes the skeleton of the shape. The *upper-bound shapes* $\mathcal{N}_S$ illustrates the largest possible point cloud that give the same global shape feature $f(S)$ as the input point cloud S. $\mathcal{C}_S$ and $\mathcal{N}_S$ reflect the robustness of PointNet, meaning that losing some non-critical points does not change the global shape signature $f(S)$ at all.

从图 7 中我们可以清楚地看到，临界点集，那些有助于最大池化特征的点集 $\mathcal{C}_S$ ，总结了形状的骨架。上限形状 $\mathcal{N}_S$ 说明了提供与输入点云 S 相同的全局形状特征 $f(S)$ 的最大可能点云。 $\mathcal{C}_S$ 并 $\mathcal{N}_S$ 反映了 PointNet 的稳健性，这意味着丢失一些非临界点根本不会改变全局形状特征 $f(S)$ 。

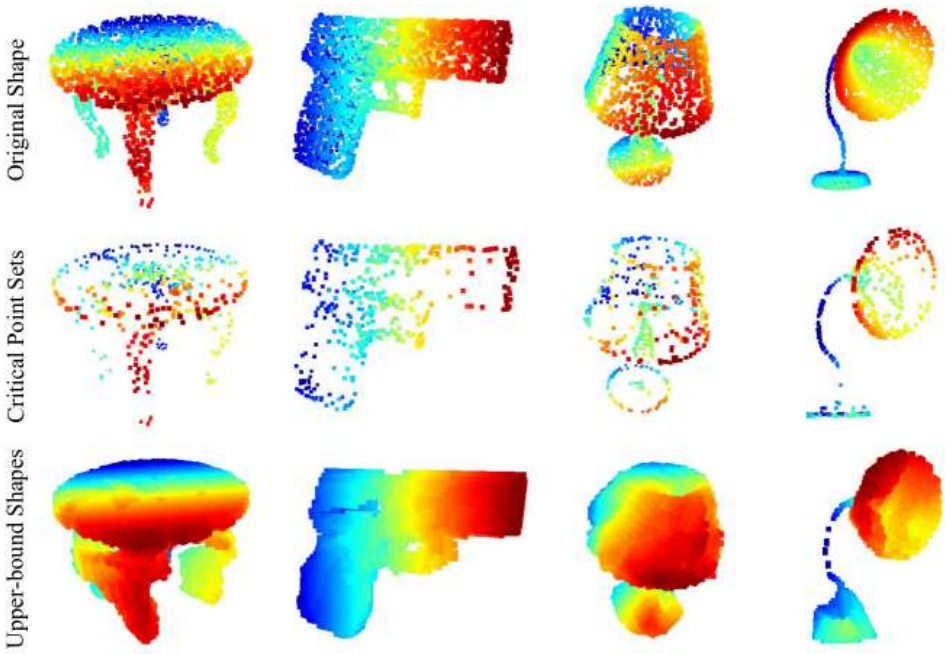


Figure 7: **Critical points and upper bound shape.** While critical points jointly determine the global shape feature for a given shape, any point cloud that falls between the critical points set and the upper bound shape gives exactly the same feature. We color-code all figures to show the depth information.

图 7：临界点和上限形状。虽然临界点共同决定了给定形状的全局形状特征，但落在临界点集和上限形状之间的任何点云都会提供完全相同的特征。我们对所有图物进行颜色编码以显示深度信息。

The $\mathcal{N}_S$ is constructed by forwarding all the points in a edge-length-2 cube through the network and select points p whose point function values $(h_1(p), h_2(p), \cdots, h_K(p))$ are no larger than the global shape descriptor.

它是 $\mathcal{N}_S$ 通过网络转发边长为 2 的立方体中的所有点并选择点函数值 $(h_1(p), h_2(p), \cdots, h_K(p))$ 不大于全局形状描述符的点 p 来构建的。

5.4 Time and Space Complexity Analysis

5,4 时间和空间复杂性分析

Table 6 summarizes space (number of parameters in the network) and time (floating-point operations/sample) complexity of our classification PointNet. We also compare PointNet to a representative set of volumetric and multi-view based architectures in previous works.

表 6 总结了我们分类 PointNet 的空间（网络中的参数数量）和时间（浮点运算/样本）复杂性。我们还将 PointNet 与以前工作中一组具有代表性的基于体积和多视图的架构进行了比较。

While MVCNN [23] and Subvolume (3D CNN) [18] achieve high performance, PointNet is orders more efficient in computational cost (measured in FLOPs/sample: *141x* and *8x* more efficient, respectively). Besides, PointNet is much more space efficient than MVCNN in terms of #param in the network (*17x* less parameters). Moreover, PointNet is much more scalable – it's space and time complexity is $O(N)$ – *linear* in the number of input points. However, since convolution dominates computing time, multi-view method's time complexity grows *squarely* on image resolution and volumetric convolution based method grows *cubically* with the vol-

ume size.

虽然 MVCNN [23] 和子卷 （3D CNN） [18] 实现了高性能，但 PointNet 在计算成本方面效率更高（以 FLOPs/样本衡量：效率分别高出 *141* 倍和 *8* 倍）。此外，就网络中的 #param 而言，PointNet 的空间效率比 MVCNN 高得多（参数减少了 *17* 倍）。此外，PointNet 的可扩展性要强得多 – 它的空间和时间复杂度是 $O(N)$ – 输入点的数量是线性的。然而，由于卷积在计算时间中占主导地位，因此多视图方法的时间复杂度与图像分辨率成正比增加，而基于体积卷积的方法则随体积大小呈立方增长。

Empirically, PointNet is able to process more than one million points per second for point cloud classification (around 1K objects/second) or semantic segmentation (around 2 rooms/second) with a 1080X GPU on TensorFlow, showing great potential for real-time applications.

根据经验，PointNet 能够在 TensorFlow 上使用 1080 倍 GPU 每秒处理超过 100 万个点，用于点云分类（约 1K 个对象/秒）或语义分割（约 2 个房间/秒），显示出实时应用的巨大潜力。

	#params	FLOPs/sample FLOPs/样本
PointNet (vanilla) PointNet（原版） PointNet 点网	0.8M 3.5M	148M 440M
Subvolume [18] 子卷 [18]	16.6M	3633M
MVCNN [23]	60.0M	62057M

Table 6: **Time and space complexity of deep architectures for 3D data classification.** PointNet (vanilla) is the classification PointNet without input and feature transformations. FLOP stands for floating-point operation. The “M” stands for million. Subvolume and MVCNN used pooling on input data from multiple rotations or views, without which they have much inferior performance.

表 6： 用于 **3D** 数据分类的深度架构的时间和空间复杂性。PointNet （vanilla） 是没有输入和特征转换的 PointNet 分类。FLOP 代表浮点运算。“M”代表百万。Subvolume 和 MVCNN 对来自多个旋转或视图的输入数据使用池化，否则它们的性能会差得多。

6 Conclusion 6 结论

In this work, we propose a novel deep neural network *PointNet* that directly consumes point cloud. Our network provides a unified approach to a number of 3D recognition tasks including object classification, part segmentation and semantic segmentation, while obtaining on par or better results than state of the arts on standard benchmarks. We also provide theoretical analysis and visualizations towards understanding of our network.

在这项工作中，我们提出了一种直接使用点云的新型深度神经网络 *PointNet*。我们的网络为许多 3D 识别任务提供了统一的方法，包括对象分类、零件分割和语义分割，同时在标准基准上获得与最先进的技术相当或更好的结果。我们还提供理论分析和可视化，以理解我们的网络。

Acknowledgement. 确认。

The authors gratefully acknowledge the support of a Samsung GRO grant, ONR MURI N00014-13-1-0341 grant, NSF grant IIS-1528025, a Google Focused Research Award, a gift from the Adobe corporation and hardware

donations by NVIDIA.

作者衷心感谢三星 GRO 赠款、ONR MURI N00014-13-1-0341 赠款、NSF 赠款 IIS-1528025、Google 重点研究奖、Adobe 公司的礼物以及 NVIDIA 的硬件捐赠。

References

[1] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2016.

[2] M. Aubry, U. Schlickewei, and D. Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1626–1633. IEEE, 2011.

[3] M. M. Bronstein and I. Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1704–1711. IEEE, 2010.

[4] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[5] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003.

[6] Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu, and E. Wong. 3d deep shape descriptor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2319–2328, 2015.

[7] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree. BlenSor: Blender Sensor Simulation Toolbox Advances in Visual Computing. volume 6939 of *Lecture Notes in Computer Science*, chapter 20, pages 199–208. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2011.

[8] K. Guo, D. Zou, and X. Chen. 3d mesh labeling via deep convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 35(1):3, 2015.

[9] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NIPS 2015*.

[10] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes.

IEEE Transactions on pattern analysis and machine intelligence, 21(5):433–449, 1999.

[11] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, volume 6, pages 156–164, 2003.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[13] Y. Li, S. Pirk, H. Su, C. R. Qi, and L. J. Guibas. Fpnn: Field probing neural networks for 3d data. *arXiv preprint arXiv:1605.06240*, 2016.

[14] H. Ling and D. W. Jacobs. Shape classification using the inner-distance. *IEEE transactions on pattern analysis and machine intelligence*, 29(2):286–299, 2007.

[15] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[16] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 37–45, 2015.

[17] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, September 2015.

[18] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2016.

[19] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. IEEE, 2009.

[20] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3384–3391. IEEE, 2008.

[21] M. Savva, F. Yu, H. Su, M. Aono, B. Chen, D. Cohen-Or, W. Deng, H. Su, S. Bai, X. Bai, et al.
Shrec'16 track large-scale 3d shape retrieval from shapenet core55.

[22] P. Y. Simard, D. Steinkraus, and J. C. Platt.
Best practices for convolutional neural networks applied to visual document analysis.
In *ICDAR*, volume 3, pages 958–962, 2003.

[23] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller.
Multi-view convolutional neural networks for 3d shape recognition.
In *Proc. ICCV, to appear*, 2015.

[24] J. Sun, M. Ovsjanikov, and L. Guibas.
A concise and provably informative multi-scale signature based on heat diffusion.
In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library, 2009.

[25] O. Vinyals, S. Bengio, and M. Kudlur.
Order matters: Sequence to sequence for sets.
arXiv preprint arXiv:1511.06391, 2015.

[26] D. Z. Wang and I. Posner.
Voting for voting in online point cloud object detection.
Proceedings of the Robotics: Science and Systems, Rome, Italy, 1317, 2015.

[27] Z. Wu, R. Shou, Y. Wang, and X. Liu.
Interactive shape co-segmentation via label propagation.
Computers & Graphics, 38:248–254, 2014.

[28] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao.
3d shapenets: A deep representation for volumetric shapes.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

[29] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas.
A scalable active framework for region annotation in 3d shape collections.
SIGGRAPH Asia, 2016.

Supplementary 补充的

Appendix A Overview

附录 A 概述

This document provides additional quantitative results, technical details and more qualitative test examples to the main paper.

本文档提供了主要论文的其他定量结果、技术细节和更多定性测试示例。

In Sec B we extend the robustness test to compare PointNet with VoxNet on incomplete input. In Sec C we provide more details on neural network architectures, training parameters and in Sec D we describe our detection pipeline in scenes. Then Sec E illustrates more applications of PointNet, while Sec F shows more analysis experiments. Sec G provides a proof for our theory on PointNet. At last, we show more visualization results in Sec H.

在第 B 节中，我们扩展了稳健性测试，以比较 PointNet 和 VoxNet 对不完整输入的处理。在章节 C 中，我们提供了有关神经网络架构、训练参数的更多详细信息，在章节 D 中，我们描述了场景中的检测管道。然后 E 部分说明了 PointNet 的更多应用，而 F 部分显示了更多的分析实验。Sec G 为我们在 PointNet 上的理论提供了证明。最后，我们在 Sec H 中展示了更多的可视化结果。

Appendix B Comparison between PointNet and VoxNet (Sec 5.2)

附录 B PointNet 和 VoxNet 之间的比较（第 5.2 节）

We extend the experiments in Sec 5.2 Robustness Test to compare PointNet and VoxNet [17] (a representative architecture for volumetric representation) on robustness to missing data in the input point cloud. Both networks are trained on the same train test split with 1024 number of points as input. For VoxNet we voxelize the point cloud to $32 \times 32 \times 32$ occupancy grids and augment the training data by random rotation around up-axis and jittering.

我们扩展了 Sec 5.2 鲁棒性测试中的实验，以比较 PointNet 和 VoxNet [17]（体积表示的代表性架构）对输入点云中缺失数据的鲁棒性。两个网络都在同一个训练测试分片上进行训练，输入有 1024 个点。对于 VoxNet，我们将点云体素化为 $32 \times 32 \times 32$ 占用网格，并通过围绕上轴随机旋转和抖动来增强训练数据。

At test time, input points are randomly dropped out by a certain ratio. As VoxNet is sensitive to rotations, its prediction uses average scores from 12 viewpoints of a point cloud. As shown in Fig 8, we see that our PointNet is much more robust to missing points. VoxNet's accuracy dramatically drops when half of the input points are missing, from 86.3% to 46.0% with a 40.3% difference, while our PointNet only has a 3.7% performance drop. This can be explained by the theoretical analysis and explanation of our PointNet – it is learning to use a collection of *critical points* to summarize the shape, thus it is very robust to missing data.

在测试时，输入点按一定比例随机丢弃。由于 VoxNet 对旋转很敏感，因此其预测使用点云 12 个视点的平均分数。如图 8 所示，我们看到我们的 PointNet 对缺失点的鲁棒性要强得多。当一半的输入点缺失时，VoxNet 的精度会急剧下降，从 86.3% 到 46.0% 有 40.3% 差异，而我们的 PointNet 只有 3.7% 性能下降。这可以通过我们的 PointNet 的理论分析和解释来解释——它正在学习使用一组关键点来总结形状，因此它对缺失数据非常健壮。

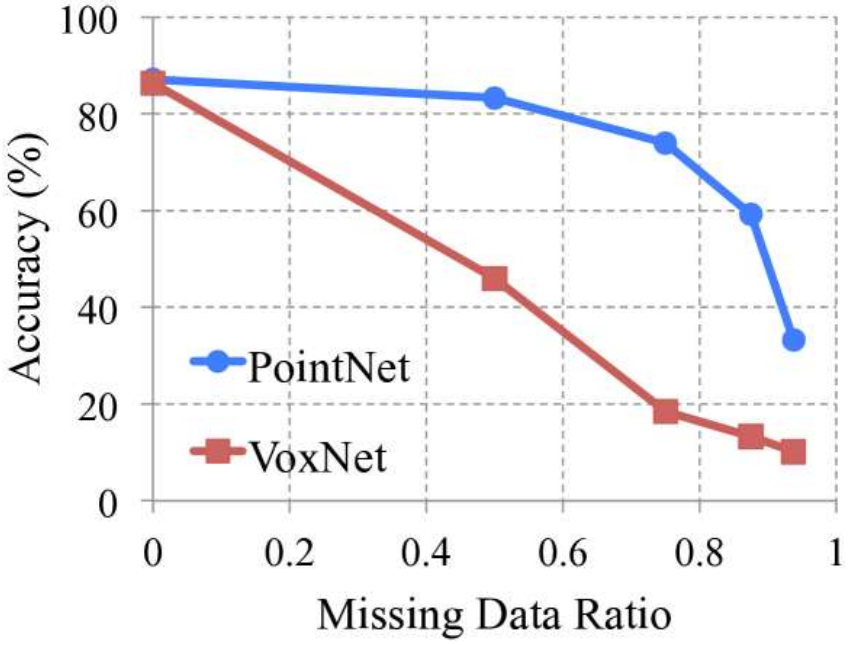


Figure 8: **PointNet v.s. VoxNet [17] on incomplete input data.** Metric is overall classification accurcacy on ModelNet40 test set. Note that VoxNet is using 12 viewpoints averaging while PointNet is using only one view of the point cloud. Evidently PointNet presents much stronger robustness to missing points.

图 8： **PointNet** 与 **VoxNet [17]** 对不完整输入数据的影响。Metric 是 ModelNet40 测试集的总体分类准确性。请注意，VoxNet 使用 12 个视点平均，而 PointNet 仅使用点云的一个视图。显然，PointNet 对缺失点的鲁棒性要强得多。

Appendix C Network Architecture and Training Details (Sec 5.1)

附录 C 网络架构和训练详细信息（第 5.1 节）

PointNet Classification Network

PointNet 分类网络

As the basic architecture is already illustrated in the main paper, here we provides more details on the joint alignment/transformation network and training parameters.

由于基本架构已在主要论文中进行了说明，因此我们在这里提供了有关联合对齐/转换网络和训练参数的更多详细信息。

The first transformation network is a mini-PointNet that takes raw point cloud as input and regresses to a 3×3 matrix. It's composed of a shared $MLP(64, 128, 1024)$ network (with layer output sizes 64, 128, 1024) on each point, a max pooling across points and two fully connected layers with output sizes $512, 256$. The output matrix is initialized as an identity matrix. All layers, except the last one, include ReLU and batch normalization. The second transformation network has the same architecture as the first one except that the output is a 64×64 matrix. The matrix is also initialized as an identity. A regularization loss (with weight 0.001) is added to the softmax classification loss to make the matrix close to orthogonal.

第一个转换网络是一个 mini-PointNet，它以原始点云作为输入并回归到矩阵。 3×3 它由每个点上的共享 $MLP(64, 128, 1024)$ 网络（层输出大小为 64、128、1024）、跨点的最大池化和两个输出大小 512 为 . 256 输出矩阵初始化为单位矩阵。除最后一个图层外，所有图层都包括 ReLU 和批量归一化。第二个转换网络与第一个转换网络具有相同的架构，只是输出是一个 64×64 矩阵。矩阵也初始化为标识。将正则化损失（权重为 0.001）添加到 softmax 分类损失中，以使矩阵接近正交。

We use dropout with keep ratio 0.7 on the last fully connected layer, whose output dimension 256, before class score prediction. The decay rate for batch normalization starts with 0.5 and is gradually increased to 0.99. We use adam optimizer with initial learning rate 0.001, momentum 0.9 and batch size 32. The learning rate is divided by 2 every 20 epochs. Training on ModelNet takes 3-6 hours to converge with TensorFlow and a GTX1080 GPU.

我们在类分数预测之前，在最后一个全连接层（其输出维度 256 ）上使用带有 keep ratio 0.7 的 dropout 。批量归一化的衰减率从 开始 0.5 ，并逐渐增加到 0.99 。我们使用 adam optimizer 和 initial learning rate 0.001 、 momentum 0.9 和 batch size 32 。学习率每 20 个 epoch 除以 2。在 ModelNet 上进行训练需要 3-6 小时才能与 TensorFlow 和 GTX1080 GPU 融合。

PointNet Segmentation Network

PointNet 分段网络

The segmentation network is an extension to the classification PointNet. Local point features (the output after the second transformation network) and global feature (output of the max pooling) are concatenated for each point. No dropout is used for segmentation network. Training parameters are the same as the classification network.

分割网络是分类 PointNet 的扩展。局部点特征（第二个变换网络后的输出）和全局特征（最大池化的输出）将针对每个点进行连接。没有 dropout 用于分段网络。训练参数与分类网络相同。

As to the task of shape part segmentation, we made a few modifications to the basic segmentation network architecture (Fig 2 in main paper) in order to achieve best performance, as illustrated in Fig 9. We add a one-hot vector indicating the class of the input and concatenate it with the max pooling layer's output. We also increase neurons in some layers and add skip links to collect local point features in different layers and concatenate them to form point feature input to the segmentation network.

关于形状部分分割的任务，我们对基本的分割网络架构进行了一些修改（主论文中的图 2），以实现最佳性能，如图 9 所示。我们添加一个 one-hot vector 来指示 input 的类，并将其与 max pooling 层的输出连接起来。我们还增加了一些层中的神经元，并添加了跳过链接来收集不同层中的局部点特征并将它们连接起来，形成到分割网络的点特征输入。

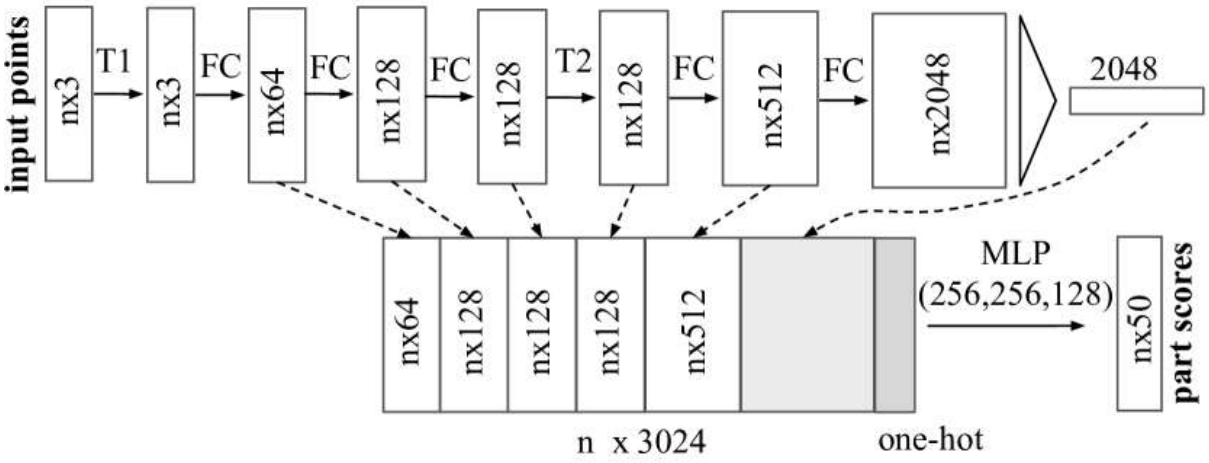


Figure 9: **Network architecture for part segmentation.** T1 and T2 are alignment/transformation networks for input points and features. FC is fully connected layer operating on each point. MLP is multi-layer perceptron on each point. One-hot is a vector of size 16 indicating category of the input shape.

图 9：用于零件分割的网络架构。T1 和 T2 是输入点和要素的对齐/变换网络。FC 是在每个点上运行的全连接层。MLP 是每个点上的多层感知器。One-hot 是一个大小为 16 的向量，指示输入形状的类别。

While [27] and [29] deal with each object category independently, due to the lack of training data for some categories (the total number of shapes for all the categories in the data set are shown in the first line), we train our PointNet across categories (but with one-hot vector input to indicate category). To allow fair comparison, when testing these two models, we only predict part labels for the given specific object category.

虽然 [27] 和 [29] 独立处理每个对象类别，但由于缺乏某些类别的训练数据（数据集中所有类别的形状总数显示在第一行），我们跨类别训练 PointNet（但使用独热向量输入来指示类别）。为了进行公平比较，在测试这两个模型时，我们只预测给定特定对象类别的零件标签。

As to semantic segmentation task, we used the architecture as in Fig 2 in the main paper.

至于语义分割任务，我们使用了主论文中图 2 中的架构。

It takes around six to twelve hours to train the model on ShapeNet part dataset and around half a day to train on the Stanford semantic parsing dataset.

在 ShapeNet 零件数据集上训练模型大约需要 6 到 12 小时，在 Stanford 语义解析数据集上训练大约需要半天时间。

Baseline 3D CNN Segmentation Network

基线 3D CNN 分割网络

In ShapeNet part segmentation experiment, we compare our proposed segmentation version PointNet to two traditional methods as well as a 3D volumetric CNN network baseline. In Fig 10, we show the baseline 3D volumetric CNN network we use. We generalize the well-known 3D CNN architectures, such as VoxNet [17] and 3DShapeNets [28] to a fully convolutional 3D CNN segmentation network.

在 ShapeNet 零件分割实验中，我们将我们提出的分割版本 PointNet 与两种传统方法以及 3D 体积 CNN 网络基线进行了比较。在图 10 中，我们展示了我们使用的基线 3D 体积 CNN 网络。我们将著名的 3D CNN 架构，如 VoxNet [17] 和 3DShapeNets [28] 推广到一个全卷积的 3D CNN 分割网络。

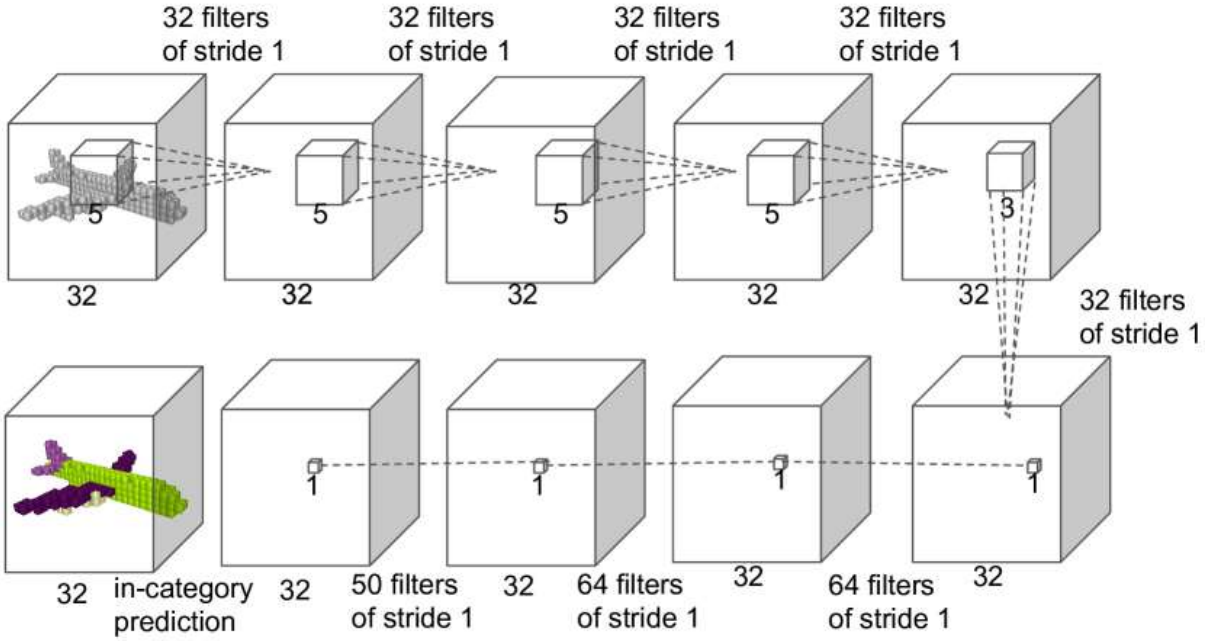


Figure 10: **Baseline 3D CNN segmentation network.** The network is fully convolutional and predicts part scores for each voxel.

图 10：基线 **3D CNN** 分割网络。该网络是完全卷积的，可以预测每个体素的部分分数。

For a given point cloud, we first convert it to the volumetric representation as a occupancy grid with resolution $32 \times 32 \times 32$. Then, five 3D convolution operations each with 32 output channels and stride of 1 are sequentially applied to extract features. The receptive field is 19 for each voxel. Finally, a sequence of 3D convolutional layers with kernel size $1 \times 1 \times 1$ is appended to the computed feature map to predict segmentation label for each voxel. ReLU and batch normalization are used for all layers except the last one. The network is trained across categories, however, in order to compare with other baseline methods where object category is given, we only consider output scores in the given object category.

对于给定的点云，我们首先将其转换为体积表示，作为具有 Resolution $32 \times 32 \times 32$ 的占用网格。然后，依次应用 5 个 3D 卷积运算，每个运算有 32 个输出通道，步幅为 1。每个体素的感受野为 19。最后，将一系列具有核大小的 $1 \times 1 \times 1$ 3D 卷积层附加到计算的特征图中，以预测每个体素的分割标签。ReLU 和批量归一化用于除最后一个图层之外的所有图层。该网络是跨类别训练的，但是，为了与给定对象类别的其他基线方法进行比较，我们只考虑给定对象类别中的输出分数。

Appendix D Details on Detection Pipeline (Sec 5.1)

附录 D 有关检测管道的详细信息（第 5.1 节）

We build a simple 3D object detection system based on the semantic segmentation results and our object classification PointNet.

我们根据语义分割结果和我们的对象分类 PointNet 构建了一个简单的 3D 对象检测系统。

We use connected component with segmentation scores to get object proposals in scenes. Starting from a random point in the scene, we find its predicted label and use BFS to search nearby points with the same label, with a search radius of 0.2 meter. If the resulted cluster has more than 200 points (assuming a 4096 point sample in a 1m by 1m area), the cluster's bounding box is marked as one object proposal. For each proposed object, it's detection score is computed as the average point score for that category. Before evaluation, pro-

posals with extremely small areas/volumes are pruned. For tables, chairs and sofas, the bounding boxes are extended to the floor in case the legs are separated with the seat/surface.

我们使用具有分割分数的连通组件来获取场景中的对象建议。从场景中的随机点开始，找到它的预测标签，并使用 BFS 搜索附近具有相同标签的点，搜索半径为 0.2 米。如果生成的集群超过 200 个点（假设在 1m x 1m 区域中有 4096 个点样本），则集群的边界框将标记为一个对象建议。对于每个建议的对象，其检测分数计算为该类别的平均分数。在评估之前，将修剪面积/体积极小的提案。对于桌子、椅子和沙发，边界框会延伸到地板上，以防腿与座椅/表面分开。

We observe that in some rooms such as auditoriums lots of objects (e.g. chairs) are close to each other, where connected component would fail to correctly segment out individual ones. Therefore we leverage our classification network and uses sliding shape method to alleviate the problem for the chair class. We train a binary classification network for each category and use the classifier for sliding window detection. The resulted boxes are pruned by non-maximum suppression. The proposed boxes from connected component and sliding shapes are combined for final evaluation.

我们观察到，在某些房间（例如礼堂）中，许多物体（例如椅子）彼此靠近，其中连接的组件将无法正确分割出单个组件。因此，我们利用我们的分类网络并使用滑动形状方法来缓解 chair 类的问题。我们为每个类别训练一个二元分类网络，并使用分类器进行滑动窗口检测。结果框由非 maximum suppression 修剪。将 connected component 和 sliding shapes 中建议的框组合在一起以进行最终评估。

In Fig 11, we show the precision-recall curves for object detection. We trained six models, where each one of them is trained on five areas and tested on the left area. At test phase, each model is tested on the area it has never seen. The test results for all six areas are aggregated for the PR curve generation.

在图 11 中，我们显示了对象检测的精度-召回率曲线。我们训练了六个模型，每个模型都在 5 个区域进行了训练，并在左侧区域进行了测试。在测试阶段，每个模型都在它从未见过的区域进行测试。所有 6 个区域的测试结果都聚合起来，用于生成 PR 曲线。

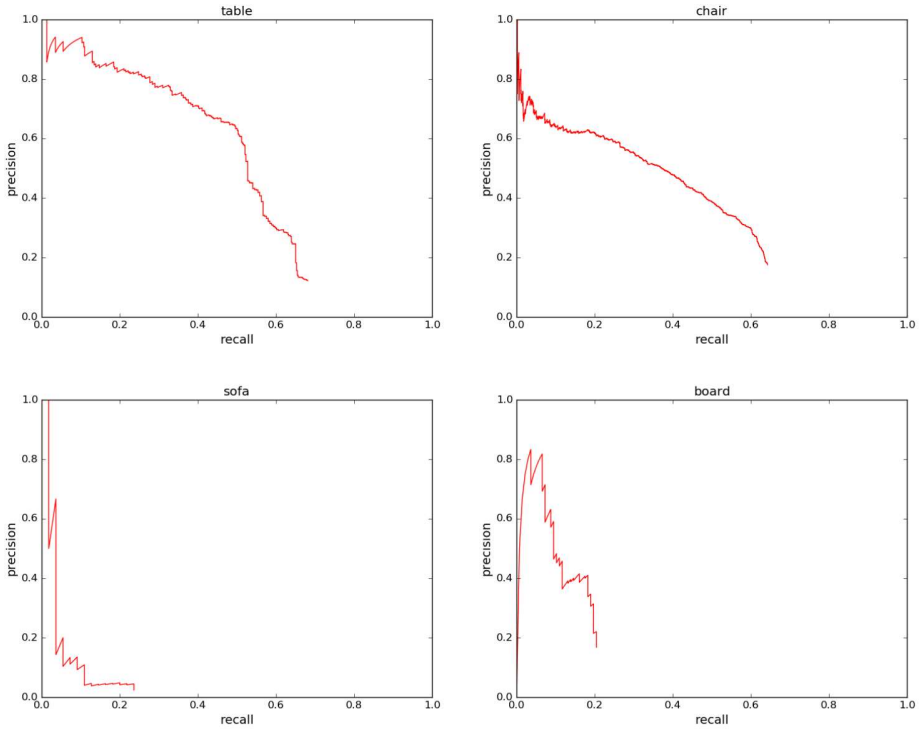


Figure 11: **Precision-recall curves for object detection in 3D point cloud.** We evaluated on all six areas for four categories: table, chair, sofa and board. IoU threshold is 0.5 in volume.

图 11：**3D** 点云中对象检测的精度-召回曲线。我们评估了四个类别的所有六个领域：桌子、椅子、沙发和板子。IoU 阈值的交易量为 0.5。

Appendix E More Applications (Sec 5.1)

附录 E 更多应用（第 5.1 节）

Model Retrieval from Point Cloud

从点云中检索模型

Our PointNet learns a global shape signature for every given input point cloud. We expect geometrically similar shapes have similar global signature. In this section, we test our conjecture on the shape retrieval application. To be more specific, for every given query shape from ModelNet test split, we compute its global signature (output of the layer before the score prediction layer) given by our classification PointNet and retrieve similar shapes in the train split by nearest neighbor search. Results are shown in Fig 12.

我们的 PointNet 会为每个给定的输入点云学习一个全局形状特征。我们预计几何相似的形状具有相似的全局特征。在本节中，我们将测试我们对形状检索应用程序的猜想。更具体地说，对于 ModelNet 测试拆分中的每个给定查询形状，我们计算其分类 PointNet 给出的全局签名（分数预测层之前的层输出），并检索按最近邻搜索划分的火车中的相似形状。结果如图 12 所示。

Figure 12: **Model retrieval from point cloud.** For every given point cloud, we retrieve the top-5 similar shapes from the ModelNet test split. From top to bottom rows, we show examples of chair, plant, nightstand and bathtub queries. Retrieved results that are in wrong category are marked by red boxes.

图 12：从点云中检索模型。对于每个给定的点云，我们从 ModelNet 测试拆分中检索前 5 个相似形状。从上到下行，我们展示了椅子、植物、床头柜和浴缸查询的示例。检索到的分类错误的结果将用红框标记。

Shape Correspondence 形状对应

In this section, we show that point features learnt by PointNet can be potentially used to compute shape correspondences. Given two shapes, we compute the correspondence between their *critical point sets* C_S's by matching the pairs of points that activate the same dimensions in the global features. Fig 13 and Fig 14 show the detected shape correspondence between two similar chairs and tables.

在本节中，我们展示了 PointNet 学习的点特征可用于计算形状对应关系。给定两个形状，我们通过匹配在全局特征中激活相同维度的点对来计算它们的关键点集 C_S 之间的对应关系。图 13 和图 14 显示了检测到的两把相似的椅子和桌子之间的形状对应关系。

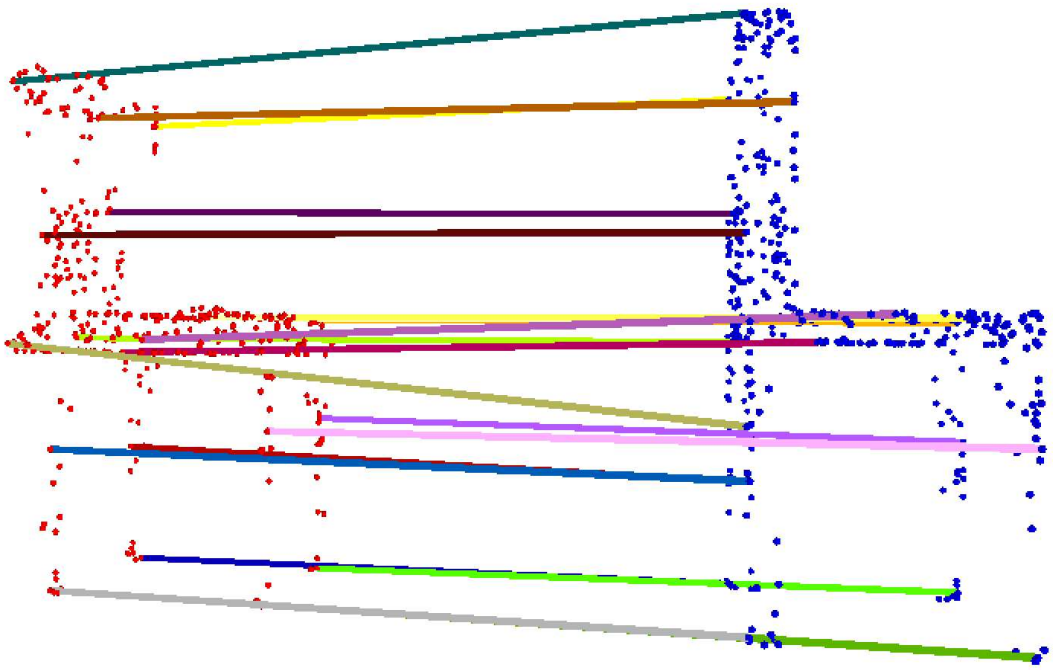


Figure 13: **Shape correspondence between two chairs.** For the clarity of the visualization, we only show 20 randomly picked correspondence pairs.

图 13： 两把椅子之间的形状对应关系。为了可视化的清晰度，我们只显示 20 个随机选取的对应对。

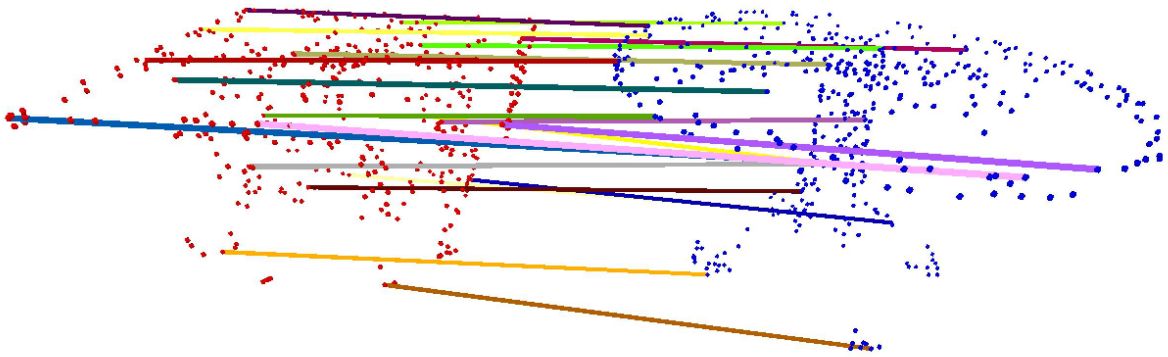


Figure 14: **Shape correspondence between two tables.** For the clarity of the visualization, we only show 20 randomly picked correspondence pairs.

图 14： 两个表格之间的形状对应关系。为了可视化的清晰度，我们只显示 20 个随机选取的对应对。

Appendix F More Architecture Analysis (Sec 5.2)

附录 F 更多架构分析（第 5.2 节）

Effects of Bottleneck Dimension and Number of Input Points

瓶颈维度和输入点数量的影响

Here we show our model's performance change with regard to the size of the first max layer output as well as the number of input points. In Fig 15 we see that performance grows as we increase the number of points however it saturates at around 1K points. The max layer size plays an important role, increasing the layer size from 64 to 1024 results in a $2-4\%$ performance gain. It indicates that we need enough point feature functions to cover the 3D space in order to discriminate different shapes.

在这里，我们显示了模型在第一个最大层输出的大小以及输入点数量方面的性能变化。在图 15 中，我们看到性能随着点数的增加而增长，但它在 1K 点左右达到饱和。最大图层大小起着重要作用，将图层大小从 64 增加到 1024 会提高性能 $2-4\%$ 。它表明我们需要足够的点特征函数来覆盖 3D 空间，以便区分不同的形状。

It's worth notice that even with 64 points as input (obtained from furthest point sampling on meshes), our network can achieve decent performance.

值得注意的是，即使有 64 个点作为输入（从网格上的最远点采样获得），我们的网络也可以实现不错的性能。

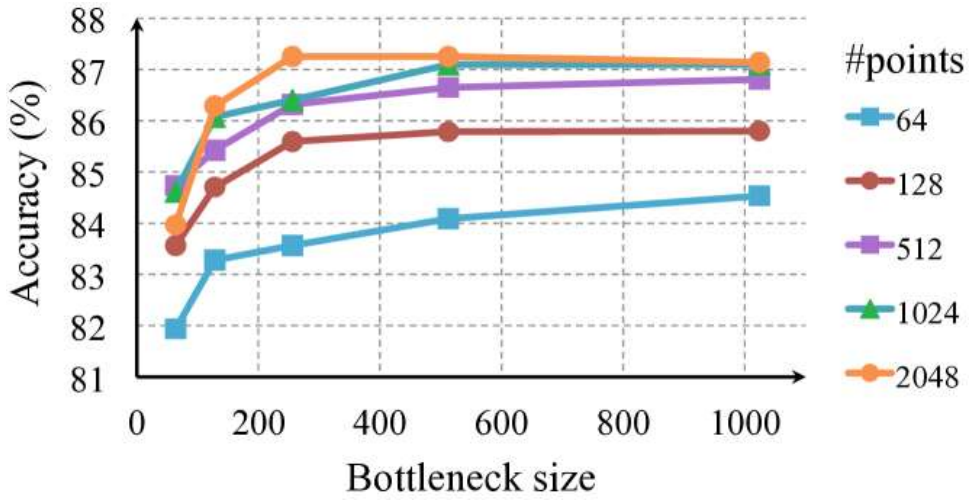


Figure 15: **Effects of bottleneck size and number of input points.** The metric is overall classification accuracy on ModelNet40 test set.

图 15： 瓶颈大小和输入点数量的影响。该指标是 ModelNet40 测试集的总体分类准确率。

MNIST Digit Classification

MNIST 数字分类

While we focus on 3D point cloud learning, a sanity check experiment is to apply our network on a 2D point clouds - pixel sets.

虽然我们专注于 3D 点云学习，但完整性检查实验是将我们的网络应用于 2D 点云 - 像素集。

To convert an MNIST image into a 2D point set we threshold pixel values and add the pixel (represented as a point with (x, y) coordinate in the image) with values larger than 128 to the set. We use a set size of 256. If

there are more than 256 pixels int he set, we randomly sub-sample it; if there are less, we pad the set with the one of the pixels in the set (due to our max operation, which point to use for the padding will not affect outcome).

为了将 MNIST 图像转换为 2D 点集，我们对像素值进行阈值处理，并将值大于 128 的像素（表示为图像中带有 (x, y) 坐标的点）添加到该集。我们使用 set size 256。如果集合中的像素超过 256 个，则随机对其进行子采样;如果数量较少，则使用集合中的一个像素填充集合（由于我们的 Max作，用于填充的点不会影响结果）。

As seen in Table 7, we compare with a few baselines including multi-layer perceptron that considers input image as an ordered vector, a RNN that consider input as sequence from pixel (0,0) to pixel (27,27), and a vanilla version CNN. While the best performing model on MNIST is still well engineered CNNs (achieving less than 0.3% error rate), it's interesting to see that our PointNet model can achieve reasonable performance by considering image as a 2D point set.

如表 7 所示，我们与一些基线进行了比较，包括将输入图像视为有序向量的多层感知器、将输入视为从像素（0,0） 到像素 （27,27） 的序列的 RNN，以及普通版本的 CNN。虽然 MNIST 上性能最好的模型仍然是设计良好的 CNN（实现低于 0.3% 错误率），但有趣的是，我们的 PointNet 模型可以通过将图像视为 2D 点集来实现合理的性能。

	input 输入	error (%) 误差 (%)
Multi-layer perceptron [22] **多层感知器 [22]**	vector 向量	1.60
LeNet5 [12]	image 图像	0.80
Ours PointNet 我们的 PointNet	point set 点集	0.78

Table 7: **MNIST classification results.** We compare with vanilla versions of other deep architectures to show that our network based on point sets input is achieving reasonable performance on this traditional task.

表 7： **MNIST** 分类结果。我们与其他深度架构的 vanilla 版本进行比较，以表明我们基于点集输入的网络在这项传统任务上实现了合理的性能。

Normal Estimation

In segmentation version of PointNet, local point features and global feature are concatenated in order to provide context to local points. However, it's unclear whether the context is learnt through this concatenation. In this experiment, we validate our design by showing that our segmentation network can be trained to predict point normals, a local geometric property that is determined by a point's neighborhood.

在 PointNet 的分段版本中，局部点特征和全局特征被连接起来，以便为局部点提供上下文。但是，目前尚不清楚上下文是否是通过这种串联来学习的。在这个实验中，我们通过证明我们的分割网络可以被训练来预测点法线，这是一种由点的邻域决定的局部几何属性，从而验证了我们的设计。

We train a modified version of our segmentation PointNet in a supervised manner to regress to the ground-truth point normals. We just change the last layer of our segmentation PointNet to predict normal vector for each point. We use absolute value of cosine distance as loss.

我们以监督方式训练分割 PointNet 的修改版本，以回归到真实点法线。我们只需更改分割 PointNet 的最后一层即可预测每个点的法向量。我们使用余弦距离的绝对值作为损失。

Fig. 16 compares our PointNet normal prediction results (the left columns) to the ground-truth normals computed from the mesh (the right columns). We observe a reasonable normal reconstruction. Our predictions are more smooth and continuous than the ground-truth which includes flipped normal directions in some region.

无花果。图 16 将我们的 PointNet 法线预测结果（左列）与从网格计算的真实法线（右列）进行了比较。我们观察到合理的正常重建。我们的预测比地面实况更平滑、更连续，地面实况包括某些区域的反转法线方向。

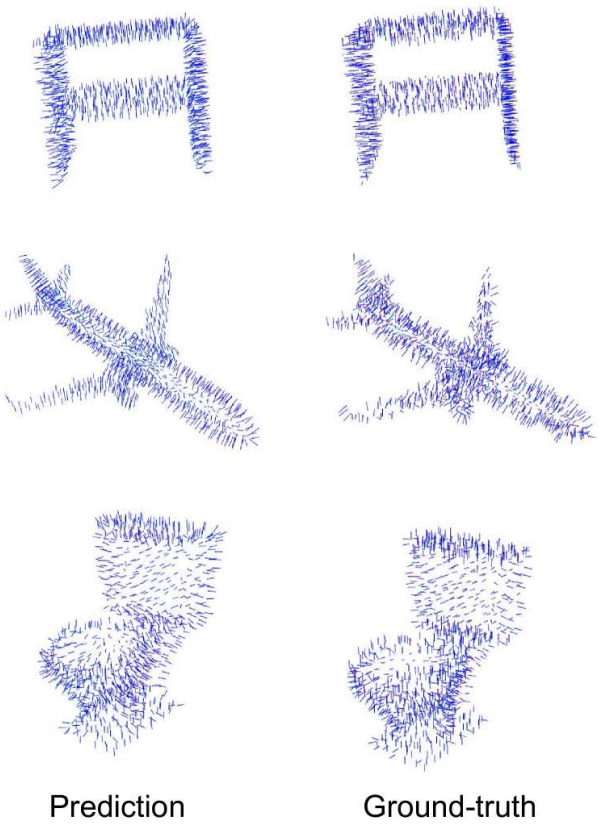


Figure 16: **PointNet normal reconstrution results.** In this figure, we show the reconstructed normals for all the points in some sample point clouds and the ground-truth normals computed on the mesh.

图 16： **PointNet** 法态重构结果。在此图中，我们显示了一些样本点云中所有点的重建法线，以及在网格上计算的真实法线。

Segmentation Robustness 分段稳健性

As discussed in Sec 5.2 and Sec B, our PointNet is less sensitive to data corruption and missing points for classification tasks since the global shape feature is extracted from a collection of *critical points* from the given input point cloud. In this section, we show that the robustness holds for segmentation tasks too. The per-point part labels are predicted based on the combination of per-point features and the learnt global shape feature. In Fig 17, we illustrate the segmentation results for the given input point clouds S (the left-most column), the *critical point sets* $\mathcal{C}_S$ (the middle column) and the *upper-bound shapes* $\mathcal{N}_S$.

正如第 5.2 节和第 B 节所讨论的，我们的 PointNet 对分类任务的数据损坏和缺失点不太敏感，因为全局形状特征是从给定输入点云的关键点集合中提取的。在本节中，我们展示了 Segmentation 任务的稳健性也适用。每个点的零件标签是根据每个点特征和学习的全局形状特征的组合来预测的。在图 17 中，我们说明了给定输入点云 S （最左列）、临界点集 $\mathcal{C}_S$ （中间列）和上限形状 $\mathcal{N}_S$ 的分割结果。

Figure 17: **The consistency of segmentation results.** We illustrate the segmentation results for some sample given point clouds S, their *critical point sets* $\mathcal{C}_S$ and *upper-bound shapes* $\mathcal{N}_S$. We observe that the shape family between the $\mathcal{C}_S$ and $\mathcal{N}_S$ share a consistent segmentation results.

图 17：分割结果的一致性。我们说明了一些样本给定点 S 云的分割结果，它们的关键点集 $\mathcal{C}_S$ 和上限形状 $\mathcal{N}_S$ 。我们观察到 $\mathcal{C}_S$ 和 $\mathcal{N}_S$ 之间的形状族共享一致的分割结果。

Network Generalizability to Unseen Shape Categories

网络泛化到看不见的形状类别

In Fig 18, we visualize the *critical point sets* and the *upper-bound shapes* for new shapes from unseen categories (face, house, rabbit, teapot) that are not present in ModelNet or ShapeNet. It shows that the learnt per-point functions are generalizable. However, since we train mostly on man-made objects with lots of planar structures, the reconstructed upper-bound shape in novel categories also contain more planar surfaces.

在图 18 中，我们可视化了 ModelNet 或 ShapeNet 中不存在的不可见类别（人脸、房子、兔子、茶壶）中新形状的关键点集和上界形状。它表明学习的每点函数是可泛化的。然而，由于我们主要在具有许多平面结构的人造物体上进行训练，因此新类别中重建的上界形状也包含更多的平面。

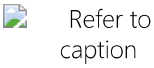


Figure 18: **The critical point sets and the upper-bound shapes for unseen objects.** We visualize the *critical point sets* and the *upper-bound shapes* for teapot, bunny, hand and human body, which are not in the ModelNet or ShapeNet shape repository to test the generalizability of the learnt per-point functions of our PointNet on other unseen objects. The images are color-coded to reflect the depth information.

图 18：看不见对象的关键点集和上限形状。我们可视化了茶壶、兔子、手和人体的关键点集和上限形状，这些形状不在 ModelNet 或 ShapeNet 形状存储库中，以测试我们的 PointNet 学习的每点函数在其他看不见的对象上的泛化性。图像采用颜色编码以反映深度信息。

Appendix G Proof of Theorem (Sec 4.3)

附录 G 定理证明 (Sec 4.3)

Let $\mathcal{X} = \{S : S \subseteq [0,1] \text{ and } |S| = n\}$. 设 $\mathcal{X} = \{S : S \subseteq [0,1] \text{ and } |S| = n\}$.

$f : \mathcal{X} \to \mathbb{R}$ is a continuous function on $\mathcal{X}$ w.r.t to Hausdorff distance $d_H(\cdot,\cdot)$ if the following condition is satisfied:

$f : \mathcal{X} \to \mathbb{R}$ 是 w.r.t 到 Hausdorff 距离 $d_H(\cdot,\cdot)$ 的 $\mathcal{X}$ 连续函数，如果满足以下条件：

$\forall \epsilon > 0, \exists \delta > 0$, for any $S, S^{'} \in \mathcal{X}$, if $d_H(S, S^{'}) < \delta$, then $|f(S) - f(S^{'})| < \epsilon$.

$\forall \epsilon > 0, \exists \delta > 0$ ，对于任何 $S, S^{'} \in \mathcal{X}$ 、 if $d_H(S, S^{'}) < \delta$ 、 则 $|f(S) - f(S^{'})| < \epsilon$ 。

We show that f can be approximated arbitrarily by composing a symmetric function and a continuous function.

我们表明， f 可以通过组合对称函数和连续函数来任意近似。

Theorem 1. *Suppose $f : \mathcal{X} \to \mathbb{R}$ is a continuous set function w.r.t Hausdorff distance $d_H(\cdot,\cdot)$. $\forall \epsilon > 0$, $\exists$ a continuous function h and a symmetric function $g(x_1, ..., x_n) = \gamma \circ MAX$, where γ is a continuous function, MAX is a vector max operator that takes n vectors as input and returns a new vector of the element-wise maximum, such that for any $S \in \mathcal{X}$,*

$$|f(S) - \gamma(MAX(h(x_1), ..., h(x_n)))| < \epsilon$$

where $x_1, ..., x_n$ are the elements of S extracted in certain order,

其中 $x_1, ..., x_n$ 是按一定顺序提取的 S 元素，

定理 1. 假设 f：X→R：f→XRf：\mathcal{X}\rightarrow\mathbb{R} 是一个连续集函数，w.r.t Hausdorff 距离 dH（·，·）下标dH··d_{H}（\cdot，\cdot）。∀ε>0for-allitalic-ε0\forall\epsilon>0， ∃\存在一个连续函数 hhh 和一个对称函数 g（x1,...,xn）=γ∘MAXgsubscriptx1...下标xnγMAXg（x_{1}，\dots，x_{n}）=\gamma\circ\mbox{MAX}，其中 γγ\gamma 是一个连续函数，MAX 是一个向量 max 运算符，它以 nnn 向量作为输入并返回元素最大值的新向量，使得对于任何 S∈XSXS\in\mathcal{X}，

Proof. 证明。 By the continuity of f, we take δ_ϵ so that $|f(S) - f(S^{'})| < \epsilon$ for any $S, S^{'} \in \mathcal{X}$ if $d_H(S, S^{'}) < \delta_\epsilon$.

通过 f 的连续性， 我们取 δ_ϵ 使得 $|f(S) - f(S^{'})| < \epsilon$ 对于任何 $S, S^{'} \in \mathcal{X}$ if $d_H(S, S^{'}) < \delta_\epsilon$.

Define $K = \lceil 1 / \delta_\epsilon \rceil$, which split $[0,1]$ into K intervals evenly and define an auxiliary function that maps a point to the left end of the interval it lies in:

define $K = \lceil 1 / \delta_\epsilon \rceil$ ，它均匀地划分 $[0,1]$ 为 K 间隔，并定义一个辅助函数，将一个点映射到它所在的区间的左端：

$$\sigma(x) = \frac{\lfloor Kx \rfloor}{K}$$

Let $\widetilde{S} = \{\sigma(x) : x \in S\}$, then 让 $\widetilde{S} = \{\sigma(x) : x \in S\}$ ， 然后

$$|f(S) - f(\widetilde{S})| < \epsilon$$

because $d_H(S, \widetilde{S}) < 1 \,/\, K \leq \delta_\epsilon$. 因为 $d_H(S, \widetilde{S}) < 1 \,/\, K \leq \delta_\epsilon$.

Let $h_k(x) = e^{-d(x, [\frac{k-1}{K}, \frac{k}{K}])}$ be a soft indicator function where $d(x, I)$ is the point to set (interval) distance. Let $\mathbf{h}(x) = [h_1(x); ...; h_K(x)]$, then $\mathbf{h} : \mathbb{R} \to \mathbb{R}^K$.

设 $h_k(x) = e^{-d(x, [\frac{k-1}{K}, \frac{k}{K}])}$ 为软指标函数，其中 $d(x, I)$ 是设置（间隔）距离的点。设 $\mathbf{h}(x) = [h_1(x); ...; h_K(x)]$ ，则 $\mathbf{h} : \mathbb{R} \to \mathbb{R}^K$ 。

Let $v_j(x_1, ..., x_n) = \max\{\widetilde{h}_j(x_1), ..., \widetilde{h}_j(x_n)\}$, indicating the occupancy of the j-th interval by points in S. Let $\mathbf{v} = [v_1; ...; v_K]$, then $\mathbf{v} : \underbrace{\mathbb{R} \times ... \times \mathbb{R}}_{n} \to \{0, 1\}^K$ is a symmetric function, indicating the occupancy of each interval by points in S.

设 $v_j(x_1, ..., x_n) = \max\{\widetilde{h}_j(x_1), ..., \widetilde{h}_j(x_n)\}$ ，指示 j 中的点对第 −个区间的占用 S 率。设 $\mathbf{v} = [v_1; ...; v_K]$ ，则 $\mathbf{v} : \underbrace{\mathbb{R} \times ... \times \mathbb{R}}_{n} \to \{0, 1\}^K$ 是一个对称函数，表示 中点对每个区间的占用 S 率。

Define $\tau : \{0, 1\}^K \to \mathcal{X}$ as $\tau(v) = \{\frac{k-1}{K} : v_k \geq 1\}$, which maps the occupancy vector to a set which contains the left end of each occupied interval. It is easy to show:

定义为 $\tau : \{0, 1\}^K \to \mathcal{X}$ $\tau(v) = \{\frac{k-1}{K} : v_k \geq 1\}$ ，它将占用向量映射到包含每个占用间隔的左端的集。它很容易显示：

$$\tau(\mathbf{v}(x_1, ..., x_n)) \equiv \widetilde{S}$$

where $x_1, ..., x_n$ are the elements of S extracted in certain order.

其中 $x_1, ..., x_n$ 是按一定顺序提取的 S 元素。

Let $\gamma : \mathbb{R}^K \to \mathbb{R}$ be a continuous function such that $\gamma(\mathbf{v}) = f(\tau(\mathbf{v}))$ for $v \in \{0, 1\}^K$. Then,

设 $\gamma : \mathbb{R}^K \to \mathbb{R}$ 为连续函数，使得 $\gamma(\mathbf{v}) = f(\tau(\mathbf{v}))$ 对于 $v \in \{0, 1\}^K$ 。然后

$$\begin{aligned} & |\gamma(\mathbf{v}(x_1, ..., x_n)) - f(S)| \\ = \; & |f(\tau(\mathbf{v}(x_1, ..., x_n))) - f(S)| < \epsilon \end{aligned}$$

Note that $\gamma(\mathbf{v}(x_1, ..., x_n))$ can be rewritten as follows:

请注意， $\gamma(\mathbf{v}(x_1, ..., x_n))$ 可以按如下方式重写：

$$\begin{aligned} \gamma(\mathbf{v}(x_1, ..., x_n)) = \; & \gamma(\mathrm{MAX}(\mathbf{h}(x_1), ..., \mathbf{h}(x_n))) \\ = \; & (\gamma \circ \mathrm{MAX})(\mathbf{h}(x_1), ..., \mathbf{h}(x_n)) \end{aligned}$$

Obviously $\gamma \circ \mathrm{MAX}$ is a symmetric function. ■

显然 $\gamma \circ \mathrm{MAX}$ 是一个对称函数。 ■

Next we give the proof of Theorem 2. We define $\mathbf{u} = \underset{x_i \in S}{\mathrm{MAX}}\{h(x_i)\}$ to be the sub-network of f which maps a point set in $[0, 1]^m$ to a K-dimensional vector. The following theorem tells us that small corruptions or extra

noise points in the input set is not likely to change the output of our network:

接下来我们给出定理 2 的证明。 我们定义为 $\mathbf{u} = \underset{x_i \in S}{\mathrm{MAX}}\{h(x_i)\}$ 子网络 f，其子网络将一个点集 $[0,1]^m$ 映射到一个 K -维向量。以下定理告诉我们，输入集中的小损坏或额外的噪声点不太可能改变我们网络的输出：

Theorem 2. *Suppose* $\mathbf{u} : \mathcal{X} \rightarrow \mathbb{R}^K$ *such that* $\mathbf{u} = \underset{x_i \in S}{MAX}\{h(x_i)\}$ *and* $f = \gamma \circ \mathbf{u}$*. Then,*

(a) $\forall S, \exists\ \mathcal{C}_S, \mathcal{N}_S \subseteq \mathcal{X}$*,* $f(T) = f(S)$ *if* $\mathcal{C}_S \subseteq T \subseteq \mathcal{N}_S$*;*

$\forall S, \exists\ \mathcal{C}_S, \mathcal{N}_S \subseteq \mathcal{X}$ ， $f(T) = f(S)$ 如果 $\mathcal{C}_S \subseteq T \subseteq \mathcal{N}_S$ *;*

(b) $|\mathcal{C}_S| \leq K$

定理 2. 假设 u：X→RK：u→XsuperscriptRK\mathbf{u}：\mathcal{X}\rightarrow\mathbb{R}^{K}，使得 u=MAXxi∈S{h（习）}usubscriptxiSMAXhsubscriptxi\mathbf{u}=\underset{x_{i}\in S}{\mbox{MAX}}\{h（x_{i}）\} 和 f=γ∘ufγuf=\gamma\circ\mathbf{u}。然后

Proof. Obviously, $\forall\, S \in \mathcal{X}$, $f(S)$ is determined by $\mathbf{u}(S)$. So we only need to prove that $\forall\, S, \exists\, \mathcal{C}_S, \mathcal{N}_S \subseteq \mathcal{X}, f(T) = f(S)\,\text{if}\,\mathcal{C}_S \subseteq T \subseteq \mathcal{N}_S$.

证明。 显然，∀S∈Xfor-allSX\forall S\in\mathcal{X}， f（S）fSf（S） 由 u（S）uS\mathbf{u}（S） 确定。所以我们只需要证明这一点 ∀S，∃CS，NS⊆X，f（T）=f（S）ifCS⊆T⊆NSformulae-sequencefor-allSsubscriptCSsubscriptNSXfTfSifsubscriptCSTsubscriptNS\forall S，\exists\，\mathcal{C}_{S}，\mathcal{N}_{S}\subseteq\mathcal{X}，f（T）=f（S）\，\mbox{if}\，\mathcal{C}_{S}\subseteq T\subseteq\mathcal{N}_{S}.

For the jth dimension as the output of $\mathbf{u}$, there exists at least one $x_j \in \mathcal{X}$ such that $h_j(x_j) = \mathbf{u}_j$, where h_j is the j th dimension of the output vector from h. Take $\mathcal{C}_S$ as the union of all x_j for $j = 1, ..., K$. Then, $\mathcal{C}_S$ satisfies the above condition.

对于作为 输出的 j $\mathbf{u}$ 第 th 维，至少存在一个 $x_j \in \mathcal{X}$ ，其中 $h_j(x_j) = \mathbf{u}_j$ h_j 是来自 h 的输出向量的 j 第 个维。将 $\mathcal{C}_S$ 作为 的所有 x_j 的 $j = 1, ..., K$ 并集。然后， $\mathcal{C}_S$ 满足上述条件。

Adding any additional points x such that $h(x) \leq \mathbf{u}(S)$ at all dimensions to $\mathcal{C}_S$ does not change $\mathbf{u}$, hence f. Therefore, $\mathcal{T}_S$ can be obtained adding the union of all such points to $\mathcal{N}_S$.

添加任何其他点 x ， $h(x) \leq \mathbf{u}(S)$ 使得在所有维度 to $\mathcal{C}_S$ 上都不会改变 $\mathbf{u}$ ，因此 f 。因此， $\mathcal{T}_S$ 可以将所有这些点的并集添加到 $\mathcal{N}_S$ 中。

■

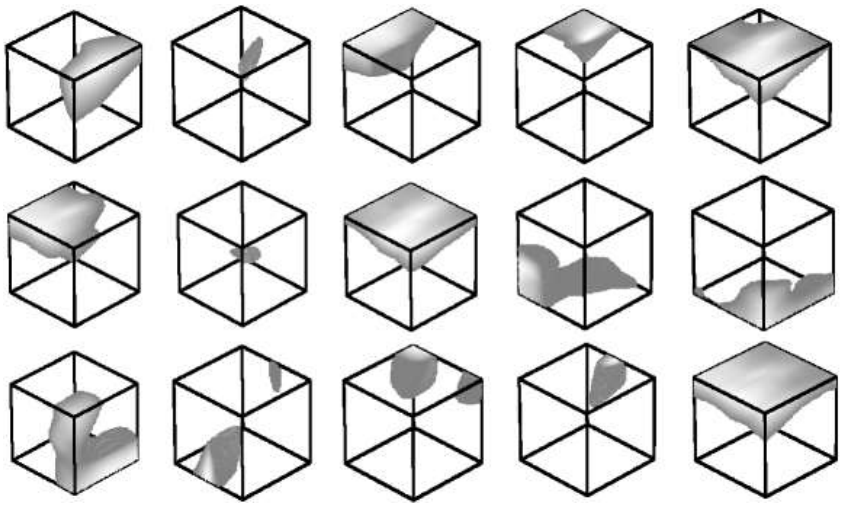


Figure 19: **Point function visualization.** For each per-point function h, we calculate the values $h(p)$ for all the points p in a cube of diameter two located at the origin, which spatially covers the unit sphere to which our input shapes are normalized when training our PointNet. In this figure, we visualize all the points p that give $h(p) > 0.5$ with function values color-coded by the brightness of the voxel. We randomly pick 15 point functions and visualize the activation regions for them.

图 19：点函数可视化。对于每个每点函数 h，我们计算位于原点处的直径为 2 的立方体中所有点 p 的值 $h(p)$，该立方体在空间上覆盖了在训练 PointNet 时输入形状被归一化到的单位球体。在此图中，我们可视化了所有使用 $h(p) > 0.5$ 按体素亮度进行颜色编码的函数值的点 p。我们随机选择 15 个点函数并可视化它们的激活区域。

Appendix H More Visualizations

附录 H 更多可视化

Classification Visualization

分类可视化

We use t-SNE[15] to embed point cloud global signature (1024-dim) from our classification PointNet into a 2D space. Fig 20 shows the embedding space of ModelNet 40 test split shapes. Similar shapes are clustered together according to their semantic categories.

我们使用 t-SNE[15] 将分类 PointNet 中的点云全局签名（1024-dim）嵌入到 2D 空间中。图 20 显示了 ModelNet 40 测试拆分形状的嵌入空间。相似的形状根据其语义类别聚集在一起。

Figure 20: **2D embedding of learnt shape global features.** We use t-SNE technique to visualize the learnt global shape features for the shapes in ModelNet40 test split.

图 20：学习形状全局特征的 **2D** 嵌入。我们使用 t-SNE 技术在 ModelNet40 测试拆分中可视化形状的学习全局形状特征。

Segmentation Visualization

细分可视化

We present more segmentation results on both complete CAD models and simulated Kinect partial scans. We also visualize failure cases with error analysis. Fig 21 and Fig 22 show more segmentation results generated on complete CAD models and their simulated Kinect scans. Fig 23 illustrates some failure cases. Please read the caption for the error analysis.

我们在完整的 CAD 模型和模拟的 Kinect 部分扫描上提供了更多的分割结果。我们还通过错误分析将故障案例可视化。图 21 和图 22 显示了在完整的 CAD 模型及其模拟的 Kinect 扫描上生成的更多分割结果。图 23 说明了一些失败的情况。请阅读错误分析的标题。

Figure 21: **PointNet segmentation results on complete CAD models.**

图 21：完整 **CAD** 模型上的 **PointNet** 分割结果。

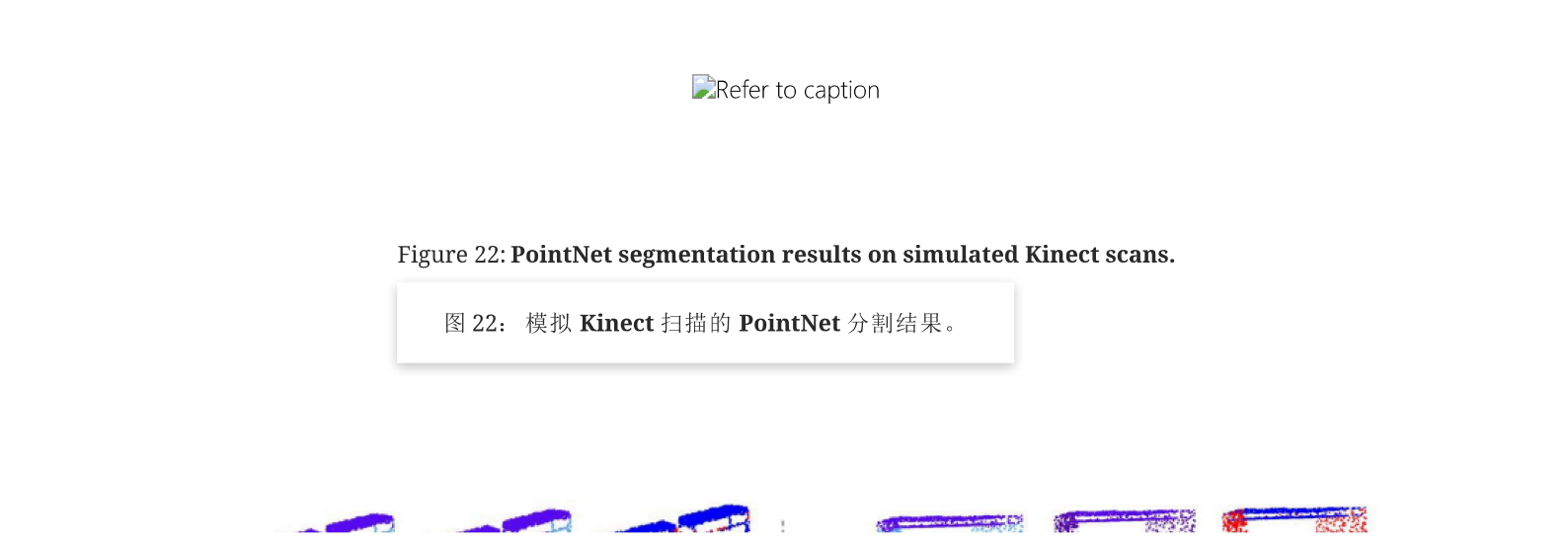


Figure 22: **PointNet segmentation results on simulated Kinect scans.**

图 22：模拟 **Kinect** 扫描的 **PointNet** 分割结果。

Figure 23: **PointNet segmentation failure cases.** In this figure, we summarize six types of common errors in our segmentation application. The prediction and the ground-truth segmentations are given in the first and second columns, while the difference maps are computed and shown in the third columns. The red dots correspond to the wrongly labeled points in the given point clouds. (a) illustrates the most common failure cases: the points on the boundary are wrongly labeled. In the examples, the label predictions for the points near the intersections between the table/chair legs and the tops are not accurate. However, most segmentation algorithms suffer from this error. (b) shows the errors on exotic shapes. For examples, the chandelier and the airplane shown in the figure are very rare in the data set. (c) shows that small parts can be overwritten by nearby large parts. For example, the jet engines for airplanes (yellow in the figure) are mistakenly classified as body (green) or the plane wing (purple). (d) shows the error caused by the inherent ambiguity of shape parts. For example, the two bottoms of the two tables in the figure are classified as table legs and table bases (category *other* in [29]), while ground-truth segmentation is the opposite. (e) illustrates the error introduced by the incompleteness of the partial scans. For the two caps in the figure, almost half of the point clouds are missing. (f) shows the failure cases when some object categories have too less training data to cover enough variety. There are only 54 bags and 39 caps in the whole dataset for the two categories

shown here.

图 23： **PointNet** 分段失败案例。在此图中，我们总结了 Segmentation 应用程序中的六种常见错误。第一列和第二列给出了预测和真实分割，而第三列给出了差异图并显示了差异图。红点对应于给定点云中错误标记的点。（a） 说明了最常见的故障情况：边界上的点被错误地标记了。在示例中，桌子/椅子腿与顶部之间交叉点附近点的标签预测不准确。但是，大多数分割算法都会遇到此错误。（b） 显示奇异形状上的误差。例如，图中所示的吊灯和飞机在数据集中非常罕见。（c） 显示小零件可以被附近的大零件覆盖。例如，飞机的喷气发动机（图中黄色）被错误地归类为机身（绿色）或飞机机翼（紫色）。（d） 显示了由形状部分固有的模糊性引起的误差。例如，图中两个表格的两个底部被归类为表腿和表基（[29] 中的类别 *other*），而真实分割则相反。（e） 说明了部分扫描的不完整性所引入的错误。对于图中的两个帽，几乎缺少一半的点云。（f） 显示某些对象类别的训练数据太少而无法涵盖足够种类时的失败情况。此处显示的两个类别在整个数据集中只有 54 个袋子和 39 个帽子。

Scene Semantic Parsing Visualization

场景语义解析可视化

We give a visualization of semantic parsing in Fig 24 where we show input point cloud, prediction and ground truth for both semantic segmentation and object detection for two office rooms and one conference room. The area and the rooms are unseen in the training set.

我们在图 24 中给出了语义解析的可视化，其中我们展示了两个办公室和一个会议室的语义分割和对象检测的输入点云、预测和地面实况。该区域和房间在训练集中不可见。

Figure 24: **Examples of semantic segmentation and object detection.** First row is input point cloud, where walls and ceiling are hided for clarity. Second and third rows are prediction and ground-truth of semantic segmentation on points, where points belonging to different semantic regions are colored differently (chairs in red, tables in purple, sofa in orange, board in gray, bookcase in green, floors in blue, windows in violet,

beam in yellow, column in magenta, doors in khaki and clutters in black). The last two rows are object detection with bounding boxes, where predicted boxes are from connected components based on semantic segmentation prediction.

图 24：语义分割和对象检测的示例。第一行是输入点云，为了清晰起见，其中的墙壁和天花板被隐藏起来。第二行和第三行是点语义分割的预测和真实值，其中属于不同语义区域的点的颜色不同（椅子为红色，桌子为紫色，沙发为橙色，板为灰色，书柜为绿色，地板为蓝色，窗户为紫色，横梁为黄色，柱子为洋红色，门为卡其色，杂物为黑色）。最后两行是使用边界框的对象检测，其中预测的框来自基于语义分割预测的连通分量。

Point Function Visualization

点函数可视化

Our classification PointNet computes K (we take $K = 1024$ in this visualization) dimension point features for each point and aggregates all the per-point local features via a max pooling layer into a single K-dim vector, which forms the global shape descriptor.

我们的分类 PointNet 计算 K （我们采用 $K = 1024$ 这种可视化方式）每个点的维度点特征，并通过最大池化层将所有每个点的局部特征聚合成一个 K -dim 向量，形成全局形状描述符。

To gain more insights on what the learnt per-point functions h's detect, we visualize the points p_i's that give high per-point function value $f(p_i)$ in Fig 19. This visualization clearly shows that different point functions learn to detect for points in different regions with various shapes scattered in the whole space.

为了更深入地了解学习的每点函数 h 检测到什么，我们在图 19 中可视化了给出高每点函数值的 $f(p_i)$ 点 p_i 。这种可视化清楚地表明，不同的点函数学习检测整个空间中散布着各种形状的不同区域中的点。