Learning Geometry-Disentangled Representation for Complementary Understanding of 3D Object Point Cloud

Mutian Xu [1 2*], **Junhao Zhang** [1*], **Zhipeng Zhou** [1], **Mingye Xu** [1 3], **Xiaojuan Qi** [2], **Yu Qiao** [1†]

[1]Guangdong-Hong Kong-Macao Joint Laboratory of Human-Machine Intelligence-Synergy Systems, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
[2]The University of Hong Kong
[3]University of Chinese Academy of Sciences
mino1018@outlook.com, {zhangjh, zp.zhou, my.xu}@siat.ac.cn, xjqi@eee.hku.hk, yu.qiao@siat.ac.cn

Abstract

In 2D image processing, some attempts decompose images into high and low frequency components for describing edge and smooth parts respectively. Similarly, the contour and flat area of 3D objects, such as the boundary and seat area of a chair, describe different but also complementary geometries. However, such investigation is lost in previous deep networks that understand point clouds by directly treating all points or local patches equally. To solve this problem, we propose Geometry-Disentangled Attention Network (GDANet). GDANet introduces Geometry-Disentangle Module to dynamically disentangle point clouds into the contour and flat part of 3D objects, respectively denoted by sharp and gentle variation components. Then GDANet exploits Sharp-Gentle Complementary Attention Module that regards the features from sharp and gentle variation components as two holistic representations, and pays different attentions to them while fusing them respectively with original point cloud features. In this way, our method captures and refines the holistic and complementary 3D geometric semantics from two distinct disentangled components to supplement the local information. Extensive experiments on 3D object classification and segmentation benchmarks demonstrate that GDANet achieves the state-of-the-arts with fewer parameters.

1 Introduction

The capacity to analyze and comprehend 3D point clouds receives interests in computer vision community due to its wide applications in autonomous driving and robotics (Rusu et al. 2008; Qi et al. 2018). Recent studies explore deep learning methods to understand 3D point clouds inspired by their great success in computer vision applications (He et al. 2015, 2016). Deep networks (Guo et al. 2016) can extract effective semantics of 3D point clouds with layered operations, in contrast to low-level handcrafted shape descriptors. The pioneer work PointNet (Qi et al. 2017a) directly processes 3D points by Multi-layer Perceptrons(MLPs) (Hornik 1991), whose main idea is to learn a spatial encoding of each point and then aggregate all point features to a global point cloud signature. PointNet++ (Qi et al. 2017b) adopts a hierarchical encoder-decoder structure to consider local regions, which downsamples point clouds in layers and gradually interpolates them to the original resolution. From another perspective, some recent efforts extend regular grid convolution (Xu et al. 2018; Li et al. 2018; Thomas et al. 2019) on irregular 3D point cloud configuration.

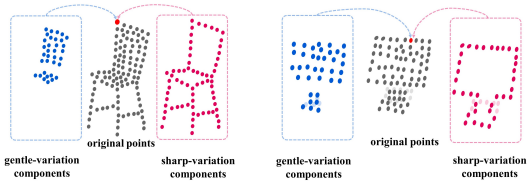

Figure 1: Examples of 3D objects, where sharp-variation component describes the contour areas, and gentle-variation component denotes the flat areas. Our method regards the features from these two disentangled components as two holistic representations, i.e., each point of the original point cloud is linked with all points of sharp and gentle variation components. This operation integrates complementary geometric information from two disentangled components.

To fully utilize geometric information, some attempts (Wang et al. 2019b; Lan et al. 2019) capture local geometric relations among center points and its neighbors. However, these works treat all points or local patches equally, which are entangled together with large redundancy, making it hard to capture the most related and key geometric interest to the network. Moreover, previous operations only capture the geometric information in local areas.

To remedy these defects, we need to disentangle point clouds into distinct components and learn the few-redundant information represented by these holistic components. In image processing, some attempts collect and combine the high-frequency (edge) and low-frequency (smooth) components with distinct characteristics filtered through digital signal processing. Similarly, the contour areas of 3D objects delineating skeletons provide basic structural information, while the flat areas depicting surfaces supply the geomet-

*M.Xu and J.Zhang contribute equally.
†Corresponding author.

ric manifold context. When operating them separately, the network performs better by learning distinct but also complementary geometric representations of point clouds. This inspires us to extract and utilize the geometric information of the contour and flat areas disentangled from 3D objects.

Here comes a challenge about how to disentangle 3D objects into such holistic representations (contour and flat area) and utilize them for better understanding point clouds. Thanks to the graph signal processing (Ortega et al. 2018; Sandryhaila and Moura 2014) who analyzes the frequency on graphs, we firstly extend this to our Geometry-Disentangle Module (GDM) that dynamically analyzes graph signals on 3D point clouds in different semantic levels and factorizes the original point cloud into the contour and flat parts of objects, respectively denoted by sharp and gentle variation components (explained at the end of Sec 4.1). Further, we design Sharp-Gentle Complementary Attention Module (SGCAM) that pays different attentions to features from sharp and gentle variation components according to geometric correlation, then respectively fuses them with each original point features instead of only operating local patches. As shown in Fig. 1, the contour and flat area have distinct but also complementary effects on reflecting the geometry of objects.

Equipped with GDM and SGCAM, we propose GDANet who captures and refines the holistic and complementary geometries of 3D objects to supplement local neighboring information. Experimental results on challenging benchmarks demonstrate that our GDANet achieves the state of the arts, and is more lightweight and robust to density, rotation and noise. Thorough visualizations and analysis verify that our model effectively captures the complementary geometric information from two disentangled components.

2 Related Work

Point Cloud Models Based on Geometry. Recently the exploration on point cloud geometry has drawn large focus. Geo-CNN (Lan et al. 2019) defines a convolution operation that aggregates edge features to capture the local geometric relations. In (Xu, Zhou, and Qiao 2020), they aggregate points from local neighbors with similar semantics in both Euclidean and Eigenvalue space. RS-CNN (Liu et al. 2019c) extends regular CNN to encode geometric relation in a local point set by learning the topology. DensePoint (Liu et al. 2019b) recognizes the implicit geometry by extracting contextual shape semantics. However, the contour and flat area of 3D objects play different but complementary roles in modeling 3D objects. All the operations mentioned above neglect this property and treat all points or local patches equally. By contrast, we present Geometry-Disentangle Module to disentangle point clouds into sharp (contour) and gentle (flat area) variation components, which are two distinct representations of 3D objects.

Attention Networks. The applications of attention mechanism in sequence-based tasks become popular (Vaswani et al. 2017), which helps to concentrate on the most relevant and significant parts. Some recent point cloud methods utilize attention to aggregate the neighboring features of each point (Liu et al. 2019a). GAC (Wang et al. 2019a) proposes a graph attention convolution that can be carved into specific shapes by assigning attention weights to different neighbor points. Different from them, our network learns to assign different attention weights to the disentangled contour and flat areas of 3D objects based on geometric correlations. In Sec 4.3, we also illustrate that not only the attention mechanism helps the network refine the disentangled feature but also our disentanglement strategy assists the attention module easily concentrate on the key geometric interests.

Disentangled Representation. Recent attempts use disentangled representations in different applications. In general, the concept of disentanglement (Bengio, Courville, and Vincent 2012) dominates representation learning, closely linking with human reasoning. In (Xiao, Hong, and Ma 2018), they separate a facial image into individual and shared factors encoding single attribute. (Huang et al. 2018) processes images by decomposing latent space into content and style space. Yet, the disentangled representation on point cloud understanding remains untouched. Our method explicitly disentangles point clouds into two components denoting contour and flat area of objects, which are fused to provide distinct and complementary geometric information.

3 Revisit Graph Signal Processing

Graph signal processing (Ortega et al. 2018; Sandryhaila and Moura 2014) is based on a graph $\mathcal{G} = (\mathcal{V}, \mathbf{A})$ where $\mathcal{V} = \{v_1, \cdots, v_N\}$ denotes a set of N nodes and $\mathbf{A} \in \mathbb{R}^{N \times N}$ denotes a weight adjacency matrix encoding the dependencies between nodes. Using this graph, we refer to the one-channel features of the data on all nodes in vertex domain as $\mathbf{s} \in \mathbb{R}^N$. Let $\mathbf{A}$ be a graph shift operator which takes a graph signal $\mathbf{s} \in \mathbb{R}^N$ as input and produces a new graph signal $\mathbf{y} = \mathbf{A}\mathbf{s}$. We also have the eigen decomposition $\mathbf{A} = V\mathbf{\Lambda}V^{-1}$, where the columns of matrix V are the eigenvectors of $\mathbf{A}$ and the diagonal eigenvalue matrix $\mathbf{\Lambda} \in \mathbb{R}^{N \times N}$ corresponds to ordered eigenvalues $\lambda_1, \cdots, \lambda_N$.

Theorem 1 (Ortega et al. 2018; Sandryhaila and Moura 2014). *The ordered eigenvalues* $(\lambda_1 \geq \lambda_2 \geq \cdots, \geq \lambda_N)$ *represent frequencies on the graph from low to high.*

Accordingly, we obtain $V^{-1}\mathbf{y} = \mathbf{\Lambda}V^{-1}\mathbf{s}$, and the graph Fourier transform of graph signal $\mathbf{s}$ and $\mathbf{y}$ are $\widehat{\mathbf{s}} = V^{-1}\mathbf{s}$, $\widehat{\mathbf{y}} = V^{-1}\mathbf{y}$, respectively. V^{-1} is the graph Fourier transform matrix. The components of $\widehat{\mathbf{s}}$ and $\widehat{\mathbf{y}}$ are considered as frequency contents of signal $\mathbf{s}$ and $\mathbf{y}$.

As stated in (Sandryhaila and Moura 2014), a graph filter is a polynomial in the graph shift:

$$h(\mathbf{A}) = \sum_{\ell=0}^{L-1} h_\ell \mathbf{A}^\ell, \tag{1}$$

where h_ℓ are filter coefficients and L is the length of the filter. It takes a graph signal $\mathbf{s} \in \mathbb{R}^N$ as the input and generates a filtered signal $\mathbf{y} = h(\mathbf{A})s \in \mathbb{R}^N$. Then $\mathbf{y} = Vh(\mathbf{\Lambda})V^{-1}\mathbf{s}$, making $V^{-1}\mathbf{y} = h(\mathbf{\Lambda})V^{-1}\mathbf{s}$ and $\widehat{\mathbf{y}} = h(\mathbf{\Lambda})\widehat{\mathbf{s}}$.

Theorem 2 (Ortega et al. 2018; Sandryhaila and Moura 2014). *The diagonal matrix* $h(\mathbf{\Lambda})$ *is the graph frequency response of the filter* $h(\mathbf{A})$, *which is*

denoted as $\widehat{h(\mathbf{A})}$. *The frequency response of* λ_i *is* $\sum_{\ell=0}^{L-1} h_\ell \lambda_i^\ell$.

4 Method

We firstly design Geometry-Disentangle Module based on graph signal processing to decompose point clouds into sharp and gentle variation components (respectively denotes the contour and flat area). Further, we propose Sharp-Gentle Complementary Attention Module to fuse the point features from sharp and gentle variation components. Last, we introduce Geometry-Disentangled Attention Network equipped with these two modules.

4.1 Geometry-Disentangle Module

Graph Construction. We consider a point cloud consisted of N points with $C-$dimensional features, which is denoted by a matrix $X = [x_1, x_2, \cdots, x_N]^T = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_C] \in \mathbb{R}^{N\times C}$, where $x_i \in \mathbb{R}^C$ represents the i-th point and $\mathbf{s}_c \in \mathbb{R}^N$ represents the c-th channel feature. Features can be 3D coordinates, normals and semantic features. We construct a graph $\mathcal{G} = (\mathcal{V}, \mathbf{A})$ through an adjacency matrix $\mathbf{A}$ that encodes point similarity in the feature space. Each point $x_i \in \mathbb{R}^C$ is associated with a corresponding graph vertex $i \in \mathcal{V}$ and $\mathbf{s}_c \in \mathbb{R}^N$ is a graph signal. The edge weight between two points x_i and x_j is

$$\mathbf{A}_{i,j} = \begin{cases} f(||x_i - x_j||_2), & ||x_i - x_j||_2 \le \tau \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

where $f(\cdot)$ is an non-negative decreasing function (e.g., Gaussian function) which must ensure that $\mathbf{A} \in \mathbb{R}^{N\times N}$ is a diagonally dominant matrix and τ is a threshold. In addition, to handle the size-varying neighbors across different points and feature scales, we normalize all edge weights as follows:

$$\tilde{\mathbf{A}}_{i,j} = \frac{\mathbf{A}_{i,j}}{\sum_j \mathbf{A}_{i,j}}, \tag{3}$$

where $\tilde{\mathbf{A}}$ is still a diagonally dominant matrix. Now as illustrated in Theorem 1, we obtain a graph $\mathcal{G} = (\mathcal{V}, \tilde{\mathbf{A}})$, where eigenvalues of $\tilde{\mathbf{A}}$ $\left(\tilde{\lambda}_1 \ge \tilde{\lambda}_2 \ge \cdots, \ge \tilde{\lambda}_N\right)$ represent graph frequencies from low to high.

Disentangling Point Clouds into Sharp and Gentle Variation Components. In 2D image processing, high frequency component corresponding to intense pixel variation (edge) in spatial domain gets high response while low frequency component (smooth area) gets very low response after being processed by a high-pass filter. Here we aim to design a graph filter on our constructed $\mathcal{G} = (\mathcal{V}, \tilde{\mathbf{A}})$ to select the points belongs to the contour and flat areas of 3D objects. Following Eq. (1), the key to design this graph filter is to construct the corresponding polynomial format of $h(\tilde{\mathbf{A}})$. Here we use the Laplacian operator as (Chen et al. 2018), where $L = 2$, $h_0 = 1$, $h_1 = -1$, making the polynomial format of the graph filter to be $h(\tilde{\mathbf{A}}) = I - \tilde{\mathbf{A}}$. This filter takes $\mathbf{s}_c \in \mathbb{R}^N$ in this graph as the input and generates a filtered graph signal $\mathbf{y}_c = h(\tilde{\mathbf{A}})\mathbf{s}_c \in \mathbb{R}^N$. Following Theorem 2, the frequency response of $h(\tilde{\mathbf{A}})$ with corresponding λ_i is

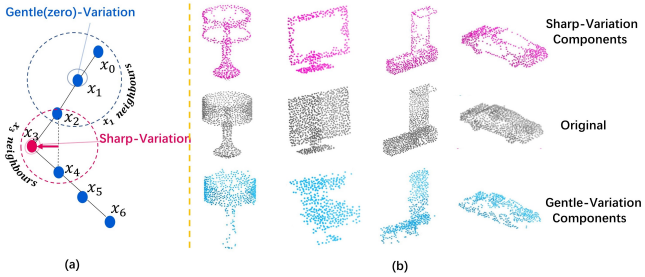

Figure 2: Visualization of the process in our Geometry-Disentangle Module and the obtained sharp and gentle variation components of some objects.

$$\widehat{h(\tilde{\mathbf{A}})} = \begin{bmatrix} 1-\tilde{\lambda}_1 & 0 & \cdots & 0 \\ 0 & 1-\tilde{\lambda}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1-\tilde{\lambda}_N \end{bmatrix}. \tag{4}$$

In this way, the eigenvalues $\tilde{\lambda}_i$ are in a descending order, which represents that frequencies are ordered ascendingly according to Theorem 1. Due to the corresponding **frequencies response** $1 - \tilde{\lambda}_i < 1 - \tilde{\lambda}_{i+1}$, the low frequency part is weakened after this filter. Hence, we call this filter $h(\tilde{\mathbf{A}}) = I - \tilde{\mathbf{A}}$ a **high-pass** filter.

Note: The eigenvalues representing frequencies is only for deducing the polynomial format of our high-pass filter $h(\tilde{\mathbf{A}})$. The implementation of this filter only requires the calculation of $\tilde{\mathbf{A}}$.

Next, we apply $h(\tilde{\mathbf{A}})$ to filter the point set X and get a filtered point set $h(\tilde{\mathbf{A}})X$. Due to $h(\tilde{\mathbf{A}}) = I - \tilde{\mathbf{A}}$, each point in $h(\tilde{\mathbf{A}})X$ can be formulated as:

$$\left(h(\tilde{\mathbf{A}})\mathrm{X}\right)_i = \mathrm{x}_i - \sum_j^N \tilde{\mathbf{A}}_{i,j}\mathrm{x}_j. \tag{5}$$

When the distance between two point x_i, x_j is less than the threshold τ, $\mathbf{A}_{i,j}$ remains non-zero value. Here $\left(h(\tilde{\mathbf{A}})\mathrm{X}\right)_i$ actually equals to the difference between a point feature and the linear convex combination of its neighbors' features, which reflects the degree of each point's **variation** to its neighbors.

Finally, we calculate the l^2-norm of Eq. (5) at every point, and the larger l^2-norm at a given point reflects sharp variation and means this point belongs to the **contour** of a 3D object, which is consistent to the edge areas obtained by a high-pass filter in 2D images. We put all original points in descending order as $X_o = [\check{x}_1, \check{x}_2, \cdots, \check{x}_N]^T$ according to l^2-norm of Eq. (5). Following this order, we select the first M points $X_s = [\check{x}_1, \check{x}_2, \cdots, \check{x}_M]^T \in \mathbb{R}^{M\times C}$ called as **sharp-variation component** and the last M points $X_g = [\check{x}_{N-M+1}, \check{x}_{N-M+2}, \cdots, \check{x}_N]^T \in \mathbb{R}^{M\times C}$ denoted by **gentle-variation component**. Fig. 2 (a) shows this process and Fig. 2 (b) visualizes sharp and gentle variation components disentangled by our trained network. We employ our Geometry-Disentangle Module on point features in different semantic levels, which is elaborated in Sec 4.4.

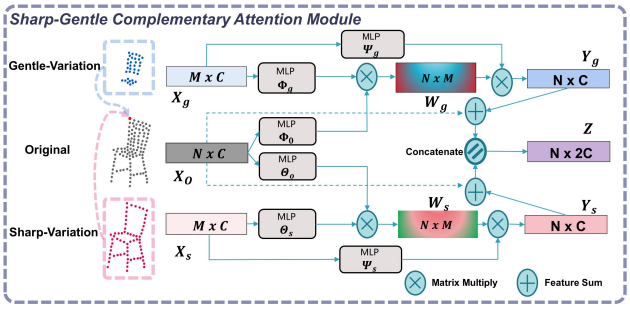

Figure 3: Visualization of Sharp-Gentle Complementary Attention Module. Features from different variation components are respectively integrated with the features from original point cloud, so that to provide complementary geometric information.

4.2 Sharp-Gentle Complementary Attention Module

The sharp and gentle variation components play different but complementary roles in representing the 3D object geometries, which should not be treated equally. However, most previous methods operate all points or local point sets equally. To solve this issue and utilize different variation components gained from Geometry-Disentangle Module, we design Sharp-Gentle Complementary Attention Module inspired by (Vaswani et al. 2017; Wang et al. 2018). It regards the features from two variation components as two holistic geometric representations and generates two attention matrices separately according to the geometric correlation. Then our module assigns the corresponding attention weights to features from two different variation components while respectively integrating them with the original input point features. The details of Sharp-Gentle Complementary Attention Module are shown in Fig. 3 and elaborated below.

Attention Matrix. According to Sec 4.1, we have original point cloud features X_o, features of sharp-variation component X_s and features of gentle-variation component X_g. These features are firstly encoded by different nonlinear functions, and then are utilized to calculate different attention matrices as the following equation:

$$W_s = \Theta_o(X_o)\cdot\Theta_s\left(X_s\right)^T, W_g = \Phi_o(X_o)\cdot\Phi_g\left(X_g\right)^T, \quad (6)$$

where different nonlinear functions Θ_o, Θ_s, Φ_o and Φ_g are implemented by different MLPs without sharing parameters. In this way, we get two learnable adjacency matrices $W_s \in \mathbb{R}^{N\times M}$ and $W_g \in \mathbb{R}^{N\times M}$, where M is the number of points in either X_s or X_g. Each row of W_s or W_g corresponds to attention weights between each original point feature and features from sharp and gentle variation components, respectively. Because the adjacency matrices W_s and W_g are computed as feature dot product, they can explicitly measure the semantic correlation or discrepancy between points.

Geometric Complementary Understanding. Next we apply the attention matrices W_s and W_g respectively to the features from sharp and gentle variation components so that the network can pay different attentions to them while fusing them with the original point features. The whole fusion procedure can be formulated as the following:

$$Y_s = X_o + W_s \cdot \Psi_s\left(X_s\right), \quad (7)$$

$$Y_g = X_o + W_g \cdot \Psi_g\left(X_g\right), \quad (8)$$

in element-wise:

$$(Y_s)_i = (X_o)_i + \sum_{j=1}^{M}(W_s)_{ij}\cdot\Psi_s((X_s)_j), \quad (9)$$

$$(Y_g)_i = (X_o)_i + \sum_{j=1}^{M}(W_g)_{ij}\cdot\Psi_g((X_g)_j), \quad (10)$$

where two different nonlinear functions Ψ_s and Ψ_g are utilized to refine X_s and X_g. They are implemented by different MLPs without sharing parameters.

Last we concatenate the features as the following equation:

$$Z = Y_s \oplus Y_g. \quad (11)$$

Accordingly, our method regards features from sharp and gentle variation components as two holistic representations, i.e., all the point features with different attention weights from these two components are respectively linked with each original input point cloud feature. Our module explicitly conveys the complementary geometric information and the most relevant and key geometric interest to the network in a holistic way for better understanding 3D point cloud.

4.3 Self-Attention or Sharp-Gentle Attention

Self-Attention is an alternative holistic fusion strategy that pays different attentions to each point feature while linking them with the original point cloud itself, instead of integrating the feature from sharp and gentle variation components.

However, self-attention unavoidably brings large redundancy due to it operates all points together, making it hard to capture the most related geometric interest to the network. Yet this issue is alleviated after disentangling point clouds, where our model easily pays different attentions to two variation components carried with few redundancy and distinct geometric information.

To verify this, we compare our module with applying self-attention fusion on original point features and visualize the attention weights distribution in Fig. 5. The attention weights in self-attention are assigned irregularly, which indicates that self-attention method is limited to capture the most relevant geometric information. By contrast, through our module, different point features from sharp and gentle variation components are assigned different weights based on the geometric correlation with the anchor point in the original input. As shown in Fig. 5, a point in one leg of desk pays more attention to the points belongs to that leg among sharp-variation component and the relatively geometry-correlated points among gentle-variation component.

Therefore, compared with self-attention, our attention module drives the network to easily capture the distinct and complementary geometric information with less redundancy from two disentangled components. The quantitative comparison is listed in Table 4.

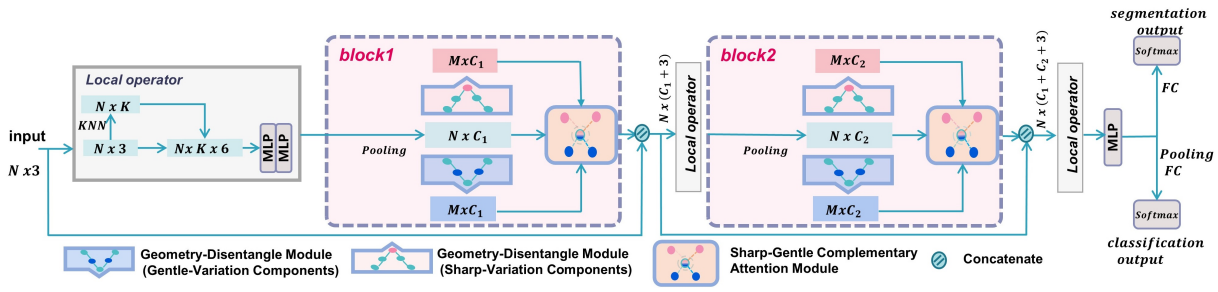

Figure 4: GDANet architecture for classification and segmentation. Our network disentangles the original point cloud into sharp and gentle variation components in different semantic levels, then fuses features from these two components with the input point features to supplement the KNN local context.

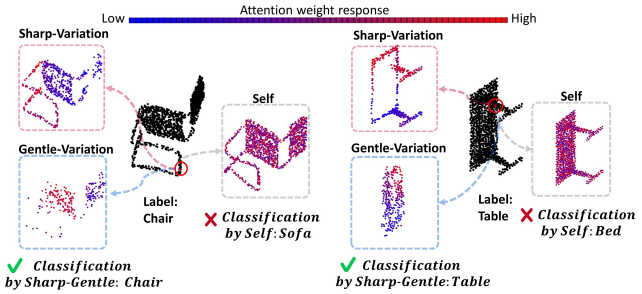

Figure 5: Visualization of the attention weights distribution on Sharp-Gentle Complementary Attention and Self-attention. We select some points in original point clouds as the anchor points (in **circle**) to investigate their attention distributions, where the points drawn in **red** are assigned higher attention weights and the points drawn in **blue** are assigned lower attention weights with respect to anchor points. We observe that anchor points pay different attentions to points based on the geometric correlation.

4.4 Geometry-Disentangled Attention Network

As illustrated in Fig. 4, we combine Geometry-Disentangle Module (GDM) and Sharp-Gentle Complementary Attention Module (SGCAM) as basic blocks to design Geometry-Disentangled Attention Network for 3D point cloud analysis. Before each block, features of each point and its K-nearest neighbors (KNN) are concatenated through the local operator. In each block, an adjacency matrix is constructed in the feature space through GDM to disentangle points into sharp and gentle variation components. Then we fuse features from these two components with the original input of the block through SGCAM. A residual connection (He et al. 2016) is applied at the end of each block. Two basic blocks are followed by another local operator. After the last MLP layer, the final global representation from max-pooling followed by fully connected and softmax layers is configured to the classification task. For the segmentation task, the outputs of the last MLP layer are directly passed through the fully connected as well as softmax layers to generate per-point scores for semantic labels. Note that each original input point is not only integrated with its nearest neighbors to capture local structure, but also linked with all of the disentangled sharp and gentle variation components that beyond the local area to describe distinct and complementary 3D geometries. Table 4 shows the quantitative comparison of applying our modules with only using KNN.

Dynamic Adjacency Matrix Calculation. Inspired by (Wang et al. 2019b), the adjacency matrix at the beginning of the GDM in each block is calculated in a dynamic way depending on the features learned in different semantic levels. Fig. 2 (b) shows that our module successfully disentangles points into sharp and gentle components in various objects. This disentanglement module is jointly optimized during learning to help the network better model the geometric structure of objects. Table 6 in Sec 5.2 suggests the quantitative comparison of this dynamic operation with the operation of pre-selecting points before training.

5 Experiments

We evaluate our network on shape classification task and part segmentation task on various datasets. Furthermore, we provide other experiments to analyze our network in depth.

5.1 3D Point Cloud Processing

Object Classification. We firstly evaluate GDANet on ModelNet40 (Wu et al. 2015). It contains 9843 training models and 2468 test models in 40 categories and the data is uniformly sampled from the mesh models by (Qi et al. 2017a). Same with (Wang et al. 2019b), the training data is augmented by randomly translating objects and shuffling the position of points. Similar to (Qi et al. 2017a,b), we perform several voting tests with random scaling and average the predictions during testing. Table 2 lists the quantitative comparisons with the state-of-the-art methods. GDANet outperforms other methods using only 1024 points as the input.

Our model is also tested on ScanObjectNN by (Uy et al. 2019), which is used to investigate the robustness to **noisy** objects with deformed geometric shape and non-uniform surface density in the real world. We adopt our model on the OBJ_ONLY (simplest variant of the dataset) and OBJ_BG (more noisy background). Sample objects of these variants are shown in Fig. 6. We retrain the methods listed in (Uy et al. 2019) and compare them with our network. The results

method (time order)	class mIOU	inst. mIOU	aero	bag	cap	car	chair	ear phone	guitar	knife	lamp	lap top	motor	mug	pistol	rocket	skate board	table
Kd-Net(Klokov and Lempitsky 2017)	77.4	82.3	80.1	74.6	74.3	70.3	88.6	73.5	90.2	87.2	81.0	94.9	57.4	86.7	78.1	51.8	69.9	80.3
PointNet(Qi et al. 2017a)	80.4	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	80.6
PointNet++(Qi et al. 2017b)	81.9	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
SyncCNN(Yi et al. 2017)	82.0	84.7	81.6	81.7	81.9	75.2	90.2	74.9	**93.0**	86.1	84.7	95.6	66.7	92.7	81.6	60.6	82.9	82.1
SCN(Xie et al. 2018)	81.8	84.6	83.8	80.8	83.5	79.3	90.5	69.8	91.7	86.5	82.9	96.0	69.2	93.8	82.5	62.9	74.4	80.8
KCNet(Shen et al. 2018)	82.2	84.7	82.8	81.5	86.4	77.6	90.3	76.8	91.0	87.0	84.5	95.5	69.2	94.4	81.6	60.1	75.2	81.3
SpiderCNN(Xu et al. 2018)	82.4	85.3	83.5	81.0	87.2	77.5	90.7	76.8	91.1	87.3	83.3	95.8	70.2	93.5	82.7	59.7	75.8	82.8
DGCNN(Wang et al. 2019b)	82.3	85.2	84.0	83.4	86.7	77.8	90.6	74.7	91.2	87.5	82.8	95.7	66.3	94.9	81.1	63.5	74.5	82.6
RS-CNN(Liu et al. 2019c)	84.0	86.2	83.5	84.8	88.8	79.6	**91.2**	81.1	91.6	88.4	**86.0**	96.0	73.7	94.1	83.4	60.5	77.7	83.6
DensePoint(Liu et al. 2019b)	84.2	86.4	84.0	85.4	90.0	79.2	91.1	81.6	91.5	87.5	84.7	95.9	74.3	94.6	82.9	**64.6**	76.8	83.7
InterpCNN(Mao, Wang, and Li 2019)	84.0	86.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3D-GCN(Lin, Huang, and Wang 2020)	82.1	85.1	83.1	84.0	86.6	77.5	90.3	74.1	90.0	86.4	83.8	95.6	66.8	94.8	81.3	59.6	75.7	82.8
GSNet(Xu et al. 2020)	83.5	85.3	82.9	84.3	88.6	78.4	89.7	78.3	91.7	86.7	81.2	95.6	72.8	94.7	83.1	62.3	81.5	83.8
GDANet(ours)	**85.0**	**86.5**	**84.2**	**88.0**	**90.6**	**80.2**	90.7	**82.0**	91.9	**88.5**	82.7	**96.1**	**75.8**	**95.7**	**83.9**	62.9	**83.1**	**84.4**

Table 1: Segmentation results (%) on ShapeNet Part dataset.

Method (time order)	Input	Acc.
PointNet(Qi et al. 2017a)	1K points	89.2
PointNet++(Qi et al. 2017b)	1K points	90.7
SCN(Xie et al. 2018)	1K points	90.0
KCNet(Shen et al. 2018)	1K points	91.0
PointCNN(Li et al. 2018)	1K points	92.2
PointWeb(Zhao et al. 2019)	1K points	92.3
Point2Sequence(Liu et al. 2019a)	1K points	92.6
DGCNN(Wang et al. 2019b)	1K points	92.9
KPConv(Thomas et al. 2019)	1K points	92.9
InterpCNN(Mao, Wang, and Li 2019)	1K points	93.0
DensePoint(Liu et al. 2019b)	1K points	93.2
Geo-CNN(Lan et al. 2019)	1K points	93.4
RS-CNN(Liu et al. 2019c)	1K points	93.6
3D-GCN(Lin, Huang, and Wang 2020)	1K points	92.1
FPConv(Lin et al. 2020)	1K points	92.5
GSNet(Xu, Zhou, and Qiao 2020)	1K points	92.9
GDANet(ours)	**1K points**	**93.8**
PointNet++(Qi et al. 2017b)	5K points+normal	91.9
PointConv(Wu, Qi, and Fuxin 2019)	1K points+normal	92.5

Table 2: Classification accuracy (%) on ModelNet40 dataset.

(a) Objects only. (b) Objects with background.

Figure 6: Visualization of objects in ScanObjectNN.

are summarized in Table 3, where our model gets the highest accuracy and the lowest performance drop from OBJ_ONLY to OBJ_BG. This proves the practicality of our method in the real world.

Shape Part Segmentation. Shape Part segmentation is a more challenging task for fine-grained shape recognition. We employ ShapeNet Part (Yi et al. 2016) that contains 16881 shapes with 16 categories and is labeled in 50 parts where each shape has 25 parts. Our network is trained with multiple heads to segment the parts of each object categories. Same voting test in classification is conducted. Table 1 summarizes the instance average, the class average and each class mean Inter-over-Union (mIoU). GDANet achieves the best performance with class mIoU of 85.0% and instance mIoU of 86.5%. It is worth mentioning that

Method	OBJ_ONLY	OBJ_BG	acc drop
PointNet(Qi et al. 2017a)	79.2	73.3	↓ 5.9
SpiderCNN(Xu et al. 2018)	79.5	77.1	↓ 5.4
PointNet++(Qi et al. 2017b)	84.3	82.3	↓ 2.0
DGCNN(Wang et al. 2019b)	86.2	82.8	↓ 3.4
PointCNN(Li et al. 2018)	87.9	85.8	↓ 2.1
GDANet(ours)	**88.5**	**87.0**	↓ 1.5

Table 3: Classification results (%) on ScanObjectNN dataset (noise robustness test).

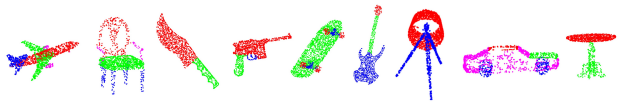

Figure 7: Part Semantic Segmentation examples.

GDANet performs better on objects with obvious geometric structure such as bag, mug and table. Fig. 7 visualizes some segmentation results.

5.2 Network Analysis

Ablation Study. The results are summarized in Table 4. When the input point cloud is fused with features from both sharp and gentle variation components, the network achieves the best accuracy with 93.4%. GDANet also surpasses the architecture of only using KNN with 1.2%↑. Eventually, our method obtains an accuracy of 93.8% with voting tests. This experiment supports our claim that the disentangled sharp and gentle variation components cooperatively provide different and complementary geometric information to supplement KNN local semantics.

Furthermore, we replace the selection of sharp and gentle variation components with random and Furthest Point Selection (FPS) in GDM. The results are listed in Table 5, where disentangling point clouds into sharp and gentle variation components gets the highest accuracy and is noticeably robust on the noisy dataset ScanObjectNN (Uy et al. 2019). Empirically, our disentanglement strategy selects points carried with informative geometries instead of noisy points, which improves the noise robustness.

Moreover, the result of dynamic adjacency matrix calculation (Sec 4.4) is shown in Table 6. We pre-compute the adjacency matrix on 3D coordinates to pre-disentangle different variation components, and fuse the features from them,

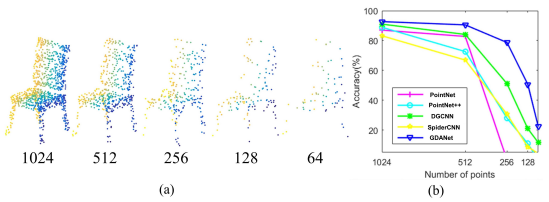

Figure 8: Density robustness test. (a). Point cloud with random point dropout. (b). Test results on ModelNet40 of using sparser points as the input to a model trained with 1024 points. For fair comparisons, all methods have no data enhancement of random point dropout during training.

knn	self	sharp	gentle	voting	acc. (%)
					91.5
✓					92.2
	✓				91.7
✓	✓				92.6
		✓	✓		92.7
✓		✓			92.7
✓			✓		92.4
✓		✓	✓		**93.4**
✓		✓	✓	✓	**93.8**

Table 4: Geometry-Disentangled complementary effect to supplement KNN information in GDANet on ModelNet40. 'knn' indicates KNN aggregation, 'self' means the input point cloud is fused with itself by self-attention, 'sharp' and 'gentle' denote the input point cloud is fused with features of sharp and gentle variation, 'voting' is the voting strategy during testing, respectively.

which gets 93.0% accuracy. Yet by dynamically calculating the adjacency matrix on semantic features in GDM at different levels, our network gains 0.8%↑. Our disentanglement module is jointly optimized during training for modeling different geometric structures. Fig. 2 shows that our disentanglement strategy successfully decompose points into sharp (contour) and gentle (flat area) variation components.

Last, we investigate the impact of the number of selected points M in GDM, which is shown in Table 7. GDANet performs best when selecting 25% of input points, which proves the benefit of our disentanglement strategy. When the number of selected points equals to the number of input, it indicates self-attention. More investigations of GDANet are included in the supplementary material.

Robustness Analysis. First, the robustness of our network on sampling density is tested by using sparser points as the input to GDANet trained with 1024 points. Fig. 8 shows the result. Although sparser points bring more difficulties, GDANet still performs consistently at all density settings.

Moreover, Table 8 summarizes the results of rotation robustness, where GDANet performs best especially with 2.7%↑ than the second best at (s/s).

Last, our model is tested on ScanObjectNN (Uy et al. 2019) that consists of noisy objects with deformed geometric shape and non-uniform surface density. Table 3 shows GDANet gains the lowest accuracy drop from OBJ_ONLY to OBJ_BG, which proves the noise robustness of GDANet.

method	ModelNet40	OBJ_ONLY	OBJ_BG
random	92.6	84.7	84.3
FPS	92.7	86.0	84.3
sharp-gentle	**93.8**	**88.1**	**86.6**

Table 5: Classification results (%) of using different point selection methods in our Geometry-Disentangle Module.

method	acc. (%)
Precomputed	93.0
Dynamic	**93.8**

Table 6: Impact of dynamic strategy.

number	1024	512	**256**	128
acc. (%)	92.6	93.2	**93.8**	92.9

Table 7: Selecting different number of points in GDM on ModelNet40.

Method	z/z	s/s
PointNet(Qi et al. 2017a)	81.6	66.3
PointNet++(Qi et al. 2017b)	90.1	87.8
SpiderCNN(Xu et al. 2018)	83.5	69.6
DGCNN(Wang et al. 2019b)	90.4	82.6
GDANet(ours)	**91.2**	**90.5**

Table 8: Accuracy (%) comparisons of rotation on ModelNet40. z/z: both training and test sets are augmented by random rotation for z axis; s/s: both training and test sets are augmented by random rotation for three axis (x,y,z).

Method	#params	acc. (%)
PointNet(Qi et al. 2017a)	3.50 M	89.2
PointNet++(Qi et al. 2017b)	1.48 M	90.7
KPConv(Thomas et al. 2019)	6.15 M	92.9
DGCNN(Wang et al. 2019b)	1.81 M	92.9
GSNet(Xu, Zhou, and Qiao 2020)	1.51 M	92.9
GDANet(ours)	**0.93 M**	**93.8**

Table 9: Comparisons of model complexity on ModelNet40.

Model Complexity. Table 9 shows the complexity by comparing the number of parameters. GDANet reduces the number of parameters by 84.9% and increases the accuracy with 0.9%↑ compared with KPConv (Thomas et al. 2019).

6 Conclusion

This work proposes GDANet for point cloud processing. Equipped with Geometry-Disentangle Module, GDANet dynamically disentangles point clouds into sharp and gentle variation components in different semantic levels, which respectively denotes the contour and flat area of a point cloud. Another core component is Sharp-Gentle Complementary Attention Module, which applies the attention mechanism to explore the relations between original points and different variation components to provide geometric complementary information for understanding point clouds. Extensive experiments have shown that our method achieves state-of-the-art performances and decent robustness.

7 Acknowledgments

This work was supported in part by Guangdong Special Support Program under Grant (2016TX03X276), in part by the National Natural Science Foundation of China under Grant (61876176, U1713208), and in part by the Shenzhen Basic Research Program (CXB201104220032A), the Joint Laboratory of CAS-HK. This work was done during Mutian Xu's internship at Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences.

References

Bengio, Y.; Courville, A.; and Vincent, P. 2012. Representation Learning: A Review and New Perspectives.

Chen, S.; Tian, D.; Feng, C.; Vetro, A.; and Kovačević, J. 2018. Fast Resampling of Three-Dimensional Point Clouds via Graphs. *IEEE Transactions on Signal Processing* 66(3): 666–681.

Guo, H.; Wang, J.; Gao, Y.; Li, J.; and Lu, H. 2016. Multi-View 3D Object Retrieval With Deep Embedding Network. *IEEE Transactions on Image Processing* 25(12): 5526–5537. doi:10.1109/TIP.2016.2609814.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *The IEEE International Conference on Computer Vision (ICCV)*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hornik, K. 1991. Approximation Capabilities of Multilayer Feedforward Networks. *Neural Netw.* 4(2): 251–257. ISSN 0893-6080. doi:10.1016/0893-6080(91)90009-T. URL https://doi.org/10.1016/0893-6080(91)90009-T.

Huang, X.; Liu, M.-Y.; Belongie, S.; and Kautz, J. 2018. Multimodal Unsupervised Image-to-image Translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Klokov, R.; and Lempitsky, V. 2017. Escape From Cells: Deep Kd-Networks for the Recognition of 3D Point Cloud Models. In *The IEEE International Conference on Computer Vision (ICCV)*.

Lan, S.; Yu, R.; Yu, G.; and Davis, L. S. 2019. Modeling Local Geometric Structure of 3D Point Clouds Using Geo-CNN. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; and Chen, B. 2018. PointCNN: Convolution On X-Transformed Points. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31*, 820–830. Curran Associates, Inc. URL http://papers.nips.cc/paper/7362-pointcnn-convolution-on-x-transformed-points.pdf.

Lin, Y.; Yan, Z.; Huang, H.; Du, D.; Liu, L.; Cui, S.; and Han, X. 2020. FPConv: Learning Local Flattening for Point Convolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lin, Z.-H.; Huang, S.-Y.; and Wang, Y.-C. F. 2020. Convolution in the Cloud: Learning Deformable Kernels in 3D Graph Convolution Networks for Point Cloud Analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, X.; Han, Z.; Liu, Y.-S.; and Zwicker, M. 2019a. Point2Sequence: Learning the Shape Representation of 3D Point Clouds with an Attention-based Sequence to Sequence Network. In *Thirty-Third AAAI Conference on Artificial Intelligence*.

Liu, Y.; Fan, B.; Meng, G.; Lu, J.; Xiang, S.; and Pan, C. 2019b. DensePoint: Learning Densely Contextual Representation for Efficient Point Cloud Processing. In *IEEE International Conference on Computer Vision (ICCV)*, 5239–5248.

Liu, Y.; Fan, B.; Xiang, S.; and Pan, C. 2019c. Relation-Shape Convolutional Neural Network for Point Cloud Analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 8895–8904.

Mao, J.; Wang, X.; and Li, H. 2019. Interpolated Convolutional Networks for 3D Point Cloud Understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Ortega, A.; Frossard, P.; Kovačević, J.; Moura, J. M. F.; and Vandergheynst, P. 2018. Graph Signal Processing: Overview, Challenges, and Applications. *Proceedings of the IEEE* 106(5): 808–828. doi:10.1109/JPROC.2018.2820126.

Qi, C. R.; Liu, W.; Wu, C.; Su, H.; and Guibas, L. J. 2018. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 918–927.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30*, 5099–5108. Curran Associates, Inc. URL http://papers.nips.cc/paper/7095-pointnet-deep-hierarchical-feature-learning-on-point-sets-in-a-metric-space.pdf.

Rusu, R. B.; Marton, Z. C.; Blodow, N.; Dolha, M.; and Beetz, M. 2008. Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems* 56(11): 927–941.

Sandryhaila, A.; and Moura, J. M. F. 2014. Discrete Signal Processing on Graphs: Frequency Analysis. *IEEE Transactions on Signal Processing* 62(12): 3042–3054. doi:10.1109/TSP.2014.2321121.

Shen, Y.; Feng, C.; Yang, Y.; and Tian, D. 2018. Mining Point Cloud Local Structures by Kernel Correlation and Graph Pooling. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Thomas, H.; Qi, C. R.; Deschaud, J.-E.; Marcotegui, B.; Goulette, F.; and Guibas, L. J. 2019. KPConv: Flexible and Deformable Convolution for Point Clouds. *Proceedings of the IEEE International Conference on Computer Vision* .

Uy, M. A.; Pham, Q.-H.; Hua, B.-S.; Nguyen, T.; and Yeung, S.-K. 2019. Revisiting Point Cloud Classification: A New Benchmark Dataset and Classification Model on Real-World Data. In *The IEEE International Conference on Computer Vision (ICCV)*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30*, 5998–6008. Curran

Associates, Inc. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; and Shan, J. 2019a. Graph Attention Convolution for Point Cloud Semantic Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local Neural Networks. *CVPR* .

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019b. Dynamic Graph CNN for Learning on Point Clouds. *ACM Trans. Graph.* 38(5): 146:1–146:12. ISSN 0730-0301. doi:10.1145/3326362. URL http://doi.acm.org/10.1145/3326362.

Wu, W.; Qi, Z.; and Fuxin, L. 2019. PointConv: Deep Convolutional Networks on 3D Point Clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3D ShapeNets: A Deep Representation for Volumetric Shapes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xiao, T.; Hong, J.; and Ma, J. 2018. ELEGANT: Exchanging Latent Encodings with GAN for Transferring Multiple Face Attributes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 172–187.

Xie, S.; Liu, S.; Chen, Z.; and Tu, Z. 2018. Attentional ShapeContextNet for Point Cloud Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xu, M.; Zhou, Z.; and Qiao, Y. 2020. Geometry Sharing Network for 3D Point Cloud Classification and Segmentation. In *AAAI*, 12500–12507.

Xu, Y.; Fan, T.; Xu, M.; Zeng, L.; and Qiao, Y. 2018. SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters. In *The European Conference on Computer Vision (ECCV)*.

Yi, L.; Kim, V. G.; Ceylan, D.; Shen, I.-C.; Yan, M.; Su, H.; Lu, C.; Huang, Q.; Sheffer, A.; and Guibas, L. 2016. A Scalable Active Framework for Region Annotation in 3D Shape Collections. *ACM Trans. Graph.* 35(6): 210:1–210:12. ISSN 0730-0301. doi:10.1145/2980179.2980238. URL http://doi.acm.org/10.1145/2980179.2980238.

Yi, L.; Su, H.; Guo, X.; and Guibas, L. J. 2017. SyncSpecCNN: Synchronized Spectral CNN for 3D Shape Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhao, H.; Jiang, L.; Fu, C.-W.; and Jia, J. 2019. PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing. In *CVPR*.

Supplementary Material for
Learning Geometry-Disentangled Representation for Complementary Understanding of 3D Object Point Cloud

Outline

This supplementary document is arranged as follows:
(1) Sec A details network parameters and implementation strategies;
(2) Sec B investigates the number of blocks consisted by our modules in the network;
(3) Sec C explores the number of selected points M in Geometry-Disentangle Module on shape part segmentation;
(4) Sec D lists the result of employing our method on PointNet for indoor scenes segmentation dataset, i.e, S3DIS (Armeni et al. 2016);
(5) Sec E shows more visualization examples of the attention weights distribution of our Sharp-Gentle Complementary Attention.

A. Network parameters and implementations

Fig.4 in the paper shows the GDANet architecture. As for the feature dimension, two MLPs (64, 64) in block1 are employed so that $C_1 = 64$. In block2, two MLPs (64, 64) produce $C_2 = 64$. After the last KNN aggregation out of block2, two MLPs (128, 128), a pooling layer and another MLPs (512) are used. At last, three fully-connected layers (256, 256, 128) with dropout keeping probability of 0.4 are followed. All layers include ReLU and batch normalization. As for segmentation network, we also concatenate the one-hot encoding (16-d) of the object label to the last feature layer.

We set the number of neighbors $k = 30$ for both classification and segmentation tasks during every KNN aggregation.

For the classification task, we use SGD with learning rate 0.1 and reduce it to 0.001 with cosine annealing. The momentum is 0.9 and the weight decay is 10^{-4}. The batch size is set to 64. For the segmentation task, Adam with learning rate 0.003 is employed and is divided by 2 each 40 epochs. The weight decay is 0 and the batch size is 64.

GDANet is implemented using Pytorch. A data-parallel training scheme is adopted on several Nvidia GeForce GTX 1080 Ti GPUs.

# blocks	1	**2**	3
acc. (%)	93.2	**93.8**	93.6

Table I: Impact of blocks numbers on ModelNet40.

# blocks	1	**2**	3
instance mIoU (%)	86.0	**86.5**	86.3
class mIoU (%)	84.4	**85.0**	84.9

Table II: Imapct of blocks numbers on ShapeNet Part.

B. GDANet with different number of blocks

As stated in Sec 4.4 of the paper, we combine Geometry-Disentangle Module and Sharp-Gentle Complementary Attention Module as basic blocks in our network. Table I and Table II show how the number of blocks influences the performance on classification and part segmentation task, respectively. When GDANet employs two basic blocks, it achieves the best performance on both tasks.

C. Selecting different number of points in GDM for shape part segmentation

Table 7 in the paper suggests the effect of selecting different number of points M in Geometry-Disentangle Module for classification task. Here we also explores this on Shape Part Segmentation task with 2048-points input. We set M=2048, 1024, 512, 256 in GDM and the results are shown in Table III, where our network achieves the best performance when selecting 25% of input points in GDM.

This experiment further proves the benefit of our disentanglement strategy. Note that when the number of selected points in GDM equals to the number of input (2048 for segmentation), it indicates applying self-attention on the original input point cloud. (Sec 4.3 of the paper).

# selected points	2048	1024	**512**	256
instance mIoU (%)	85.7	86.3	**86.5**	86.2
class mIoU (%)	83.3	84.6	**85.0**	84.5

Table III: Different number of points in GDM on ShapeNet Part.

Method	mIoU(%)
PointNet	41.1
PointNet + Self-Attention	47.5
PointNet + InSem-SP (Liu et al. 2020)	52.5
PointNet + Our Blocks	**52.3**

Table IV: Plugging our basic blocks into PointNet on S3DIS Area-5.

D. Employing our method on PointNet for indoor scenes segmentation

To further verify the generalization of our proposed modules on large-scale point clouds, we directly replace two MLP layers of PointNet (Qi et al. 2017) with two blocks of our own network without changing the original number of channels in PointNet, and test it on indoor scenes segmentation task. We choose S3DIS as the dataset, which covers six large-scale indoor areas from three different buildings for a total of 273 million points annotated with 13 classes. Following recent methods, we use Area-5 as the test scene to measure the generalization ability of our method. The results are shown in Table IV, where our own blocks improve the segmentation mIoU of PointNet from 41.1% to 52.3% (**11.2**%↑), while replacing with self-attention only brings 8.4%↑. Our method also achieves the comparable performance with the recent SOTA plug-in method (Liu et al. 2020). This proves the generalization of our method on large-scale point clouds.

E. More visualizations of the attention weights distribution on Sharp-Gentle Complementary Attention

We provide more visualization examples of the attention weights distribution on Sharp-Gentle Complementary Attention (Sec 4.3 of the main paper). As shown in Fig. I, different points from sharp and gentle variation components are assigned different weights based on the geometric correlation with the anchor points in original input point clouds.

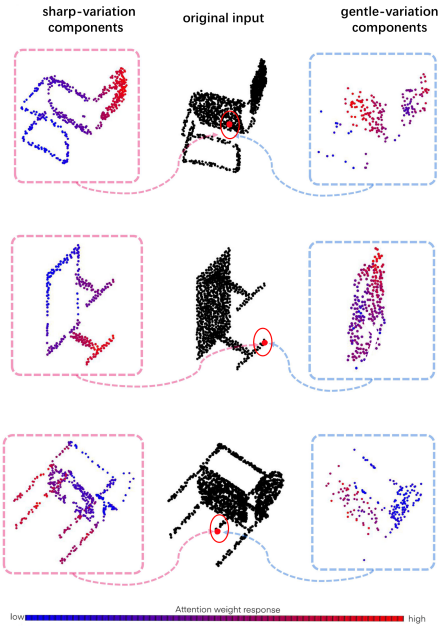

Figure I: Visualization of the attention weights distribution on Sharp-Gentle Complementary Attention and Self-attention. We select some points in original point clouds as the anchor points (in **circle**) to investigate their attention distributions, where the points drawn in **red** are assigned higher attention weights and the points drawn in **blue** are assigned lower attention weights with respect to anchor points. We observe that anchor points pay different attentions to points based on the geometric correlation.

References

Armeni, I.; Sener, O.; Zamir, A. R.; Jiang, H.; Brilakis, I.; Fischer, M.; and Savarese, S. 2016. 3d semantic parsing of large-scale indoor spaces. In *CVPR*.

Liu, J.; Yu, M.; Ni, B.; and Chen, Y. 2020. Self-Prediction for Joint Instance and Semantic Segmentation of Point Clouds. In *ECCV*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.