Self-Prediction for Joint Instance and Semantic Segmentation of Point Clouds

Jinxian Liu[1,2*], Minghui Yu[1,2*], Bingbing Ni[1,2†], and Ye Chen[1,2]

[1] Shanghai Jiao Tong University
{liujinxian,1475265722,nibingbing,chenye123}@sjtu.edu.cn
[2] Huawei Hisilicon
{liujinxian1,yuminghui2,nibingbing,chenye17}@hisilicon.com

Abstract. We develop a novel learning scheme named Self-Prediction for 3D instance and semantic segmentation of point clouds. Distinct from most existing methods that focus on designing convolutional operators, our method designs a new learning scheme to enhance point relation exploring for better segmentation. More specifically, we divide a point cloud sample into two subsets and construct a complete graph based on their representations. Then we use label propagation algorithm to predict labels of one subset when given labels of the other subset. By training with this Self-Prediction task, the backbone network is constrained to fully explore relational context/geometric/shape information and learn more discriminative features for segmentation. Moreover, a general associated framework equipped with our Self-Prediction scheme is designed for enhancing instance and semantic segmentation simultaneously, where instance and semantic representations are combined to perform Self-Prediction. Through this way, instance and semantic segmentation are collaborated and mutually reinforced. Significant performance improvements on instance and semantic segmentation compared with baseline are achieved on S3DIS and ShapeNet. Our method achieves state-of-the-art instance segmentation results on S3DIS and comparable semantic segmentation results compared with state-of-the-arts on S3DIS and ShapeNet when we only take PointNet++ as the backbone network.

Keywords: Self-Prediction, Instance Segmentation, Semantic Segmentation, Point Cloud, State-of-the-art, S3DIS, ShapeNet

1 Introduction

With the growing popularity of low-cost 3D sensors, e.g., LiDAR and RGB-D cameras, 3D scene understanding is tremendous demand recently due to its great application values in autonomous driving, robotics, augmented reality, etc. 3D data provides rich information about the environment, however, it is hard for traditional convolutional neural networks (CNNs) to process this irregular data.

Fortunately, many ingenious works [20, 21, 33, 24, 36, 23, 5, 10, 14, 8, 42, 13, 15] are proposed to directly process point cloud, which is the simplest 3D data format. This motivates us to work with 3D point clouds.

The key to better understanding a 3D scene is to learn more discriminative point representations. To this end, many works [14, 33, 27, 37, 8] elaborately design various point convolution operators to capture semantic or geometric relation among points. DGCNN [33] proposes to construct a KNN graph and an operator named EgdeConv to process this graph, where semantic relation among points is explicitly modeled. RelationShape [14] attempts to model geometric point relation in local areas, hence local shape information is captured. Other methods also share similar design philosophy. Although good segmentation performance is achieved by explicitly modeling points relation, lack of constraint/guidance on relation exploring limits the network from reaching its full potential. Hence a constraint is urgently needed to enforce/guide/encourage this relation exploring and helps the network learn more representative features.

3D Instance and semantic segmentation are two of the most important tasks in 3D scene understanding. Many works [25, 3, 1, 9, 40, 41, 38] tackle these two tasks separately. And some works [17, 19] address these two tasks in a serial fashion, where instance segmentation is usually formulated as a post-processing task of semantic segmentation. However, this formulation often gets a sub-optimal solution since the performance of instance segmentation highly depends on the performance of semantic segmentation. Actually, these two tasks could be associated and cooperate with each other as proved in ASIS [32] and JSIS3D [18]. They propose to couple these two tasks in a parallel fashion. ASIS makes instance segmentation benefit from semantic segmentation through learning semantic-aware instance embeddings. Semantic features of the points belonging to the same instance are fused to make more accurate semantic predictions. However, extra parameters and computation burden are introduced during inference. JSIS3D combines these two tasks in a simple way. They formulate it as a simple multi-task problem and just train the two tasks simultaneously. A multi-value conditional random fields model is proposed to jointly optimize class labels and object instances. However, it is a time consuming post-processing scheme and cannot be optimized end-to-end. Moreover, performance improvements achieved by ASIS and JSIS3D are both limited.

To address these two issues, we propose a novel learning scheme named Self-Prediction to constrain the network to fully capture point relation and a unified framework that equipped with this scheme to associate instance and semantic segmentation. The framework of our method is shown in Figure 1, which contains a backbone network and three heads named instance-head, semantic-head and Self-Prediction head respectively. The instance-head learns instance embeddings for instance clustering and the semantic-head outputs semantic embeddings for semantic prediction. In Self-Prediction head, the instance and semantic embeddings for each point are combined. We then concatenate semantic and instance labels to form a multi-label for every point. After that, we divide the point cloud into two groups with one group's labels being discarded. Given the com-

bined embeddings of the whole point cloud and labels of one group, we construct a complete graph and then predict semantic and instance labels simultaneously for the other group using label propagation algorithm. It should be noted that bidirectional propagation among the two groups are performed. Through this procedure of multi-label Self-Prediction, the instance and semantic embeddings are associatively enhanced. The process of Self-Prediction incorporates embedding similarity of points, which enforces the network to explore effective relation among points and learn more discriminative representations. The three heads are jointly optimized at training time. During inference, our Self-Prediction head is discarded, and no computation burden and network parameters are introduced. Our framework is demonstrated to be general and effective on different backbone networks such as PointNet, PointNet++, etc. Significant performance improvements over baseline are achieved on both instance and semantic segmentation. By only taking PointNet++ as the backbone, our method achieves state-of-the-art instance segmentation results and comparable semantic segmentation results compared with state-of-the-art networks.

2   Related Work

Instance Segmentation in 3D Point Clouds. A pioneer work for instance segmentation in 3D point clouds can be found in [31], which uses similarity matrix to yield proposals followed by confidence map for pruning proposals and utilizes semantic map for assigning labels. ASIS [32] proposes to associate instance segmentation and semantic segmentation to achieve semantic awareness for instance segmentation. JSIS3D [18] introduces a multi-value CRF model to jointly optimize class labels and object instances. However, their performance is quite limited. Encouraged by the success of RPN and RoI, GSPN [41] generates proposals by reconstructing shapes and proposes Region-based PointNet to get final segmentation results. 3D-SIS [4] is also a proposal-based method. However, proposal-based methods are usually two-stages and need pruning proposals. 3D-BoNet [38] directly predicts point-level masks for instances within detected object boundaries. It is single-stage, anchor free and computationally efficient. However, there is a limitation on adaptation to different types of input point clouds. In this work, we propose a unified framework equipped with an efficient learning scheme to simultaneously improve instance and semantic segmentation significantly.

Semantic Segmentation in 3D Point Clouds. PointNet [20] is the first to directly consume raw point clouds which processes each point identically and independently and then aggregates them through global max pooling. It well respects order invariances of input data and achieves strong performance. Pointnet++ [21] applies PointNet in a recursive way to learn local features with increasing contextual scales thus it achieves both robustness and detailed features. Attention [36, 39, 29, 43] has also been paied to aggregate local features effectively. RSNet [6] proposes a lightweight local dependency module to efficiently model local structures in point clouds, which is composed by slice pooling
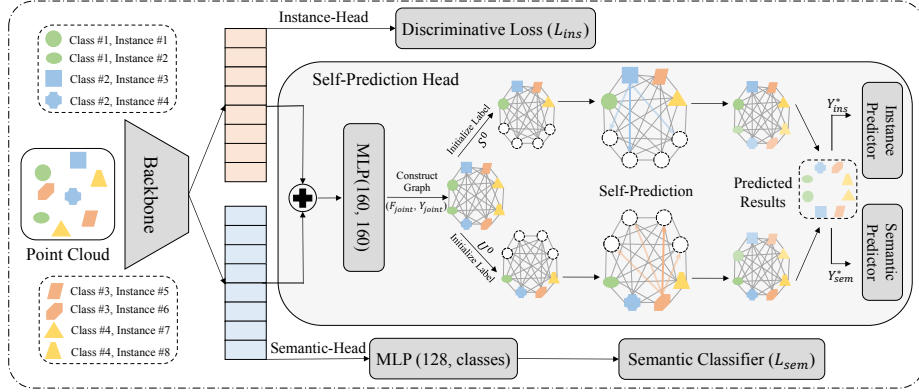
Fig. 1. The overall framework of our method. The input point cloud goes through a backbone network to extract instance and semantic features for instance and semantic segmentation respectively. These two features are then combined to construct a complete graph to perform bidirectional Self-Prediction in Self-Prediction head.

layers, RNN layers and slice unpooling layers. SPLATNET [24] utilizes sparse bilateral convolutional layers to maintain efficiency and flexibility. PointCNN [10] explores $\mathcal{X}$-transformation to promote both weighting input point features and permutation of points into a latent and potentially canonical order. Graph convolutions [23, 9, 27] are also proposed for improving semantic segmentation task. SPG [9] adapts graph convolutional network on compact but rich representations of contextual relationship between object parts. SEGCloud [25] combines advantages of neural network and conditional random field to get coarse to fine semantics on points. DGCNN [33] tries to capture local geometric structures by a new operation named EdgeConv, which generates edge features describing relation between a point and its neighbors. [7] shares the same idea of edge features, which constructs an edge branch to hierarchically integrate point features and edge features. Different from these methods, PointConv [34] proposes a density re-weighted convolution which can closely approximate 3D continuous convolution on 3D point set. KPConv [26] uses kernel points located in Euclidean Space to define the area where each kernel weight is applied, which well models local geometry. Our method can take most of these models as backbone network and achieve better segmentation performance.

Label Propagation Alogrithms. Label propagation is derived from unsupervised learning. [35] is an earlier attempt to address this issue where the labeled data act as sources that push out labels through unlabeled data and then developed by [45], which introduces consistency assumption to guide label propagation. It is mentioned in [28] that since scale parameter σ will affect the performance significantly, to address this issue, LNP uses overlapped linear neighborhood patches to approximate the whole graph. How to automatically learn optimal σ is worthwhile exploring. [45] proposes to learn parameter σ by minimum spanning tree heuristic and entropy minimization. Label propagation

algorithms are designed to enhance models with unlabeled samples. Motivated by our intention of enhancing point relation exploring, we design a new learning scheme to restrict the results predicted by label propagation algorithm be identical with their ground truth.

3   Methodology

We design a novel learning scheme named Self-Prediction to strengthen our backbone networks and learn more discriminative representations for better segmentation. Among a point cloud sample, this proposed scheme encourages the network to capture more effective relation between points by predicting the label of a part of points when given labels of rest points and all points embeddings. Equipped with Self-Prediction, a unified framework is proposed to combine instance and semantic segmentation and conduct these two tasks in a mutually reinforced fashion. The overall framework of our method is shown in Figure 1. In this section, we first introduce our proposed general Self-Prediction scheme. Then we present how to use this scheme to conduct instance and semantic segmentation jointly, and describe the overall framework. Finally, we summarize the global optimization objectives of our method.

3.1   Self-Prediction

Self-Prediction is an auxiliary task paralleled with instance and segmentation tasks, and is designed to enforce backbone networks to learn more strong and discriminative representations. To get better segmentation performance, many existing works [33, 43, 27, 26] elaborately design convolution operators to capture relation, geometric and shape information contained in point clouds. Our common goal is to learn more discriminative representations. However, we take a new perspective. We think that if the learned representations can be utilized to predict instance/semantic labels of a part of a point cloud when given labels of rest points in a point cloud, it can be considered to have fully exploited the relation information and be representative enough. Hence we naturally formulate a Self-Prediction task, i.e., equally divide a point cloud into two groups, and then perform bidirectional prediction between the two groups given their representations. By constraining the network to perform well on Self-Prediction task, we get more strong features and perform better on specific tasks, i.e., instance and semantic segmentation.

   Given a point cloud example that contains N points $X = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_N}\}$, each point $\mathbf{x_i} \in \mathbb{R}^h$ can be represented by coordinates, color, normal, etc. h is the dimension of features of input point. For each point x_i, its class label is represented by a one-hot vector. We formulate a label matrix $\mathbf{Y} \in \mathcal{Y}$, where each row of matrix $\mathbf{Y}$ denotes the one-hot label of point $\mathbf{x_i}$ and $\mathcal{Y}$ denotes the set of $N \times C$ matrix (C is the number of classes) with non-negative elements.

   We equally divide a point cloud into two groups, i.e., $X_S = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_M}\}$ with its label matrix $\mathbf{Y}_{1:M}$ and $X_U = \{\mathbf{x_{M+1}}, \mathbf{x_{M+2}}, ..., \mathbf{x_N}\}$ with its label ma-

trix $\mathbf{Y}_{M+1:N}$. We use label propagation algorithm to perform bidirectional Self-Prediction between point subsets X_S and X_U, i.e., propagating labels from X_S to X_U and from X_U to X_S inversely. Firstly, we construct a complete graph $\mathbf{W} \in \mathbb{R}^{N \times N}$, each element of which is defined by Gaussian similarity function:

$$\mathbf{W}_{ij} = exp(-\frac{d(\varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j))}{2\sigma^2}). \tag{1}$$

φ is the backbone network and $\varphi(\mathbf{x}_i)$ denotes extracted features of point x_i. d is Euclidean distance measure function and σ is the length scale parameter used to adjust the strength neighbors. We set σ to 1 in all our experiments. Then we normalize the constructed graph by computing Laplacian matrix:

$$\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}, \tag{2}$$

where $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii}$ to be the sum of the i-th row of $\mathbf{W}$, i.e., $\mathbf{D}_{ii} = \sum_{j=1}^{N} \mathbf{W}_{ij}$. To predict labels of X_U when given labels of X_S and labels of X_S when given labels of X_U respectively, we have to prepare two initial label matrices $\mathbf{S^0}$ and $\mathbf{U^0}$ by padding $\mathbf{Y}_{1:M}$ and $\mathbf{Y}_{M+1:N}$ with zero vectors correspondingly. Specifically, $\mathbf{S^0}$ and $\mathbf{U^0}$ are represented by:

$$\mathbf{S^0} = [\mathbf{Y}_1^T, ..., \mathbf{Y}_M^T, \mathbf{0}^T, ..., \mathbf{0}^T]^T,$$
$$\mathbf{U^0} = [\mathbf{0}^T, ..., \mathbf{0}^T, \mathbf{Y}_{M+1}^T, ..., \mathbf{Y}_N^T]^T, \tag{3}$$

where $\mathbf{Y}_i$ denotes the i-th row of label matrix $\mathbf{Y}$. The Self-Prediction procedure is conducted by label propagation algorithm, the iterative version of which is as follows:

$$\mathbf{S^{(t+1)}} = \alpha\mathbf{L}\mathbf{S^{(t)}} + (1 - \alpha)\mathbf{S^0},$$
$$\mathbf{U^{(t+1)}} = \alpha\mathbf{L}\mathbf{U^{(t)}} + (1 - \alpha)\mathbf{U^0}, \tag{4}$$

where α is a parameter used to control the propagation proportion, i.e., how much the initial label matrix has effect on propagated results. Following the common setting [12], we set α to 0.99 in all our experiments. $\mathbf{S^{(t)}} \in \mathcal{Y}$ and $\mathbf{U^{(t)}} \in \mathcal{Y}$ are the t-th iteration results. We will get the final results $\mathbf{S^*}$ and $\mathbf{U^*}$ by iterating Equation 4 until convergence. In practice, we directly use the closed form of the above iteration version that proposed in [45] to get propagated/predicted results. We present the closed form expression as follows:

$$\mathbf{S^*} = (\mathbf{I} - \alpha\mathbf{L})^{-1}\mathbf{S^0},$$
$$\mathbf{U^*} = (\mathbf{I} - \alpha\mathbf{L})^{-1}\mathbf{U^0}, \tag{5}$$

where $\mathbf{I} \in \mathbb{R}^{N \times N}$ is the identity matrix. It should be noted that $\mathbf{S^*}_{M+1:N}$ and $\mathbf{U^*}_{1:M}$ are valid propagated results. We can predict label of x_i by $\arg\max \mathbf{U^*}_i$ when $1 < i \le M$ and $\arg\max \mathbf{S^*}_i$ when $M < i \le N$. We formulate the final self-predicted results $\mathbf{Y^*} \in \mathcal{Y}$ as:

$$\mathbf{Y^*} = [\mathbf{U^*}_{1:M}^T, \mathbf{S^*}_{M+1:N}^T]^T. \tag{6}$$

Finally, we use ground truth label matrix $\mathbf{Y}$ as supervised signal to train this Self-Prediction task.

3.2   Associated Learning Framework

As shown in Figure 1, our proposed framework contains one backbone network and three heads. The backbone network can be almost all existing point cloud learning architectures. We take PointNet, PointNet++ as examples in our work. Based on the backbone network, three heads are utilized to perform instance segmentation, semantic segmentation and Self-Prediction task respectively.

Taken a point cloud X as input, the backbone network output a feature matrix $F \in \mathbb{R}^{N \times H}$, where H denotes dimension of output features. Instance-head takes F as input and transform it into point-wise instance embeddings $F_{ins} \in \mathbb{R}^{N \times H_{ins}}$, where H_{ins} is dimension of instance embeddings and set to 32 in all our experiments. We adopt the same discriminative loss function as [32] and [18] to supervise instance segmentation. If a point cloud example contains K instances and the k-th ($k \in 1, 2, ...K$) instance contains N_k points, we denote $\mathbf{e}_j \in \mathbb{R}^{H_{ins}}$ as the instance embedding of the j-th point and $\boldsymbol{\mu}_k \in \mathbb{R}^{H_{ins}}$ as the mean embedding of the k-th instance. Hence the instance loss is written as :

$$\mathcal{L}_{var} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{N_k} \sum_{j=1}^{N_k} \left[\|\boldsymbol{\mu}_k - \mathbf{e}_j\|_2 - \delta_{\mathrm{v}} \right]_+^2, \tag{7}$$

$$\mathcal{L}_{dist} = \frac{1}{K(K-1)} \sum_{k=1}^{K} \sum_{m=1, m \neq k}^{K} \left[2\delta_{\mathrm{d}} - \|\boldsymbol{\mu}_k - \boldsymbol{\mu}_m\|_2 \right]_+^2, \tag{8}$$

$$\mathcal{L}_{reg} = \frac{1}{K} \sum_{k=1}^{K} \|\boldsymbol{\mu}_k\|_2, \tag{9}$$

$$\mathcal{L}_{ins} = \mathcal{L}_{var} + \mathcal{L}_{dist} + 0.001 \cdot \mathcal{L}_{reg} \tag{10}$$

where $[x]_+ = max(0, x)$, δ_{v} and δ_{d} are margins for $\mathcal{L}_{var}$ and $\mathcal{L}_{dist}$ respectively. Instance labels are obtained by conducting mean-shift clustering [2] on instance embeddings during inference.

The semantic-head takes feature matrix F as input and learns a semantic embedding matrix $F_{sem} \in \mathbb{R}^{N \times H_{sem}}$ to further perform point-wise classification that supervised by cross-entropy loss. H_{sem} is dimension of point semantic embedding and set to 128 in all our experiments.

In Self-Prediction head, we combine instance and semantic embeddings and jointly self-predict instance and semantic labels. Specifically, we concatenate F_{ins} and F_{sem} along the axis of features and transform it into a joint embedding matrix $F_{joint} \in \mathbb{R}^{H_{joint}}$, where H_{joint} is dimension of joint embeddings and set to 160 in all our experiments. For each point in X, we transform its semantic and instance label into one-hot form respectively. Instance label of each point denotes which instance it belongs to. This instance label is semantic-agnostic, i.e., we cannot infer the semantic label of a point from its instance label. We assume that a dataset contains C_{sem} semantic classes and the input point cloud sample X contains C_{ins} instances. Then we denote the semantic label matrix and instance label matrix as $\mathbf{Y}_{sem} \in \mathcal{Y}_{sem}$ and $\mathbf{Y}_{ins} \in \mathcal{Y}_{ins}$ respectively, where $\mathcal{Y}_{sem}$

is the set of $N \times C_{sem}$ matrix with non-negative elements and $\mathcal{Y}_{ins}$ is the set of $N \times C_{ins}$ matrix with non-negative elements. Given the two label matrices, we formulate a multi-label matrix $\mathbf{Y}_{joint} \in \mathcal{Y}_{joint}$ by concatenating semantic label and instance label of each point, where $\mathcal{Y}_{joint}$ is the set of $N \times (C_{sem} + C_{ins})$ matrix with non-negative elements. In other words, one can infer which semantic class and instance each point belongs to from the $\mathcal{Y}_{joint}$. We finally carry out Self-Prediction described in Section 3.1 based on the joint feature matrix F_{joint} and multi-label matrix $\mathbf{Y}_{joint}$. We slice the self-predicted results $\mathbf{Y}_{joint}^{*} \in \mathcal{Y}_{joint}$ into semantic results $\mathbf{Y}_{sem}^{*} \in \mathcal{Y}_{sem}$ and instance results $\mathbf{Y}_{ins}^{*} \in \mathcal{Y}_{ins}$, which are then supervised by semantic ground truth $\mathbf{Y}_{sem}$ and instance ground truth $\mathbf{Y}_{ins}$ respectively. It should be noted that our Self-Prediction is conducted among one point cloud sample every time, hence it does not matter that the meaning of instance label varies from sample to sample.

Instance-head, semantic-head and Self-Prediction head are jointly optimized. Instance-head and semantic-head are aimed to get segmentation results. Our proposed Self-Prediction head incorporates similarity relation among points and enforces the backbone to learn more discriminative representations. These three heads cooperate with each other and get better segmentation performance. We want to emphasize that our Self-Prediction head is discarded and only instance-head and semantic-head are used during inference, hence no extra computational burden and space usage are introduced.

3.3   Optimization Objectives

We train the instance-head with the instance loss $\mathcal{L}_{ins}$ that formulated in Equation 10. The semantic-head is trained by classical cross-entropy loss and supervised by semantic label $\mathbf{Y}_{sem}$, which is written as:

$$\mathcal{L}_{sem} = -\frac{1}{N} \sum_{i=1}^{N} [\mathbf{Y}_{sem}]_i \log \mathbf{p}_i, \qquad (11)$$

where $\mathbf{p}_i$ denotes output probability distribution computed by softmax function.

Given the jointly self-predicted results $\mathbf{Y}_{ins}^{*}$ and $\mathbf{Y}_{sem}^{*}$, we train our Self-Prediction head also by cross-entropy loss, which is formulated as:

$$\mathcal{L}_{sp} = -\frac{1}{N} \sum_{i=1}^{N} ([\mathbf{Y}_{ins}]_i * \log \mathbf{q}_i + [\mathbf{Y}_{sem}]_i * \log \mathbf{r}_i), \qquad (12)$$

where $\mathbf{q}_i$ and $\mathbf{r}_i$ are output probability distribution (computed by softmax) of the i-th row of $\mathbf{Y}_{ins}^{*}$ and $\mathbf{Y}_{sem}^{*}$ respectively. The output probability distribution is also computed by softmax function.

The three head are jointly optimized and the overall optimization objective is a weighted sum of above three losses:

$$\mathcal{L} = \mathcal{L}_{ins} + \mathcal{L}_{sem} + \beta \mathcal{L}_{sp}, \qquad (13)$$

where β is used to balance contributions of the three above terms such that they contribute equally to the overall loss. β is set to 0.8 in all our experiments.

4   Experiments

4.1   Experiment Settings

Datasets   Stanford 3D Indoor Semantics Dataset (S3DIS) is a large scale real scene segmentation benchmark and contains 6 areas with a total of 272 rooms. Each 3D RGB point is annotated with an instance label and a semantic label from 13 categories. Each room is typically parsed to about 10-80 object instances. ShapeNet part dataset contains 16681 samples from 16 categories. There are totally 50 parts, and each category contains 2-6 parts. The instance annotations are got from [31], which is used as ground truth instance label.

Evaluation Metrics   On S3DIS dataset, following the common evaluation settings, we validate our method in a 6-fold cross validation fashion over the 6 areas, i.e., 5 areas are used for training and the left 1 area for validation each time. Moreover, test results on Area 5 are reported individually due to no overlaps between Area 5 and left areas, which is a better way to show generalization ability of methods. For evaluation of semantic segmentation, we use mean IoU (mIoU) across all the categories, class-wise mean of accuracy (mAcc) and point-wise overall accuracy (oAcc) as metrics. We take the same evaluation metric as [32] for instance segmentation. Apart from common used metric mean precision (mPrec) and mean recall (mRec) with IoU threshold 0.5, coverage and weighted coverage (Cov, WCov) [11, 22, 46] are taken. Cov is the average instance-wise IoU between prediction and ground truth. WCov means Cov that is weighted by the size of the ground truth instances. On ShapeNet, part-averaged IoU (pIoU) and mean per-class pIoU (mpIoU) are taken as evaluation metrics for semantic segmentation. Following [31, 32], we only provide qualitative results of part instance segmentation on ShapeNet.

Implementation Details   For experiments on S3DIS, we follow the same setting as PointNet [20], where each room is split into blocks of area $1m \times 1m$. Each 3D point is represented by a 9-dim vector, (XYZ, RGB and normalized locations as to the room). We sample 4096 points for each block during training and all points are used for testing. We have mentioned above that we construct a graph and then divide the point cloud into two groups to perform Self-Prediction in Self-Prediction head. In practice, we partition the point cloud into more than two groups for acceleration. Specifically, we divide every block equally into 8 groups according to their instance labels, i.e., guarantee points of each instance are averagely distributed in every group. As a result, points of each semantics are also averagely distributed in every group. And then 4 pairs are randomly paired to conduct Self-Prediction. We train all models on S3DIS for 100 epochs with SGD optimizer and batch size 8. The base learning rate is set to 0.01 and divided by 2 every 20 epochs. For instance head, we set δ_v to 0.5 and δ_d to 1.5 following the same setting as [32] and [18]. The loss weight coefficient β for $\mathcal{L}_{sp}$ is set to 0.8. BlockMerging algorithm [32, 18] is used to merge instances from different blocks during inference, and bandwidth is set to 0.8 for mean-shift clustering.

For experiments on ShapeNet, input point cloud is represented only by coordinates. In Self-Prediction head, input point cloud is divided into 4 groups. We

train all models for 200 epochs with Adam optimizer and batch size 16. The base learning rate is set to 0.001 and divided by 2 every 20 epochs. Other settings are the same as experiments conducted on S3DIS.

4.2   Segmentation Results on S3DIS

We report instance and semantic segmentation results in Table 1 and Table 2 respectively, where results of Area 5 and 6-fold cross validation are all shown. Baseline results in tables denote that we train our backbone network with only instance-head and semantic-head. All baseline results for PointNet and Point-Net++ in the table are got from vanilla results of [32], which are almost the same as ours. In all tables, InsSem-SP denotes complete version of our method, i.e., performing instance and semantic Self-Prediction jointly. To prove effectiveness of our proposed Self-Prediction scheme and our associated framework more clearly, we report the results of Ins-SP and Sem-SP in Table 1 and Table 2 respectively. Ins-SP means that we only perform instance Self-Prediction by taking $\mathbf{F}_{ins}$ and $\mathbf{Y}_{ins}$ as input. Sem-SP means that we only perform semantic Self-Prediction by taking $\mathbf{F}_{sem}$ and $\mathbf{Y}_{sem}$ as input.

Backbone	Method	mPrec	mRec	mCov	mWCov
	Rseults on Area 5				
PN	Baseline [32]	42.3	34.9	38.0	40.6
	ASIS [32]	44.5	37.4	40.4	43.3
	Ours (Ins-SP)	48.2	39.9	44.7	47.6
	Ours (InsSem-SP)	**51.1**	**43.6**	**49.2**	**51.8**
PN++	Baseline [32]	53.4	40.6	42.6	45.7
	ASIS [32]	55.3	42.4	44.6	47.8
	Ours (Ins-SP)	58.9	46.3	52.8	54.9
	Ours (InsSem-SP)	**60.1**	**47.2**	**54.1**	**56.3**
	Results 6-fold CV				
PN	Baseline [32]	50.6	39.2	43.0	46.3
	ASIS [32]	53.2	40.7	44.7	48.2
	Ours (Ins-SP)	55.1	44.3	48.9	50.1
	Ours (InsSem-SP)	**56.6**	**45.9**	**51.8**	**52.2**
PN++	Baseline [32]	62.7	45.8	49.6	53.4
	ASIS [32]	63.6	47.5	51.2	55.1
	Ours (Ins-SP)	65.9	53.2	58.0	60.7
	Ours (InsSem-SP)	**67.5**	**54.6**	**60.4**	**63.0**

Table 1. Instance segmentation results on S3DIS dataset.

From Table 1 and Table 2, we can observe that our method improves the baseline based on all three backbone networks on both instance and semantic segmentation tasks significantly. For example, our method improve baseline by 8.3 mPrec, 8.7 mRec, 11.2 mCov, 11.2 mWCov in instance segmentation and 7.7 mIoU, 9.5 mAcc, 3.9 oAcc in semantic segmentation on Area 5 when we use PointNet as backbone. Effectiveness of proposed Self-Prediction scheme is

fully proved by comparing the results of Ins-SP with baseline in Table 1 and the results of Sem-SP with baseline in Table 2. Moreover, performance is further improved when we conduct instance and semantic Self-Prediction jointly. In Figure 2, we show some visualization results of baseline and Ours (InsSem-SP) based on PointNet++. We observe that our method achieves obvious more accurate predictions and performs better instance/semantic class boundaries.

Backbone	Method	mIoU	mAcc	oAcc	mIoU	mAcc	oAcc
		Area 5			6-fold CV		
PN	Baseline [32]	44.7	52.9	83.7	49.5	60.7	80.4
	ASIS [32]	46.4	55.7	84.5	51.1	62.3	81.7
	Ours (Sem-SP)	48.0	58.6	85.5	52.3	64.5	83.0
	Ours (InsSem-SP)	**52.4**	**62.4**	**87.6**	**54.8**	**67.4**	**84.8**
PN++	Baseline [32]	50.8	58.3	86.7	58.2	69.0	85.9
	ASIS [32]	53.4	60.9	86.9	59.3	70.1	86.2
	Ours (Sem-SP)	55.9	63.6	87.3	61.1	72.2	87.3
	Ours (InsSem-SP)	**58.8**	**65.9**	**89.2**	**64.1**	**74.3**	**88.5**

Table 2. Semantic segmentation results on S3DIS dataset.

Based on the baseline, ASIS associates instance and semantic segmentation, and designs a module to make these two tasks cooperate with each other. Obvious improvements are achieved by ASIS compared with baseline, while our method performes significantly better. Another advantage of our method is that our proposed Self-Prediction head is formulated as a loss function and will be taken off during inference, hence no extra computation burden and space usage are introduced compared with baseline.
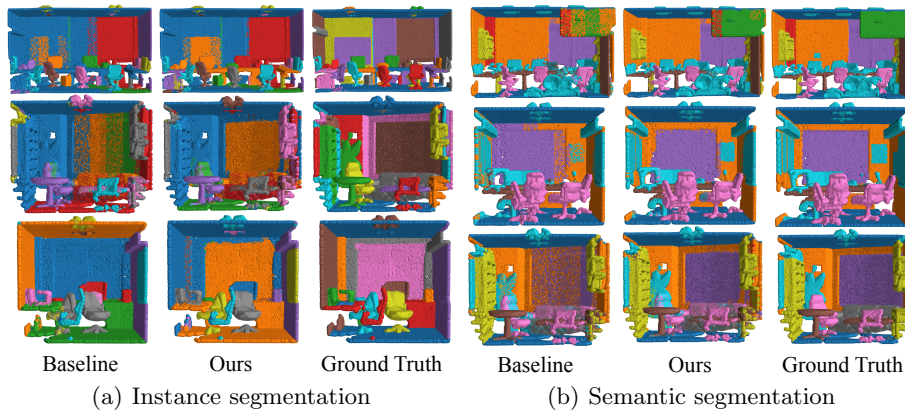

(a) Instance segmentation   (b) Semantic segmentation

Fig. 2. Visualization results of instance and semantic segmentation. Our method obviously performs better than baseline. Best viewed in color.

Compare with state-of-the-arts We also compare our method with other state-of-the-art methods. Instance segmentation results are shown in Table 3,

from which we see that our method achieves state-of-the-art performance. To the best of our knowledge, 3D-BoNet [38] is the best published method for instance segmentation in 3D point cloud. Obviously better performance compared with 3D-BoNet is achieved by our method, especially for mean recall. JSNet [44] achieves excellent performance by designing a feature fusion module based on PointConv [34]. Compared with JSNet, our method (PN++) performs better especially for mCov and mWCov. For semantic segmentation, results are shown in Table 4. Our method achieves comparable results compared with state-of-the-art methods when we only use PointNet++ as backbone. Even better performance on Area 5 is achieved compared with PointCNN, which is an excellent point cloud learning architecture. Moreover, our method is general and can use the most advanced architectures as the backbone to achieve superior performance.

Method	mPrec	mRec	mCov	mWCov
Results on Area 5				
SGPN (PN) [31]	36.0	28.7	32.7	35.5
3D-BoNet [38]	57.5	40.2	-	-
Ours (PN++)	**60.1**	**47.2**	**54.1**	**56.3**
Results on 6-fold CV				
SGPN (PN) [31]	38.2	31.2	37.9	40.8
PartNet [16]	56.4	43.4	-	-
3D-BoNet [38]	65.6	47.6	-	-
JSNet [44]	66.9	53.9	54.1	58.0
Ours (PN++)	**67.5**	**54.6**	**60.4**	**63.0**

Table 3. Instance segmentation results of state-of-the-art methods on S3DIS dataset.

Method	mIoU	mAcc	oAcc	mIoU	mAcc	oAcc
	Area 5			6-fold CV		
RSNet [6]	-	-	-	56.5	66.5	-
JSNet [44]	54.5	61.4	87.7	61.7	71.7	**88.7**
SPGraph [9]	58.0	66.5	86.5	62.1	73.0	85.5
PointCNN [10]	57.3	63.9	85.9	65.4	75.6	88.1
PCCN [30]	58.3	**67.0**	-	-	-	-
PointWeb [43]	60.3	66.6	87.0	**66.7**	**76.2**	87.3
GACNet [29]	**62.9**	-	87.8	-	-	-
Ours (PN++)	58.8	65.9	**89.2**	64.1	74.3	88.5

Table 4. Semantic segmentation results of state-of-the-art methods on S3DIS dataset.

4.3   Segmentation Results on ShapeNet

We provide qualitative results of part instance segmentation in Figure 3 following [31] and [32]. As shown in Figure 3, our method successfully segments instances of the same part, such as different legs of the chair. Semantic segmentation results are shown in Table 5, from which we observe that our method

achieves significant improvements over baselines. And more improvements compared with ASIS are achieved by our methods. In addition to PointNet and PointNet++, we add a stronger network DGCNN [33] in this dataset as our backbone. Obvious performance improvement over baseline also can be observed based on this backbone.
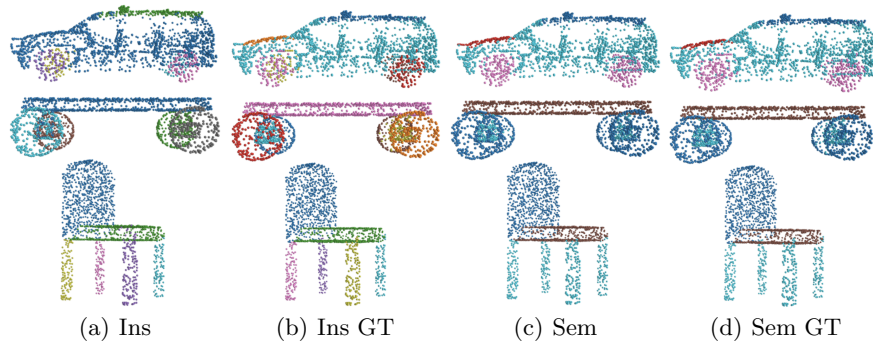

| (a) Ins | (b) Ins GT | (c) Sem | (d) Sem GT |

Fig. 3. Visualization results of our method on ShapeNet. (a) Instance segmentation results. (b) Instance segmentation ground truth. (c) Semantic segmentation results. (d) Semantic segmentation ground truth.

Method	pIoU	mpIoU
PointNet (*RePr*)	83.3	79.7
PointNet++ (*RePr*)	84.5	80.5
DGCNN (*RePr*)	85.2	82.3
ASIS (PN)	84.0	-
ASIS (PN++)	85.0	-
Ours (InsSem-SP, PN)	84.5	81.5
Ours (InsSem-SP, PN++)	85.8	82.6
Ours (InsSem-SP, DGCNN)	**86.2**	**83.1**

Table 5. Semantic segmentation results on ShapeNet dataset. *RePr* denotes our reproduced results. All models in the table are trained without normal information.

4.4   Ablation Study

In this section, we analyze some important components and hyper-parameters of our methods. All experiments in this section are performed on S3DIS Area 5 using PointNet as backbone.

Component Analyses. As shown in Section 4.2, the effectiveness of our proposed Self-Prediction scheme and joint learning framework has been proved. We further discuss how much our method benefits from bidirectional Self-prediction and class-averaged group dividing way. To this end, two corresponding experiments are conducted: 1) we only perform unidirectional Self-Prediction, and the direction is randomly selected among the two directions, 2) we randomly divide point cloud into groups rather than dividing according to instance labels in Self-Prediction head. Experimental results are reported in Table 6, where mPrec, mRec for instance segmentation and mIoU, mAcc for semantic segmentation are

shown. We can draw a conclusion that bidirectional Self-Prediction bring visible improvements compared with unidirectional Self-Prediction and randomly grouping will slightly degrade the performance.

Method	mPrec	mRec	mIoU	mAcc
Unidirectional	49.9	40.7	51.0	60.8
Randomly Dividing	50.5	42.1	51.6	61.3
Ours (InsSem-SP)	**51.1**	**43.6**	**52.4**	**62.4**

Table 6. Component analyses. Results on S3DIS Area 5 are shown.

Parameter Analyses. Three important parameters introduced by our method are analyzed in this section. The first is β used to balance the weight of $\mathcal{L}_{sp}$. The analysis results are shown in Figure 4(a), from which we can see that our method is not sensitive to this parameter and works very well in a wide range (0.4-1.4). The second parameter is the number of divided groups G to make a trade off between performance and training speed. We show the results in Figure 4(b), from which we see that the performance is relatively stable and not sensitive to G in a reasonable range. The last parameter is α used to control propagation portion in the process of label propagation. Although we follow the common setting [12] ($\alpha = 0.99$) in all our experiments, we still conduct experiments to analyze the sensitivity to this parameter of our method. As shown in Figure 4(c), our method outperforms baseline in a large range, i.e., $\alpha > 0.5$.
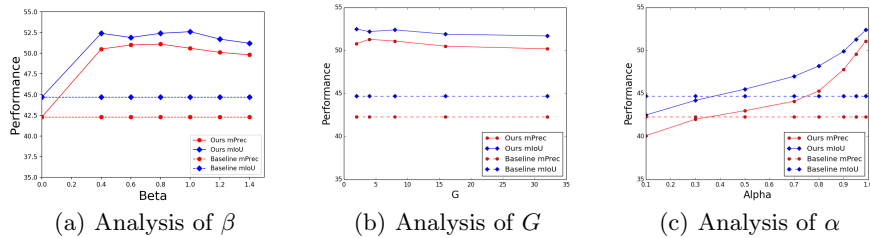

(a) Analysis of β (b) Analysis of G (c) Analysis of α

Fig. 4. Results of parameter analyses. mPrec for instance segmentation and mIoU for semantic segmentation are shown in figure. Dotted lines represent results of baseline.

5 Conclusion

In this paper, we present a novel learning scheme named Self-Prediction to enforce relation exploring, and a joint framework for associating instance and semantic segmentation of point clouds. Extensive experiments prove that our method can be combined with popular networks significantly improve their performance. By only taking PointNet++ as the backbone, our method achieves state-of-the-art or comparable results. Moreover, our method is a general learning framework and easy to apply to most existing learning networks.

References

1. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: CVPR (2019)
2. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Mach. Intell. (2002)
3. Graham, B., Engelcke, M., van der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. In: CVPR (2018)
4. Hou, J., Dai, A., Nießner, M.: 3d-sis: 3d semantic instance segmentation of RGB-D scans. In: CVPR (2019)
5. Hua, B., Tran, M., Yeung, S.: Pointwise convolutional neural networks. In: CVPR (2018)
6. Huang, Q., Wang, W., Neumann, U.: Recurrent slice networks for 3d segmentation of point clouds. In: CVPR (2018)
7. Jiang, L., Zhao, H., Liu, S., Shen, X., Fu, C.W., Jia, J.: Hierarchical point-edge interaction network for point cloud semantic segmentation. In: ICCV (2019)
8. Lan, S., Yu, R., Yu, G., Davis, L.S.: Modeling local geometric structure of 3d point clouds using geo-cnn. In: CVPR (2019)
9. Landrieu, L., Simonovsky, M.: Large-scale point cloud semantic segmentation with superpoint graphs. In: CVPR (2018)
10. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. In: NeurIPS (2018)
11. Liu, S., Jia, J., Fidler, S., Urtasun, R.: SGN: sequential grouping networks for instance segmentation. In: ICCV (2017)
12. Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S.J., Yang, Y.: Learning to propagate labels: Transductive propagation network for few-shot learning. In: ICLR (2019)
13. Liu, Y., Fan, B., Meng, G., Lu, J., Xiang, S., Pan, C.: Densepoint: Learning densely contextual representation for efficient point cloud processing. In: ICCV (2019)
14. Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: CVPR (2019)
15. Mao, J., Wang, X., Li, H.: Iccv (2019)
16. Mo, K., Zhu, S., Chang, A.X., Yi, L., Tripathi, S., Guibas, L.J., Su, H.: Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In: CVPR (2019)
17. Pham, Q., Hua, B., Nguyen, D.T., Yeung, S.: Real-time progressive 3d semantic segmentation for indoor scenes. In: WACV (2019)
18. Pham, Q., Nguyen, D.T., Hua, B., Roig, G., Yeung, S.: JSIS3D: joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In: CVPR (2019)
19. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from RGB-D data. In: CVPR (2018)
20. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017)
21. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: NIPS (2017)
22. Ren, M., Zemel, R.S.: End-to-end instance segmentation with recurrent attention. In: CVPR (2017)
23. Shen, Y., Feng, C., Yang, Y., Tian, D.: Mining point cloud local structures by kernel correlation and graph pooling. In: CVPR (2018)

24. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M., Kautz, J.: Splatnet: Sparse lattice networks for point cloud processing. In: CVPR (2018)
25. Tchapmi, L.P., Choy, C.B., Armeni, I., Gwak, J., Savarese, S.: Segcloud: Semantic segmentation of 3d point clouds. In: 3DV (2017)
26. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: Kpconv: Flexible and deformable convolution for point clouds. In: ICCV (2019)
27. Wang, C., Samari, B., Siddiqi, K.: Local spectral graph convolution for point set feature learning. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV (2018)
28. Wang, F., Zhang, C.: Label propagation through linear neighborhoods. IEEE Transactions on Knowledge and Data Engineering (2007)
29. Wang, L., Huang, Y., Hou, Y., Zhang, S., Shan, J.: Graph attention convolution for point cloud semantic segmentation. In: CVPR (2019)
30. Wang, S., Suo, S., Ma, W., Pokrovsky, A., Urtasun, R.: Deep parametric continuous convolutional neural networks. In: CVPR (2018)
31. Wang, W., Yu, R., Huang, Q., Neumann, U.: SGPN: similarity group proposal network for 3d point cloud instance segmentation. In: CVPR (2018)
32. Wang, X., Liu, S., Shen, X., Shen, C., Jia, J.: Associatively segmenting instances and semantics in point clouds. In: CVPR (2019)
33. Wang, Y., Sun, Y., Liu, Z., Sarma, S., Bronstein, M., Solomon, J.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics **38** (2018)
34. Wu, W., Qi, Z., Li, F.: Pointconv: Deep convolutional networks on 3d point clouds. In: CVPR (2019)
35. Xiaojin, Z., Zoubin, G.: Learning from labeled and unlabeled data with label propagation. Tech. Rep., Technical Report CMU-CALD-02–107, Carnegie Mellon University (2002)
36. Xie, S., Liu, S., Chen, Z., Tu, Z.: Attentional shapecontextnet for point cloud recognition. In: CVPR (2018)
37. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y.: Spidercnn: Deep learning on point sets with parameterized convolutional filters. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV (2018)
38. Yang, B., Wang, J., Clark, R., Hu, Q., Wang, S., Markham, A., Trigoni, N.: Learning object bounding boxes for 3d instance segmentation on point clouds. In: NeurIPS (2019)
39. Yang, J., Zhang, Q., Ni, B., Li, L., Liu, J., Zhou, M., Tian, Q.: Modeling point clouds with self-attention and gumbel subset sampling. In: CVPR (2019)
40. Ye, X., Li, J., Huang, H., Du, L., Zhang, X.: 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In: ECCV (2018)
41. Yi, L., Zhao, W., Wang, H., Sung, M., Guibas, L.J.: GSPN: generative shape proposal network for 3d instance segmentation in point cloud. In: CVPR (2019)
42. Zhang, Z., Hua, B.S., Yeung, S.K.: Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In: ICCV (2019)
43. Zhao, H., Jiang, L., Fu, C., Jia, J.: Pointweb: Enhancing local neighborhood features for point cloud processing. In: CVPR (2019)
44. Zhao, L., Tao, W.: Jsnet: Joint instance and semantic segmentation of 3d point clouds. In: AAAI (2020)
45. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NeurIPS (2004)
46. Zhuo, W., Salzmann, M., He, X., Liu, M.: Indoor scene parsing with instance segmentation, semantic labeling and support relationship inference. In: CVPR (2017)