PUPS: Point Cloud Unified Panoptic Segmentation

Shihao Su [*1], **Jianyun Xu** [*2], **Huanyu Wang** [1],
Zhenwei Miao [2], **Xin Zhan** [2], **Dayang Hao** [2], **Xi Li** [1, 3, 4†]

[1]College of Computer Science and Technology, Zhejiang University
[2]Alibaba Group
[3]Shanghai Institute for Advanced Study, Zhejiang University
[4]Shanghai AI Laboratory
shihaocs@zju.edu.cn, xujianyun.xjy@alibaba-inc.com, huanyuhello@zju.edu.cn,
{zhenwei.mzw, zhanxin.zx}@alibaba-inc.com, haodayang@gmail.com, xilizju@zju.edu.cn

Abstract

Point cloud panoptic segmentation is a challenging task that seeks a holistic solution for both semantic and instance segmentation to predict groupings of coherent points. Previous approaches treat semantic and instance segmentation as surrogate tasks, and they either use clustering methods or bounding boxes to gather instance groupings with costly computation and hand-crafted designs in the instance segmentation task. In this paper, we propose a simple but effective **p**oint cloud **u**nified **p**anoptic **s**egmentation (PUPS) framework, which use a set of point-level classifiers to directly predict semantic and instance groupings in an end-to-end manner. To realize PUPS, we introduce bipartite matching to our training pipeline so that our classifiers are able to exclusively predict groupings of instances, getting rid of hand-crafted designs, e.g. anchors and Non-Maximum Suppression (NMS). In order to achieve better grouping results, we utilize a transformer decoder to iteratively refine the point classifiers and develop a context-aware CutMix augmentation to overcome the class imbalance problem. As a result, PUPS achieves **1st** place on the leader board of SemanticKITTI panoptic segmentation task and state-of-the-art results on nuScenes.

1 Introduction

As one of the most challenging problems in computer vision, panoptic segmentation (Kirillov et al. 2019b) seeks a holistic solution to both semantic segmentation and instance segmentation. Shortly after the researchers proposed the question in image data, two LiDAR datasets (Behley, Milioto, and Stachniss 2020; Fong et al. 2021) for autonomous driving extend the research area to point cloud data. These emerging challenges aim at assigning points with groupings of countable *thing* instances and uncountable *stuff* classes, revealing that perception system of autonomous vehicles demands understanding of the environment in terms of both semantic level and instance level through point cloud sensors.

To solve point cloud panoptic segmentation, previous efforts can be divided into two streams: proposal-based methods and proposal-free methods. As the name indicates,

---

[*]These authors contributed equally.
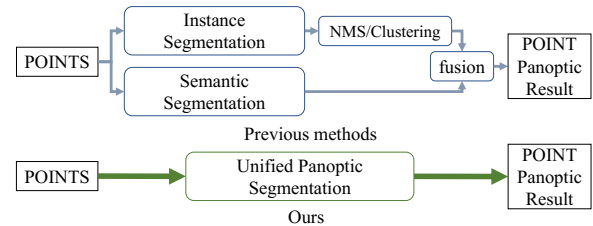[†]Corresponding author

Figure 1: Illustration of previous framework and PUPS.

proposal-based methods rely on proposals generated by an object detection head to get instance segmentation and employ an extra semantic branch for semantic segmentation. Besides their cascaded structure, this stream of methods involves lots of hand-crafted components such as proposals and non-maximum suppression (NMS). As for proposal-free methods, they introduce clustering-based methods in their instance branch based on the predicted offsets to instance centers. Similarly, an extra semantic branch is attached for semantic segmentation. Although outstanding results have been achieved by these methods on different benchmarks, there are two main drawbacks as shown in the upper part of Figure 1: 1) they treat semantic and instance segmentation as surrogate tasks, which does not truly solve panoptic segmentation holistically; 2) their instance branch involves many hand-crafted components and post-processing, which is complicated and time-consuming.

Inspired by recent developments in image segmentation (Cheng, Schwing, and Kirillov 2021; Wang et al. 2021; Li et al. 2022c; Zhang et al. 2021), we propose PUPS, a simple but effective **p**oint cloud **u**nified **p**anoptic **s**egmentation framework to solve the challenges above. In essence, the aim of point cloud panoptic segmentation is to predict groupings of coherent points. PUPS unifies point cloud instance and semantic segmentation as a classifier-assigning problem. More specifically, PUPS allocates a set of point-level classifiers, learning to assign them to exclusive instances or semantic classes. By utilizing bipartite matching in the training phase, we achieve PUPS in an end-to-end manner and it is able to predict exclusive groupings with no hand-crafted design or post-processing as shown in Figure 1.

In addition to predicting exclusive groupings, we adopt two designs to produce better grouping results. First, we utilize a transformer decoder to refine the classifiers. In each stages of refinement, our point-level classifiers query point features from backbone and generate refined classifiers. Afterwards, the classifiers integrate the feature of corresponding instances and semantics, enhancing their ability to distinguish between groupings. Moreover, we employ a classifier self-attention to incorporate global relations into the classifiers. After the classifiers are refined, new point groupings are produced and can be further used to refine the classifiers in the next stage together with the refined classifiers. Second, in order to alleviate class imbalance and train the classifiers more sufficiently, we design a context-aware CutMix (Yan, Mao, and Li 2018; Xu et al. 2021; Li et al. 2022b) augmentation. We cut instances from training scans and mix them with the instances in the current scan based on their background to avoid damaging their context so that performance is improved in return.

To evaluate the effectiveness of our proposals, we conduct extensive experiments on two point cloud panoptic segmentation datasets. Our method ranks 1st on the leader board of SemanticKITTI (Behley, Milioto, and Stachniss 2020) and achieves state-of-the-art results on nuScenes (Caesar et al. 2020).

To sum up, the contributions of this paper are listed below:

- To the best of our knowledge, PUPS is the first simple but effective **p**oint cloud **u**nified **p**anoptic **s**egmentation framework, using a set of point-level classifiers to directly predict semantic and instance groupings.

- To get rid of post-processing and hand-crafted designs, we introduce bipartite matching in our training so that the classifiers are able to exclusively predict groupings.

- We utilize a transformer decoder to iteratively refine the classifiers with point features to produce more accurate groupings of points.

- To encounter class imbalance, we adopt a context-aware CutMix strategy to enhance the performance of segmentation by preserving the context of instances.

- We achieve rank 1 performance on the leader board of SemanticKITTI panoptic segmentation task and SOTA results on nuScenes.

2   Related Work

Panoptic Segmentation aims to divide an input sample into countable *thing* instances or uncountable *stuff* classes. The output of panoptic segmentation is to assign element-wise label with both instance ID and semantic class. For input modal of LiDAR point cloud or image, panoptic segmentation models follow two typical frameworks: proposal-based and proposal-free.

2.1   LiDAR Point Cloud Panoptic Segmentation

Proposal-based methods   This kind of methods are usually formulated in a two-stage manner: segmentation after detection (Milioto et al. 2020; Hurtado, Mohan, and Valada 2020). Moreover, SemanticKITTI (Behley, Milioto, and Stachniss 2020) and nuScenes (Caesar et al. 2020) report results by joining state-of-the-art point cloud object detection methods and point cloud semantic segmentation methods. Taking in range-view images, EfficientLPS (Sirohi et al. 2021) utilizes a instance branch to predict classes, bounding boxes and masks for *thing* classes and fuse semantic feature to predict *stuff* classes. After post-processing, the range-view result is projected back to point-wise result. It is worth noting that hand-crafted components such as anchors or NMS are often involved in these methods.

Proposal-free methods   As for proposal-free ones, they usually predict instance centers and point-wise offset to centers to output panoptic segmentation result (Zhou, Zhang, and Foroosh 2021; Hong et al. 2021). Recently, Panoptic-PHNet (Li et al. 2022b) introduces a K-NN transformer to predict more accurate offsets. Additionally, proposal-free methods (Gasperini et al. 2021; Hong et al. 2021; Li et al. 2022b) often involve clustering algorithms to cluster points to instances. GP-S3Net (Razani et al. 2021) propose a novel graph-based clustering method to effectively predict instances from over-segmented clusters.

PUPS directly groups point cloud without any bounding box proposals, hand-crafted post-processing or clustering algorithms. It is worth noting that these hand-crafted components in both proposal-based/free methods require lots of computation and careful tuning.

2.2   Image Panoptic Segmentation

Proposal-based methods   This stream of methods follow the pipeline that bounding boxes are first obtained and masks of each bounding box are predicted afterwards, such as Panoptic-FPN (Kirillov et al. 2019a). These methods fuse their masks of *thing* classes and masks of *stuff* classes with merging modules (Liu et al. 2019; Li et al. 2019; Porzi et al. 2019a).

Proposal-free methods   Proposal-free methods solves panoptic segmentation by employing two separate branches to predict semantic masks and group pixels to instances. One of the most popular grouping methods is instance center regression, which predicts pixel-level offsets to instance centers (Neven et al. 2019; Cheng et al. 2020). Recently, following DETR (Carion et al. 2020), multiple works introduce bipartite matching in their training (Zhang et al. 2021; Wang et al. 2021; Cheng, Schwing, and Kirillov 2021; Li et al. 2022a), simplifying the process of panoptic segmentation.

Inspire by the methods that introduce bipartite matching into their training pipeline, we propose PUPS, the first framework on point cloud data which is able to exclusive predict panoptic groupings of points and can be trained in an end-to-end manner.

3   Method

In this section, we first state the definition of point cloud panoptic segmentation in Section 3.1. Then, we present the network architecture of PUPS in Section 3.2, along with its two core components, i.e., bipartite matching (Section 3.3) and classifier refinement (Section 3.4). Lastly, we show a context-aware CutMix for instances in Section 3.5.
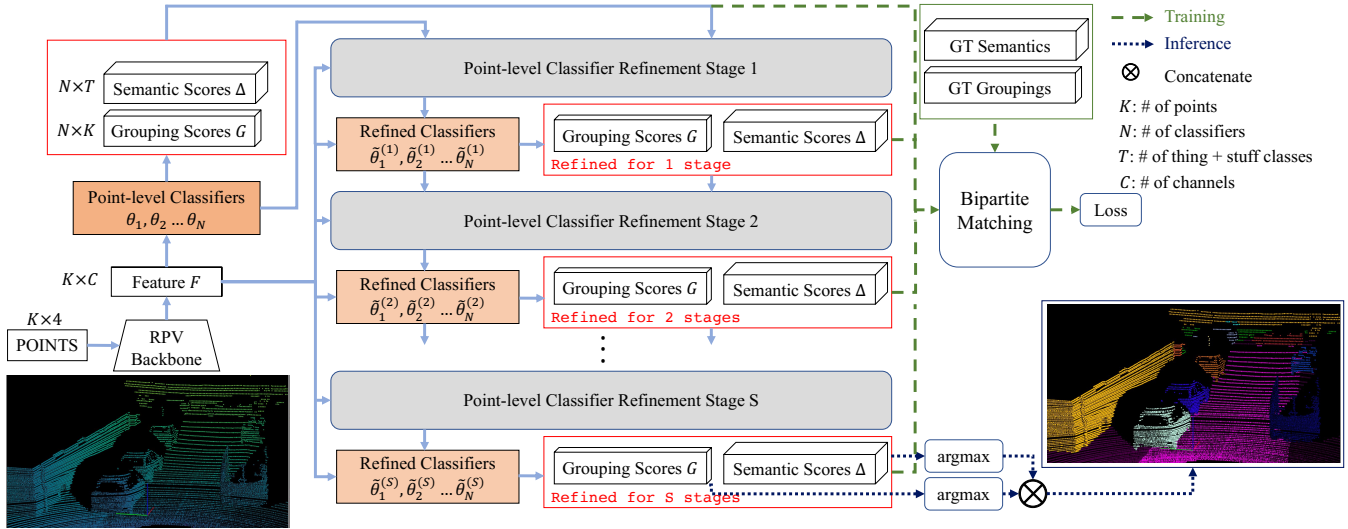
Figure 2: Pipeline of PUPS. Point features are first encoded by a RPV backbone (Xu et al. 2021) and fed into the unrefined classifiers to get initial groupings and semantics. Then, point features activated by corresponding groupings are integrated into the classifiers and a self-attention is applied to the classifiers to produce refined classifiers. For simplicity, we omit the superscripts of classifiers in Section 3.4. With the refined classifiers, more accurate groupings and semantics are obtained. To clarify, groupings and semantics of all stages will be supervised by ground-truth with bipartite matching in training and only the groupings and semantics of the last stage will be used to output segmentation results in inference.

3.1 Problem Formulation

Point cloud panoptic segmentation aims at grouping a point cloud $P \in \mathbb{R}^{K \times 4}$ of K points into a set of *thing* instances and *stuff* classes, among which *thing* instances refer to countable objects (e.g. person, car, bicycle) and *stuff* classes refers to uncountable backgrounds (e.g. road, terrain, vegetation). As shown in Equation 1, ground-truth groupings in a point cloud are defined as:

$$\{y_i\}_{i=1}^M = \{(g_i, c_i)\}_{i=1}^M, \qquad (1)$$

where $g_i \in \{0,1\}^K$ is a ground truth binary mask indicating which points belong to group i, c_i is the semantic class of group i, and M is the number of ground truth groupings in the point cloud. Note that the groupings are mutually exclusive, i.e. , each point in a point cloud belongs to either an instance of *things* or a background *stuff*. In this way, each point is assigned to a group ID and a semantic class.

3.2 Point Cloud Unified Panoptic Segmentation

Given the definition in Section 3.1, we allocate N learnable point-level classifiers to predict the groupings for both distinct *thing* instances and background *stuff* in a unified manner. We denote the learnable parameters of the classifiers as $\theta = \{\theta_i \mid \theta_i \in \mathbb{R}^C\}_{i=1}^N$, where C is the number of channels. As shown in the inference pipeline of Figure 2, K points are fed into a backbone to extract point-wise feature F. Using the feature F and the parameters θ, PUPS generate two vital scores for panoptic segmentation results: grouping scores $G = \{\hat{g}_i \mid \hat{g}_i \in [0,1]^K\}_{i=1}^N$ and semantic scores $\Delta = \{\Delta_i \mid \Delta_i \in \mathbb{R}^T\}_{i=1}^N$, where T is the number of *thing* and *stuff* classes.

First of all, grouping scores G indicate the probability of the K points belonging to N groups, which are used to decide the group ID for each point. Similarly, semantic scores Δ indicates the probability of the N groupings belonging to T semantic classes, which are used to assign semantic classes to groupings, and further decide semantic classes for each point.

Specifically, with the parameters θ of classifiers and the feature F of points, we utilize a simple matrix multiplication and a sigmoid, denoted as $\delta(\cdot, \cdot)$, to obtain grouping scores G for each point with respect to each classifier:

$$\hat{g}_i = \delta(\theta_i, F), i = 1, \ldots, N, \qquad (2)$$

where $F \in \mathbb{R}^{K \times C}$ is the point-level features of K points.

Similarly, we utilize $\delta(\cdot, \cdot)$ to predict a semantic score $\Delta_i \in \mathbb{R}^T$ for each point-level classifier, where T is the number of *thing* and *stuff* classes:

$$\Delta_i = \delta(\psi, \theta_i), i = 1, \ldots, N. \qquad (3)$$

To clarify, $\psi \in \mathbb{R}^C$ stands for a set of learnable parameters.

To output the result of panoptic segmentation, PUPS assigns semantic class c_i to $\hat{g}_i$ and groupings to points by:

$$\hat{c}_i = \arg\max \Delta_i, \qquad (4)$$

$$\hat{z}_{i,k} = \begin{cases} 1 & \text{if} \quad \hat{g}_{i,k} = \max_i (\hat{g}_{i,k}) \\ 0 & \text{otherwise} \end{cases}, \qquad (5)$$

$$\{\hat{y}_i\}_{i=1}^N = \{(\hat{z}_i, \hat{c}_i)\}_{i=1}^N, \qquad (6)$$

Our prediction is organized similarly to Equation 1 and group ID and semantic class for each point is obtained.

Now, PUPS is able to predict the groupings directly. Instead of adding post-processing at test time, we use bipartite

matching (Section 3.3) in our training pipeline to prevent multiple classifiers from predicting the same instance.

3.3 Bipartite Matching

One solution to the aforementioned problem is to make our point-level classifiers learn from exclusive ground truth groupings. Therefore, a one-to-one mapping from the M ground-truth groupings to the N classifiers is needed. Inspired by recent application of bipartite matching in object detection (Carion et al. 2020) and image segmentation (Cheng, Schwing, and Kirillov 2021; Li et al. 2022c; Zhang et al. 2021), bipartite matching is able to assign one ground-truth to only one prediction according to a cost matrix. This one-to-one rule plays a vital role in exclusively predicting the point cloud panoptic segmentation results for the reason that there are no classifier assigned to learn the same *thing* instance or *stuff* class, reducing the possibility of duplicate predictions. It also prevents the classifiers from only focusing on easy groupings because all ground truths are mapped, reducing the bias of the model. Since instance ID prediction is not required for *stuff* classes, several classifiers are constantly mapped to the ground truth of *stuff* classes.

Cost Computation   To match the predictions $\{\hat{s}_i\}_{i=1}^N = \{(\hat{g}_i, \Delta_i)\}_{i=1}^N$ and ground-truths $\{y_i\}_{i=1}^M \{(g_i, c_i)\}_{i=1}^M$, we compute the cost matrix based on their pairwise accordance in terms of both points and groups. For simplicity, we term the match cost and the training loss between prediction and ground-truth with the same notation:

$$\mathcal{L}_{\text{match}} = \alpha\mathcal{L}_{\text{dice}} + \beta\mathcal{L}_{\text{focal}} + \gamma\mathcal{L}_{\text{CE}}, \tag{7}$$

where dice loss $\mathcal{L}_{\text{dice}}$ (Milletari, Navab, and Ahmadi 2016) and cross entropy loss ($\mathcal{L}_{\text{CE}}$) are for point-wise accordance between $\hat{g}_i$ and g_j ($\forall 1 \leq i \leq N, 1 \leq j \leq M$). The focal loss (Lin et al. 2017) is for group classification between Δ_i and c_j ($\forall 1 \leq i \leq N, 1 \leq j \leq M$). For classifiers that are not assigned with any ground truth, they are masked as negative.

In summary, Section 3.2 resolves the problem of predicting the groupings and Section 3.3 enables the classifiers to learn from exclusive ground truth so that there is no need to apply post-processing to remove duplicate predictions or clustering algorithms to coalesce segmented groupings.

3.4 Classifier Refinement

Although the process is simple as stated in Section 3.2, it is challenging to accurately classify the unordered points into groups. Being different from classifying points into semantic classes, panoptic segmentation demands a further step to discriminate instance information within one semantic class. Moreover, the predicting process is purely point-based, i.e., points are treated as isolated only with implicit spatial information, and appearance encoded in their backbone features. Thus, classifying the points into groups only once may introduce noises from other groups

To fulfill the aforementioned demand and overcome the problem, we employ a transformer decoder with S stages to refine the point-level classifiers with point features as shown in Figure 2. In each stage of the transformer decoder, the process of refinement are divided into three parts: 1) classifier feature query. It gathers point features for each classifier. 2) classifier update. It updates the classifiers with the gathered features. 3) classifier self-attention. It further models the context information between classifiers.

Classifier Feature Query   First, we query the point features with grouping scores from Equation 2 using the parameters of the point-level classifiers fed into this stage. The grouping score serves as an attention map for the ith classifier with respect to every single point in the point cloud so that the most related features are collected as instance and semantic information. With the point features and their attention, the discriminative instance and semantic feature $F^{\theta_i} \in \mathbb{R}^C$ of the grouping i is obtained by:

$$F_{\theta_i} = \frac{1}{K}\sum_{k=1}^{K}\hat{g}_{i,k} \cdot F_k. \tag{8}$$

For simplicity, we omit the superscript used in Figrue 2 and $\theta = \{\theta_i\}_{i=1}^N$ stands for the parameters of the classifiers fed into this stage.

Classifier Update   Since the objective of classifiers is to learn the distinct groupings of points, we integrate the discriminative instance and semantic features into the parameters of the classifiers so that they are able to retrieve instance points that are missing in the current groups and rule out noisy ones, thus producing more accurate results. Specifically, we first project the features into the space of classifiers' parameter and employ a learnable momentum m to control the extent of integration. The calculation of the projection and momentum m is as followed:

$$m = 1 - \sigma(\varphi_1(F_{\theta_i})) \tag{9}$$
$$\tilde{\theta}_i = (1 - m) \cdot \varphi_2(F_{\theta_i}) + m \cdot \theta_i, \tag{10}$$

where σ is a non-linear function sigmoid and φ_1, φ_2 are linear transformations.

Classifier Self-attention   Lastly, besides the integration of classifier parameters and their corresponding local information gathered by the attention maps, we apply self-attention to incorporate global relation into the parameters of the classifiers. We utilize a multi-head self-attention (Vaswani et al. 2017) to model the relation between classifiers. The relation helps classifiers distinguish between each other and understand the context of the point cloud, reducing the probability that their groupings share a large overlap and enhancing the grouping accuracy.

Eventually, the point features and the refined classifiers are again feed into the next stage of the decoder. The refined grouping scores generated by Equation 2 in the next stage is able to gather more points from the instance and suppress noisy ones more accurately, producing better instance feature. The refined semantic scores produced by Equation 3 assign better semantic classes for the groupings.

3.5 Context-aware CutMix

In object detection and segmentation, class imbalance is a common issue, leading to performance degradation in the

Method	PQ	PQ†	SQ	RQ	PQ$^{\text{Th}}$	SQ$^{\text{Th}}$	RQ$^{\text{Th}}$	PQ$^{\text{St}}$	SQ$^{\text{St}}$	RQ$^{\text{St}}$
R.Net + P.P.	37.1	45.9	75.9	47.0	20.2	75.2	25.2	49.3	76.5	62.8
KPC + P.P.	44.5	52.5	80.0	54.4	32.7	81.5	38.7	53.1	79.0	65.9
Panoptic-PolarNet	54.1	60.7	81.4	65.0	53.3	87.2	60.6	54.8	77.2	68.1
DS-Net	55.9	62.5	82.3	66.7	55.1	87.2	62.8	56.5	78.7	69.5
EfficientLPS	57.4	63.2	83.0	68.7	53.1	87.8	60.5	*60.5*	79.5	*74.6*
GP-S3Net	60.0	*69.0*	82.0	72.1	65.0	86.6	**74.5**	56.4	78.7	70.4
Panoptic-PHNet	61.5	67.9	*84.8*	72.1	63.8	*90.7*	70.4	59.9	*80.5*	73.3
PUPS (ours)	*62.2*	65.8	84.2	*72.8*	*65.7*	90.6	72.7	59.6	79.5	73.1
PUPS § (ours)	**65.7**	**70.3**	**85.7**	**75.8**	**68.1**	**91.6**	*74.3*	**63.9**	**81.4**	**76.9**

Table 1: Comparison of LiDAR panoptic segmentation performance on SemanticKITTI test set, in which PQ is the primary metric for comparison. R.Net, P.P. and KPC refer to RangeNet++ (Milioto et al. 2019), Point Pillars (Lang et al. 2019) and KPConv (Thomas et al. 2019), respectively. § represents results of model ensemble and test-time augmentation (TTA). Bold refers to best result and italic refers to second best result. All scores are in [%]. Our method rank 1st on the leader borad of SemanticKITTI[1]

minor classes. A trivial solution to this issue is to cut objects out of training set to form a sample database. Before an input is fed into a network, the objects are sampled from the database and mix with the existing ones.

Context-aware Mixing We suggest mixing instances in accordance with their context in light of the aforementioned scenario. After an instance is sampled from the database, context-aware mixing translate the instance to the nearest contextual point. Contextual points are the points that most possibly exist underneath an instance, e.g., *car* instances are most possibly on top of *road* and *parking* but not *pole*. Panoptic segmentation methods are able to model the relation between instances and background since their objective is to distinguish among them. Thus, preserving the context is beneficial to the recognition of instances. As a result, the grouping results of the mixed classes are enhanced.

4 Experiment

To evaluate PUPS, we conduct experiments on two popular LiDAR point cloud datasets: SemanticKITTI(Behley, Milioto, and Stachniss 2020) and nuScenes(Fong et al. 2021).

4.1 Datasets and Evaluation Metric

SemanticKITTI proposes the first panoptic segmentation challenge on point cloud data. It contains 22 data sequences splited into 3 parts: 10 for training, 1 for validation and 11 for testing. There are 8 *thing* classes and 11 *stuff* classes.

nuScenes is a large-scale dataset for autonomous driving, which contains LiDAR data of 1000 scenes. The 1000 scenes are divided into 3 parts: 750 for training, 100 scenes for validation and 150 scenes for testing. There are 10 *thing* classes and 6 *stuff* classes.

Evaluation Metric Mean Panoptic Quality (PQ) (Kirillov et al. 2019b) is adopted as the primary evaluation metric for our experiment. As shown in Equation 11, PQ of a specific class can be decomposed into Segmentation Quality (SQ) and Recognition Quality (RQ):

$$\text{PQ}_c = \underbrace{\frac{\sum_{(p,g)\in\text{TP}_c} \text{IoU}(p,g)}{|\text{TP}_c|}}_{\text{segmentation quality (SQ)}} \times \underbrace{\frac{|\text{TP}_c|}{|\text{TP}_c| + \frac{1}{2}|\text{FP}_c| + \frac{1}{2}|\text{FN}_c|}}_{\text{recognition quality (RQ)}},$$

(11)

where TP_c is the set of matched predicted masks and ground truth masks of class c, FP_c is the set of unmatched predicted masks of class c, FN_c is set of unmatched ground truth masks of class c, and $\text{IoU}(p,g)$ is the intersection-over-union of predicted mask p and ground truth mask g. Mean PQ is the average of PQ of all classes and we additionally report PQ$^{\text{Th}}$, SQ$^{\text{Th}}$ and RQ$^{\text{Th}}$ of *thing* classes, PQ$^{\text{St}}$, SQ$^{\text{St}}$ and RQ$^{\text{St}}$ of *stuff* classes and PQ† (Porzi et al. 2019b), where PQ of *stuff* classes is replaced by their IoU in the calculation.

4.2 Implementation Details

Settings and Hyper-parameters Our implementation is based on MMDetection3D (MMDetection3DContributors 2020). Specifically, we train our models for 80 epochs with a batch size of 4. The learning rate is set to 0.002 initially and decrease with a factor of 0.1 after 50 epochs. We adopt AdamW (Loshchilov and Hutter 2017) with a weight decay of 0.05 as our optimizer. In addition to our proposed Cut-Mix augmentation, we apply random flipping along x- and y- axis, random rotation along z- axis and random scaling. Unless specified, the point feature dimension is set to 128 and the number of classifiers is set to 100. The number of refinement stages is 3. As for training, the losses are included in Equation 7 and the coefficients α, β, γ are set to 4, 1, 1 respectively.

Backbone We employ the backbone of RPVNet (Xu et al. 2021) in PUPS. It fuse points, voxels and range-view feature, and extract representative features. We follow the same backbone architecture as RPVNet but change the output feature dimension to 128. The voxel size is set to 5 cm and 10 cm for SemanticKITTI and nuScenes respectively.

---

[1]Our 1st place performance is assessed on Aug 12th, 2022 in *https://competitions.codalab.org/competitions/24025#results*.

Method	PQ	PQ†	SQ	RQ	PQ$^{\text{Th}}$	SQ$^{\text{Th}}$	RQ$^{\text{Th}}$	PQ$^{\text{St}}$	SQ$^{\text{St}}$	RQ$^{\text{St}}$
R.Net + P.P.	36.5	-	73.0	44.9	19.6	69.2	24.9	47.1	75.8	59.4
KPC + P.P.	41.1	-	74.3	50.3	28.9	69.8	33.1	50.1	*77.6*	62.8
DS-Net	57.7	63.4	77.6	68.0	61.8	78.2	68.8	54.8	77.1	67.3
Panoptic-PolarNet	59.1	64.1	78.3	70.2	65.7	87.4	74.7	54.3	71.6	66.9
EfficientLPS	59.2	65.1	75.0	69.8	58.0	78.0	68.2	**60.9**	72.8	71.0
Panoptic-PHNet	61.7	-	-	-	69.3	-	-	-	-	-
GP-S3Net	63.3	67.5	81.4	**75.9**	70.2	86.2	*80.1*	58.3	**77.9**	*71.9*
PUPS (ours)	*64.4*	*68.6*	*81.5*	74.1	*73.0*	*92.6*	79.3	58.1	73.5	70.4
PUPS § (ours)	**66.3**	**70.2**	**82.5**	*75.6*	**74.6**	**93.4**	**80.3**	*60.2*	74.5	**72.2**

Table 2: Comparison of LiDAR panoptic segmentation performance on SemanticKITTI validation set, in which PQ is the primary metric for comparison. R.Net, P.P. and KPC refer to RangeNet++ (Milioto et al. 2019), Point Pillars (Lang et al. 2019) and KPConv (Thomas et al. 2019), respectively. § represents results of model ensemble and test-time augmentation (TTA). Bold refers to best result and italic refers to second best result. All scores are in [%].

Method	PQ	PQ†	SQ	RQ	PQ$^{\text{Th}}$	SQ$^{\text{Th}}$	RQ$^{\text{Th}}$	PQ$^{\text{St}}$	SQ$^{\text{St}}$	RQ$^{\text{St}}$
PanopticTrackNet	51.4	56.2	80.2	63.3	45.8	81.4	55.9	60.4	78.3	75.5
DS-Net	55.9	62.5	82.3	66.7	55.1	87.2	62.8	56.5	78.7	69.5
GP-S3Net	61.0	67.5	84.1	72.0	56.0	85.3	65.2	66.0	82.9	78.7
EfficientLPS	62.0	65.6	83.4	73.9	56.8	83.2	68.0	70.6	83.8	83.6
Panoptic-PolarNet	63.4	67.2	83.9	75.3	59.2	84.1	70.3	70.4	83.6	83.5
Panoptic-PHNet	**74.7**	**77.7**	*88.2*	**84.2**	74.0	*89.0*	**82.5**	**75.9**	**86.8**	**86.9**
PUPS (ours)	**74.7**	*77.3*	**89.4**	*83.3*	**75.4**	**91.8**	*81.9*	*73.6*	*85.3*	*85.6*

Table 3: Comparison of LiDAR panoptic segmentation performance on nuScenes validation set. Bold refers to best result and italic refers to second best result. All scores are in [%].

# of Stages	PQ	SQ	RQ	$\mathcal{L}_{\text{train}}$
1	62.1	80.6	72.0	0.213
2	63.5	81.0	73.2	0.139
3	**64.4**	**81.5**	**74.1**	0.125
4	63.2	80.8	72.9	0.113
5	63.0	80.5	73.0	0.101

Table 4: Ablation study of number of refining stages on validation set of SemanticKITTI. $\mathcal{L}_{\text{train}}$ denotes training loss of models. All scores are in [%].

4.3 Main Results

Results on SemanticKITTI As shown in Table 1 and 2, we surpass all existing methods in PQ of both test set and validation set, and show significant advantages in the performance of *thing* classes. As for test set, we improve PQ of Panoptic-PHNet (Li et al. 2022b) from 61.5% to 62.2% and achieve a gain of 1.9% in PQ$^{\text{Th}}$. As for validation set, we outperform GP-S3Net (Razani et al. 2021) by a margin of 1.1% in PQ and 2.8% in PQ$^{\text{Th}}$. Compared with clustering-based methods DS-Net (Hong et al. 2021), and Panoptic-PolarNet (Zhou, Zhang, and Foroosh 2021) in addition to Panoptic-PHNet and GP-S3Net, our method achieve an increase of over 6% in PQ of test set. With respect to range-image-based method EfficientLPS (Sirohi et al. 2021), PUPS outperforms by 4.8% in PQ of test set.

The results of combined methods (row 1 and row 2) presented in the tables are obtained by training a detection head and semantic head as stated in the dataset. Moreover, following Panoptic-PHNet, we report results of model ensemble and test-time augmentation. Additionally, we provide class-wise performance of PUPS in supplementary material.

Results on nuScenes In this section, we compare the results of PUPS on nuScenes with results of previous methods. As listed in Table 3, our method achieves state-of-the-art results on validation set.

4.4 Ablation Study

Ablation on Network Components To verify the effectiveness of PUPS, we gradually apply our proposed components to a vanilla network. As shown in Table 5, M1 refers to a vanilla network with no refinement on the classifier or CutMix augmentation. M2 is trained with classifier refinement and M3 is trained with context-aware CutMix. The performance of M4 shows that both classifier refinement and context-aware CutMix contribute to the high performance.

Ablation on Number of Stages As shown in Table 4, trials with different number of stages reveals that PUPS with 3 stages achieves the best result. We observe that there exists over-fitting concerning the decrease of training loss as the number of stages increase. It suggests that models for larger datasets may benefit from more stages.

Model Variant	Context-aware CutMix	Classifier Refinement	PQ (%)	PQ† (%)	SQ (%)	RQ (%)	PQTh (%)	SQTh (%)	RQTh (%)	PQSt (%)	SQSt (%)	RQSt (%)
M1			44.5	49.3	68.0	54.4	40.5	74.6	47.0	47.4	63.2	59.7
M2		✓	50.9	55.1	74.2	60.5	43.6	76.5	49.5	56.2	72.5	56.2
M3	✓		55.9	60.7	75.1	66.2	64.4	90.5	71.8	49.7	63.9	62.1
M4	✓	✓	**64.4**	**68.6**	**81.5**	**74.1**	**73.0**	**92.6**	**79.3**	**58.1**	**73.5**	**70.4**

Table 5: Ablation study on proposed components of PUPS. The results are reported on the SemanticKITTI validation set.

Ablation on Number of Classifiers Table 6 contains result from different number of classifiers. It shows that 100 classifiers achieve the best result. On the one hand, insufficient number of classifiers is harmful to performance on both *thing* classes and *stuff* classes since bipartite assignment may assign the classifiers to inconsistent semantic. On the other hand, excessive number of classifiers benefit from consistency in assignment and perform better in *thing* classes. However, since the number of classifiers for background classes is fixed, excessive instance classifiers may lead to under-segmented background.

# of Classifiers	PQ	SQ	RQ	PQTh	PQSt
50	63.3	80.0	73.0	69.8	56.2
100	**64.4**	**81.5**	**74.1**	73.0	**58.1**
150	63.7	81.0	73.6	73.7	56.3
200	63.8	81.0	73.9	**73.9**	56.5

Table 6: Ablation study of number of classifiers on validation set of SemanticKITTI. All scores are in [%].

Ablation on CutMix Strategies In addition to our proposed context-aware CutMix, there is another CutMix strategy in point cloud object detection and segmentation: random CutMix (Yan, Mao, and Li 2018; Xu et al. 2021; Li et al. 2022b). They alleviate class imbalance by randomly mixing the sampled instances in the current scan. To validate the effectiveness of our context-aware CutMix, we compare performance by applying the strategies on M1 in Table 7. As shown in Table 7, our context-aware CutMix achieve a gain of 5.1% in PQ and outperform by a large margin on *thing* classes. It verifies our design on preserving context information of instances to enhance performance.

CutMix Type	PQ	SQ	RQ	PQTh	SQTh	RQTh
Random	50.8	72.8	61.7	57.5	86.4	66.4
Context-aware	**55.9**	**75.1**	**66.2**	**64.4**	**90.5**	**71.8**

Table 7: Ablation study of cutmix strategy on validation set of SemanticKITTI. All scores are in [%].

4.5 Analysis and Visualization

Spatial Distributions of Predictions As stated in Section 3.3 and 3.4, our classifiers are able to distinguish between instances, hence being capable of predicting panoptic segmentation results directly. Considering that the predictions are in 3D space, it is better to present an illustration more intuitively. Therefore, we plot the centers of the predictions in bird-eye view (BEV). As shown in Figure 3, each subplot stands for the spatial distribution of a classifier's predictions on *car* in a 100m × 100m square. The distributions follow certain patterns: 1) the arc-shaped patterns reveal accordance with the rotation of LiDAR sensors. 2) the positions where dense predictions located demonstrate spatial spacing, verifying the ability of the classifiers to predict exclusive instances.
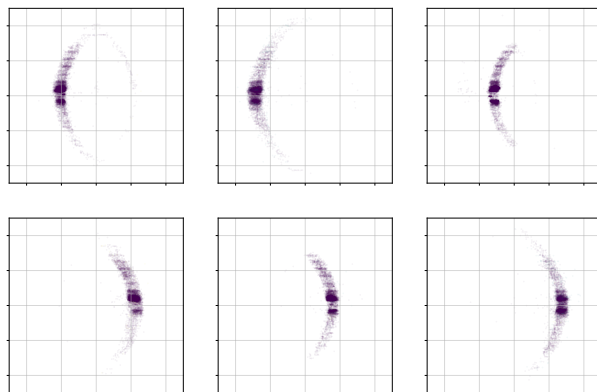

Figure 3: Spatial distributions of predictions by classifiers. The centers are projected into a 100m × 100m x-y plane. Results are obtained from SemanticKITTI test set.

5 Conclusion

In this paper, we develop a unified panoptic segmentation framework, dubbed PUPS, for point cloud data, which is capable to exclusively predict panoptic results without any hand-crafted post-processing and achieves state-of-the-art performance. PUPS allocates a set of classifiers to learn how to group coherent points directly and introduces bipartite matching to enable end-to-end training. Moreover, PUPS employs a transformer decoder to refine the groupings and resolve class imbalance problem by designing a context-aware cutmix augmentation. PUPS is the first to provide a holistic and end-to-end solution for point cloud panoptic segmentation. We hope that PUPS can inspire more researchers to delve into the development of unified segmentation for point cloud in autonomous driving.

Acknowledgements

References

Behley, J.; Milioto, A.; and Stachniss, C. 2020. A Benchmark for LiDAR-based Panoptic Segmentation based on KITTI. In *arXiv preprint arXiv:2003.02371*.

Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuScenes: A Multimodal Dataset for Autonomous Driving. In *CVPR*, 11618–11628.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-End Object Detection with Transformers. In *ECCV*, 213–229.

Cheng, B.; Collins, M. D.; Zhu, Y.; Liu, T.; Huang, T. S.; Adam, H.; and Chen, L.-C. 2020. Panoptic-DeepLab: A Simple, Strong, and Fast Baseline for Bottom-Up Panoptic Segmentation. In *CVPR*.

Cheng, B.; Schwing, A. G.; and Kirillov, A. 2021. Per-Pixel Classification is Not All You Need for Semantic Segmentation. In *NeurIPS*.

Fong, W. K.; Mohan, R.; Hurtado, J. V.; Zhou, L.; Caesar, H.; Beijbom, O.; and Valada, A. 2021. Panoptic nuScenes: A Large-Scale Benchmark for LiDAR Panoptic Segmentation and Tracking. *arXiv preprint arXiv:2109.03805*.

Gasperini, S.; Mahani, M. N.; Marcos-Ramiro, A.; Navab, N.; and Tombari, F. 2021. Panoster: End-to-End Panoptic Segmentation of LiDAR Point Clouds. *IEEE Robotics Autom. Lett.*, 3216–3223.

Hong, F.; Zhou, H.; Zhu, X.; Li, H.; and Liu, Z. 2021. LiDAR-Based Panoptic Segmentation via Dynamic Shifting Network. In *CVPR*, 13090–13099.

Hurtado, J. V.; Mohan, R.; and Valada, A. 2020. MOPT: Multi-Object Panoptic Tracking. In *CVPR Workshop*.

Kirillov, A.; Girshick, R.; He, K.; and Dollar, P. 2019a. Panoptic Feature Pyramid Networks. In *CVPR*.

Kirillov, A.; He, K.; Girshick, R. B.; Rother, C.; and Dollár, P. 2019b. Panoptic Segmentation. In *CVPR*, 9404–9413.

Lang, A. H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; and Beijbom, O. 2019. PointPillars: Fast Encoders for Object Detection From Point Clouds. In *CVPR*, 12697–12705.

Li, F.; Zhang, H.; xu, H.; Liu, S.; Zhang, L.; Ni, L. M.; and Shum, H.-Y. 2022a. Mask DINO: Towards A Unified Transformer-based Framework for Object Detection and Segmentation. arXiv:2206.02777.

Li, J.; He, X.; Wen, Y.; Gao, Y.; Cheng, X.; and Zhang, D. 2022b. Panoptic-PHNet: Towards Real-Time and High-Precision LiDAR Panoptic Segmentation via Clustering Pseudo Heatmap. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11809–11818.

Li, Y.; Chen, X.; Zhu, Z.; Xie, L.; Huang, G.; Du, D.; and Wang, X. 2019. Attention-Guided Unified Network for Panoptic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, Z.; Wang, W.; Xie, E.; Yu, Z.; Anandkumar, A.; Alvarez, J. M.; Luo, P.; and Lu, T. 2022c. Panoptic SegFormer: Delving Deeper Into Panoptic Segmentation With Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1280–1289.

Lin, T. Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal Loss for Dense Object Detection. *PAMI*, PP(99): 2999–3007.

Liu, H.; Peng, C.; Yu, C.; Wang, J.; Liu, X.; Yu, G.; and Jiang, W. 2019. An End-To-End Network for Panoptic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Loshchilov, I.; and Hutter, F. 2017. Fixing Weight Decay Regularization in Adam. *CoRR*, abs/1711.05101.

Milioto, A.; Behley, J.; McCool, C.; and Stachniss, C. 2020. LiDAR Panoptic Segmentation for Autonomous Driving. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 8505–8512.

Milioto, A.; Vizzo, I.; Behley, J.; and Stachniss, C. 2019. RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. In *IROS*, 4213–4220.

Milletari, F.; Navab, N.; and Ahmadi, S. A. 2016. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*.

MMDetection3DContributors. 2020. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d. Accessed: 2022-05-20.

Neven, D.; Brabandere, B. D.; Proesmans, M.; and Gool, L. V. 2019. Instance Segmentation by Jointly Optimizing Spatial Embeddings and Clustering Bandwidth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Porzi, L.; Bulo, S. R.; Colovic, A.; and Kontschieder, P. 2019a. Seamless Scene Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Porzi, L.; Bulò, S. R.; Colovic, A.; and Kontschieder, P. 2019b. Seamless Scene Segmentation. In *CVPR*, 8277–8286.

Razani, R.; Cheng, R.; Li, E.; Taghavi, E.; Ren, Y.; and Bingbing, L. 2021. GP-S3Net: Graph-Based Panoptic Sparse Semantic Segmentation Network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 16076–16085.

Sirohi, K.; Mohan, R.; Büscher, D.; Burgard, W.; and Valada, A. 2021. EfficientLPS: Efficient LiDAR Panoptic Segmentation. *CoRR*, abs/2102.08009.

Thomas, H.; Qi, C. R.; Deschaud, J.; Marcotegui, B.; Goulette, F.; and Guibas, L. J. 2019. KPConv: Flexible and Deformable Convolution for Point Clouds. In *ICCV*, 6410–6419.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *NeurIPS*, 5998–6008.

Wang, H.; Zhu, Y.; Adam, H.; Yuille, A.; and Chen, L.-C. 2021. MaX-DeepLab: End-to-End Panoptic Segmentation With Mask Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5463–5474.

Xu, J.; Zhang, R.; Dou, J.; Zhu, Y.; Sun, J.; and Pu, S. 2021. RPVNet: A Deep and Efficient Range-Point-Voxel Fusion Network for LiDAR Point Cloud Segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 16024–16033.

Yan, Y.; Mao, Y.; and Li, B. 2018. SECOND: Sparsely Embedded Convolutional Detection. *Sensors*, 3337.

Zhang, W.; Pang, J.; Chen, K.; and Loy, C. C. 2021. K-Net: Towards Unified Image Segmentation. In *NeurIPS*.

Zhou, Z.; Zhang, Y.; and Foroosh, H. 2021. PanopticPolarNet: Proposal-Free LiDAR Point Cloud Panoptic Segmentation. In *CVPR*, 13194–13203.