Our code is available at
https://github.com/shenglandu/PushBoundary

Push-the-Boundary: Boundary-aware Feature Propagation for Semantic Segmentation of 3D Point Clouds

Shenglan Du, Nail Ibrahimli, Jantien Stoter, Julian Kooij, Liangliang Nan
Delft University of Technology
Julianalaan 134, Delft 2628BL, The Netherlands
{shenglan.du, n.ibrahimli, j.e.Stoter, j.f.p.kooij, liangliang.nan}@tudelft.nl

Abstract

Feedforward fully convolutional neural networks currently dominate in semantic segmentation of 3D point clouds. Despite their great success, they suffer from the loss of local information at low-level layers, posing significant challenges to accurate scene segmentation and precise object boundary delineation. Prior works either address this issue by post-processing or jointly learn object boundaries to implicitly improve feature encoding of the networks. These approaches often require additional modules which are difficult to integrate into the original architecture.

To improve the segmentation near object boundaries, we propose a boundary-aware feature propagation mechanism. This mechanism is achieved by exploiting a multi-task learning framework that aims to explicitly guide the boundaries to their original locations. With one shared encoder, our network outputs (i) boundary localization, (ii) prediction of directions pointing to the object interior, and (iii) semantic segmentation, in three parallel streams. The predicted boundaries and directions are fused to propagate the learned features to refine the segmentation. We conduct extensive experiments on the S3DIS and SensatUrban datasets against various baseline methods, demonstrating that our proposed approach yields consistent improvements by reducing boundary errors.

1. Introduction

Semantic segmentation of 3D point clouds aims to assign each point a semantic category (*e.g.*, building, tree, window), which is a fundamental yet challenging task in 3D computer vision. Successful interpretation of semantics from 3D point clouds serves as a crucial prerequisite for various applications, such as autonomous driving [42], robotics [5], and urban environment modeling [3].

Driven by the success of deep learning and Fully Convolutional Networks (FCNs) in 2D image recognition, many

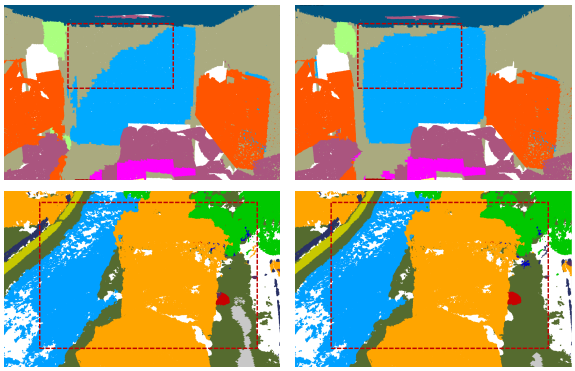


Figure 1: Comparison between the segmentation results from the previous FCN-based network [32] and ours on both indoor (top) and outdoor (bottom) scenes. Left: standard FCNs lose boundary details at the object level. Right: our approach refined the segmentation at object boundaries.

works have been proposed for semantic segmentation of 3D point clouds by transforming points into regular grids or voxels as input to standard FCNs [9, 20, 25, 27, 35, 38]. Such methods introduce extra computational costs and information loss which leads to suboptimal performance. To avoid that, the seminal PointNet [23] directly consumes point clouds and extracts point features through a sequence of shared Multi-Layer Perceptrons (MLPs). Following PointNet, a number of point-based deep learning frameworks have been introduced [12, 17, 18, 32, 34]. Despite their strong performances in the semantic segmentation task, they suffer from an often-overlooked limitation in feature propagation: the loss of local information in decoding. Specifically, since most existing networks adopt FCN-like architectures with the encoding-decoding strategy, the existence of pooling layers in the encoder can capture hierarchical semantic features with increased receptive fields. Though beneficial for object-level recognition, it leads to coarse feature maps at lower resolutions. In the decoder, these coarse features are then propagated back to the orig-

inal resolution via nearest-neighbor upsampling, which ignores point-level variations among different semantic categories. As a result, networks lose object boundary details and fail to generate accurate predictions. Figure 1 illustrates the segmentation outputs with blurred object boundaries.

Several works have made fruitful attempts to address semantic segmentation in both 2D and 3D domains. One attempt is to construct contextual affinity through graphical models such as Markov Random Fields (MRFs) and Conditional Random Fields (CRFs) [4, 45]. However, these methods introduce MRFs and CRFs as additional modules that are challenging to be integrated into the networks. Considering that boundaries play an important role in semantic segmentation, as they naturally indicate the transition between objects of different semantic categories, another line of methods exploits boundary information in the networks [6, 7, 10, 13, 44]. Most of these methods incorporate boundary detection as an auxiliary branch for semantic segmentation. With one shared encoder, the two tasks implicitly improve each other. However, such methods do not seek to explicitly tackle semantic segmentation. Another drawback is that extra encoding layers are needed to merge features from the two tasks, which is more challenging for the network to optimize as more parameters are involved.

Contrary to existing works, our work is mainly motivated by the attempt to *refine semantic segmentation by explicitly pushing the boundary towards desired directions*. Predicting a direction scheme to refine semantic segmentation has been recently studied in 2D image recognition [21, 41], although in these works directions mainly serve as post-processing tools to refine generated semantic labels. To overcome such limitations, we propose a novel end-to-end framework for joint boundary detection, direction prediction, and semantic segmentation. The proposed network has one FCN feature encoder while jointly giving three streams of point-wise predictions: (i) a boundary label (i.e., binary prediction), (ii) a direction vector that originates from the closest boundary and points to the object's interior, and (iii) a semantic class label. We demonstrate that even though the FCN architecture is primarily optimized for semantic segmentation, it does provide discriminative features for the two tasks of boundary detection and direction prediction.

The key to our framework is a lightweight guiding mechanism that effectively fuses the boundary and direction information to refine the segmentation. Our motivation is to reduce the loss of local information in the decoder of the framework. Within decoding layers, we propose to guide the feature propagation using the predicted direction (i.e., pointing from boundary to interior) near object boundaries. In such a way, feature mixture from different semantic classes is prevented while semantic boundaries are explicitly pushed along the desired direction. Our contributions can be summarized in two folds:

- To the best of our knowledge, the proposed network is the first end-to-end framework for joint semantic segmentation, boundary detection, and direction prediction in the 3D domain. The tasks of boundary detection and direction prediction can appropriately improve segmentation output.
- We introduce a novel boundary-aware feature upsampling strategy to guide feature propagation towards the predicted directions near object boundaries, which can be easily plugged into existing frameworks.

2. Related Work

2.1. Point Cloud Semantic Segmentation

Several methods have been proposed for semantic segmentation of point clouds. They can be roughly divided into projection-based, voxel-based, and point-based methods.

Projection-based methods first project points onto specified 2D planes and then apply 2D FCNs to recognize 3D objects [9, 27, 38, 39]. These methods do not fully utilize the data due to the loss of information in the z^{th} dimension in 2D projection, limiting their ability to analyze complex 3D objects in 3D scenes.

Voxel-based methods discretize point clouds over a volumetric 3D grid for shape classification and semantic segmentation using standard 3D FCNs [20, 25, 35]. Such methods require data transformation, which introduces extra computational costs and information loss due to the limited resolution of the grid structure. Besides, the voxel representation also has a high demand for GPU memory, limiting its application to single objects and small scenes.

Compared with the above two types of methods, *point-based methods* directly take point clouds as input to the networks and have demonstrated promising performances on various datasets. PointNet [23] is the first successful attempt in this direction, which adopts a sequence of MLPs and max-pooling operators to learn global features that can be used for 3D shape classification and scene segmentation. PoinNet++ [24] recursively applies PointNet over a nested partitioning of the points to learn fine-grained details of local geometrical structures. Besides MLPs, a number of works [14, 18, 32, 34] explore 3D convolutional operators on point clouds. Another line of works have also exploited Graph Convolutional Networks (GCNs) and self-attention mechanisms for gaining contextual knowledge of the points from local neighborhood graph patches [15, 17, 26, 33, 43]. Our work is built upon FCN-like architectures such as PointNet++ [24] and KP-Conv [32], aiming to tackle the loss of local boundary details in these networks.

2.2. Boundary-aware Segmentation

Standard FCN-like networks for semantic segmentation fail to model point-level accurate object boundaries, mainly

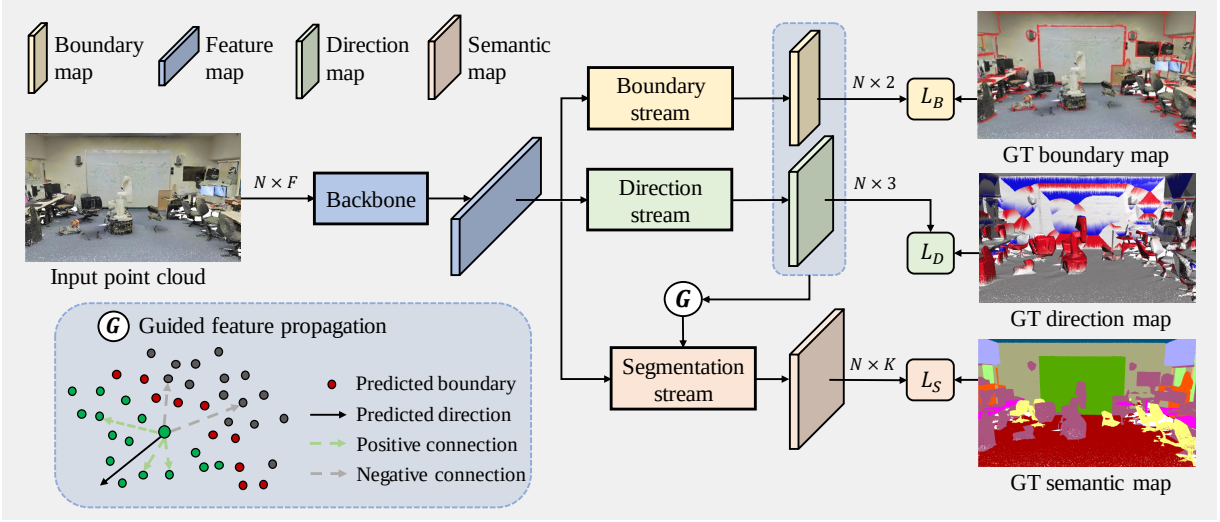


Figure 2: An overview of the network architecture. Our network can adopt various FCN-like feature encoders (*e.g.*, KP-Conv [32], PointNet++ [24]) for learning point features. N denotes the number of input points, F the input feature dimension, K the output category numbers, and GT the ground truth. L_B, L_D, and L_S are the network losses for the boundary, direction, and segmentation streams, respectively. In the guided feature propagation, a positive connection means the vector of the point pair has a positive cosine angle with the predicted direction, while a negative connection means a negative cosine angle.

due to the significant loss of low-level features both in the feature encoding and decoding layers. Many works have been introduced to sharpen the object boundaries.

In 2D image recognition, Boundary Neural Field [2] formalizes a global energy model to enhance semantic segment coherence with predicted boundary cues. Other works combine the semantic segmentation task and the boundary detection task into one network [6, 10, 19, 28, 29, 40], in which the two tasks share a common feature encoder and are expected to mutually improve each other.

In the 3D domain, boundary-aware strategies have also been adopted by a few works. JSENet [44] employs extra feature enhancement modules that require a curriculum learning strategy for good segmentation performance. CBL [30] introduces additional loss terms to contrast the features across the scene boundaries. Contrary to them, our work focuses on interior information passage in network decoding. We show that the guided feature propagation naturally recovers information near boundaries, which does not require exterior constraints or additional training modules. Several works [7, 36] address 3D boundary detection by adaptive methods or pre-process modules. Instead, our method does not treat boundaries as intermediate results. It jointly learns boundary cues for guiding the feature propagation, which can be easily plugged into existing networks.

2.3. Graphical Models for Segmentation

In the 2D domain, some works have proposed to integrate graphical models, such as MRFs and CRFs, into the networks for segmentation refinement [4, 16, 45]. Based on the fact that pixels presenting similar features tend to have the same semantic labels, this line of work formulates segmentation as a probabilistic inference problem using graphical models. While these methods can sharpen the segmentation mask near object boundaries, integrating additional MRFs and CRFs into the networks is still challenging. Besides, this strategy requires extra computational costs and is therefore slow in speed.

3. Methodology

Our network consists of one FCN feature encoder followed by branches of three task streams: (i) boundary detection, (ii) direction prediction (i.e., for each point, we predict a direction vector from its closest boundary to the object interior), and (iii) semantic segmentation. The outputs of (i) and (ii) are then fused to guide the feature propagation in a boundary-aware manner. Figure 2 overviews our architecture. In the following, we explain each part in detail.

3.1. Architecture

We use KP-Conv [32] as our backbone network, which directly applies fully convolutional layers over 3D point clouds using kernel-point convolution. In Section 4.1, we also provide segmentation experiments adopting other baseline networks as backbones, such as PointNet++ [24].

Boundary detection stream. For the input point cloud $P \in R^{N \times F}$, where N is the number of points and F is the input feature dimension, this stream predicts a binary map $B \in R^{N \times 2}$, with 1 for the points on boundaries and 0 for interior points. Note that we refer to semantic boundaries

(i.e., boundaries between different semantic categories) in this task. Due to that boundary points only account for a small portion of the whole point set, we use the weighted binary cross-entropy loss to supervise this task.

Direction prediction stream. Predicting a direction scheme to refine semantic segmentation has been recently studied [21, 41] in the 2D domain. These works require separate learning for direction vectors, which then serve as a post-processing technique to refine the segmentation output. In this work, we extend the direction scheme to the 3D domain and learn it in an end-to-end manner. The direction stream predicts a direction map $D \in R^{N\times 3}$, with each row (d_x, d_y, d_z) giving a unit direction vector pointing from the closest boundary to the object interior. Designed in this way, the learned D generates for each point a pointer towards its homogeneous region interior. We consider direction prediction as a regression task and thus adopt Mean Squared Error loss for supervision. The predicted directions are further fused with boundary predictions to guide the feature propagation in the semantic segmentation task stream.

Semantic segmentation stream. Given a point cloud $P \in R^{N\times F}$, the semantic segmentation branch outputs a semantic mask $S \in R^{N\times K}$, where K is the total number of semantic categories. This stream predicts a probability distribution for each point over K categories. We supervise this task using the standard cross-entropy loss.

3.2. Boundary-aware Feature Propagation

FCNs adopt pooling layers that progressively subsample points and obtain high-level semantic features in the latent space. In the subsequent decoding stage, features are propagated from sub-sampled points to the original points. A common propagation strategy is based on inverse distance interpolation in a local k-Nearest Neighborhood (kNN). For example, PointNet++ [24] uses $k = 3$ for feature propagation, while KP-Conv [32] uses $k = 1$ that reduces to the nearest neighbor feature upsampling. Standard feature propagation assumes that spatially close points also share the semantic affinity. However, it ignores drastic semantic transitions across object boundaries. Points at the boundary can inherit information from different objects, thus resulting in blurred feature representation, which is not favorable for the final segmentation.

We use the predicted boundaries and directions to guide the feature propagation in decoding layers, where features are encouraged to pass along the desired direction for generating purified feature maps. The guided feature propagation is illustrated in Figure 3. In the FCN decoder, features are up-sampled from sparser points in high layers to denser points in low layers. We propagate features of points $\mathbf{x_j}$ in l^{th} layer to points $\mathbf{x_i}$ in $l-1^{th}$ layer. The feature propagation is performed by

$$f^{l-1}(\mathbf{x_i}) = \sigma(\Phi(\frac{\sum_{j=1}^{k} w(\mathbf{x_j}) f^{l}(\mathbf{x_j})}{\sum_{j=1}^{k} w(\mathbf{x_j})}) \oplus f_{skip}^{l-1}(\mathbf{x_i})), \quad (1)$$

where $\mathbf{x} \in R^3$ denotes a 3D point and k is the total number of neighboring points. The decoder features $f^l(\mathbf{x_j})$ of neighboring points $\mathbf{x_j}$ at l^{th} layer are first interpolated based on corresponding weights w, and then concatenated with skip-linked encoder features that will pass through a MLP layer and a Relu activation function to obtain the new features $f^{l-1}(\mathbf{x_i})$ for $\mathbf{x_i}$. f_{skip} denotes the skip-linked features, $\Phi(\cdot)$ a MLP operator, and $\sigma(\cdot)$ the ReLU activation function. We compute an adaptive weight term w to guide the feature propagation process,

$$\begin{aligned} w(\mathbf{x_j}) &= \max(0, w_s(\mathbf{x_j}, \mathbf{x_i}) + \alpha w_c(\mathbf{x_j}, \mathbf{x_i})), \quad (2)\\ w_s(\mathbf{x_j}, \mathbf{x_i}) &= \exp \frac{-||\mathbf{x_j} - \mathbf{x_i}||_2}{r},\\ w_c(\mathbf{x_j}, \mathbf{x_i}) &= \exp(P_b(\mathbf{x_i}) - 1) \cos(\mathbf{x_j} - \mathbf{x_i}, \mathbf{d_i}), \end{aligned}$$

where we linearly combine the spatial similarity w_s and cosine similarity w_c between $\mathbf{x_j}$ and $\mathbf{x_i}$. $P_b(\mathbf{x_i})$ is the predicted boundary probability of point $\mathbf{x_i}$, $\mathbf{d_i}$ the predicted direction of $\mathbf{x_i}$, and $\cos(\cdot)$ the cosine angle of two vectors. $\max(\cdot)$ is used for eliminating negative weights generated in the computation. The constant coefficients r and α are used to balance the two terms. The computed weights are further normalized, i.e.,

$$w(\mathbf{x_j}) = \frac{w(\mathbf{x_j})}{\sum_{j=1}^{k} w(\mathbf{x_j})}. \quad (3)$$

Our propagation strategy favors semantic segmentation in three folds. (i) For points near boundaries (i.e., with $P_b \approx 1$), we assign significant importance to the neighbors aligned with the direction while eliminating the influence of opposed neighbors, encouraging the features to propagate along the desired direction. (ii) For interior points (i.e., with $P_b \approx 0$), the second cosine term is largely reduced. All neighbors majorly contribute to feature propagation according to their distances to the center. Therefore, we facilitate smooth segmentation in the object interior. (iii) Our designed weight term is a continuous function over B and D, which helps the gradient backpropagation of the network. In Section 4.3, we conduct ablation studies to further verify the effectiveness of the proposed boundary-aware feature propagation mechanism.

3.3. Network Supervision

Our network involves three tasks (see Figure 2), which are supervised using the three loss terms, respectively.

Boundary detection loss. Boundary detection is a binary classification task. Due to the sparsity of boundary

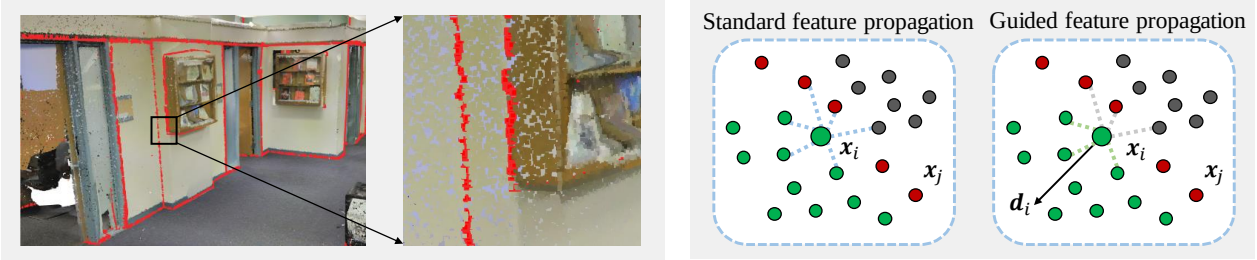


Figure 3: Feature propagation near object boundaries. Left: drastic semantic transitions happen across the object boundaries within a small local neighborhood. Right: two ways of feature propagation. We propagate features from neighboring points $\mathbf{x_j}$ in the higher layer to the center $\mathbf{x_i}$ in the lower layer. *Standard feature propagation* ignores object boundaries (red dots), which leads to the mixed feature representation of $\mathbf{x_i}$. In contrast, *our proposed strategy* uses the predicted direction vector $\mathbf{d_i}$ to guide the feature propagation. Features from neighboring $\mathbf{x_j}$ aligned with $\mathbf{d_i}$ are encouraged to propagate to the center $\mathbf{x_i}$ (green dashed lines), while feature propagation from the opposed neighbors is constrained (grey dashed lines).

points in the dataset, we adopt the weighted binary cross entropy loss to supervise this task,

$$L_B = -\sum_{i=1}^{N}[\,\beta\hat{b_i}\log b_i + (1-\beta)(1-\hat{b_i})\log(1-b_i)], \quad (4)$$

where $\hat{b_i}$ denotes the GT binary label, i.e., 1 for boundary points and 0 for interior points. b_i is the network softmax output of the i^{th} point. We use a coefficient β to balance the boundary class and the object interior class.

Direction prediction loss. Unlike previous work of 2D image segmentation [41] that predicts discrete directions by evenly dividing the entire direction into a set of ranges, we perform continuous prediction in the whole space. Therefore, our predictions are more adaptable for complex 3D scenes. Specifically, we tackle direction prediction as a regression task and adopt MSE loss for supervision, i.e.,

$$L_D = \sum_{i=1}^{N} \|\mathbf{d_i} - \hat{\mathbf{d}}_\mathbf{i}\|_2^2, \quad (5)$$

where $\mathbf{d_i} \in R^3$ is the predicted direction, and $\hat{\mathbf{d}}_\mathbf{i} \in R^3$ is the GT direction (both directions have a unit magnitude). We also tested with the dot product loss of $\mathbf{d_i}$ and $\hat{\mathbf{d}}_\mathbf{i}$, while our experiments suggested that the MSE loss gives robust direction predictions that are less variant to dataset scales.

Semantic segmentation loss. We use the standard cross entropy loss to supervise the semantic segmentation stream,

$$L_S = -\sum_{i=1}^{N} y_i^s \log p^s(\mathbf{x_i}), \quad (6)$$

where $y_i^s \in R^K$ denotes the one-hot vector of the Ground Truth (GT) semantic label s of the i^{th} point. $p^s(\mathbf{x_i})$ is the predicted probability of the i^{th} point over the GT category obtained from the network softmax layer.

Accordingly, the total network loss is given by

$$L = L_S + \lambda_1 L_B + \lambda_2 L_D, \quad (7)$$

where we use λ_1 and λ_2 to balance between different losses.

3.4. Implementation Details and Parameters

Our method is implemented in Pytorch [22]. Regarding the hyperparameters, we set $\alpha = 1.0$ and $r = 0.125$ for the guided feature propagation in Equation 2. For supervising the boundary stream in Equation 4, we set $\beta = 0.6$. For the weights in Equation 7, we set $\lambda_1 = 3.0$ and $\lambda_2 = 0.3$.

4. Experiments

Evaluation setup. We evaluate the effectiveness of our approach in semantic segmentation of 3D point clouds using two state-of-the-art network architectures as our backbone: PointNet++ [24] and KP-Conv [32]. Pointnet++ recursively applies point-wise MLP operators over a nested partitioning of the point clouds to obtain hierarchical point features. In contrast, KP-Conv explores point convolution in 3D Euclidean space. Both approaches employ FCN-like architectures with the encoding-decoding strategy.

Our experiments are carried out on datasets of both indoor and outdoor scenes. We use standard metrics including Overall Accuracy (OA), mean Intersection over Union (mIoU), and per-category IoU scores for the evaluation. To achieve a fair per-category comparison, we use the standard unweighted cross-entropy loss for all the experiments, and we adopt the same experimental settings as in the original corresponding baseline approaches.

Ground-truth boundary maps and direction maps. Both GT boundary maps and direction maps are directly derived from the raw dataset. To generate GT boundary maps, we use kNN search for each point in the dataset (we empirically set $k = 4$). If its semantic label differs from any of its neighbors, we recognize it as a boundary point. To generate

Method	OA(%)	**mIoU(%)**	ceiling	floor	wall	beam	col.	window	door	table	chair	sofa	book.	board	clutter
PointNet++ [37] (w/o N)	83.5	53.6	89.6	**97.6**	74.6	0.0	4.7	55.1	**19.9**	78.2	68.9	**66.3**	41.7	55.6	**44.2**
+ Ours	**83.6**	**53.9**	**89.7**	97.2	**75.4**	0.0	**7.4**	**60.8**	14.1	**79.0**	**69.5**	65.8	**42.1**	**57.0**	42.6
PointNet++ [37] (w/ N)	83.9	53.9	91.4	96.6	76.0	0.0	8.0	**53.7**	16.9	81.8	**70.5**	63.8	48.9	49.4	43.6
+ Ours	**84.3**	**55.1**	**91.5**	**97.2**	76.0	0.0	**13.8**	53.4	**19.1**	**83.6**	70.4	**65.4**	**49.6**	**50.3**	**45.4**
KP-Conv rigid [32] (w/o N)	-	65.4	92.6	97.3	81.4	0.0	16.5	54.5	69.5	**80.2**	90.1	66.4	**74.6**	**63.7**	58.1
+ Ours	89.7	**67.1**	**94.0**	**97.9**	**82.6**	0.0	**23.3**	**56.6**	**75.4**	80.1	**91.1**	**75.7**	74.4	62.3	**59.1**
KP-Conv rigid [32] (w/ N)	89.2	65.6	**94.0**	97.9	81.6	0.0	20.0	54.5	64.4	80.1	91.6	77.4	73.8	59.2	59.0
+ Ours	**89.6**	**67.2**	93.9	97.9	**82.7**	**0.2**	**23.8**	**55.0**	**73.7**	**80.5**	**91.8**	**77.7**	**74.1**	**63.2**	59.0

Table 1: Semantic segmentation results achieved on the S3DIS dataset [1], following instructions of the officially released code of each method. These methods are trained using either 3D coordinates and color information (w/o N) or with additional normal information (w/ N). We adopt unweighted cross-entropy loss for all the segmentation experiments to achieve a fair comparison. Overall Accuracy (OA, %), mean Intersection over Union (mIoU, %), and per-category IoU scores are reported. We report averaged scores over five training runs for PointNet++ [37] based networks due to performance variations.

GT direction maps, we search the closest boundary point for each point, and the direction is assigned as the direction pointing from the closest boundary point to the current point. We use the normalized direction vectors (i.e., with a length of 1) as our GT for training.

4.1. Semantic Segmentation on Indoor Scenes

For indoor scenes, we use the challenging Stanford Large-Scale 3D Indoor Spaces (S3DIS) dataset [1] for evaluation. S3DIS is a large-scale dataset for semantic indoor scene parsing. Each point is annotated with a semantic label from 13 categories (e.g., ceiling, table, window). Since the S3DIS dataset contains massive points that are impractical to be directly segmented, both the two baseline methods were intentionally designed to train and test on downsampled data. We follow the same data sampling strategies as the baseline networks. Furthermore, to align with the previous works [17, 23, 32, 43], we also use Area 5 for testing and the rest areas for training.

In Table 1, we report the performance comparison between the two baselines and our corresponding networks. These methods are either trained using only 3D coordinates and color information (w/o N), or using 3D coordinates, color information, and normals (w/ N). We consider training the networks using normal information as they help discriminate between boundary and non-boundary points. Compared to the baselines, our boundary-aware feature propagation mechanism achieves consistent improvements in terms of both OA and mIoU scores. Trained using only 3D coordinates and color information, our method obtains mIoU gains of 0.3% and 1.7% with the backbones PointNet++ and KP-Conv, respectively. When trained using 3D coordinates, color information, and normals, our method obtains mIoU gains of 1.2% and 1.6% with the backbones PointNet++ and KP-Conv, respectively. Specifically, we observe significant improvements in categories such as column, window, door, and board, for which accurate boundary delineation is difficult to achieve in the baseline methods. In some scenarios our method can also propagate segmentation errors to a larger extent, leading to a decrease in mIoU scores of certain categories (e.g., board). Nevertheless, the qualitative results show that our segmentation outputs more regularized shapes (Section 4.4). In *Supplementary Material*, we present more comparisons with other state-of-the-art works in the S3DIS benchmark.

4.2. Semantic Segmentation on Outdoor Scenes

For outdoor scenes, we use the most recent SensatUrban dataset [11] for evaluation. SensatUrban is a large-scale UAV photogrammetry point cloud dataset, consisting of nearly three billion points with fine-grained semantic labels (*e.g.*, building, high vegetation, rail) collected from three UK cities. Following the SensatUrban benchmark, we use the Birmingham block 1 and 5, and the Cambridge block 7 and 10 for validation. We use the Birmingham block 2 and 8, and the Cambridge block 15, 16, 22, and 27 for testing. The rest tiles are used for training. We experiment on SensatUrban using KP-Conv [32] as our backbone.

In Table 2, we present the performance of our approach in comparison with KP-Conv and other state-of-the-art methods. Our method has achieved an OA of 93.9% and an mIoU of 59.7%, which outperforms all the seminal competitors on this benchmark. The results demonstrate the capability of our approach to generalize to large-scale outdoor urban scenes. Compared to the state-of-the-art baseline KP-Conv [32], using our boundary-aware feature propagation strategy improves its mIoU by a margin of 2.1%. Regarding per category performance, improvements in most categories can also be observed, especially in the categories that exhibit distinct geometric boundaries, such as bridges, roads, and footpaths. The minority categories such as rail and bike cannot be successfully recognized by most methods listed in the table, which is due to the class imbalance in the SensatUrban dataset. We also observe a performance drop in the water category (20.8% IoU compared to baseline). This is attributed to the fact that water in urban scenes often has

Method	OA(%)	**mIoU(%)**	ground	veg.	building	wall	bridge	parking	rail	traffic.	street.	car	footpath	bike	water
PointNet [23]	80.8	23.7	68.0	89.5	80.1	0.0	0.0	4.0	0.0	32.0	0.0	35.1	0.0	0.0	0.0
PointNet++ [24]	84.3	32.9	72.5	94.2	84.8	2.7	2.1	25.8	0.0	31.5	11.4	38.8	7.1	0.0	56.9
TangentConv [31]	77.0	33.3	71.5	91.4	75.9	35.2	0.0	45.3	0.0	26.7	19.2	67.6	0.0	0.0	0.0
SPGraph [17]	85.3	37.3	69.9	94.6	88.9	32.8	12.6	15.8	**15.5**	30.6	23.0	56.4	0.5	0.0	44.2
SparseConv [8]	88.7	42.7	74.1	97.9	94.2	63.3	7.5	24.2	0.0	30.1	34.0	74.4	0.0	0.0	54.8
RandlaNet [12]	89.8	52.7	80.1	98.1	91.6	49.0	40.8	51.6	0.0	56.7	33.2	80.1	32.6	0.0	71.3
KP-Conv [32]	93.2	57.6	**87.1**	**98.9**	95.3	74.4	28.7	41.4	0.0	56.0	54.4	85.7	40.4	0.0	**86.3**
Ours	**93.8**	**59.7**	85.8	98.9	**96.8**	**79.3**	**49.7**	**52.4**	0.0	**62.1**	**57.5**	**86.8**	**42.0**	0.0	65.5

Table 2: Semantic segmentation results achieved on SensatUban dataset [11], evaluated on the Birmingham block 2, 8, and the Cambridge block 15, 16, 22, and 27. Overall Accuracy (OA, %), mean Intersection over Union (mIoU, %), and per-category IoU scores are reported. The results of the seven competing networks are from the original SensatUrban benchmark paper [11]. To achieve a fair comparison, we train our method using only the 3D coordinates and color information.

irregular geometrical boundaries, which leads to noisy estimates in both boundary detection and direction prediction, yielding less accurate segmentation.

4.3. Ablation Study

In this section, we present ablation studies to support our contributions in detail. We use the S3DIS dataset as we can access the GT semantic labels of all the points. We conduct our ablation studies adopting KP-Conv [32] as our backbone network. Same as in the previous works [17, 23, 32, 43], we use Area 5 for testing and the rest areas for training. All the experiments are conducted under the same hyperparameter settings as in Section 4.1.

Table 3 presents the results of the ablation experiments. From this table, we observe that adding the boundary stream to the baseline improves mIoU by 1.0% and OA by 0.1%. Adding the direction stream to the baseline improves mIoU by 0.8% but slightly decreases the OA (0.1%), and combining the three streams with the standard feature propagation achieves a 0.2% gain in OA and 0.7% gain in mIoU over the baseline. Compared to adding only the boundary stream, adding both the boundary stream and the direction stream to the network shows a decrease of 0.3% in mIoU. Even though, the direction stream brings important information for segmentation refinement in the next step. Using the proposed boundary-aware feature propagation mechanism, we observe an increase in both metrics, i.e., 0.4% in OA and 1.6% in mIoU. This reveals that more effective features are inherited to produce purified feature maps.

4.4. Qualitative Evaluation

In this section, we present qualitative results of semantic scene segmentation achieved on both S3DIS [1] and SensatUrban [11] datasets.

Figure 4 presents the segmentation results on the indoor dataset S3DIS. Our joint learning and boundary-aware feature propagation strategy effectively reduce segmentation errors near boundaries for several object categories, such as window, column, and board. To further understand

Method	OA(%)	**mIoU(%)**
(1) Baseline network	89.2	65.6
(2) Baseline + Boundary Stream	89.3	66.6
(3) Baseline + Direction Stream	89.1	66.2
(4) Full network with SFP	89.4	66.3
(5) Full network with GFP	89.6	67.2

Table 3: mIoU scores of the ablated networks from (1) to (5). SFP denotes standard feature propagation and GFP denotes guided feature propagation. All networks are trained using 3D coordinates, color information, and normals.

why simply guiding the feature propagation with the predicted directions significantly improves the performance, we go beyond numbers by visualizing what is learned in the boundary detection and the direction prediction streams. The visualization results reveal that even though the predicted boundaries and directions do not completely match the ground truth, they still provide informative guidance for feature propagation in the decoding layers.

Figure 5 demonstrates the segmentation results on the SensatUrban dataset, where the ground-truth labels are not presented since we do not have access to the true labels in the test set. However, the comparison with baseline results still shows that our method achieves better boundary localization for minor categories such as parking and footpath.

4.5. Limitations

Our proposed approach jointly performs boundary prediction, direction prediction, and semantic segmentation. By adopting a lightweight guiding mechanism for feature propagation, our approach can produce sharp feature maps that effectively reduce segmentation errors in boundary regions. However, it has several limitations. First, feature propagation is performed within a neighborhood, which improves the feature consistency of object interiors on a local level. Nevertheless, it cannot guarantee global-level feature consistency within a long range. Second, since our work focuses on guiding the features to recover boundary infor-

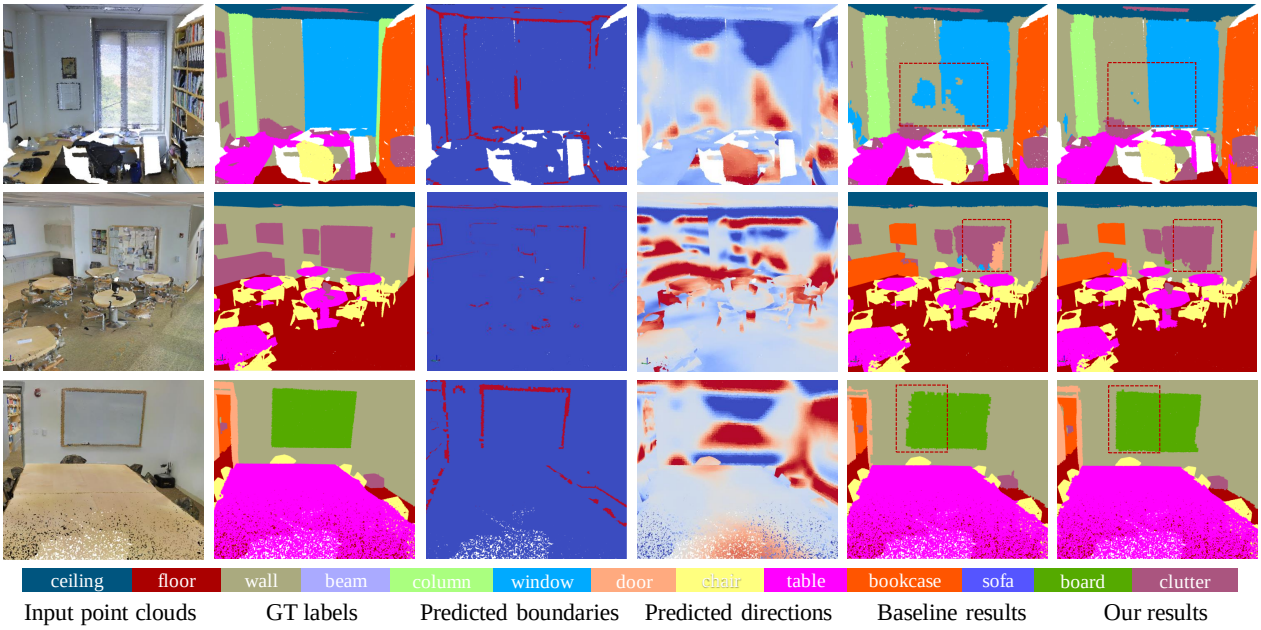


Figure 4: Qualitative results on the S3DIS dataset [1]. KP-Conv [32] is used as our backbone, trained using 3D coordinates, colors and normals. In the third column, the predicted boundary points are rendered in red. In the fourth column, we visualize the Z-component (i.e., vertical direction) of the predicted vectors, where blue indicates -1 (down) and red indicates +1 (up).

mation in the decoding layers, it naturally cannot cope with the information loss in the feature encoding layers. Last, our approach requires extra pre-processing steps (i.e., generating GT boundary maps and direction maps for training).

5. Conclusion

We have presented a novel boundary-aware feature propagation mechanism to improve semantic segmentation of 3D point clouds, in a way such that the boundaries are pushed to the desired locations. Our network jointly learns boundary maps, direction maps, and point-wise semantic labels end-to-end. Extensive studies on the S3DIS and SensatUrban datasets have demonstrated the effectiveness of our approach. Our experiments and analysis reveal two factors that contribute to the improvements in semantic segmentation. First, the joint learning of the three tasks mutually improves the shared feature encoder. Second, the predicted boundaries and directions are effective in guiding the points to inherit features from more homogeneous regions, which compensates for the loss of local boundary information in the decoding layers of FCNs. Our approach is particularly effective for objects with clear geometric boundaries such as doors, windows, and street paths.

Nevertheless, the achieved improvements are still limited because the locally-performed feature propagation cannot optimize the segmentation output on a global scale. Besides, since the three downstream tasks have different levels of complexity, combining them into a single network limits the capability of the shared feature encoder to learn discriminative features for all the tasks. In the future, we will explore the adaptive boundary detection method [36] to improve semantic segmentation. We would also like to extend our boundary-aware feature propagation mechanism using self-supervised learning techniques.

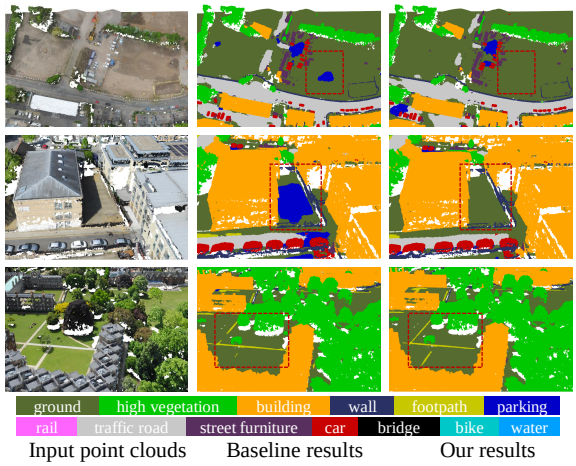


Figure 5: Qualitative results of semantic segmentation on the SensatUrban dataset [11], trained using 3D coordinates and colors.

Acknowledgement

This work was supported by the 3D Urban Understanding Lab funded by the TU Delft AI Initiative.

References

[1] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1534–1543, 2016.

[2] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. Semantic segmentation with boundary neural fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3602–3610, 2016.

[3] Filip Biljecki, Jantien Stoter, Hugo Ledoux, Sisi Zlatanova, and Arzu Çöltekin. Applications of 3d city models: State of the art review. *ISPRS International Journal of Geo-Information*, 4(4):2842–2889, 2015.

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[5] Jonathan Crespo, Jose Carlos Castillo, Oscar Martinez Mozos, and Ramon Barber. Semantic information for robot navigation: A survey. *Applied Sciences*, 10(2):497, 2020.

[6] Henghui Ding, Xudong Jiang, Ai Qun Liu, Nadia Magnenat Thalmann, and Gang Wang. Boundary-aware feature propagation for scene segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6819–6829, 2019.

[7] Jingyu Gong, Jiachen Xu, Xin Tan, Jie Zhou, Yanyun Qu, Yuan Xie, and Lizhuang Ma. Boundary-aware geometric encoding for semantic segmentation of point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1424–1432, 2021.

[8] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9224–9232, 2018.

[9] Haiyun Guo, Jinqiao Wang, Yue Gao, Jianqiang Li, and Hanqing Lu. Multi-view 3d object retrieval with deep embedding network. *IEEE Transactions on Image Processing*, 25(12):5526–5537, 2016.

[10] Zeeshan Hayder, Xuming He, and Mathieu Salzmann. Boundary-aware instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5696–5704, 2017.

[11] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Towards semantic segmentation of urban-scale 3d point clouds: A dataset, benchmarks and challenges. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4977–4987, 2021.

[12] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020.

[13] Zeyu Hu, Mingmin Zhen, Xuyang Bai, Hongbo Fu, and Chiew-lan Tai. Jsenet: Joint semantic segmentation and edge detection network for 3d point clouds. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 222–239, 2020.

[14] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 984–993, 2018.

[15] Li Jiang, Hengshuang Zhao, Shu Liu, Xiaoyong Shen, Chi-Wing Fu, and Jiaya Jia. Hierarchical point-edge interaction network for point cloud semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10433–10441, 2019.

[16] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, 24, 2011.

[17] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4558–4567, 2018.

[18] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31:820–830, 2018.

[19] Yun Liu, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Xiang Bai. Richer convolutional features for edge detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3000–3009, 2017.

[20] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[21] Davide Mazzini and Raimondo Schettini. Spatial sampling network for fast scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[23] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[24] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[25] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3577–3586, 2017.

[26] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on

graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3693–3702, 2017.

[27] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[28] Jinming Su, Jia Li, Yu Zhang, Changqun Xia, and Yonghong Tian. Selectivity or invariance: Boundary-aware salient object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3799–3808, 2019.

[29] Towaki Takikawa, David Acuna, Varun Jampani, and Sanja Fidler. Gated-scnn: Gated shape cnns for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5229–5238, 2019.

[30] Liyao Tang, Yibing Zhan, Zhe Chen, Baosheng Yu, and Dacheng Tao. Contrastive boundary learning for point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8489–8499, 2022.

[31] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3887–3896, 2018.

[32] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019.

[33] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10296–10305, 2019.

[34] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2589–2597, 2018.

[35] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[36] Mingye Xu, Zhipeng Zhou, Junhao Zhang, and Yu Qiao. Investigate indistinguishable points in semantic segmentation of 3d point cloud. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3047–3055, 2021.

[37] Xu Yan. Pointnet/pointnet++ pytorch. `https://github.com/yanx27/Pointnet_Pointnet2_pytorch`, 2019.

[38] Ze Yang and Liwei Wang. Learning relationships for multi-view 3d object recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7505–7514, 2019.

[39] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 186–194, 2018.

[40] Zhiding Yu, Chen Feng, Ming-Yu Liu, and Srikumar Ramalingam. Casenet: Deep category-aware semantic edge detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5964–5973, 2017.

[41] Yuhui Yuan, Jingyi Xie, Xilin Chen, and Jingdong Wang. Segfix: Model-agnostic boundary refinement for segmentation. In *European Conference on Computer Vision*, pages 489–506, 2020.

[42] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE access*, 8:58443–58469, 2020.

[43] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021.

[44] Mingmin Zhen, Jinglu Wang, Lei Zhou, Shiwei Li, Tianwei Shen, Jiaxiang Shang, Tian Fang, and Long Quan. Joint semantic segmentation and boundary detection using iterative pyramid contexts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13666–13675, 2020.

[45] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537, 2015.

— Supplementary Material —
Push-the-Boundary: Boundary-aware Feature Propagation for Semantic Segmentation of 3D Point Clouds

In this document, we provide more details on network implementation, additional ablation studies on the guided feature propagation, the comparison with baselines in terms of complexity and runtime, and the comparison with more state-of-the-art works on and S3DIS benchmark. In the end, we present more qualitative results.

A. Details in Implementation

Data sampling. Both S3DIS and SensatUrban are large datasets of 3D scenes, which are too big for the baseline networks to directly process. To handle such data, PointNet++ randomly samples a fixed number of points (i.e., 4096) in each scene for network training. KP-Conv first uses grid sampling to reduce the number of input points. For training, it randomly samples 3D spheres in the scenes as inputs for network batches. For testing, spheres are sampled regularly with a potential term to ensure that points can be seen by the network multiple times from different sphere locations. In the end, semantic predictions on grid-sampled clouds are transferred to the original clouds. We follow the same data sampling strategies as the baseline networks, which are further detailed in Table 1.

Network training. We keep all training settings the same as the baseline networks, such as the momentum for gradient descent optimization, base learning rate, and learning rate schedule. This is detailed in Table 1.

	Settings	PointNet++	KP-Conv	
		S3DIS	S3DIS	SensatUrban
Data	sampling strategy	random	grid + sphere	
	points per scene	4096	-	-
	grid size	-	5cm	20cm
	sphere radius	-	1.5m	9.0m
Training	base learning rate	0.001	0.01	0.01
	scheduler	70% per 10 epochs	98% per 1 epoch	
	momentum	-	0.98	0.98
	batch size	16	6	6
	steps per epoch	2973	300	600
	epochs	32	500	550

Table 1: Details of data pre-processing and network training.

B. Additional Ablation Studies on Guided Feature Propagation

In this section, we further investigate the effectiveness of our proposed feature propagation mechanism applied to various decoding layers.

Figure 1 shows a detailed illustration of the network architecture using KP-Conv as the backbone. Note that only the semantic segmentation stream is visualized for clarity. In the decoder, features are up-sampled four times, i.e., from the 5^{th} layer to the 4^{th} layer, from the 4^{th} layer to the 3^{rd} layer, etc. In the upsampling, we replace the Standard Feature Propagation (SFP) with Guided Feature Propagation (GFP) to evaluate the network performance.

Feature propagation details. We propagate features of neighboring points $\mathbf{x_j}$ in the l^{th} layer to points $\mathbf{x_i}$ in the $l-1^{th}$ layer. For GFP, the features are propagated using the weighting functions introduced in the Methodology section of the paper (see Equation (2) in Section 3.2). For SFP, we use the inverse distance weighting in a local neighborhood, i.e.,

$$w(\mathbf{x_j}) = \frac{1}{||\mathbf{x_j} - \mathbf{x_i}||_2 + \epsilon}, \tag{1}$$

where ϵ denotes a threshold that prevents the denominator to be 0.

Neighborhood size. For SFP, we use a fixed radius to search for the local neighbors since it is more robust to varying point densities compared to kNN. The base radius in the 1^{st} layer is set to 6.25cm and increases two times in every next layer. For GFP, as the predicted directions and boundaries naturally help to reduce outlier influences, we use kNN instead. We empirically use $k = 8$.

Table 2 presents the segmentation scores obtained by applying GFP in different feature upsampling layers. By adopting GFP for the 1^{st} upsampling layer and SFP for the rest layers, we achieve the highest mIoU score, which has already been reported in the paper. When replacing SFP

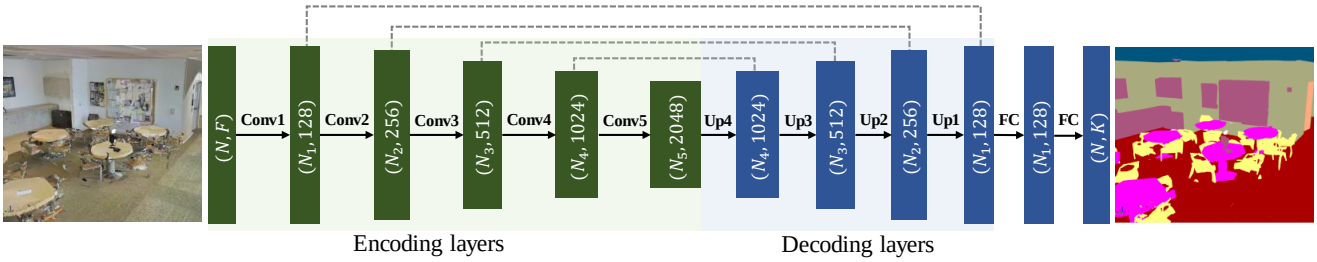


Figure 1: Network architecture details, adopting KP-Conv as the backbone. Only the segmentation stream is visualized for clarity. *Conv* denotes the convolutional-like operation for feature aggregation. *Up* denotes the feature upsampling process. *FC* denotes the fully connected layer. The grey dashed lines denote the skipped links. N is the number of points, F the dimension of the input feature, K the number of categories in the output, and N_i the number of points in each layer. Note that $N_1 = N$.

GFP layers	SFP layers	OA(%)	**mIoU(%)**
-	up1, up2, up3, up4	89.4	66.3
up1	up2, up3, up4	89.6	67.2
up1, up2	up3, up4	89.7	66.9
up1, up2, up3	up4	89.6	66.6
up1, up2, up3, up4	-	89.4	66.0

Table 2: OA and mIoU scores of the ablated networks adopting GFP in various decoding layers. SFP: standard feature propagation. GFP: our guided feature propagation. The experiments are conducted on the S3DIS dataset using KP-Conv as the backbone, trained using 3D coordinates, color information and normals.

with GFP in higher layers, we observe a slight drop in the performance. When adopting GFP for all the upsampling layers, the performance reaches its lowest. Our insights into this behavior is that, GFP is designed for recovering local boundary details, which explains its effectiveness when applied to the low-level upsampling layers. While in high layers, SFP is more effective because it well preserves global features in the feature propagation process.

C. Convergence and Efficiency

We compare the convergence and efficiency of our network with the baseline. The results are reported in Table 3.

Our networks have a significantly larger number of parameters compared to the baselines since we have separate decoders for the three downstream tasks. Thus, it is slower in both training and testing. Compared to PointNet++, the increase in training time and inference time are 23% and 15%, respectively. Compared to KP-Conv, we notice an average increase of 55% on training time and 33% on inference time.

Though with the higher complexity, we have observed a faster convergence of our network. This is demonstrated in

		PointNet++	KP-Conv	
		S3DIS	S3DIS	SensatUrban
#Params	Baseline	0.97	24.38	24.38
(M)	Ours	2.19	32.78	32.78
Training time	Baseline	0.52	0.09	0.13
(sec./batch)	Ours	0.64	0.14	0.20
Inference time	Baseline	0.71	0.05	0.08
(sec./batch)	Ours	0.82	0.07	0.10

Table 3: Comparison of running time against the baseline on the S3DIS and SensatUrban datasets. The total number of learnable parameters is the indicator of model complexity. We use the average running time per batch as the efficiency indicator. All experiments are conducted with an NVIDIA RTX2080Ti GPU.

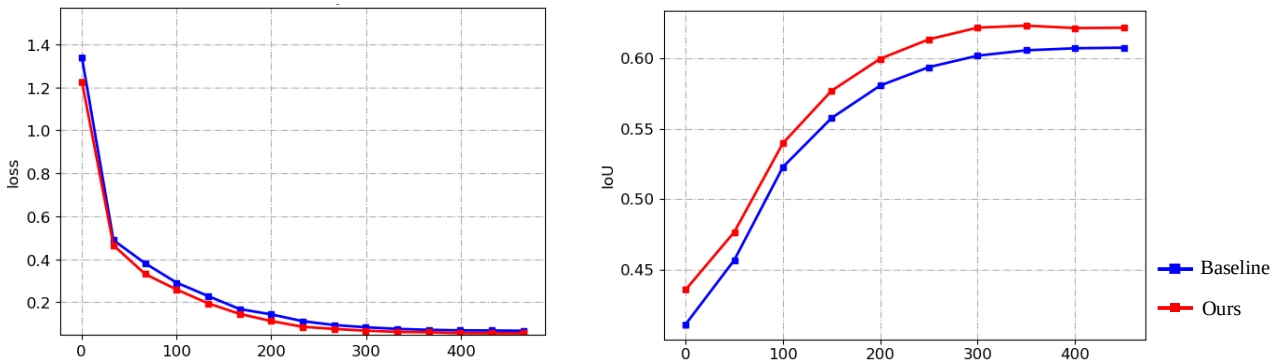
Figure 2, where we visualize the segmentation loss and validation curve of the networks. We can see that our network converges slightly faster than the baseline (i.e., 300 epochs VS. 400 epochs).

D. More Comparisons on S3DIS Benchmark

Table 4 presents the performance comparison with more state-of-the-art works on the S3DIS benchmark. Although our performance is not the best compared to Point Transformer [43], our work helps to gain insights into feature propagation in the decoding to improve commonly used networks. Our experiments and comparison have validated the effectiveness of the proposed feature propagation strategy.

E. More Qualitative Results

In this section, we present more qualitative results on both indoor and outdoor scenes. Figure 3 shows the visual results achieved on the SensatUrban dataset. Figure 4 and Figure 5 demonstrate the results of qualitative comparison on the S3DIS dataset adopting KP-Conv and PointNet++ as



(a) Segmentation losses (b) Validation curves

Figure 2: Comparison of network convergence. The horizontal axes denote the number of training epochs. The statistics are obtained on the S3DIS dataset using KP-Conv as the backbone, trained using 3D coordinates, color information and normals.

Method	OA(%)	**mIoU(%)**
PointNet [23]	-	41.1
TangentConv [31]	82.5	52.8
SPGraph [17]	86.4	58.0
BGENet [7]	-	61.4
RandLA-Net [12]	87.2	62.4
IAFNet [36]	88.4	64.6
JSENet [13]	-	67.7
Point Transformer [43]	90.8	70.4
PointNet++ [37] (w/o N)	83.5	53.6
+ ours	83.6	53.9
PointNet++ [37] (w/ N)	83.9	53.9
+ ours	84.3	55.1
KP-Conv rigid [32] (w/o N)	-	65.4
+ ours	89.7	67.1
KP-Conv rigid [32] (w/o N)	89.2	65.6
+ ours	89.6	67.2

Table 4: Semantic segmentation performance comparison with other state-of-the-art works on the S3DIS dataset. BGENet, IAFNet, and JSENet also consider boundaries. The results of RandLA-Net are from CBL paper [30].

the backbone, respectively. In all these comparisons, our guided feature propagation mechanism demonstrates consistent improvement in semantic segmentation.

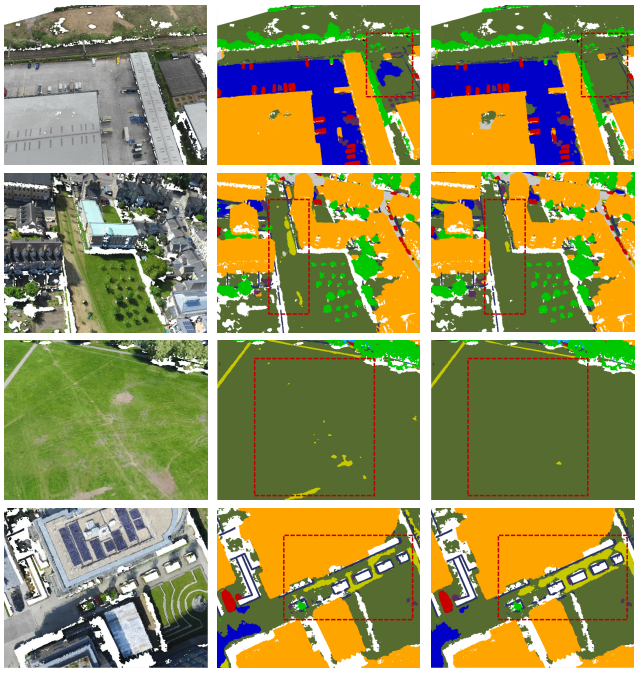


Figure 3: Qualitative results of semantic segmentation on the SensatUrban dataset, trained using 3D coordinates and colors. Our approach obtains more accurate boundaries (the 1st and 2nd rows). It also performs better in recognizing minor classes with distinct geometric boundaries such as footpaths (the 3rd and 4th rows).

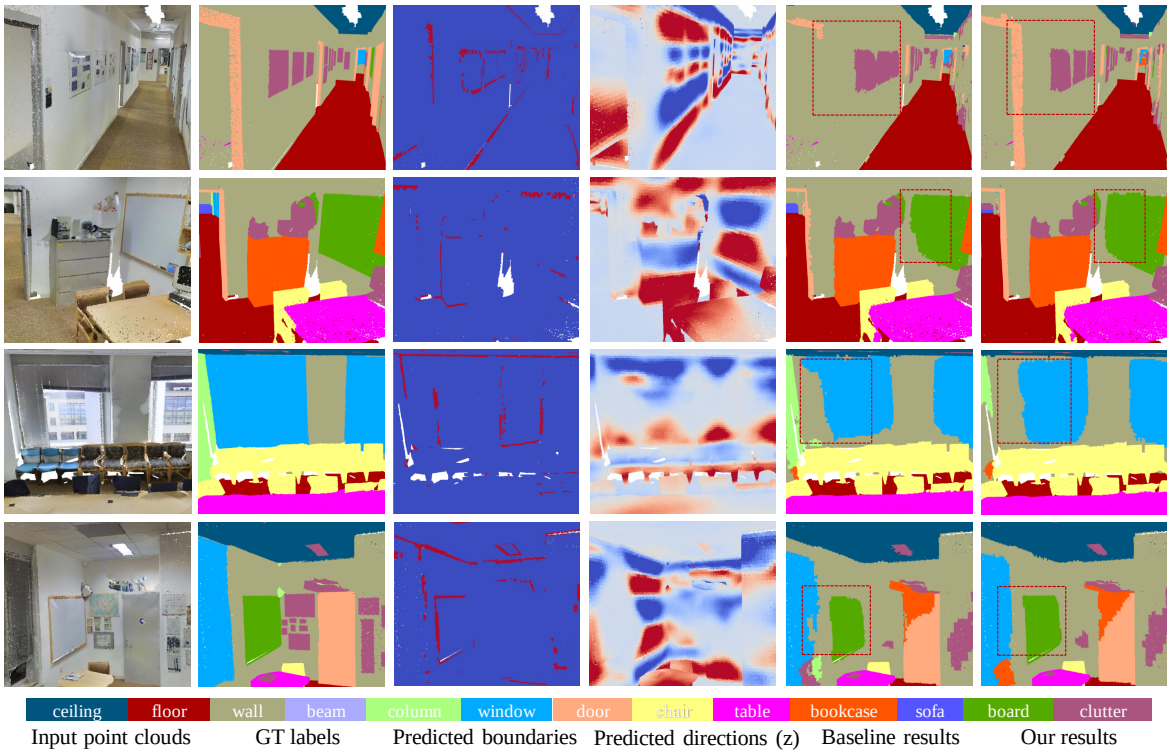


Figure 4: Qualitative results on the S3DIS dataset using KP-Conv as the backbone, trained using 3D coordinates, colors and normals. The binary boundary labels are visualized.

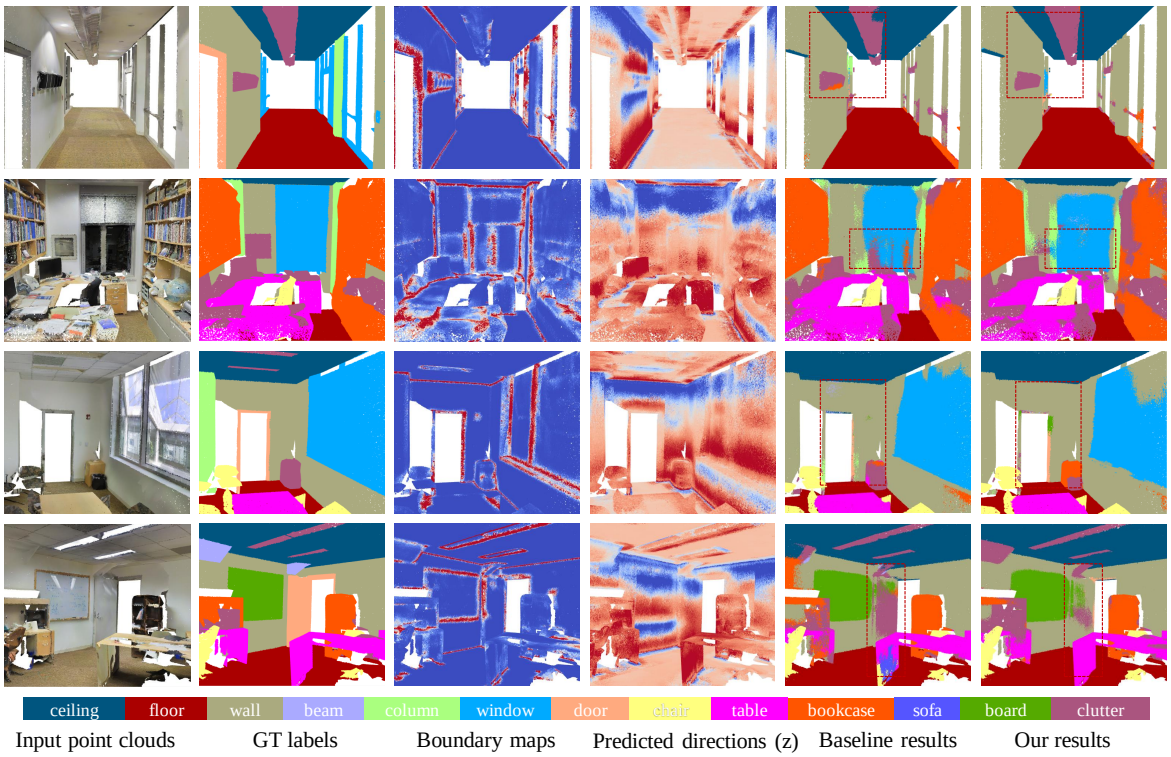


Figure 5: Qualitative results on the S3DIS dataset using PointNet++ as the backbone, trained using 3D coordinates, colors and normals. The boundary probability maps instead of binary labels are visualized: blue for *low*, and red for *high*.