University of Zurich[UZH]

VISUALIZATIONANDMULTIMEDIALAB

Learning Semantic Labeling of PTX scans

Master Thesis
Visualization and MultiMedia Lab
Department of Informatics
University of Zürich

by
Weiyi Wang
19-764-786

Supervisors:
Prof. Dr. Renato Pajarola
Lizeth J. Fuentes Perez

July 2022

Contents

Zusammenfassung v

Abstract vi

Acknowledgement vii

1 Introduction 1

2 Related Work 3

2.1 3D Semantic Segmentation . . . 3

2.2 Image Semantic Segmentation . . . 6

3 Approach 9

3.1 Point Cloud . . . 9

3.2 Networks . . . 11

3.2.1 U-Net . . . 11

3.2.2 FPN . . . 11

3.3 Loss Function . . . 12

3.3.1 Cross-entropy . . . 14

3.3.2 Focal Loss . . . 14

3.3.3 Dice Loss . . . 15

3.3.4 IoU . . . 15

4 Experiments 17

4.1 Datasets . . . 17

4.1.1 Matterport3D Dataset . . . 17

4.1.2 Structured3D Dataset . . . 19

4.1.3 2D-3D-S Dataset . . . 19

4.2 Experiment setup . . . 20

4.3 Experiment results . . . 21

4.3.1 Evaluation on Structured3D dataset . . . 21

4.3.2 Evaluation on 2D-3D-S dataset . . . 23

4.3.3 Network comparison . . . 25

4.3.4 Viewer Interface . . . 25

5 Discussion and Conclusion **27**
5.1 Limitations 27
5.2 Future work 28

Zusammenfassung

Semantische Informationen von 3D-Scans sind ein wichtiger Meilenstein für viele Applikationen zum Verstehen von Szenen, wie z.B. autonomes Fahren und 3D-Rekonstruktion. Das Ziel dieser Arbeit ist es, die semantische Segmentierung von Punktwolken in Innenräumen mit Hilfe von Deep-Learning-Techniken zu erhalten. Die semantische Segmentierung von 3D-Punktwolken mit Deep Learning steht noch vor einigen Herausforderungen. In dieser Arbeit wird Deep Learning nicht direkt auf 3D-Punktwolken angewandt, sondern es werden zuerst die Panoramabilder aus 3D-Punktwolken extrahiert. Diese Panoramabilder werden dann als Eingabe für ein CNN-Modell zur semantischen Segmentierung verwendet. Schließlich ordnen wir die erhaltenen Segmentierungspanoramen der 3D-Punktwolke zu und entwickeln dann eine Benutzeroberfläche zur Visualisierung der Punktwolke und der entsprechenden Panoramabilder. Deshalb kann die segmentierte Punktwolke in der Schnittstelle visuell analysiert werden.

Abstract

Semantic information of 3D scans is a fundamental step for many scene understanding applications such as autonomous driving and 3D reconstruction. The goal of this work is to obtain the semantic segmentation of indoor point clouds using deep learning techniques. While semantic segmentation with deep learning of 3D point clouds still faces several challenges. In this work, instead of directly using deep learning on 3D point clouds, we first extract the panoramic images from 3D point clouds. These panoramic images are then used as input to a CNN model for semantic segmentation. Finally, we map the obtained segmentation panoramas to the 3D point cloud and develop a viewer interface to visualize the point cloud and its corresponding panoramic images. Therefore, the segmented point cloud can be analyzed in the interface visually.

Acknowledgement

I am deeply grateful to all the people who supported me during this work. Firstly, I would like to thank my supervisor Lizeth F. Perez, who advised me at any time, supported me in case of problems, and searched for solutions together. Further, I want to thank Prof. Dr. Renato Pajarola and the University of Zurich for making this thesis possible.

Finally, I am also very grateful to my boyfriend and my friend Zhiwei Dou, who supported me throughout the process of the thesis.

1 Introduction

Rapid development in 3D acquisition technologies, such as LiDARs and RGB-D cameras, have accelerated the utilization of 3D data owing to its rich geometric, shape, and scale information. Compared with 2D images, 3D point clouds provide more geometric information and are the preferred representation for many scene understanding applications, including autonomous driving, augmented reality, remote sensing, and robotics. The semantic information of 3D scans plays a crucial role in scene understanding. Given the set of point clouds, the objective is to categorize each point into a set of semantic labels and cluster points with similar features in the same region. However, manually annotating point clouds is still tedious and time-consuming. This project, therefore, intends to mitigate the human effort by automatically providing the semantic labels of 3D scans.

In the last decade, machine learning techniques and deep neural networks, particularly, are the state-of-the-art for most segmentation tasks in computer vision. With their capability to learn fine and coarse details, convolutional neural networks have surpassed most of the other approaches for semantic segmentation. However, deep learning on 3D point clouds still faces significant challenges, such as the lack of annotated ground truth and the sparse and unstructured format of 3D point clouds. Most previous works convert point clouds into regular 3D voxel grids or collections of images and views before feeding them to deep neural networks, which is memory consuming and introduces artifacts.

For this reason, we propose an alternative approach that avoids the difficulties induced by semantic segmentation for 3D point clouds. We focus on the semantic segmentation of panoramic images using deep learning techniques since the semantic segmentation of 2D images has made considerable progress. We generate the panoramic images from point clouds in pre-processing and then use these panoramic images as input to the neural network. In addition, The PTX format has an ordered grid-like structure and maintains all the points of the grid. Thus, we can re-projected the segmentation results into 3D point clouds directly.

We investigate and perform experiments on several indoor datasets with U-Net and FPN. Our approach has two advantages over 3D semantic segmentation methods. Firstly, our method benefits from some existing datasets for image segmentation, which provide panoramic images that can be used directly for training. These panoramic images significantly reduce the need for 3D data for training purposes and eliminate the errors and artifacts introduced when extracting panoramas from 3D data. Secondly, 2D segmentation has lower memory complexity and better segmentation quality.

The main contributions of this work are listed below.

- Plan out the project and get familiarized with the current work.
- Investigate the indoor datasets and generate the panoramic images for training.
- Implement the neural networks for semantic segmentation.
- Train and test the networks on the panoramic images and then map the results into 3D point clouds.
- Develop a demo that loads a PTX point cloud and its corresponding panoramic images.

The remainder of this dissertation is organized as follows. Chapter 2 reviews the deep learning-based methods for 2D and 3D semantic segmentation. Chapter 3 presents the networks and the evaluation metrics utilized in this work. Then, we explain the datasets and demonstrate the results of our experiments in Chapter 4. The results are concluded and further discussed regarding their limitations and further improvements in chapter 5.

2 Related Work

Before deep learning was applied to computer vision, researchers used methods such as random forest and graph cuts for semantic segmentation. The design of convolutional neural networks introduced deep learning techniques to image segmentation and demonstrated outstanding performance. In recent years, deep neural networks have also been proposed for 3D point clouds segmentation. This chapter gives the literature review that covers deep learning-based methods for both 3D and 2D semantic segmentation. The first section 2.1 shows the deep learning methods for 3D data. Section 2.2 provides a brief overview of different state-of-the-art image segmentation methods.

2.1 3D Semantic Segmentation

For 3D semantic segmentation, the traditional methods include edge-based, region-based, graph-based, model-based, and machine learning-based methods, such as region growing, conditional random field, RANSAC, and support vector machines. Many deep learning methods derived from image segmentation and mesh segmentation have been used for point cloud segmentation recently. The 3D semantic segmentation with deep learning methods can be roughly divided into four categories: projection-based, discretization-based and point-based methods.

For projection-based methods, which benefit from 2D segmentation methods, many existing works aim to project 3D point clouds into multi-view and spherical images and then process 2D semantic segmentation. The final pixel-wise labeling is then re-projected to 3D point clouds. For example, the first work of this method was proposed by Lawin et al. [1], in which the point clouds were transformed into multi-view representations. In [2], Boulch et al. first generated RGB-D perspective images using multiple camera positions and training on 2D segmentation networks. Wu et al. [3] project the point clouds into spherical images and apply an end-to-end network based on SqueezeNet and Conditional Random Field (CRF). SqueezeSeg [4] and RangeNet++ [5] are further designed to improve the segmentation results. This method is scalable and can handle large-scale point clouds with millions of points. However, the performance of multi-view and spherical segmentation methods is sensitive to viewpoint selection and occlusions.

Discretization-based methods convert the point clouds into a dense or sparse discrete representation such as voxel. The point clouds are voxelized into 3D grids for dense representation and input to a 3D CNN. Huang et al. [6] is the pioneer of applying 3D convolutional neural networks on a set of occupancy voxels for voxel-wise segmentation. These intermediate data are generated from point clouds, and all points within a voxel are

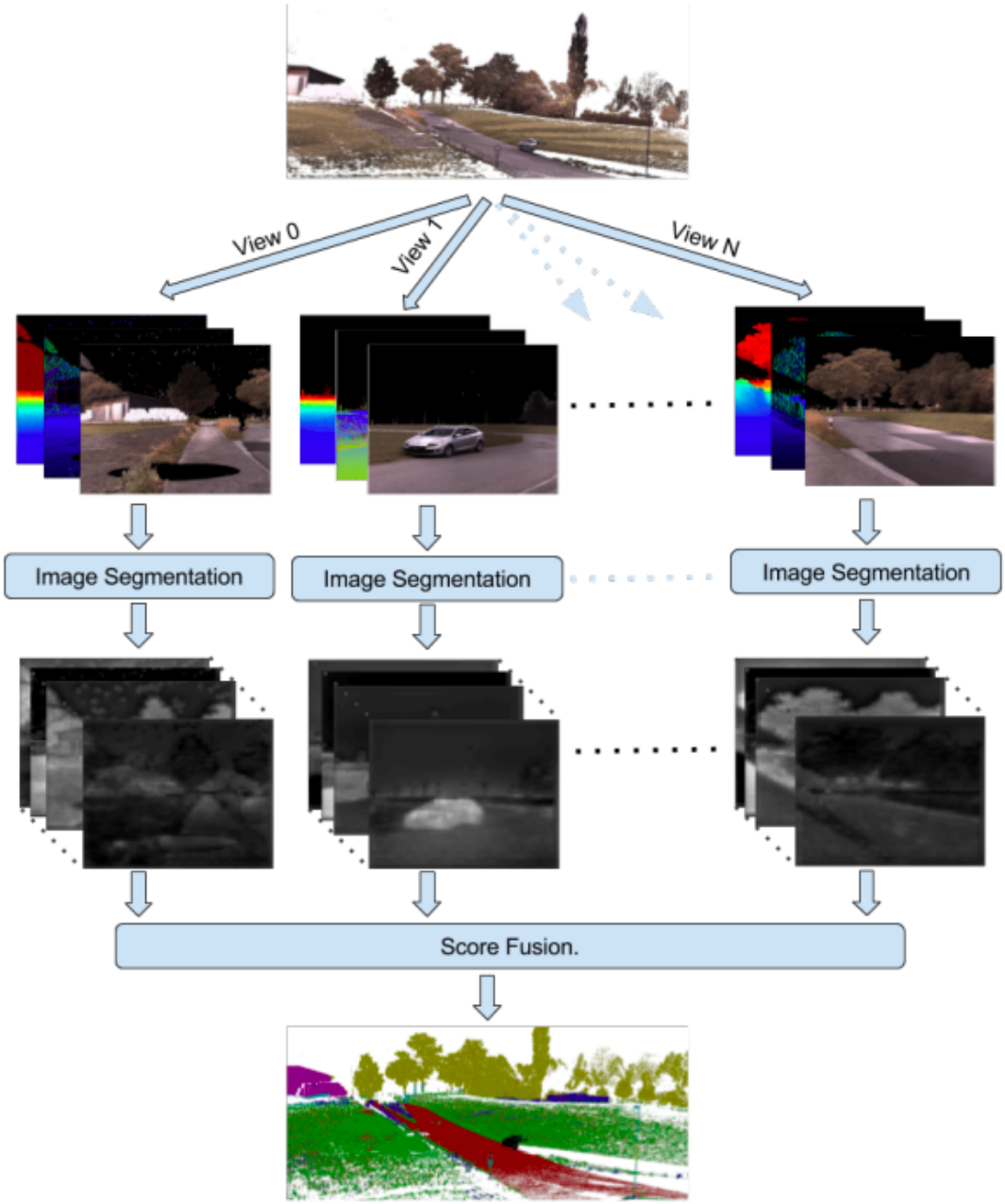
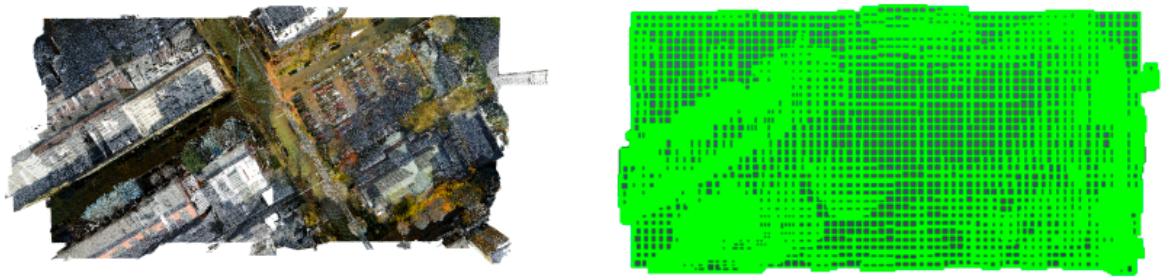


Figure 2.1: Multi-view projection methods from [1]. The input point cloud is projected into multiple virtual camera views with RGB color, depth and surface normal images.

assigned the same semantic label. Figure 2.2 shows the original input point cloud and the generated voxels. Fully-Convolutional Point Network (FCPN) [7] can process large-scale unorganized point clouds and produce



(a) Input point cloud (b) Dense voxel representation

Figure 2.2: Illustration for dense voxelization from [6]

an ordered structure or map predictions onto the input cloud. Dai et al. [8] proposed ScanComplete, which is able to feed incomplete 3D scan to a fully convolutional model and predict a complete 3D model with per-voxel semantic labeling. However, the performance of these volumetric representation methods is sensitive to the granularity of the voxels and computation expensive. Moreover, the voxelization introduces artifacts. The submanifold sparse convolutional networks developed by graham et al. [9] address the issues by restricting the output of convolution to be only related to occupied voxels. Other works, for instance, the Sparse Lattice Networks (SPLATNet) [10], and LatticeNet [11] interpolate a raw point cloud to a sparse lattice to process high-dimensional data effectively.

Unlike projection-based and discretization-based methods, the point-based methods directly work on the point clouds regardless of their orderless and unstructured nature. PointNet [12] proposed by Qi et al. is the first approach introduced to learn per-point features using Multi-Layer Perceptions (MLP) instead of convolution neural networks. The key of PointNet is a symmetric max pooling function used to reduce the vector of global features. Figure 2.3 illustrates the architecture of PointNet. The problem of PointNet is that it fails to capture local structures between neighboring points, which can lead to information loss when dealing with large amounts of data. Based on PointNet, a series of point-based networks have been proposed recently. PointNet++ [13], a hierarchical neural network architecture, improves the performance of PointNet by exploiting metric space distances. However, this model fails to provide features between sampling points. PointWeb [14] solves this problem by densely constructing a locally fully-linked web to extract contextual features from the local neighborhood. Hu et al. [15] describes how their network, RandLA-Net, utilizes random point sampling to achieve memory and computation efficiency for large-scale point cloud segmentation. To effectively apply convolution layers to point cloud segmentation, Hua et al. [16] proposed a point-wise convolution operator, which can be applied to each point of the point cloud. Wang et al. [17] proposed the Parametric Continuous Convolution (PCCN), which spans the continuous vector space with parameterized kernel functions.

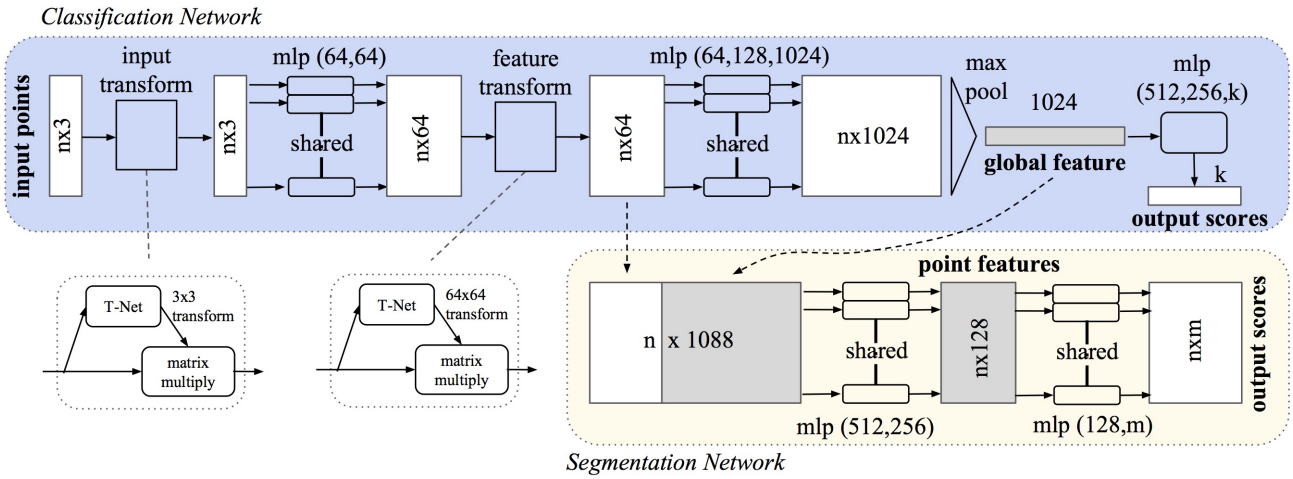


Figure 2.3: PointNet architecture from [12].

Several works tried to capture context features and geometric structures to achieve point cloud segmentation with Recurrent Neural Networks (RNN). Engelmann et al. [18] process input data with multi-scale blocks simultaneously and grid blocks to obtain input-level context. Then, the block features obtained from PointNet are sequentially fed into Consolidation Units (CU) or Recurrent Consolidation Units (RCU) to obtain output-level context. In addition, several methods have been combined together to learn semantic segmentation on point clouds. Dai et al. [19] presented 3DMV, a joint 3D-multi-view prediction network that combines the geometry and RGB data. The feature maps extracted from RGB images are projected into a volumetric feature grid and fused with 3D geometric features by a differentiable back projection layer. PGCNet proposed by Sun et al. [20] tackles the computational consuming problem by treating patches as input graph nodes and combines features of adjacent nodes via the dynamic graph U-Net (DGU) module.

2.2 Image Semantic Segmentation

In 2014, the first CNN-based models, the fully convolutional networks [21] proposed by Long et al. promoted the original CNN structure and performed dense prediction without a fully connected layer (Figure 2.4). This model is a milestone in image segmentation. The FCN can take an image of arbitrary size and outputs a spatial segmentation map of the same size by upsampling the features from the encoder using skip connection and bilinear interpolation. However, FCN consists of pooling layers, which reduce the size of the image as it passes through the network. To efficiently retain the position information discarded in the pooling layer, we need to upsample it using an interpolation technique, deconvolutional layers.

Noh et al. [22] introduced DeconvNet (Figure 2.5) on semantic segmentation based on deconvolution. Their model consists of an encoder using convolutional layers adopted from the VGG 16-layer network and a deconvolutional network composed of deconvolution and up-pooling layers that takes the feature vector as input and predicts segmentation masks. Another encoder-decoder structure is SegNet, proposed by Badrinarayanan et al.

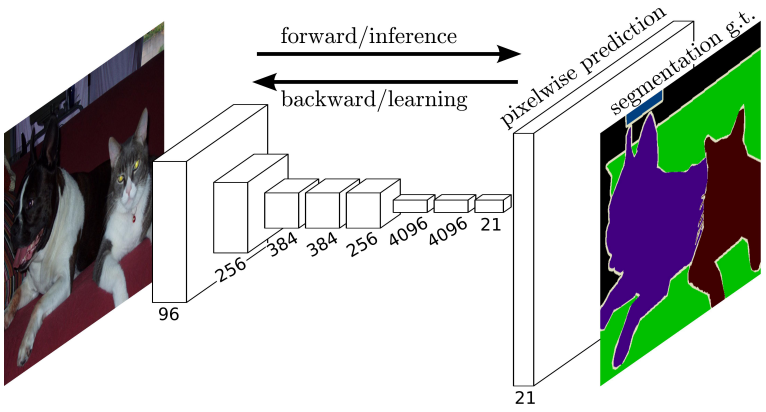


Figure 2.4: The fully convolutional network for image segmentation from [21].

[23], which is similar to DeconvNet [22]. The SegNet significantly reduces the number of trainable parameters with max-pooling indexed used in the decoder part. U-Net [24] and V-Net [25], are two typical encoder-decoder structures. Ronneberger et al. [24] proposed the U-Net, which concatenates the features from the encoder at the decoder using skip connections. There are various extensions of U-Net. For instance, Zhou et al. [26] developed a nested U-Net architecture, UNet++. V-Net, proposed by Milletari et al. [25] is designed for 3D medical image segmentation and used in other areas for segmentation. They introduced a new loss function based on the Dice coefficient, enabling the model to deal with class imbalance.

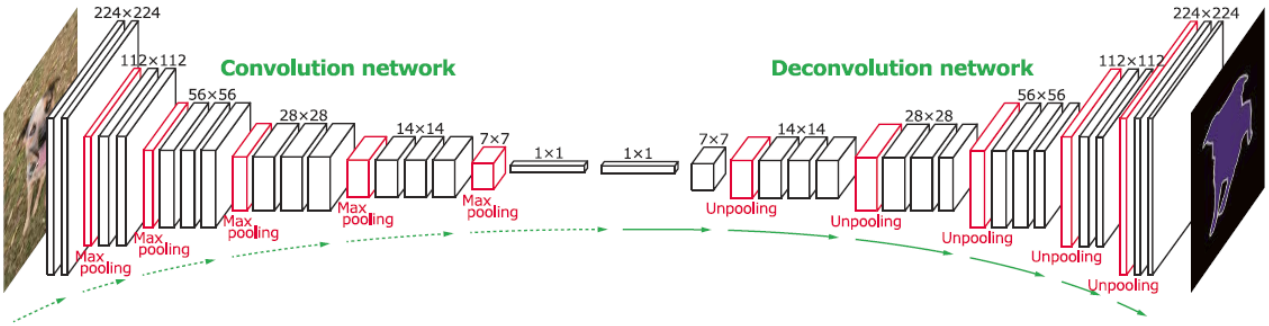


Figure 2.5: DeconvNet, a encoder-decoder structure for semantic segmentation from [22].

Another group of networks for segmentation is based on multi-scale analysis. One of the most well-known models is the Feature Pyramid Network (FPN) proposed by Lin et al. [27]. The pyramid hierarchy structure is designed to construct feature pyramids in deep learning. Zhao et al.[28] developed the Pyramid Scene Parsing Network (PSPN) in order to learn the global context representation of a scene. Other promising models, such as Adaptive Pyramid Context Network (APC-Net) [29], and Dynamic Multi-scale Filters Network [30] are also using multi-scale analysis.

Recent better-performing semantic segmentation CNN models based on dilated convolution introduce the dila-

tion rate to convolutional layers. This parameter defines a spacing between the weights of the kernel to reduce the parameters while enlarging the receptive field. Dilated convolutions, such as DeepLab family [31], have

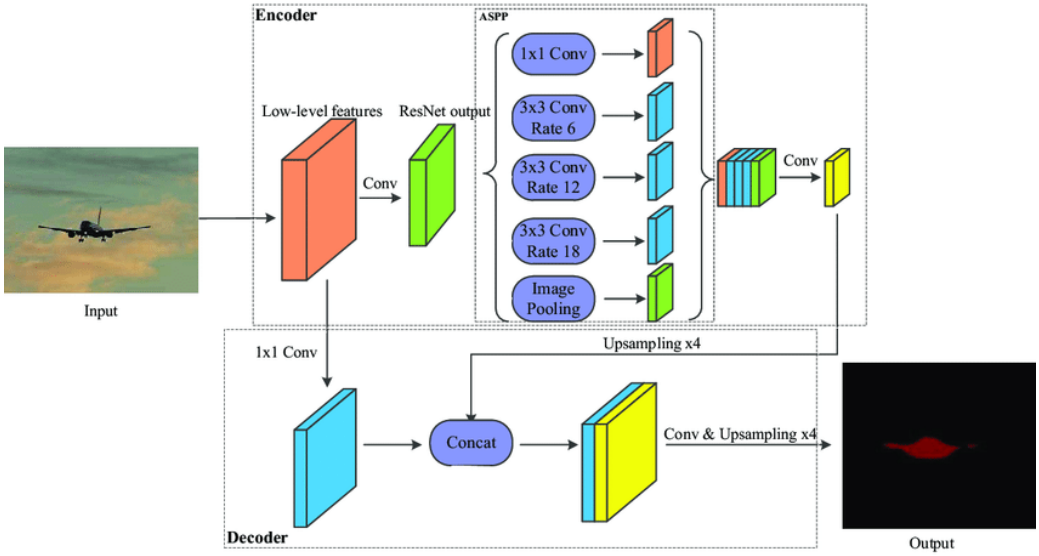


Figure 2.6: The DeepLab model from [31].

been popular in real-time segmentation, and many recent publications report the use of this technique. Figure 2.6 illustrate the DeepLab model.

Despite the CNN models, RNNs are made up of Long Short Term Memory (LSTM) blocks that model the short/long term dependencies among pixels. RNNs generate binary masks and the class probabilities sequentially to improve the prediction of the segmentation map. Visin et al. [32] presented an RNN-based architecture called ReSeg to model structure information of local generic features extracted from CNNs. The network is based on ReNet [33], and the ReNet layers are stacked on top of the output of the FCN. ReNet layers are then followed by upsampling layers to recover the original resolution of the image. Byeon et al. [86] developed a pixel-level segmentation and classification for scene labeling using two-dimensional long-short term memory (LSTM) recurrent neural network. Some networks combine RNN with CNN or FCN. The CNN is used for encoding, and LSTM is applied for decoding. One limitation of RNN-based models is that they will be slower than the CNN-based models since the sequential calculation cannot be parallelized easily.

3 Approach

The main objective of this project is to use machine learning techniques to extract the semantic labels for indoor panoramic images. The original data are PTX point clouds. Therefore, we first generate the 360-view panoramic images from the 3D PTX scans. The convolutional neural networks then take the generated panoramic images as input. The semantic outputs are mapped to 3D to reconstruct the annotated point clouds. Figure 3.1 illustrates the detailed pipeline of our work. Section 3.1 introduces the point clouds used for pre-processing. The networks for training the panoramic images are presented in Section 3.2. Section 3.3 provides several loss functions and the evaluation metrics which are widely used in image semantic segmentation.

3.1 Point Cloud

A point cloud is a collection of data points in space representing a 3D shape or object. In this work, the point clouds are PTX format, an ASCII-based format for saving point cloud data. PTX file has an ordered grid-like structure and maintains all the points in the grid, even those in shadows where no coordinates have been calculated. The point cloud data is stored in its coordinate system, and a transformation matrix for each point is provided in the header of the file. The first ten lines are the header of the PTX file, in which the first two lines are the number of rows and columns of the point cloud. Lines 3 to 6 of the header represent the position of the origin and primary axes (X, Y, Z) of the scanner coordinate system. The following four lines are the 4×4 transformation matrix, where the first three lines contain the rotation matrix and line 10 contains the translation vector. Each line following the header describes the information about one point in the cloud. Depending on whether or not the color information is saved, the stored information for each point is 4 (x, y, z, intensity) or 7 values (x, y, z, intensity, red, green, blue). The X, Y, Z coordinates are always in meters. The intensities use the decimal range [0, 1], and R, G, B values are in the range [0, 255].

The PTX files used in this work are indoor point clouds with RGB color information, and the coordinate of the missing points are represented as 0 0 0. Figure 3.2 illustrate the PTX point cloud being used. With the regular structure of the PTX file, extracting a panoramic image with a 360 view is straightforward. The points in the point cloud data are stored column by column (bottom to top), starting from the bottom left corner, which is the first point in the PTX file. The pixels of the generated panoramic images are inserted row by row (left to right), starting from the bottom left corner. Hence, it is possible to map the panoramic image of the scene with the 3D points directly. To access the point cloud and build the panoramic images correctly, we iterate over the points in the point cloud column by column and insert the current pixel at the transposed position. Figure 3.3 visualize the generated panoramic images corresponding to the point cloud in Figure 3.2.

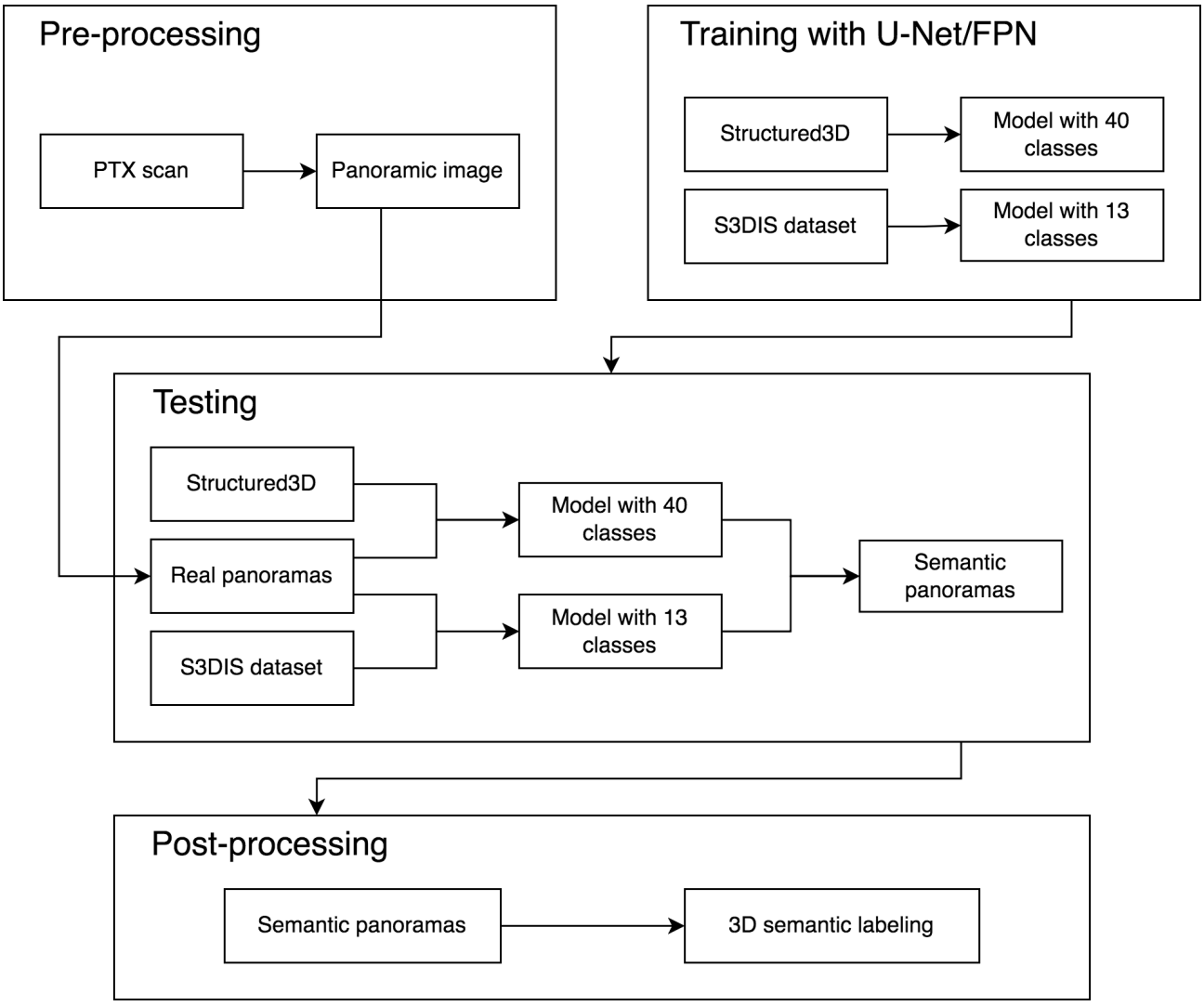


Figure 3.1: The pipeline of this work.

Figure 3.2: Example of the PTX point cloud being used.

Figure 3.3: Generated panoramic image for the PTX scan.

3.2 Networks

In the past few years, deep learning models have significantly improved the performance of image segmentation. Deep learning-based image segmentation models are far more accurate than traditional image segmentation algorithms, such as thresholding, clustering, graph cuts, and Markov random fields. Most of the neural networks that perform segmentation use an encoder-decoder architecture or a dilated network based on the fully convolutional network (FCN) [21]. This section introduces the neural networks used in this work for semantic segmentation.

3.2.1 U-Net

As the development of FCN, U-Net, proposed by Ronneberge et al. [24], was first developed for biomedical image segmentation using a symmetric U-shaped architecture. The network consists of a contracting path and an expansive path, which gives it the u-shaped architecture. The contracting path is a typical fully convolutional network that consists of repeated application of convolutions, each followed by a rectified linear unit (ReLU) and a max pooling operation. During the contraction, the spatial information is reduced while feature information is increased. The expansive pathway combines the feature and spatial information through a sequence of up-convolutions and concatenations with high-resolution features from the contracting path. The U-NET in the original paper is shown in Figure 3.4. The size of the input image in the original paper is 572x572. The size of our input images differs from that in the original paper, but the core components remain the same.

3.2.2 FPN

The feature pyramid network (FPN), developed by Facebook AI Research [27], is a feature extractor that presents a framework for building feature pyramids based on the convolutional neural network. FPN inputs

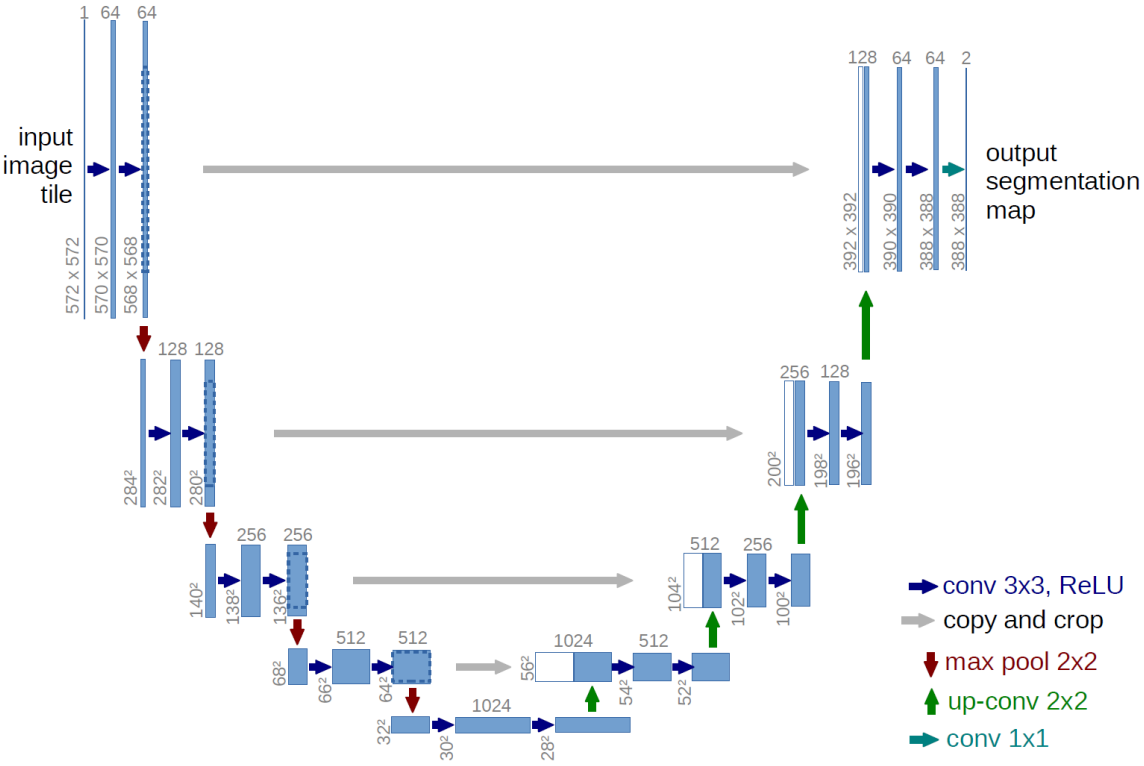


Figure 3.4: U-Net.

a single-scale image of arbitrary size and outputs proportionally sized feature maps at multiple levels. It constructs with a bottom-up pathway, a top-down pathway, and lateral connections. The bottom-up pathway resembles the encoder of the U-Net, which is the fully convolutional network where we downsample the image. The top-down pathway progressively upsamples feature maps from the deepest layer to generate higher resolution features and then merges the same-sized feature maps from the bottom-up pathway. In the lateral connections, FPN applies a 1×1 convolution layer to the feature maps from the bottom-up pathway to reduce the channel dimensions, whereas U-Net simply copies and combines the feature maps. The main difference between U-Net and FPN is that FPN has multiple prediction layers. That is, each upsampling layer makes predictions using a 3×3 convolution in the top-down pathway. The outputs for all prediction layers have 256 channels. The pyramid has scales from 1/32 to 1/4 resolution, each of which shares the classifiers.

3.3 Loss Function

Deep Learning algorithms optimize and learn the objective using gradient descent approaches. A well-designed loss function enhances the precision and convergence of deep learning models. The most commonly used loss functions for segmentation are based on either the cross-entropy loss, Dice loss [25] or a combination of the two. In this section, we introduce the loss functions and evaluation metrics that are widely used for image segmentation, especially when class imbalance is present.

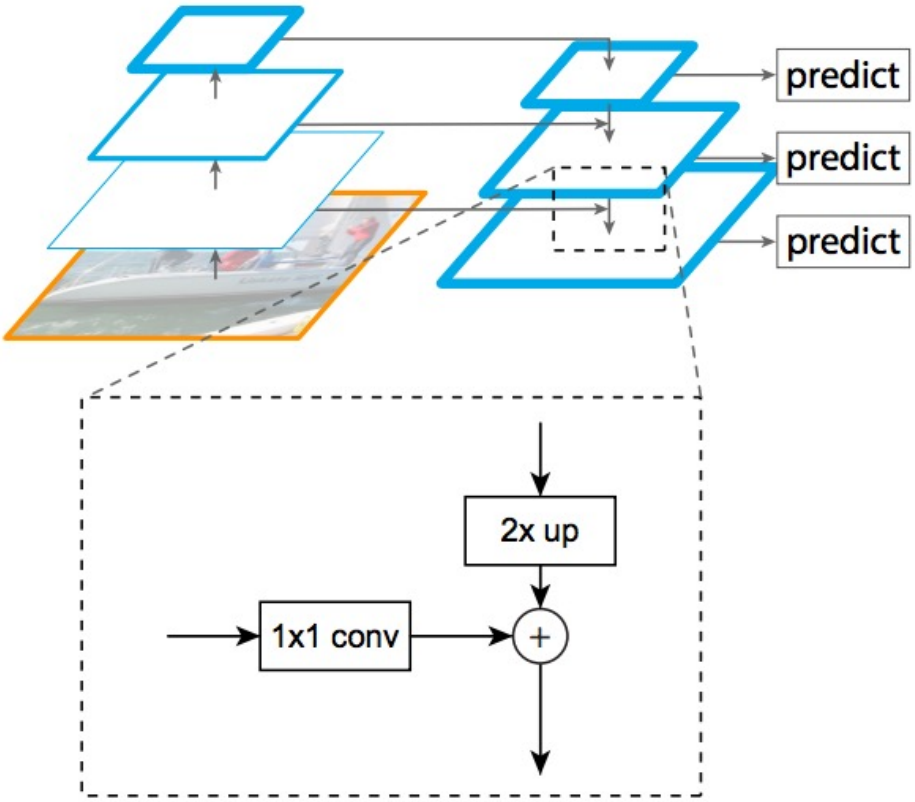


Figure 3.5: Feature pyramid network.

3.3.1 Cross-entropy

The cross-entropy loss measures the difference between the actual class and the predicted probability distributions or labels. For binary classification, the binary cross entropy loss is defined as follows:

$$\mathcal{L}_{BCE}(y, \hat{y}) = -[ylog(\hat{y}) + (1-y)log(1-\hat{y})], \tag{3.1}$$

where y represents the ground truth label and $\hat{y}$ represents the predicted label, $y, \hat{y} \in \{0,1\}^N$. The multi-class categorical cross entropy loss is given by:

$$\mathcal{L}_{CCE}(y, p) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}(y_{i,c} \cdot log(p_{i,c})), \tag{3.2}$$

where $y_{i,c}$ is the one-hot encoded ground truth labels, $p_{i,c}$ is the predicted probability for each class, and the C and N are the total number of classes and pixels, respectively. This loss function calculates the losses for all the pixels and assigns equal weights to each category, which leads to biased training results in class imbalance problems, particularly in the field of image segmentation. The neural network trained using this loss function intends to output the classes of large objects over the classes of small objects. The weighted cross-entropy loss addresses this problem by introducing more weights to the less dominant classes. It is computed as:

$$\mathcal{L}_{W-BCE}(y, \hat{y}) = -[w \cdot ylog(\hat{y}) + (1-y)log(1-\hat{y})], \tag{3.3}$$

where w represents the weight. Set $w > 1$ to decrease the number of false negatives and $w < 1$ to decrease the number of false positives. However, the weights need to be manually adjusted, making tuning more difficult.

3.3.2 Focal Loss

The Focal loss [34] is a variant of the binary cross entropy loss by adding a modulating factor to address the issue of class imbalance. It focuses on training the hard examples and down-weights the easy examples. For binary cases, to derive the Focal loss, we write the cross-entropy as:

$$CE(p, y) = \begin{cases} -log(p), & if\ y = 1 \\ -log(1-p), & if\ y = -1. \end{cases} \tag{3.4}$$

The estimated probability is:

$$p_t = \begin{cases} p, & if\ y = 1 \\ 1-p, & if\ y = -1. \end{cases} \tag{3.5}$$

We can then rewrite the cross-entropy as $CE(p, y) = CE(p_t) = -log(p_t)$. Hence, the Focal loss with the modulating factor $(1-p_t)^\gamma$ can be defined as:

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma log(p_t) = -\alpha_t(1-p_t)^\gamma \cdot \mathcal{L}_{BCE}, \tag{3.6}$$

where $\gamma \in [0, 5]$ is the focusing parameter and $\alpha \in [0, 1]$ is a parameter to balance the class weights. $\mathcal{L}_{BCE}$ is the binary cross-entropy loss. When $\gamma = 1$, the Focal loss is the cross-entropy loss. For multi-class segmentation, the categorical Focal loss is:

$$FL(p_{t,c}) = -\alpha(1 - p_{t,c})^{\gamma} \cdot \mathcal{L}_{CCE}, \tag{3.7}$$

where $p_{t,c}$ is the estimated probability for each class c and $\mathcal{L}_{CCE}$ is the categorical cross-entropy loss.

3.3.3 Dice Loss

Dice loss is a region-based loss function that is determined by the Dice similarity coefficient (DSC), a metric for evaluating the similarity of two samples. It takes on values between 0 and 1, with higher values indicating more similarity. The Dice's coefficient is two times the area of the intersection of the prediction and the ground truth, divided by the sum of the areas of the prediction and the ground truth.

$$D = \frac{2|A \cap B|}{|A| + |B|}, \tag{3.8}$$

where $|A|$ and $|B|$ are the number of elements in each set. The Dice similarity coefficient is also known as the F1 score, which can be defined in terms of true positives (TP), false positives (FP), and false negatives (FN):

$$D = \frac{2TP}{2TP + FP + FN}. \tag{3.9}$$

The dice loss is calculated as:

$$\mathcal{L}_{DSC} = 1 - DSC = 1 - \frac{2|A \cap B| + \epsilon}{|A| + |B| + \epsilon}. \tag{3.10}$$

Here, ϵ is added as a smoothing factor in order to smooth the loss and gradient and ensure the function is not divided by 0.

The dice loss focuses more on learning the foreground region and is adapted to handle class imbalance. The loss calculated for a fixed size region of positive samples is the same regardless of the image size. However, the training loss is unstable, particularly in the case of small objects.

3.3.4 IoU

Intersection over Union (IoU) [35], known as the Jaccard index, is one of the most popular evaluation metrics for tasks such as segmentation, object detection, and tracking. It computes the number of pixel overlaps between the predicted outputs and the ground-truth masks divided by the total number of pixels found in both the prediction and the target.

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{|I|}{|U|}, \tag{3.11}$$

where A and B are the prediction and ground truth segmentation. The IoU score is calculated separately for each class and then averaged over all classes to yield a global mean IoU score for our semantic segmentation prediction. In segmentation tasks, IoU is used as an evaluation metric but not a loss function as it is not differentiable. Figure 3.6 shows the IoU and Dice similarity coefficient.

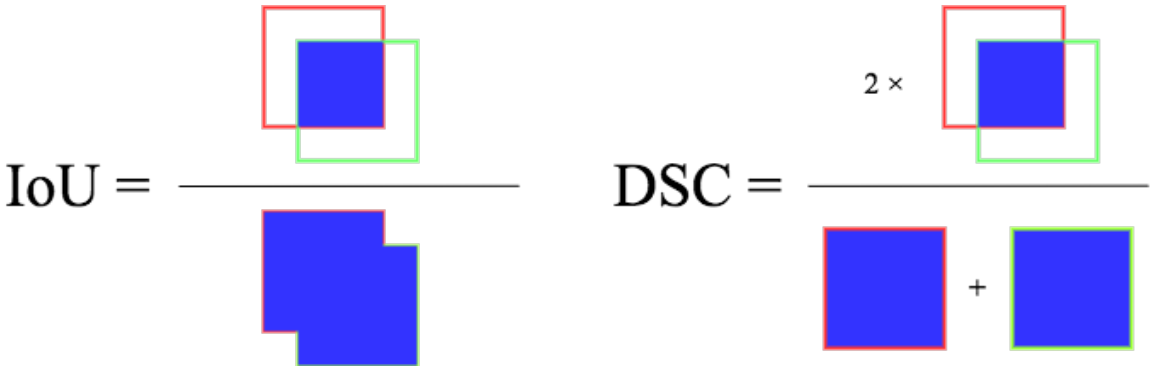


Figure 3.6: Comparison between IoU and DSC.

4 Experiments

4.1 Datasets

In this section, we introduce the datasets and the pre-processing steps we have applied for each dataset. Since we train the models using panoramic images, the datasets need to contain the labeled 3D meshes or labeled point clouds, from which we could extract the panoramas. In addition, datasets containing just indoor panoramic images and their corresponding semantic segmentation images can be utilized directly for training. However, these dataset suffers when mapping the segmented images to point clouds without 3D information. The number of datasets which contain both the labeled 3D data, the panoramic images and the semantic segmentation panoramas are limited. We have investigated a number of datasets, including the Matterport3D dataset [36], Structured3D dataset [37], 2D-3D-S dataset [38] and ScanNet dataset [39]. Due to the fact that each dataset comprises distinct data types, the pre-processing applied to these datasets varies.

4.1.1 Matterport3D Dataset

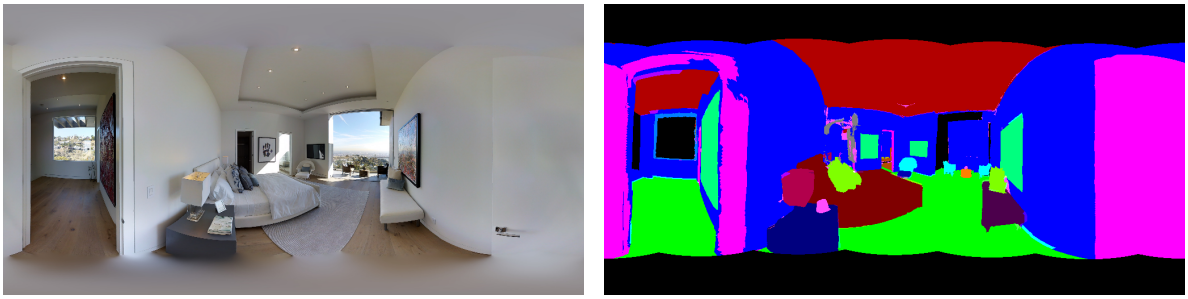
The Matterport3D dataset is a large-scale indoor RGB-D dataset consisting of 10,800 panoramic views. Their scans are from 90 distinct buildings, including offices, homes and other types of scenes. The data provided directly are 90 3D textured meshes for each building, along with RGB-D images and skybox images.

The RGB-D images for each panorama are small images taken by a Matterport camera from 18 views (horizontal, upward and downward views with 60 degree's rotation around the direction of gravity). The matterport3D dataset has 40 semantic categories which are different from the most common object category: NYUv2 40 labels. Figure 4.1 shows an example mesh and its corresponding semantic segmentation provided by Matterport3D dataset.

For pre-processing, the first step is to stitching the 18-views small images into a single panoramic image. Then extract the semantic images from the 3D meshes and combine them into semantic panoramas. Figure 4.2(a) and Figure 4.2(b) visualize the generated panorama and corresponding semantic panorama respectively. However, the resulting semantic panoramic images have some serious drawbacks. A number of segmented panoramic images has large black areas (outdoor scenes), artifacts, noise as well as mislabeled objects, all of which have significant impact on the training. Furthermore, there is no color label mapping file provided, making it difficult to match colors to semantic labels.

(a) 3D house mesh (b) 40 labels semantic segmentation

Figure 4.1: Semantic voxel label segmentation provided by Matterport3D dataset.



(a) RGB panorama (b) Semantic panorama

Figure 4.2: Panoramic images generated for Matterport3D dataset.

4.1.2 Structured3D Dataset

The Structured3D dataset is a synthetic dataset containing 21,835 rooms in 3,500 scenes, and more than 196k panoramic image. The rooms are manually designed by professional designers that are subsequently used to render the panoramas. For each room, they offer panoramic images with different amounts of furniture and lighting effects. The Structured3D dataset uses NYUv2 40 labels.

Compared to the real datasets, the synthetic datasets are consistent, free of noise and artifacts. However, these datasets lack annotated ground truth structures. We can only reconstruct the 3D room layout from the panoramas in the Structured3D dataset, while the objects can not be projected into 3D without depth information. Therefore, we use the Structured3D dataset for training. In this work, we use around 10,000 pairs of ground-truth panoramas and the corresponding semantic panoramas from the fully furnished rooms. See Figure 4.3(a) and Figure 4.3(b) for a visualization. For convenience, the panoramic and semantic images of the fully equipped rooms are extracted into two folders before training.

(a) RGB panorama (b) Semantic panorama

Figure 4.3: Panoramic images provided by Structure3D dataset.

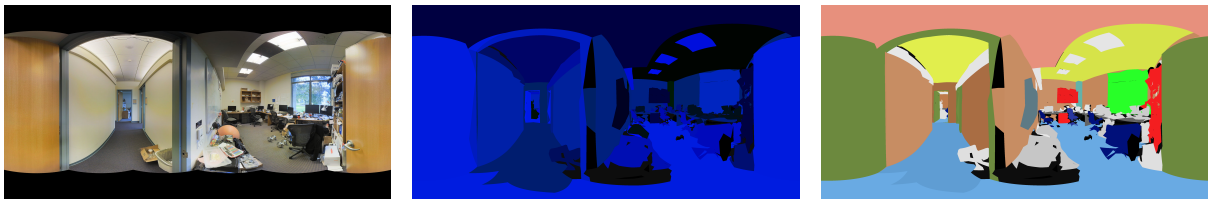
4.1.3 2D-3D-S Dataset

The Stanford 2D-3D-Semantics Dataset (2D-3D-S) provides semantically annotated panoramic images, 3D meshes (Figure 4.4) and 3D point clouds (S3DIS Dataset) from 6 large-scale indoor scenes. The 6 area scans are separated into different rooms, each represented by a single annotated point cloud. The panoramic data contains 1,413 panoramic images (Figure 4.5(a)), their corresponding semantic annotations (Figure 4.5(c)) and global XYZ files, which represent the ground truth location of each pixel in the meshes and can be used in the 3D reconstruction tasks after training. The S3DIS dataset has 13 categories, which are ceiling, floor, wall, beam, column, window, door, table, chair, sofa, bookcase, board and clutter.

We also utilize this dataset for training, since using synthetic data for training and real data for testing can significantly change the test performance. We use panoramic images in area 5 for testing and train our networks on the rest. For pre-processing, we extract the labels for semantic images into npy files via the provided json file, which consists of the labels, instance number, room type, room number and area number for each class.

Because some unclassified black regions on the top and bottom of the panoramic images are annotated as floor or ceiling in the semantic images, we encode these areas as ”unknown” to prevent them from impacting the training results. Creating the resulting images is difficult because of the lack of color mapping file. The testing semantic results are mapped to the relevant NYUv2 label colors for better visualization. 3.4.

Figure 4.4: 3D textured mesh provided by 2D-3D-S dataset.



(a) RGB panorama (b) Semantic panorama for training (c) Semantic panorama for visualization

Figure 4.5: Panoramic images provided by 2D-3D-S dataset. Note that the semantic panorama in the middle is used for learning and the panorama in the right is for better visualization.

4.2 Experiment setup

The architecture is implemented with Pytorch and runs on NVIDIA GeForce RTX 2080 GPUs and NVIDIA Tesla K80 GPUs on Science Cluster. We evaluate the performance of the model using the dice similarity coefficient (DSC) and the mean IoU as described in section 3.3. Since the results trained on the Matterport3D dataset are poor, we conducted several training experiments on the Structured3D and 2D-3D-S datasets with U-Net and FPN and two loss functions, dice loss and focal loss. The encoders for these two models are the resnet50. Both models are trained for 60 epochs with an initial learning rate of 0.001 and a batch size of 2. We partition the training data into 80% training set and 20% validation set. For the Structured3D dataset, the panoramic images and segmentation masks are provided in a png file format with size of 512×1024. For the 2D-3D-S dataset, the provided panoramic images and segmentation masks are both png file formats with

size of 2048×4096. The labels of the masks are transformed to npy file format in pre-processing. Before training, we resize the panoramic images to 256×512 for U-Net and 512×1024 for FPN due to the limitation of GPU memory. Table 4.1 summarise the settings for the dataset. The images are augmented with spatial and RGB augmentation techniques such as flipping, random cropping, and random brightness using the library albumentations.

Dataset	Size	#Training images	Resize	
			U-Net	FPN
Structured3D	512x1024	9,688	256x512	256x512
2D-3D-S	2048x4096	1,040	256x512	512x1024

Table 4.1: Parameters of the dataset.

4.3 Experiment results

This section describes the results of our U-Net and FPN using the Structured3D dataset, followed by the results of the 2D-3D-S dataset and the quantitative evaluations. To visualize the PTX scans and the semantic results, we implement a viewer interface which is introduced in the last part of this section.

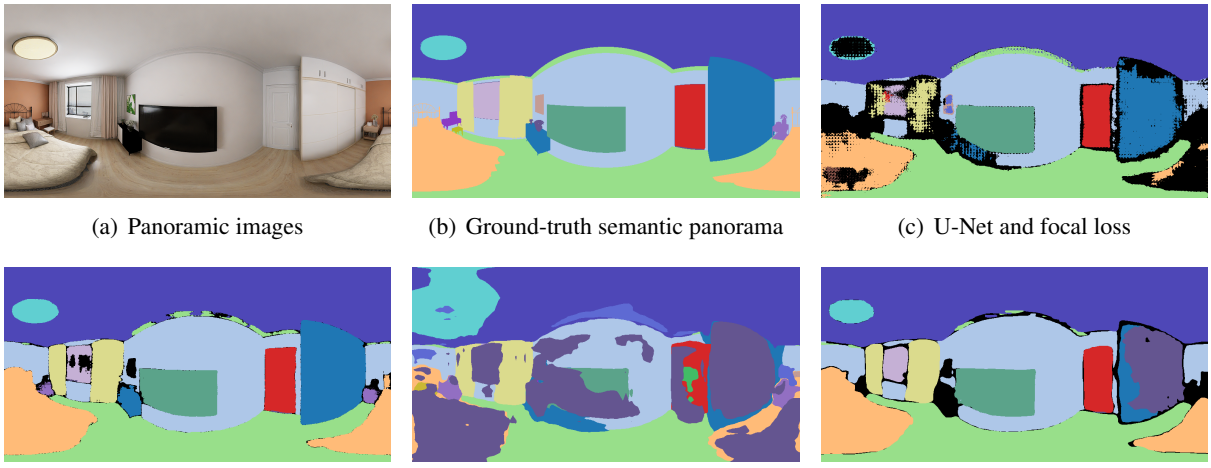
4.3.1 Evaluation on Structured3D dataset

We investigate how the performance of the semantic results changes when different networks and loss functions are applied. As shown in Table 4.2, the best performance is observed using U-Net with the dice loss. The Qualitative comparison of the models trained on the Structured3D dataset is shown in Figure 4.6. As observed from the figures, both networks with dice loss outperform those using focal loss. Since the Structured3D dataset has 40 categories and some categories are defined as otherstructure, otherfurniture and otherprop, we find that with some objects such as closet annotated as otherprop in some images, the models tend to make wrong decisions when classifying these objects. As shown in Figure 4.7, the shelving closet is annotated as otherprop (color purple) in the ground-truth semantic panoramas, while the U-Net segments it as the closet (color blue) and a mixture of both for FPN, which is the main reason for the low average IoU score. We also find that FPN has some small gaps between different objects opposite to the U-Net. The expected results of focal loss are supposed to be better than dice loss, as it is more stable in dealing with class imbalance issues. The opposite results may be due to the fact that the model is overwhelmed by the multiple annotations of the same kind of objects as described above.

To test the panoramic images generated from PTX scans, we use the model with the best performance, that is, the U-Net with dice loss. However, testing the image generated from an actual scan using a model trained on synthetic data changes the performance significantly. Figure 4.8 visualizes the testing result, from which we can find that the semantic prediction is noisy and misclassified.

Network	U-Net + Focal loss	U-Net + Dice loss	FPN + Focal loss	FPN + Dice loss
Avg IoU	0.389	0.439	0.317	0.435
DSC	0.779	0.877	0.633	0.871

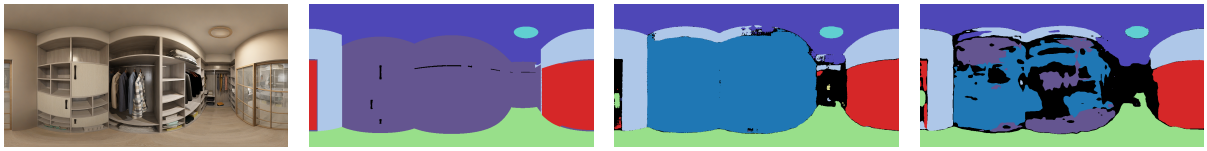
Table 4.2: The quantitative result of U-Net and FPN on Structured3D dataset.



(a) Panoramic images (b) Ground-truth semantic panorama (c) U-Net and focal loss

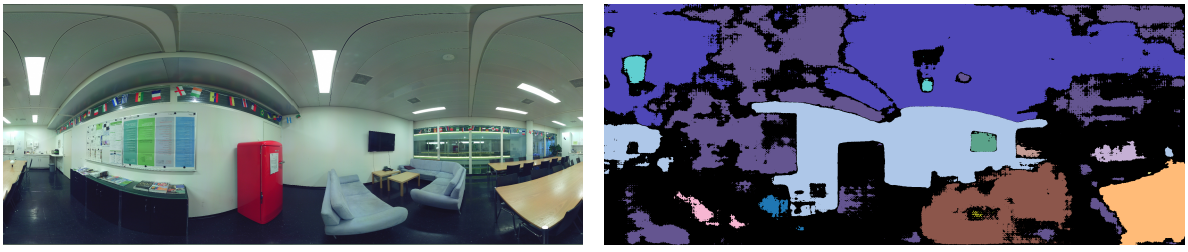
(d) U-Net and dice loss (e) FPN and focal loss (f) FPN and dice loss

Figure 4.6: Predicted semantic results for Structured3D dataset.



(a) Panoramic image (b) Semantic panorama (c) U-Net and dice loss (d) FPN and dice loss

Figure 4.7: Examples of misclassified object on Structured3D dataset.



(a) Panoramic image (b) Semantic panorama

Figure 4.8: Predicted semantic result for PTX scan with the model trained on Structured3D dataset.

3D-2D-S	NYUv2	Color
ceiling	ceiling	(78, 71, 183)
floor	floor	(152, 223, 138)
wall	wall	(174, 199, 232)
beam	other furniture	(82, 84, 163)
column	other structure	(94, 106, 211)
window	window	(197, 176, 213)
door	door	(214, 39, 40)
table	table	(255, 152, 150)
chair	chair	(188, 189, 34)
sofa	sofa	(140, 86, 75)
bookcase	bookcase	(148, 103, 189)
board	whiteboard	(178, 127, 135)
clutter	otherprop	(100, 85, 144)

Table 4.3: The color mapping of 3D-2D-S dataset for visualization purpose.

4.3.2 Evaluation on 2D-3D-S dataset

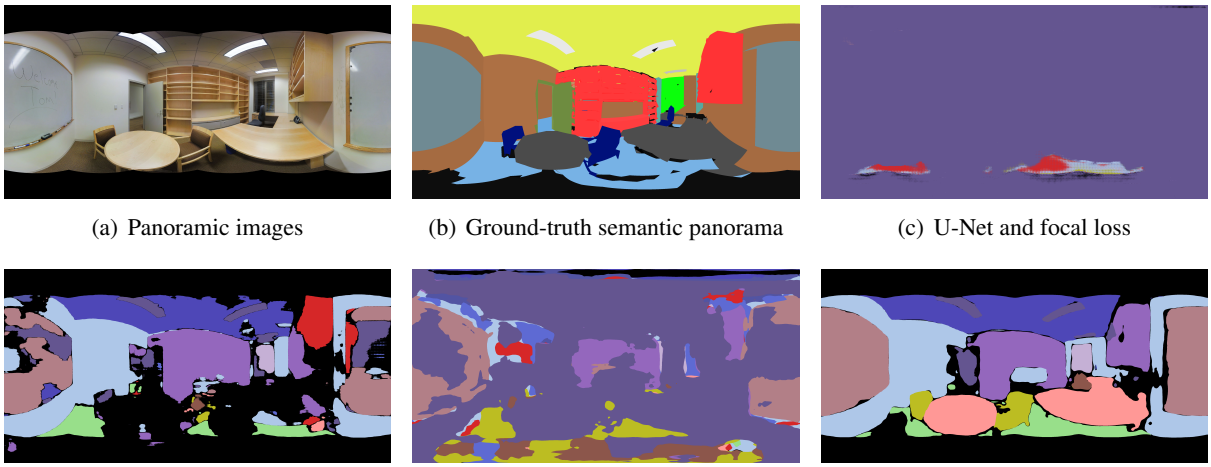
For the 2D-3D-S dataset, due to the lack of color mapping, the testing panoramic images are colored with the corresponding colors of the NYUv2 classes to have consistent colors for visualization purpose. For the categories not included in the NYUv2 classes, we map them into a random color in the NYUv2 color map. Table 4.3 shows the color mapping for 2D-3D-S dataset. Here, we have 13 classes, and the quantitative result of the U-Net and FPN on the 2D-3D-S dataset are shown in Table 4.4. When training with real scan data, the performance of our models decreased significantly. FPN with the dice loss achieves the best IoU and DSC scores for this dataset. The semantic results of the models trained on the 2D-3D-S dataset are illustrated in Figure 4.9. Similar to the Stuctured3D dataset, networks with dice loss outperform the networks using focal loss. The U-Net and FPN using focal loss fail to learn the dataset and have large areas recognized as clutter. The FPN achieves better performance in both the segmentation quality and the computational effectiveness for the 2D-3D-S dataset.

The model for testing the panoramic images generated from PTX scans utilized is the FPN with dice loss. As shown in Figure 4.8, the semantic prediction performs better than the model trained on the Structured3D dataset, and even the Structured3D model has higher IoU and DSC scores. The reason behind this is that the images in the 2D-3D-S dataset are generated from real scans. The 3D semantic segmentation is shown in Figure 4.11.

In general, semantic segmentation of indoor scenes is complex because some categories are hard to distinguish, such as whiteboards on white walls and tiny objects. In addition, for real scan data, the semantic masks are noisy and have irregular object shapes. The number of classes and the available images for training is also limited and crucial for the performance of the model. Note that we are training with the basic segmentation networks. Consequently, the results can be improved further, which will be introduced in the next chapter.

Network	U-Net + Focal loss	U-Net + Dice loss	FPN + Focal loss	FPN + Dice loss
Avg IoU	0.099	0.280	0.166	0.384
DSC	0.199	0.561	0.332	0.767

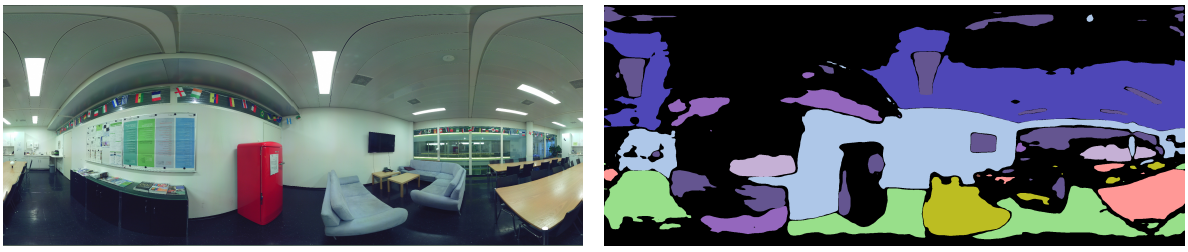
Table 4.4: The quantitative result of U-Net and FPN on 2D-3D-S dataset.



(a) Panoramic images (b) Ground-truth semantic panorama (c) U-Net and focal loss

(d) U-Net and dice loss (e) FPN and focal loss (f) FPN and dice loss

Figure 4.9: Predicted semantic results for 2D-3D-S dataset.



(a) Panoramic image (b) Semantic panorama

Figure 4.10: Predicted semantic result for PTX scan with the model trained on 2D-3D-S dataset.

Figure 4.11: 3D semantic segmentation.

4.3.3 Network comparison

The size of U-Net with resnet50 as the encoder and input size of $3 \times 512 \times 1024$ has 82,148,776 parameters, while FPN has 26,121,064 parameters. As shown in Table 4.5, the average training time per epoch of U-Net and FPN on the structured3D dataset using the same GPU and parameter settings is 3.6h and 3h, respectively. In comparison, FPN is lighter and faster than U-Net. Therefore, we can use a larger batch size or image size when training with FPN. The average training time per epoch for the 3D-2D-S dataset with U-Net and FPN is almost the same. However, the images for U-Net are down-sized to 256×512, whereas the panoramas are resized to 512×1024 for FPN.

Netwrok	Structured3D		2D-3D-S	
	Training time/epoch	Validation time/epoch	Training time/epoch	Validation time/epoch
U-Net	1.5h	1h	20min	13min
FPN	1h	1h	12min	13min

Table 4.5: Training and validation time for U-Net and FPN on the Structured3D and 2D-3D-S dataset.

4.3.4 Viewer Interface

We design a viewer interface for better visualization of the 3D point clouds, and the panoramic images based on the Easy3D library [40]. Due to the fact that the Easy3D library can only visualize 3D scenes, we load the texture from the image and create a mesh with a single quad to show it in the interface. Figure 4.12 shows the viewer interface, where the point cloud is shown in the main window, and the corresponding panoramic image is in a new window.

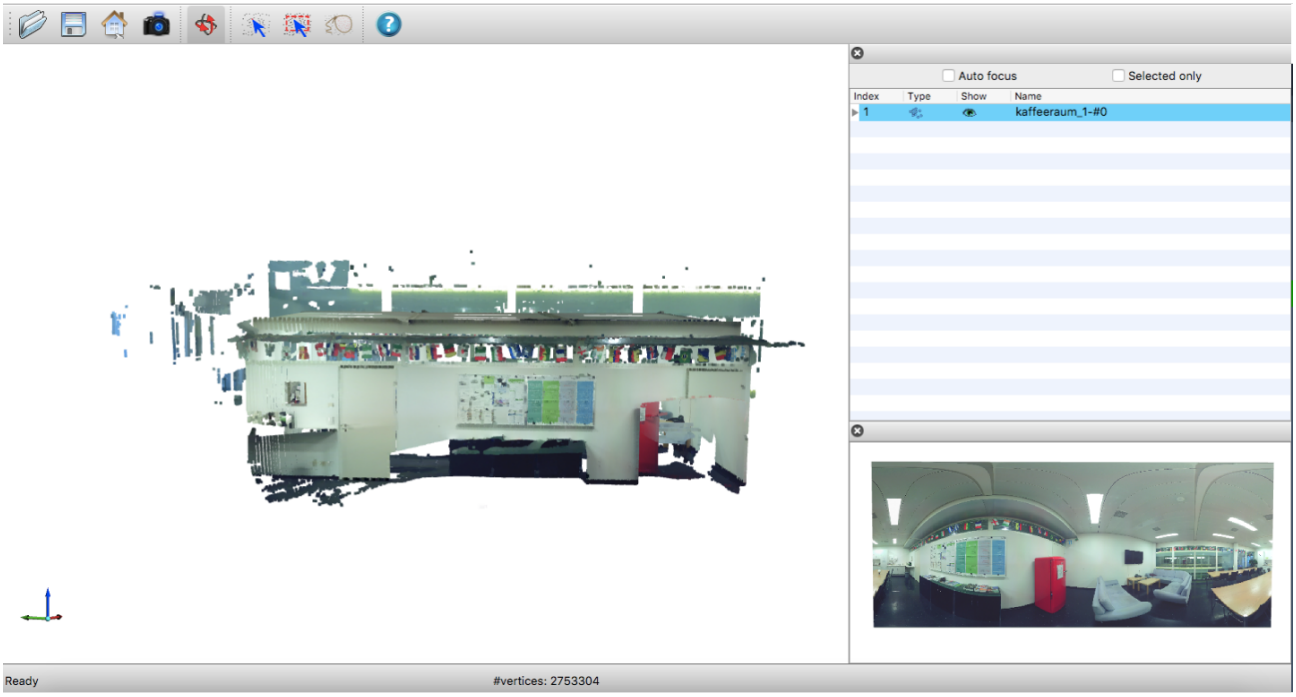


Figure 4.12: Viewer interface.

5 Discussion and Conclusion

Semantic labeling of 3D point clouds with 3D data or overlapping perspective images is computationally expensive, while panoramic images have advantages for a wide range of computer vision systems. In this work, we propose an approach for semantic segmentation of 3D point clouds that avoids the limitations of 3D CNNs. Our approach first projects the point cloud onto a panoramic image. The corresponding panoramas are then fed into a 2D CNN as the input for semantic segmentation. Consequently, we reproject the semantic panoramic images to the 3D point clouds to obtain the 3D semantic results. We further applied different loss functions for semantic segmentation to deal with class imbalance. The networks with dice loss outperform other models.

The training experiments are performed on the Structured3D dataset and 2D-3D-S dataset. The Structured3D dataset is a synthetic dataset, while the 2D-3D-S dataset is from real-world scans. Using synthetic data for training and real scan data for testing change the test performance significantly. The testing semantic for the PTX point cloud using the model trained on the 2D-3D-S dataset has a lower IoU score but performs better than the Structured3D dataset. Moreover, the image contains some objects of the same kind but with different annotations. The networks trained on these data are overwhelmed, which leads to a low IoU score.

Furthermore, we implement a viewer interface to visualize the 3D point cloud and the corresponding panoramic image. The following sections address the limitations of this work and explain possible future extensions.

5.1 Limitations

The amount of labeled 3D indoor data is limited, and most datasets do not provide panoramic images. The annotated panoramas generated from 3D scans can be noisy and decrease the performance of the training results. The 2D-3D-S dataset provides 1,413 panoramic images with 13 classes. Training with 13 classes requires additional panoramic images for better performance. The synthetic dataset provides a large amount of data. However, the testing results can be poor when testing on real-world scanning dataset.

A technical limitation of this work is the GPU devices. When feeding a high-resolution image to the neural network, we have to set the batch size to 1 or downsize the image due to limited GPU memory.

5.2 Future work

In order to overcome the previously mentioned limitations, there exist several possible approaches. The most promising ones are:

- Improve the architecture of the neural network or apply existing networks, such as DeepLab [31] family.
- Train with RGB-D images or use the multi-stream RGB+D network.
- Apply a better loss function to handle class imbalance.
- Apply post-processing techniques, such as graph-cut and conditional random field (CRF).

Furthermore, an interesting continuation of this work would be to generalize the results trained on the synthetic dataset to the real-world scanning dataset. Synthetic data usually contains more images for training, which overcomes the issue of limited data. This can be realized by fine-tuning the network using the data from real-world 3D scans after training on the synthetic dataset.

Bibliography

[1] Felix Järemo Lawin et al. “Deep projective 3D semantic segmentation”. In: *International Conference on Computer Analysis of Images and Patterns*. Springer. 2017, pp. 95–107.

[2] Alexandre Boulch, Bertrand Le Saux, and Nicolas Audebert. “Unstructured point cloud semantic labeling using deep segmentation networks.” In: *3dor@ eurographics* 3 (2017), pp. 1–8.

[3] Bichen Wu et al. “Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1887–1893.

[4] Bichen Wu et al. “Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 4376–4382.

[5] Andres Milioto et al. “Rangenet++: Fast and accurate lidar semantic segmentation”. In: *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2019, pp. 4213–4220.

[6] Jing Huang and Suya You. “Point cloud labeling using 3d convolutional neural network”. In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE. 2016, pp. 2670–2675.

[7] Dario Rethage et al. “Fully-convolutional point networks for large-scale point clouds”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 596–611.

[8] Angela Dai et al. “Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4578–4587.

[9] 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*. 2018, pp. 9224–9232.

[10] Hang Su et al. “Splatnet: Sparse lattice networks for point cloud processing”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2530–2539.

[11] Radu Alexandru Rosu et al. “Latticenet: Fast point cloud segmentation using permutohedral lattices”. In: *arXiv preprint arXiv:1912.05905* (2019).

[12] Charles R Qi et al. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[13] Charles Ruizhongtai Qi et al. “Pointnet++: Deep hierarchical feature learning on point sets in a metric space”. In: *Advances in neural information processing systems* 30 (2017).

[14] Hengshuang Zhao et al. “Pointweb: Enhancing local neighborhood features for point cloud processing”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 5565–5573.

[15] Qingyong Hu et al. “Randla-net: Efficient semantic segmentation of large-scale point clouds”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11108–11117.

[16] 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*. 2018, pp. 984–993.

[17] Shenlong Wang et al. “Deep parametric continuous convolutional neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2589–2597.

[18] Francis Engelmann et al. “Exploring spatial context for 3D semantic segmentation of point clouds”. In: *Proceedings of the IEEE international conference on computer vision workshops*. 2017, pp. 716–724.

[19] Angela Dai and Matthias Nießner. “3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 452–468.

[20] Yuliang Sun et al. “PGCNet: patch graph convolutional network for point cloud segmentation of indoor scenes”. In: *The Visual Computer* 36.10 (2020), pp. 2407–2418.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[22] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. “Learning deconvolution network for semantic segmentation”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1520–1528.

[23] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. “Segnet: A deep convolutional encoder-decoder architecture for image segmentation”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[24] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[25] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. “V-net: Fully convolutional neural networks for volumetric medical image segmentation”. In: *2016 fourth international conference on 3D vision (3DV)*. IEEE. 2016, pp. 565–571.

[26] Zongwei Zhou et al. “Unet++: A nested u-net architecture for medical image segmentation”. In: *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2018, pp. 3–11.

[27] Tsung-Yi Lin et al. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[28] Hengshuang Zhao et al. “Pyramid scene parsing network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2881–2890.

[29] Junjun He et al. “Adaptive pyramid context network for semantic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7519–7528.

[30] Junjun He, Zhongying Deng, and Yu Qiao. “Dynamic multi-scale filters for semantic segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 3562–3572.

[31] Liang-Chieh Chen et al. “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.4 (2017), pp. 834–848.

[32] Francesco Visin et al. “Reseg: A recurrent neural network-based model for semantic segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2016, pp. 41–48.

[33] Francesco Visin et al. “Renet: A recurrent neural network based alternative to convolutional networks”. In: *arXiv preprint arXiv:1505.00393* (2015).

[34] Tsung-Yi Lin et al. “Focal loss for dense object detection”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[35] Jiahui Yu et al. “Unitbox: An advanced object detection network”. In: *Proceedings of the 24th ACM international conference on Multimedia*. 2016, pp. 516–520.

[36] Angel Chang et al. “Matterport3d: Learning from rgb-d data in indoor environments”. In: *arXiv preprint arXiv:1709.06158* (2017).

[37] Jia Zheng et al. “Structured3d: A large photo-realistic dataset for structured 3d modeling”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 519–535.

[38] I. Armeni et al. “Joint 2D-3D-Semantic Data for Indoor Scene Understanding”. In: *ArXiv e-prints* (Feb. 2017). arXiv: 1702.01105 [cs.CV].

[39] Angela Dai et al. “Scannet: Richly-annotated 3d reconstructions of indoor scenes”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5828–5839.

[40] Liangliang Nan. “Easy3D: a lightweight, easy-to-use, and efficient C++ library for processing and rendering 3D data”. In: *Journal of Open Source Software* 6.64 (2021), p. 3255. DOI: 10.21105/joss.03255. URL: https://doi.org/10.21105/joss.03255.