


Learning Shapes for the Efficient Segmentation of 3D Medical Images using Point Cloud

Mohammad Sadil Khan
Master Thesis
M.Sc in Machine Learning and Data Mining
University Jean Monnet

Supervisor: Razmig Kéchichian, Sébastien Valette, Julie Digne

Creatis
Insa Lyon
June 27, 2022

Contents

1 Abstract **1**

2 Introduction **1**
2.1 Lab Overview 1
2.2 Thesis Task 1

3 Background and Related Work **3**
3.1 3D Representation 3
3.2 Medical Image Segmentation 4
3.3 Point Cloud 6
3.3.1 Point Cloud Introduction 6
3.3.2 Point Cloud Segmentation 9

4 Model **14**
4.1 Canny Edge Detection 14
4.2 RandLa-Net 15
4.3 Loss Function 18
4.4 Evaluation Metrics 20

5 Contributions **21**
5.1 Modified RandLa-Net with Feature Extractor 21
5.2 ComboLoss: 22

6 Experiments **22**
6.1 Dataset 22
6.2 Point Cloud Extraction 23
6.3 Point Cloud Segmentation 24
6.3.1 Train 24
6.3.2 Results 24
6.4 Voxel Segmentation 30

7 Conclusions **33**

Acknowledgement

I would like to deepest thanks and gratitude to my supervisors Razmig Kéchichian, Sébastien Valette and Julie Digne. Their dedication, constant supervision , keen interest and above all an overwhelming attitude towards me have made it possible to complete my work. Their timely advice, meticulous scrutiny, helpful suggestions and out-of-the-box approaches have helped me to a great extent to accomplish my task.

I owe a deep sense of gratitude to my parents, family and friends. Without their steady support and encouragement, none of it would have been possible.

Acronym

CNN Convolutional Neural Network 2

DL Deep Learning 4

SDF Signed Distance Function 4

CT Computed Topography 5

MRI Magnetic resonance imaging 5

MLP MutiLayer Perceptron 6

GNN Graph Neural Network 4

LFA Local Feature Aggregation 15

LSE Local Spatial Encoding 15

CE Cross Entropy 18

WCE Weighted Cross Entropy 19

CMCE Cost Matrix Cross Entropy 25

IOU Intersection Over Union 19

KNN K-Nearest Neighbor 12

List of Tables

1 Different Modalities in Visceral Dataset. CTce-ThAb is used for our task. 22
2 Parameters for training the RandLa-Net Model. 24
3 Results of Liver Segmentation for different weights in Cross Entropy Loss. 27
4 Results of Lungs Segmentation for different weights in Cross Entropy Loss. 27
5 Results of Liver Segmentation with and without Feature Extractor. . 28
6 Results of Lungs Segmentation with and without Feature Extractor. . 28
7 Experiment Results for multi-class segmentation. CL is the Cross Entropy Loss with combination weights ($w_r = 0.3 + \frac{0.4}{r+0.02} + \frac{0.3}{r}$). FE(n) is RandLa-Net with Feature Extractor with $n \times n \times n$ intensity shape as explained Section 5.1. DR(n) is the random downsampling rate (In every encoder block $\frac{1}{n}$% points are randomly removed). Comb weight is the weight for CL. CMCE is the cost matrix cross entropy as explained in Section 4.3.A. The best results are in red. 30

List of Figures

1 Framework of our current working model. H, W, D are the height, width and depth of an image. First we extract N points using edge detection and then we train a large scale point cloud segmentation network and finally during inference we map the voxels into point cloud space as query points and output the segment labels. 2
2 Different types of 3D representations of data (Image from [1]). 3
3 Coronal view of abdomen in CT Modality in Visceral Dataset. 5
4 Coronal view of abdomen in MRI Modality in Visceral Dataset. 5
5 Extracted Point Cloud of lungs from visceral dataset using canny edge detection. 7
6 Voxelization of a point cloud (Image from [18]). 11
7 Voxelization and memory footprint (Image from [18]). 11
8 RandLa-Net Architecture. FC is the fully connected layer, LFA is the local feature aggregation, RS is random sampling, MLP is shared multilayer perceptron, US is upsampling and DP is dropout. (Image from [29]) 15
9 Random Sampling in RandLa-Net. The downsampling rate is a hyper-parameter and has significant influence on model performance (Image from [29]). 16
10 Feature Aggregation Module. The topleft panel shows Local Spatial Encoding that extracts features and positional information from neighbors.The topright panel shows attentive pooling to extract important features from neighboring points. The bottom layer is Dilated Residual Block which expands the receptive field (Image from [29]). 17
11 Illustration of dilated residual block which expands the receptive field at each step (Image from [29]). 18
12 Modified RandLaNet with Feature Extractor. 21
13 Proposed Feature Extractor. It is a convolutional operation followed by an FC layer. This operation is done for every N points. 21
14 2D slices of a visceral image from different view with ground truth segmentation task. The images are from contrast enhanced CT scans. For multi-class segmentation task five organs are chosen - lungs, liver, bladder, left and right kidney. 23
15 **Left**: Visualization of one slice of a volumetric image from visceral. **Middle**: Edge Detection of that slice. **Right**: Point cloud extracted from the volume image. The edges in the middle are considered as points representing the organ. For the ease of visualization, background points are excluded from the point cloud. 24
16 Class distribution of point cloud extracted from visceral dataset. The background is the most dominating class, covering almost 87% of the points. 25
17 Weights for every class in CE **without taking into account** prediction cost. 26
18 Weights for every class in CE **for taking into account** prediction cost. 26
19 Confusion Matrix before applying CMCE. 26
20 Confusion Matrix after applying CMCE. 26

21 Confusion Matrix for multiclass segmentation. RandLa-Net with Feature Extractor($7 \times 7 \times 7$) has been used with CMCE Loss with downsampling factor 2. We took the mean of the confusion matrix for all 20 images. 26
22 L2 Distance of latent vectors from mean lungs latent vector. 29
23 Cosine similarity of latent vectors from mean lungs latent vector. . . . 29
24 Illustration of voxel segmentation. 3D bounding box is formed using the min and max coordinate from the predicted label. These points are mapped to voxel space and all the voxels in between the bounding box are selected. These new query voxels are mapped to point cloud space and added to the input point cloud and passed to the trained model for predicting segmentation label. 31
25 Voxel Segmentation of Lungs after mapping the voxels in the point cloud space and adding it to the input point cloud. RandLa-Net has been used for training the binary lungs segmentation task with CMCE Loss. FE is the Feature Extraction Layer. 31
26 Voxel Segmentation of Liver after mapping the voxels in the point cloud space and adding it to the input point cloud. RandLa-Net has been used for training the binary liver segmentation task with CMCE Loss. FE is the Feature Extraction Layer. 32

1 Abstract

In this report, we present a novel approach for 3D medical image segmentation using point clouds. 3D Convolutional Neural Networks have been the most dominating networks in medical image processing but they require large memory footprints and training samples. Hence we used point clouds to represent the image instead of voxels. Point clouds are lightweight and contain shape and smoother surface information. We extracted the point clouds from 3D voxel images using canny edge detection. We modified RandLa-Net[29], an attention-based point cloud segmentation network with a feature extraction layer to aggregate local geometrical features with spatial point features for our large-scale point cloud segmentation task. Our proposed model performed better than the original network in multi-class as well as binary point cloud segmentation tasks in Visceral dataset. Finally, we propose a model-independent step to perform the image segmentation of the original 3D volumetric images in Visceral dataset by mapping voxels in the point cloud space and adding it to the input point cloud before being passed to the trained model. We performed many experiments on the weights of the Cross-Entropy loss function for the class imbalance problem as well as the intrinsic architectural properties of the model architecture like downsampling factor and distinct latent vector learning that can be improved to perform better segmentation.

2 Introduction

2.1 Lab Overview

CREATIS (Centre de Recherche en Acquisition et Traitement de l'Image pour la Santé) is a Biomedical Research Lab based in Lyon, France. It is composed of about 200 persons with members of different domains. It focuses on two main tasks in medical image analysis - (1) how imaging technology can be used to address major health challenges and (2) how theoretical challenges in biomedical imaging problems related to image, speech, and modeling can be tackled. The laboratory consists of four research teams which focus on the above challenges through various interdisciplinary methods. I worked in the team Myriad during my internship. The team focuses on medical image analysis and learning (Segmentation, Reconstruction, etc) and Modeling and Simulation.

2.2 Thesis Task

Work on the segmentation of unstructured data has already been done in the framework of a previous internship funded by the ANR TOPACS project. This work led to a 2D image segmentation approach, using mainly unstructured points as input data for the segmentation, based on Occupancy Network[6]. This internship continues this work to develop a novel approach to segmenting organs in 3D medical images using point clouds. The task is divided into three parts.

- **Point Cloud Extraction**: Extraction of point clouds from 3D voxel images.
- **Point Cloud Segmentation:** Segmentation of the extracted point clouds.
- **Voxel Segmentation:** Segmentation of the 3D voxels using the model trained on point cloud segmentation.

Medical image segmentation is a crucial task to support the computer-aided diagnosis of diseases for optimal treatment options or robotic surgeries. With the rapid advancements of Deep Learning Methods in 3D computer vision, 3D Convolutional Neural Network (CNN)s are the most suitable choices for 3D medical image segmentation. However 3D CNNs faces several challenges. Deep Neural Networks require lots of training samples to train for a task. However, the most common challenge in medical image segmentation is the rarity of annotated medical datasets images. 3D CNNs also have a lot of trainable parameters. Usually, high-resolution scan volumes in medical dataset are of $512 \times 512 \times 512$ dimension, which is computationally very expensive and requires a high memory footprint. Generally, the images are downsampled to a lower size to fit in the model at the cost of information loss.([1, 2])

With the advent of deep networks for point cloud segmentation[5, 14, 18–21, 29], it is possible to segment large-scale organs points efficiently. However, point cloud segmentation faces the challenges of no-connectivity and unstructuredness.[3] In this report we present the approaches to overcome those challenges and segment the point clouds. We used the Canny edge detection method to extract the surface point clouds from voxels. Then we use RandLa-Net[29] with a feature extraction layer, to perform segmentation of the input point cloud. Finally, we will use the learned features of the points to segment the voxels in the original 3D image by mapping voxels into the point cloud space (Figure 1).

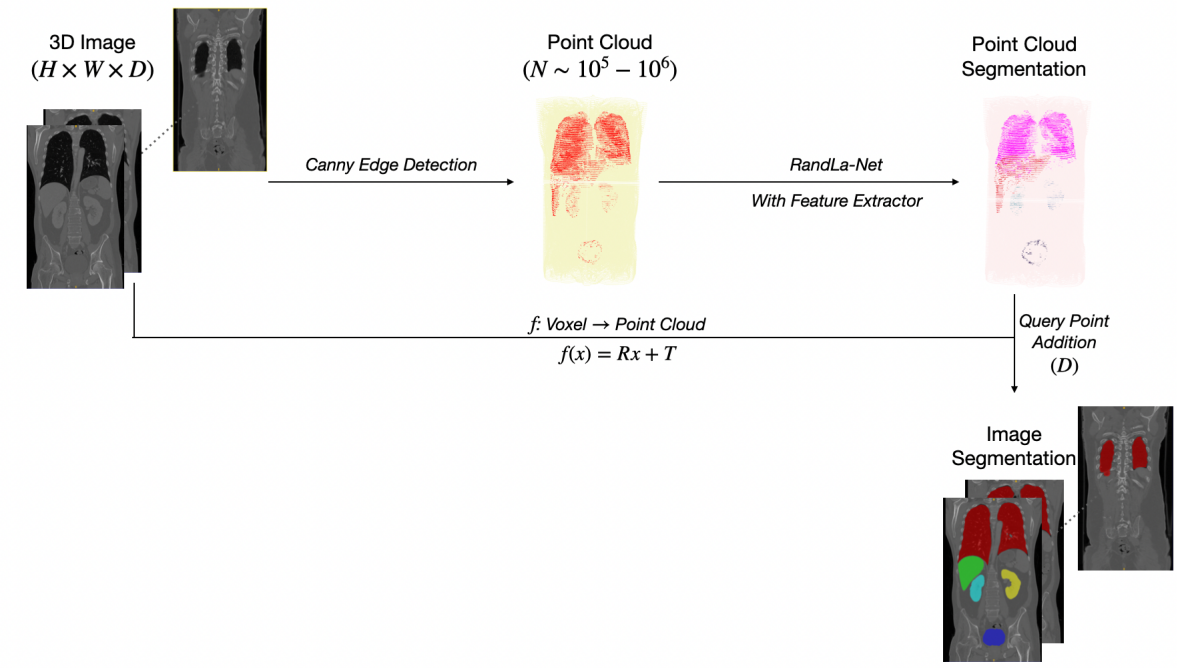

Figure 1: Framework of our current working model. H, W, D are the height, width and depth of an image. First we extract N points using edge detection and then we train a large scale point cloud segmentation network and finally during inference we map the voxels into point cloud space as query points and output the segment labels.

The rest of the report is organized in the following way. Section 3 discusses the related works in medical image segmentation and prior works in point clouds. Some background concepts necessary for point cloud processing are also discussed in Section 3. Section 4 presents the details of the models, loss functions and evaluation metrics used for the project. Section 5 explains our proposed modification of the RandLa-Net

architecture. Section 6 details all the experiments and results of the internship while Section 7 concludes with the future direction of our thesis idea.

3 Background and Related Work

Image segmentation is the task of assigning labels to every pixel in an image. It is a method to divide the image into different groups called image segments for further analysis of an image. Segments of the same nature have the same labels. Image Segmentation has vast applications in Medical Imaging, video surveillance, autonomous driving, etc.

3.1 3D Representation

3D datasets are acquired using various devices leading to different representations of 3D data. There are three most popular type of representations.

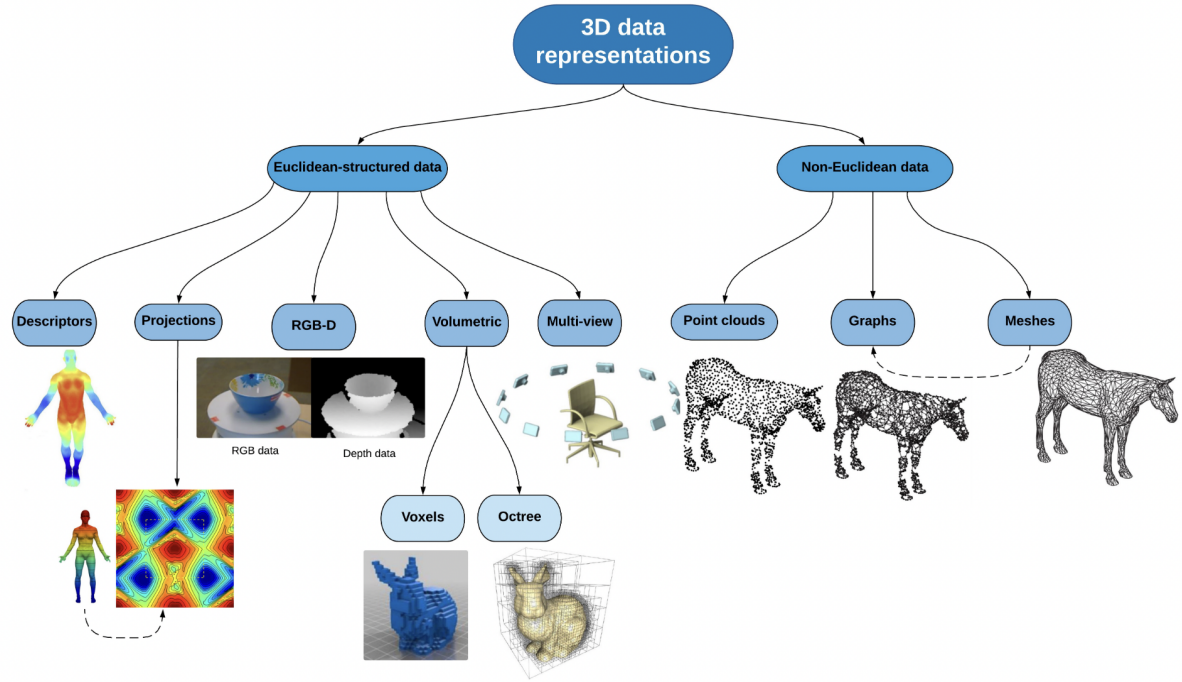

Figure 2: Different types of 3D representations of data (Image from [1]).

- **Voxel Representation:** A 3D image can be represented in a regular 3D grid. Voxels which are a 3D equivalent of pixels are the simplest approach used to represent the distribution of the object throughout the 3D scene. Despite its simplicity, it faces some limitations. Voxel-represented 3D data also encodes occupied and non-occupied parts of the scene which is not efficient and requires more memory. Due to its cubically growing nature (n^3), deep learning methods such as *3D UNet*[13] on voxel-based 3D datasets require high computational power and memory footprints due to convolution operations and large numbers of parameters. Due to this, images are often downscaled to lower resolutions($258 \times 258 \times 258$) with the risk of losing important information on smaller objects of the scene. Also, voxels do not capture finer shape details

of the object. Octree-based representations solve the problems of memory limitations. In octree-based representation, a 3D space is hierarchically divided into several cubes and each cube is either outside or inside the object. It is a very powerful method to represent the data as it has bigger cubes representing the same space of multiple voxels. It has finer shape details than voxel-based methods. However, both these methods do not preserve the intrinsic properties of shapes and surface smoothness.

- **Mesh Representation:** A 3D Mesh consists of vertices, edges, and a set of polygons called faces. A face is a closed set of edges and can be triangular (three edges) or quadrilateral(four edges). The vertices contain information about intensity and coordinates. These vertices contain a connectivity list which contains the information about the edges between two vertices. Learning on 3D meshes comes with some challenges. First of all, Deep Learning (DL) methods[34] are not yet capable enough to handle irregular representations of meshes with noisy and missing data which is very common for the mesh-based dataset. A mesh can be transformed into a graph based representation where graph nodes correspond to the vertices and edges can be extracted from the connectivity list. From there Graph Neural Network (GNN) can be used to exploit the geometry of the 3D scene to learn about the objects.

- **Point Cloud Representation:** A point cloud is a set of points in the 3D space that approximates the 3D geometry of space. Each point in the shape contains its coordinates along with intensity information. Point clouds are usually acquired using sensing technologies. Despite its simple representation, it is hard for DL algorithm like *PointNet*[14] to learn from point cloud data due to irregularity or no connectivity problem. In Section 3.3 Point Cloud, it is explained in details.

- **Boundary Representation:** Besides the above mentioned representations, one other representation which is increasingly getting popular is Occupany Networks[6]. In this representation, a classification model is learned using existing data representation (point cloud, voxel, or mesh) and the object is constructed using the class boundary. In the *Occupancy Networks*[6], an encoder is learnt on either point-based, voxel-based or range-based dataset. After the encoder learns the global properties of the 3D object, points are sampled from the 3D space and are used for query using the encoded vector. It uses conditional batch normalization to condition the network on the encoded vector. In *DeepSDF*[7], a feed-forward neural network is learnt to represent Signed Distance Function (SDF). An SDF is a continuous function that outputs the distance of a point from the closest point on the surface.

$$SDF(x) = \inf_{y} \|x - y\|_n, \forall y \in M \tag{1}$$

where M is the set of points on the boundary. SDF is negative if a point is inside the object and positive if it is outside. The surface of the object can be thought of as the decision boundary of the learned classifier.

3.2 Medical Image Segmentation

Medical Image Segmentation is the task of segmenting regions of interest(i.e organs, tissues) in a medical image. It is useful for the analysis of the anatomical structure of

organs, locating the tumors, lesions, abnormalities or measuring the growth of tissues and it is an important step for clinical diagnosis and computer-aided assistance. Due to different sensors and devices being used for acquiring medical images there are many modalities in medical dataset.

- **CT:** Computed Topography (CT) scans are an imaging modality (Figure 3) that uses X-rays to obtain the internal structural information of the human body and is based on the property that objects absorb X-rays differently. Dense tissues appear white in CT scans since it blocks some x-rays and soft tissues, hollow space and fluids appear black since rays can pass easily. Hounsfield scale can be used in CT images for interpreting the type of tissues depending upon CT intensity value. The artifacts present in CT images are *Streak Artifacts* which are caused when an object is moved or there are errors in data sampling and *Motion Artifacts* which are caused near ill-defined boundaries of an object during movement along the scanning plane.[11, 12]

- **MRI:** Magnetic resonance imaging (MRI) is an imaging modality (Figure 4) that uses a magnetic field and radio waves to create a detailed image of organs and tissues. Some of the artifacts present are *RF Noise, Partial volume, Motion, Gibbs Ringing, etc.* It is generally more expensive than CT scans and prone to spatial non-homogeneity of intensity due to the nonhomogeneity of the magnetic field.

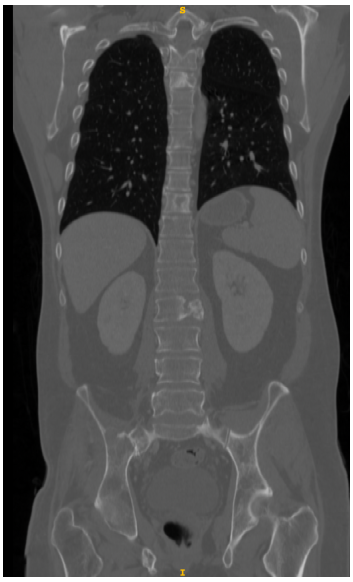

Figure 3: Coronal view of abdomen in CT Modality in Visceral Dataset.

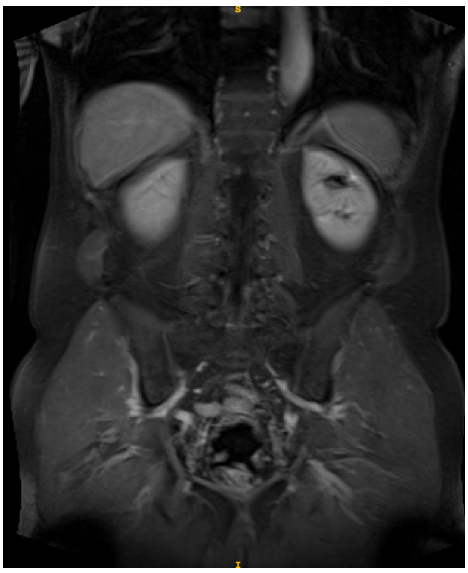

Figure 4: Coronal view of abdomen in MRI Modality in Visceral Dataset.

Although datasets for organ CT scans are widely available, medical segmentation is a challenging task because of small training datasets, noises and artifacts[2, 4]. In the following part, we present some methods to perform medical image segmentation.

Earlier Methods: Earlier segmentation techniques were based on thresholding on histogram features[8], edge-based segmentation[9], graph-cut segmentation[35], and level-set methods[37]. While these methods are simple and interpretable, it suffers from a lack of generality.

CNN: With the rise of DL, it is possible to create architectures inspired from CNN for effective segmentation and extend it to 3D CNN. In CNN, a set of filters are used in every convolution layer to learn about the visual features of the 3D voxel image. These filters work on the local receptive field to learn the features from a small set of input data. A local receptive field is a region of input data that a kernel or filter is exposed to and is defined by the kernel size. The pooling layers in CNN downsample the feature maps increasing the receptive field and reducing the parameter sizes for the next convolution layers. Due to this local learning approach, in lower layers, CNN learns lower-level features such as lines, edges, and contours, and in the higher layer, due to accumulated receptive fields, it can learn high-level features such as shapes, boundaries, etc.

3DCNN: Medical images generally contain noise with blurry boundaries and that is why it is challenging to segment based on low-level image features or just high-level features. *U-Net*[10] solves this problem by creating skip connections to fuse high-level features with low-level features. Skip connections are widely used to preserve information through the concatenation of the encoding layer and corresponding decoding layer output. Although it suffers from the semantic gap between low-resolution and high-resolution features, it can be solved by using a convolution operation between the skip connections. *3D UNet*[13] extends the idea of *U-Net*[10] for 3D volume. It uses 3D convolutional layer as well as 3D pooling layers. Due to its computational challenges, 3D-UNet downsamples the input shape to $132 \times 132 \times 116$ compared to 512×512 in UNet. It is also limited to three downsampling layers. *VNet*[38] uses residual connections in between the layers to achieve high performance since residual connections can solve vanishing gradient problems and accelerate network convergence.

Challenges: Despite its remarkable advances in segmentation tasks in the medical domain in segmentation tasks, 3D CNNs have a lot of parameters and is computationally expensive. Reducing the input size causes the loss of important information. 3DCNN also requires a large number of training samples which is rare in the medical dataset due to the lack of sufficient experts in annotation and privacy-related problems. Instead, point clouds require lower memory footprints to represent 3D data and MutiLayer Perceptron (MLP) based models can be efficiently trained for segmentation in much less time (Section 3.3.2.D Theorem 1). Also, sampling can be performed in various steps of the model architecture to reduce the number of points to minimize memory requirements which can learn richer representation for the organs with less amount of time[28, 29]. This makes point cloud learning an ideal choice for our thesis task over voxel-based learning. Our proposed modification of RandLaNet[29] model with feature extraction layer has 1.4M parameters compared to 3D UNet's 20M parameters.

3.3 Point Cloud

3.3.1 Point Cloud Introduction

A. Point Cloud:
A Point Cloud is a set of points in 3D space which can represent the boundary or the whole object (including inside points). In a point cloud, the points are unordered and are not restricted by any grid which means a point cloud can be expressed in an infinite way (using translation). Each point can have 3D coordinates and feature vectors ($P = \{(X_i, F_i)\}_{i=1}^{i=N}, X_i \in \mathbb{R}^3, F_i \in \mathbb{R}^d$). Other information such as SDF can

be approximated in the pre-processing step.

B. Properties of Point Cloud in $\mathbb{R}^3$:

- *Unordered:* Unlike images or arrays, point cloud is unordered. It has no restriction to be confined within a boundary. This causes a problem for CNN type architecture to learn since CNN uses convolutional operations which requires ordered and regular array like representation of the input. Point cloud networks are generally invariant to the $N!$ number of permutations in input.

- *Irregularity:* Points are not sampled uniformly from an image which means different organs can have dense point points while other sparse[5, 14]. This causes class imbalance problems in point cloud dataset.

- *Connectedness:* Since points are not connected like graph structure and neighbouring points contain meaningful spatial and geometry information of the organ, networks must learn to pass information from points to points.

C. Point Cloud Generation:

Point clouds are generated by 3D Scanners like time-of-flight sensors and depth cameras or photogrammetry software. Time-of-flight sensors use the reflected laser beams from sensors to the object to capture the surface of the object. For our task, we used edge detection to capture high gradient points from the voxel images. Figure 5 shows one such example.

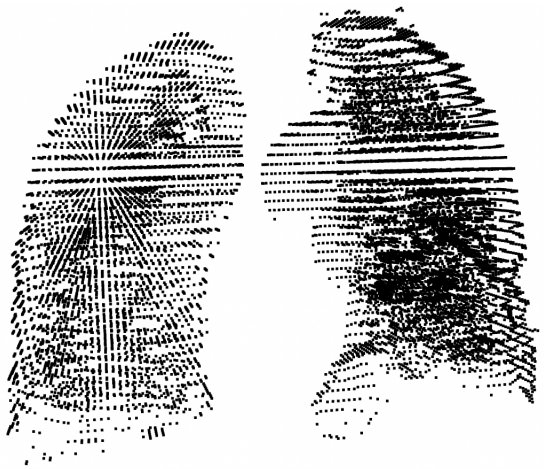

Figure 5: Extracted Point Cloud of lungs from visceral dataset using canny edge detection.

D. Point Cloud Sampling:

Point Cloud Sampling is the method of choosing a subset of point clouds. Sampling methods were used in our model RandLa-Net[29] to reduce the number of points for faster learning. This is an essential step in the large-scale point cloud processing, since learning features for all the points can be time consuming. Instead, features can be learnt for small point clouds and for other points, it can be aggregated using neighboring features. There are different sampling algorithms available. Let N be the number of points, M is the sampled number of points chosen with $N > M$, D is the maximum number of points in a 3D voxel grid ($N >> D$) and K is the number of nearest neighbour($N >> K$).

1. Heuristic Sampling

- *Grid Sampling:* In Grid Sampling, a 3D voxel grid is used over the point cloud and each occupied voxels extract one point based on averages or most frequently occurring classes. This sampling results in a uniform sample. The time complexity of the grid sampling is $O(ND)$. By averaging the points on the surface, grid sampling loses smooth boundary information.

- *Random Sampling:* One of the simplest sampling methods, Random Sampling takes M random points from a point cloud of N points ($N > M$). Time complexity is $O(M)$ which makes it efficient to use in large-scale point cloud networks.

- *Farthest Point Sampling:* It iteratively extracts set of points $P = \{p_1, p_2, \cdots, p_M\}$ such that p_j is the farthest point from the first $j-1$ points in P. The time complexity is $O(M^2N)$ which makes it unsuitable for large scale point cloud processing.

- *Inverse Density Importance Sampling:* In IDIS, density is calculated for every point by adding the distance between the point and its nearest neighbors. $density(x) = \sum_{y \in KNN(x)} \|x - y\|_2^2$. So N points are reordered according to the inverse of the density and top M points are selected which means lower density points are more likely to be chosen than high dense points. Time complexity is $O((K+N)logN)$. This sampling can control density but is sensitive to outliers and noise.

2.. Learning Based Sampling

- *Generator Based Sampling:* Generator Based Sampling(GS) learns to generate a small subset of point clouds from the original point cloud. For a point cloud set P and a task T, GS tries to find $S \subset P$ by minimizing the objective function f such that $S^* = argmin_S(f(T(S))$. It is an end-to-end trainable model. But at inference stage, it uses FPS[19] to match subsets with original point cloud. It takes up to 20 minutes to sample 10% of 10^6 points.

- *Gumbel Subset Sampling:* Gumbel Subset Sampling [28] uses attention mechanism to choose a representative and task-specific subset of the point cloud. Given an input set $X_i \in \mathbb{R}^{N_i \times c}$, the task is to choose a suitable $X_{i+1} \in \mathbb{R}^{N_{i+1} \times c}, N_{i+1} \leq N_i$ and $X_{i+1} = y \cdot softmax(WX_i^T), W \in \mathbb{R}^{N_{i+1} \times N_i}$. It is completely end-to-end learnable and can be used in any segmentation network.

E. Point Cloud Feature Aggregation:
Since point clouds lack connectedness, feature aggregation is a method to pass information from points to points. Feature Aggregation layer has been used in RandLa-Net with attention mechanism. It is the backbone of every MLP-based point cloud networks. Here we present the mathematical formula for it. In General, let $P = \{p_1, p_2, p_3, \ldots, p_n\}$ be the the set of points, $p_i \in \mathbb{R}^3$. $F = \{f_1, f_2, f_3, \ldots, f_n\}$ be the set of features for points in P, $f_i \in \mathbb{R}^d$. Then the feature aggregation function $G : \mathbb{R}^3 \times \mathbb{R}^d \to \mathbb{R}^{d'}$ is defined by

$$G(p_i, f_i) = R(\{A(\Delta p_{ij}, f_j) | j \in N(i)\} \tag{2}$$

where A is the feature transformer (MLP) and R is the reduction function(MAX,AVG,SUM). $N(i)$ is the set of indices of points that are neighbors p_i and Δp_{ij} is the effect of p_j on p_i. $\Delta p_{ij} = p_j - p_i$. Alternatively $\Delta f_{ij} = f_j - f_i$ can also be used instead of Δp_{ij} and

one example of A is $A(\Delta p_{ij}, f_j) = MLP(concat(\Delta p_{ij}, f_j))$. In RandLa-Net, attention scores have been added with the concat function to add weights to the aggregating features. Almost all the components in feature aggregation functions are susceptible to changes but all the functions can achieve state-of-the art results given the right parameters and settings[30].

3.3.2 Point Cloud Segmentation

Point Cloud Segmentation is the task for grouping objects or assigning labels to every points in the point cloud. It is one of the most challenging tasks and a research topic in deep learning since point clouds are noisy, unstructured and lack connectedness property. All the methods are categorized into four categories.

(A) *Edge Based Methods*
Edges describe the intrinsic characteristics of the boundary of any 3D object. Edge-based methods locate the points which have rapid changes in the neighborhood. Bhanu[5] proposed three approaches for detecting the edges of a 3D object. The first approach is calculating the gradient. The second approach is fitting 3D lines to a set of points(i.e neighboring points) and detecting the changes in the unit direction vector from a point to the neighboring points. The third approach is a surface normal approach where changes in the normal vectors in the neighborhood of a point determine the edge point. Edge models are fast and interpretable but they are very sensitive to noise and sparse density of point clouds and lack generalization capability. Learning on incomplete point cloud structure with edge-based models does not give good accuracy. In Medical image datasets especially MRI data, the organ boundaries sometimes do not have high gradient points compared to CT data which means for every modality, we have to find new thresholds in edge-based methods.

(B) *Region Based Methods*
Region-based methods use the idea of neighborhood information to group points that are similar thus finding similarly grouped 3D objects and maximizing the dissimilarity between different objects. Compared to edge-based methods, these methods are not susceptible to noise and outliers but they suffer from inaccurate border segmentation. There are two types of region-based methods.

- *Seeded-region Methods(bottom up):* Seeded region segmentation is a fast, effective and very robust image segmentation method. It starts the segmentation process by choosing manually or automatically in preprocessing step, a set of seeds which can be a pixel or a set of pixels and then gradually adding neighbouring points if certain conditions satisfy regarding similarity[5, 15]. The process finishes when every point belongs to a region. Seeded-based segmentation is very much dependent upon the choice of seed points. Inaccurate choices often lead to under-segmentation or over-segmentation.
- *Unseeded-region Methods(top-down):* Unlike seeded-based methods, unseeded methods have a top-down approach. The segmentation starts with grouping all the points into one region. Then the difference between all the mean point values and chosen point value is calculated. If it is more than the threshold then the point is kept otherwise the point is different than

the rest of the points and a new region is created and the point is added into the new region and removed from the old region. The challenges are over-segmentation and domain-knowledge which is not present in complex scenes[5].

(C) *Attribute Based Methods*
Attribute-based methods use the idea of clustering. The approach is to calculate attributes for points and then use a clustering algorithm to perform segmentation. The challenges in these methods are how to find a suitable attribute that contains the necessary information for segmentation and to define proper distance metrics. Some of the attributes can be normal vectors, distance, point density, or surface texture measures. It is a very robust method but performs poorly if points are large-scale and attributes are multidimensional.[5]

(D) *Deep Learning Based Methods*
The main challenge in point cloud segmentation is find good latent vector which can contain sufficient information for segmentation task. Deep Learning methods offers the best solution to learn good representations. Neural networks being a universal approximator can theoretically approximate the target function for segmentation. The following theorem justifies how MLPs can approximate the function for the segmentation task given enough neurons.
Theorem 1: *Given a set of point clouds* $X = \{\{x_1, x_2, \ldots, x_n\}, n \in \mathbb{Z}^+, x_i \in [0,1]^m\}$, *let* $f : X \to R$ *be a continuous function with respect to hausdorff distance*$(d_H(\cdot,\cdot))$.$\forall \epsilon > 0, \exists \eta$, *a continuous function and a symmetric set function* $g(x_1, x_2, \ldots x_n) = \gamma \circ MAX$ *such that* $\forall S \subset X$.

$$\left| f(S) - \gamma\left(\underset{x_i \in S}{MAX}(\eta(x_i)) \right) \right| < \epsilon$$

γ *is a continuous function and* MAX *is an elementwise max operation which takes an input* k *number of vectors and return a vector with element wise maximum. In practice* γ *and* η *are MLP[14].*
The DL methods for point cloud segmentation can be divided into following ways.

(a) **Projection-Based Networks:** Following the success of 2d CNNs, projection-based networks use the projection of 3D point clouds into 2d images from various views/angles. Then 2D CNN techniques are applied to it to learn feature representations and finally features are aggregated with multi-view information for final output[3, 16]. In [17], tangent convolutions are used. For every point, tangent planes are calculated and tangent convolutions are based on the projection of local surface geometry on the tangent plane. This gives a tangent image which is an $l \times l$ grid where 2d convolutions can be applied. Tangent images can be computed even on a large-scale point cloud with millions of points. Compared to voxel-based models, multi-view models perform better since 2D CNN is a well-researched area and multi-view data contain richer information than 3D voxels even after losing depth information. The main challenges in multi-view methods are the choice of projection plane and the occlusion which can affect accuracy.

(b) **Voxel-Based Networks:** Voxel-based methods convert the 3D point clouds into voxel-based images. Figure 6 shows an example. The points

which make up the point cloud are unstructured and unordered but CNN requires a regular grid for convolution operation. Voxelization is done in the following steps.

- A bounding box of the point cloud is calculated which defines the entire space that is to be divided.
- Then the space is divided into a fixed-size grid. Each grid is called 3D cuboids.
- The point cloud is divided into different grids with each 3D cuboid containing several points and these 3D cuboids become voxels that represent the subset of points.
- Features are calculated from the subset of points inside a voxel.

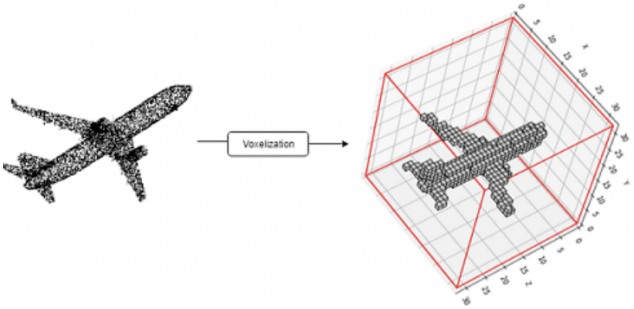

Figure 6: Voxelization of a point cloud (Image from [18]).

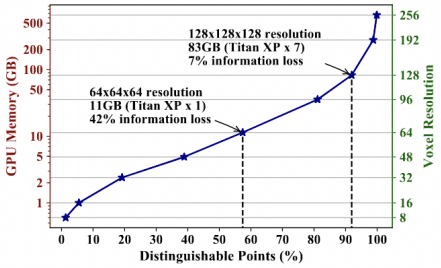

Figure 7: Voxelization and memory footprint (Image from [18]).

Voxelization creates quantization artifacts and loses smooth boundary information. It is a computationally expensive preprocessing step and memory footprints increase cubically due to the cubical growth of voxels. If voxel resolution is low, many points will belong to a voxel and will be represented by a single voxel so these points will not be *differentiable*. A point is *differentiable* if it exclusively occupies one voxel grid. Figure 7 summarizes the memory requirements for if we want to retain higher number of differentiable points which will mean lower information loss[18]. To retain 90% of the differentiable points, GPU memory is more than 82 GB and voxel resolution is $128 \times 128 \times 128$ which is a huge computational overload. After voxelization, 3D CNNs can be applied for learning features for segmentation. Techniques mentioned in Section 3.2 can be used for segmentation. In a similar approach, *Point-Voxel CNN* [18] uses CNN and MLP bases fusion learning. It first voxelizes the point cloud and uses convolution for feature learning and then devoxelize the voxels for voxel-to-point mapping(i.e interpolation is used to create distinct features of a voxel for the points that belong to the voxel). The features of a point cloud are then aggregated with the features learned using MLP. Voxel-based networks for point cloud segmentation have the same challenges as mentioned in Section 3.2 3DCNN.

(c) **Point-Based Networks:** Point-Based Networks work on raw point cloud data. They do not require voxelization or projection. *PointNet*[14] is a breakthrough network that takes input as raw point clouds and outputs labels for every point. It uses permutation-invariant operations like point-wise MLP and symmetric layer, Max-Pooling layer for feature aggregation

layer. It achieves state-of-the-art performance on benchmark datasets. But *PointNet* lacks local dependency information and so it does not capture local information. The max-pooling layer captures the global structure and loses distinct local information. Inspired by *PointNet* many new networks are proposed to learn local structure. *PointNet++*[19] extends the *PointNet* architecture with an addition of local structure learning method. The local structure information passing idea follows the three basic steps (1) Sampling (2) Grouping (3) Feature Aggregation Layer (Section 3.3.1.E lists some Feature Aggregation functions) to aggregate the information from the points in the nearest neighbors. *Sampling* is choosing M centroids from N points in a point cloud ($N > M$). Random Sampling or Farthest Point Sampling are two such methods for sampling centroids. *Grouping* refers to sample representative points for a centroid using K-Nearest Neighbor (KNN). It takes the input (1) set of points $N \times (d+C)$, with N is the number of points,d coordinates and C feature dimension and (2) set of centroids $N_1 \times d$. It outputs $N_1 \times K \times (d+C)$ with K is the number of neighbors. These points are grouped in a local patch. The points in the local patches are used for creating local feature representation for centroid points. These local patches work like receptive fields. *Feature Aggregation Layer* takes the feature of the points in the receptive field and aggregate them to output $N_1 \times (d+C)$. This process is repeated in a hierarchical way reducing the number of points as it goes deeper. This hierarchical structure enables the network to be able to learn local structures with an expanding receptive field. Most of the research in this field has gone into developing an effective feature aggregation layer to capture local structures. *PointWeb*[20]
creates a new module *Adaptive Feature Adjustment* to enhance the neighbor features by adding the information about the impact of features on centroid features and the relation between the points. It then combines the features and uses MLP to create new representations for centroid points. Despite their initial successes the following methods achieve higher performance due to their advanced local aggregation operators.

(d) **Graph-Based Networks:** A point cloud is unstructured, unordered, and has no connectivity properties. But it can be transformed into a graph structure by adding edges to the neighbors. Graph structures are good for modeling correlation and dependency amidst points through edges. GNN based networks use the idea of graph construction, local structure learning using expanding receptive field, and global summary structure. *PointGNN*[21] creates a graph structure using KNN and applies pointwise MLP on them followed by feature aggregation. It updates vertex features along with edge features at every iteration. [22] introduces super point graphs to segment large-scale point clouds. It first creates a partition of geometrically similar objects (i.e planes, cubes) in an unsupervised manner and applies graph convolutions for contextual segmentation. *DGCNN*[23] introduces the Edge Convolution operation for dynamically updating the vertices and edges thus updating the graph itself. [24] creates a *Point Global Context Reasoning* module to capture the global contextual information from the output of any segmentation network by creating a graph from the output embedding vectors.

(e) **Transformer and Attention-Based Networks:** Transformers and attention mechanism are a major breakthrough in NLP tasks. This has lead to research in attention mechanism in 2D CNN[25]. Attention follows the following derivation.

$$y_i = \sum_{x_j \in R(x_i)} \alpha(x_i, x_j) \odot \beta(x_j) \tag{3}$$

where $\odot$ is the Hadamard product, $R(x_i)$ is the local footprint of x_i (i.e a receptive field, one such example can be nearest neighbors). $\beta(x_j)$ produces a feature vector from x_j that is adaptively aggregated using the vector of $\alpha(x_i, x_j)$, where $\alpha(x_i, x_j) = \gamma(\delta(x_i, x_j))$. δ combines the features of x_i and x_j and γ explores the relationship between x_i and x_j expressively. In NLP, γ, δ and β is known as *Query,Key and Value*. Some examples of δ function can be

- $\delta(x_i, x_j) = f_1(x_i) + f_2(x_j)$
- $\delta(x_i, x_j) = f_1(x_i) - f_2(x_j)$
- $\delta(x_i, x_j) = f_1(x_i) \odot f_2(x_j)$
- $\delta(x_i, x_j) = [f_1(x_i); f_2(x_j)]$ (; denotes concatenation)

In Matrix Form: Let P be the set of points in a point cloud ($P \in \mathbb{R})^{\mathbb{N}\times\mathbb{F}}$ where F is the feature channels). $Q, K \in \mathbb{R}^{N\times C_k}, V \in \mathbb{R}^{N\times C_v}$.

$$Q = PW_q, K = PW_k, V = PW_v$$
$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{C_k}}\right)V$$

The time complexity of original attention is $O(N^2C_v)$ and space complexity $O(N^2 + NC_k + NC_v)$ which quadritically increases as N increases. Attention mechanisms can be categorized into two categories. In *Self Attention* Query, key, and value are derived from the same input meaning model uses attention scores for making better representation from a single representation. In *Cross Attention*, Value and Key come from input and Query comes from another input ($Q = P_1W_q, K = P_2W_k, V = P_2W_v$). It is used to query and learn conditional representation using the attention score of other feature vectors. In a point cloud segmentation network, attention is generally used for the local aggregation layer for giving adaptive weights to different features and can be used in point-based or graph-based networks. *Point Transformer*[26] uses Equation 3 for local feature aggregation layer after extracting neighbors. Each of the component in Equation 2 is approximated by an MLP. Furthermore, it uses an encoder-decoder-like structure. In each layer of the encoder, points are downsampled by a certain factor and in decoders, the points are upsampled with skip connections added for preventing information leaks. *3D Medical Point Transformer*[27] uses an Edge Convolution module for computing query value. It uses *Lambda Attention* layer which is

$$Attention(Q, K, V) = Q(softmax(K^T)V)$$

which has $O(NC_kC_v)$ time complexity and $O(NC_k + C_kC_v + C_kC_v)$ space complexity. Although it uses a cost-effective attention layer, 3DMedPT

can not perform large-scale point cloud segmentation due to computational challenges. *Point Attention network*[28] uses a new end-to-end subsampling method for downsampling the number of points and is permutation invariant and robust to noises and outliers.

4 Model

4.1 Canny Edge Detection

Our first task was to generate point clouds from the voxel images. So, we extracted edges using canny edge detection method from the 3D image to define the point clouds. These points are mostly on the surface of the organs. Canny Edge Detection is a multi-step algorithm to detect edges from an image. It follows the following steps

- *Noise Removal:* Due to image sensors, images often contain random noise, which results in pixel being different than neighboring pixels. This will result in false detection of edges. So Gaussian Filter is used to smoothen the image and remove the noise.

$$I = I_N * g(x, y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2+y^2}{2\sigma^2}} * I_N$$

  where I_N is the noisy image.

- *Gradiant Calculation:* Edges occur in places where there is large change of pixels' intensity. To detect it, gradiants I_x and I_y are calculated in x and y directions by convolving I with sobel kernel K_x and K_y.

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, K_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

  Magnitude M and slope θ can be calculated as follows

$$|G| = \sqrt{I_x^2 + I_y^2} \qquad \theta(x, y) = tan^{-1} \frac{I_y}{I_x}$$

- *Non Max Suppression:* After applying the sobel kernel on the smooth image I, edges can be seen but some edges are thin and some are thick. Non-max suppression make the images thin. The algorithm checks all the pixels and finds the one with the maximum value in the edge directions.

- *Hysteresis Thresholding:* Canny detection implements two types of pixels; strong, weak and non-relevant.

  - Strong pixels have high intensity to be considered as a pixel in edge
  - Weak pixels do not have high enough intensity to be considered as strong pixels but not small enough intensity to be considered as non-relevant for edge detection.
  - Any other pixels are considered as non-relevant pixels.

  The double threshold $\{\alpha, \beta\}$ holds for:

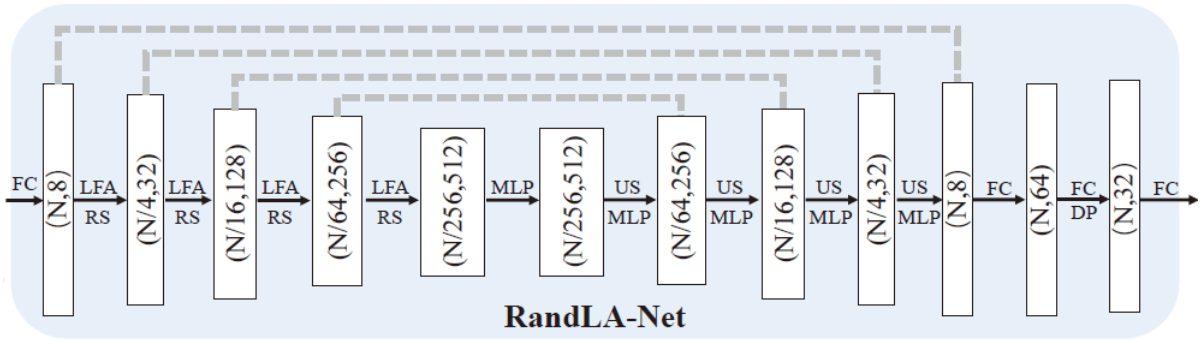

Figure 8: RandLa-Net Architecture. FC is the fully connected layer, LFA is the local feature aggregation, RS is random sampling, MLP is shared multilayer perceptron, US is upsampling and DP is dropout. (Image from [29])

- High threshold β is used to detect strong pixels.
- Low threshold α is used to detect non-relevant pixels.
- All pixels having intensity between α and β is weak pixel.

After this step, weak points are considered strong if at least one of the neighboring pixels is a strong one.

4.2 RandLa-Net

Our second task was to segment the extracted point cloud. Large-scale point cloud segmentation is a challenging task because of huge computational requirements and effective embedding learning. RandLa-Net[29] is an efficient and lightweight neural architecture that segments every point in large-scale point clouds. It is an encoder-decoder-like architecture that uses random sampling to downsample the input point cloud in the encoder and upsample the point cloud in decoder blocks. It uses random sampling compared to other sampling methods because of faster computation. Although random sampling can discard key points necessary for efficient point cloud segmentation, RandLa-Net implements attention-based local feature aggregation to effectively share features of points that are removed into the neighbor points. Figure 8 is the architecture of RandLa-Net The reasons we choose RandLa-Net for our thesis tasks are

- It is lightweight and achieves state-of-the-art results compared to existing methods. The random sampling method reduces the computation.
- The proposed Local Feature Aggregation (LFA) can expand into larger receptive fields using Local Spatial Encoding (LSE) with attentive pooling of point and neighbor features.
- The network consists of Shared MLP without any need of graph reconstuction or voxelization.
- The encoder-decoder architecture with downsampling is preferable for our task since it aims to generate discriminative latent vectors using small samples which represent our objects of interest.

A. Random Sampling

As explained in Section 3.3.1.D *Point Cloud Sampling*, random sampling is extremely fast (time complexity $O(N)$). FPS, IDIS and GS are too computationally expensive for large scale point clouds. It is invariant to any changes to the points as well as the permutation of points. The random-sampling block is added in encoder part. To compensate for the loss information, the author has added LFA module(Figure 9).

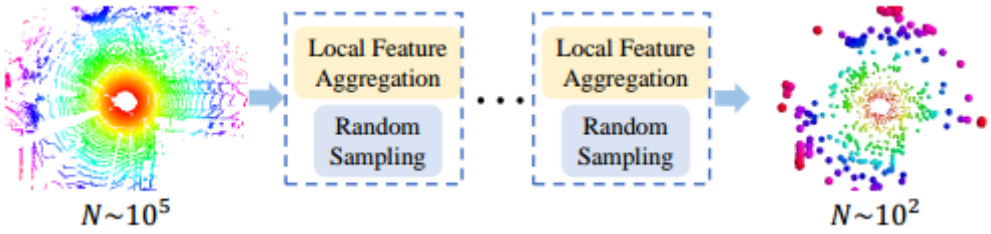

Figure 9: Random Sampling in RandLa-Net. The downsampling rate is a hyperparameter and has significant influence on model performance (Image from [29]).

B. Architecture

RandLa-Net consists of 4 encoder and 4 decoder layers (Figure 8). Each encoder layer consists of LFA modules which is shown in the bottom panel of Figure 10. LFA modules aggregate the local features and gradually expands the receptive field to perform global feature passing. Every LFA module is followed by a random sampling step. Let the input shape be $N \times d_n$, where N is the number of points in the point clouds ($N \approx 10^6$–10^7) and $d_n \in \mathbb{R}, d \geq 3$). d_n can contain the coordinates with other features like intensity, gradient or normal.

Positional Encoding: Since point clouds are unstructured, positional encoding layer embeds the positional information in an 8 dimensional vector ($3 \rightarrow 8$). This layer describes the location of a point by mapping the position/index of a point into a vector and assigning unique representation for every point. In this way, positional encoding layer makes the network more permutation-invariant.

Encoding Layer: The encoding layer progressively reduces the number of points and increases the point features. The point cloud is downsampled at each encoding layer after the dilated residual block by downsampling factor 4 ($N \rightarrow \frac{N}{4} \rightarrow \frac{N}{4^2} \rightarrow \frac{N}{4^3} \rightarrow \frac{N}{4^4}$). The per-point feature dimension is increased gradually ($8 \rightarrow 32 \rightarrow 128 \rightarrow 256 \rightarrow 512$).

Decoding Layer: In each decoder layer, points are upsampled. In each encoder layer, when a point is removed, it is stored as a reference. In subsequent decoding layer, (i.e the layer with which a skip connection is added from an encoder in Figure 8) for query reference point, KNN is used to find the nearest neighbors in the input set of points. Afterwards, features of the nearest points are copied to the target point. Subsequently, the feature maps are concatenated with the feature maps produced by corresponding encoding layers through skip connections. Then a shared MLP is applied to the concatenated feature maps.

Final Output Layer:

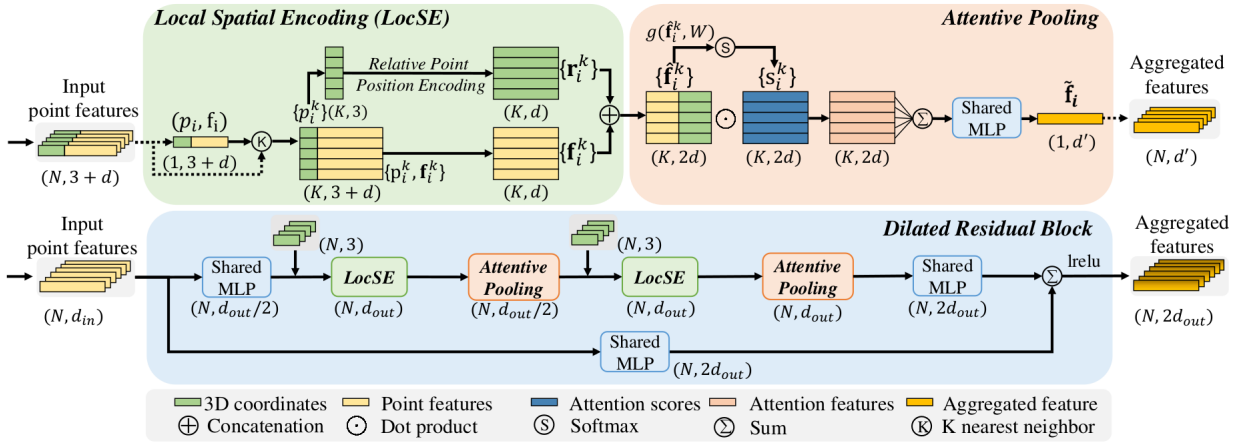

Figure 10: Feature Aggregation Module. The topleft panel shows Local Spatial Encoding that extracts features and positional information from neighbors.The topright panel shows attentive pooling to extract important features from neighboring points. The bottom layer is Dilated Residual Block which expands the receptive field (Image from [29]).

The segmentation label is predicted through three fully connected layers $(N, 64) \rightarrow (N, 32) \rightarrow (N, C)$, where C is the number of classes.

C. Local Feature Aggregation

LFA module consists of three neural units (1) LSE (2) Attentive Pooling (3) Dilated Residual Block.

C.1. Local Spatial Encoding

Let $P = \{p_1, p_2, \ldots, p_n\}, p_i \in \mathbb{R}^3$ and $F = \{f_1, f_2, \ldots, f_n\}, f_i \in R^d$ be the point set and feature set accordingly. LSE units embed the features and the spatial information from the neighbourhood points. This helps the network learn the complex local geometrical structures with as increasing receptive field.

For every point p_i, first K-Nearest Algorithm is used for finding K neighbor points. Let the set of neighbor points, $N(p_i) = \{p_1^{(i)}, p_2^{(i)}, \ldots, p_K^{(i)}\}$ and the set of features for the neighbor points be $N(f_i) = \{f_1^{(i)}, f_2^{(i)}, \ldots, f_n^{(i)}\}$. At first positional features for every point in $N(p_i)$ is encoded as follows.

$$r_k^{(i)} = MLP\left(p_i; p_k^{(i)}; (p_i - p_k^{(i)}); ||p_i - p_k^{(i)}||\right), r_k^{(i)} \in \mathbb{R}^r \tag{4}$$

; is the concatenation layer and $||\cdot||$ is the l_2 distance between neighbor and center points. $r_k^{(i)}$ not only just concatenates two positions but also the effect of one point on another point in terms of distance is also added. Once $r_k^{(i)}, \forall k = 1, 2, \ldots, K$ is computed it is concatenated with corresponding features in $N(f_i)$.(See Figure 10 Top Left). $\hat{F} = \{\hat{F}_1, \hat{F}_2, \ldots, \hat{F}_i\}, \hat{F}_i = \{\hat{f}_k^{(i)}\}_{k=1}^{k=K}, \hat{f}_k^{(i)} = \{r_k^{(i)}; f_k^{(i)}\}$.

C.2 Attentive Pooling

Attentive pooling aggregates the set of neighboring point features $\hat{F}$ with adaptive weights. Existing methods use mean or max pooling, resulting in the loss of important information. Attention mechanism will automatically learn important features. Given $\hat{F}_i = \{\hat{f}_1^{(i)}, \ldots, \hat{f}_k^{(i)}\}$, first attention scores are computed using a shared MLP g such

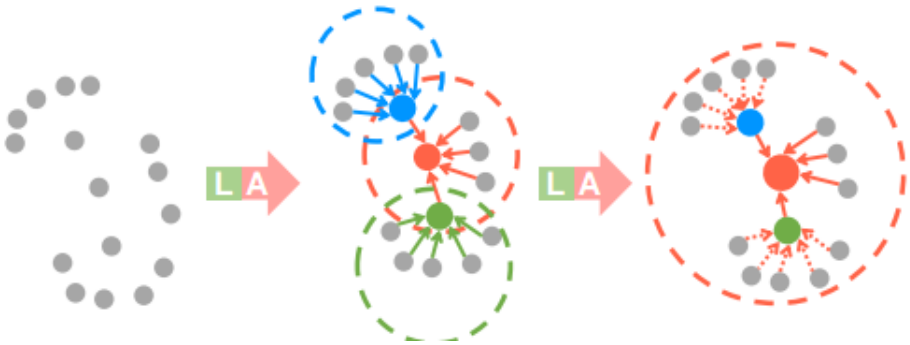

Figure 11: Illustration of dilated residual block which expands the receptive field at each step (Image from [29]).

that

$$s_k^{(i)} = g(\hat{f}_k^{(i)}, W) \tag{5}$$

where W is the weight of the MLP. After learning the attention scores feature for point p_i, f_i is updated with concatenated neighbor features.

$$\bar{f}_i = MLP(\sum_{k=1}^{K}(\hat{f}_k^{(i)} \odot s_k^{(i)})) \tag{6}$$

Together with LSE and Attentive pooling, the model learns informative features with geometric patterns for point p_i.

C.3 Dilated Residual Block

Since the point cloud is downsampled, it is necessary to expand the receptive field to preserve geometric details. Inspired by Resnet architecture, the author stacks several LSE and attentive pooling in one block before downsampling. In Figure, 11, the red points observe K features from neighboring points after the first LSE and Attentive Pooling layer and then in the next step it learns from K^2 features(See Figure 10 bottom layer). However, the more layers are added, the more the model is likely to be over-fitted. In the original paper as well as in our modified architecture(Figure 12), only two layers of LSE and Attentive pooling are used.

4.3 Loss Function

Loss functions measure the error between the ground truth and the predicted segmentation label. It is one of the most important parts of deep learning tasks. During the training phase, a loss function allows the model to learn meaningful representations and make predictions closer to ground truth. Let g_i^c be the ground truth binary indicator of class label c (i.e $g_i^c = 1$ if ith point is of class c else $g_i^c = 0$) and s_i^c is the predicted segmentation probability for class c and let N be the number of points and C be the number of classes. The loss functions used in our experiments are listed below.

A. Cross-Entropy:

Cross Entropy (CE) is the widely used loss function for classification and segmentation task. It is derived from (Kullback-Leibler) KL divergence which measures the

similarity between two probability distributions[31]. It is defined by

$$L_{CE} = -\frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c log s_i^c \tag{7}$$

Cross-Entropy loss is sensitive to class imbalance (i.e the importance of majority class can dominate other minority classes), which can lead to wrong prediction of segmentation labels for minority classes as the model will focus more on the performance of majority class and less on the loss of minority class points. Weighted Cross Entropy (WCE) loss is an extension of the CE Loss which deals with class imbalance. It is defined by

$$L_{WCE} = -\frac{1}{N}\sum_{c=1}^{C} w^c \sum_{i=1}^{N} g_i^c log s_i^c \tag{8}$$

w^c is the weight for the class c. Three types of weighing methods have been used in our experiments with RandLa-Net.

1. *Frequency Method:* Let $C_N = \{n_1, n_2, \ldots, n_k\}, \forall k = 1, 2, \ldots, C$ be the number of classes in a dataset. Let the ratio set be $R = \{r_1, r_2, \ldots, r_k\}$, where $r_i = \frac{n_i}{\sum_{i=1}^{C} n_i}$. Then the weights are $w_c = \frac{1}{r_c}$.
2. *Cost Matrix Cross Entropy (CMCE):* The cost matrix M_C for cross entropy is defined by

$$\begin{bmatrix} w^{(11)} & w^{(12)} & \ldots & w^{(1C)} \\ w^{(21)} & w^{(22)} & \ldots & w^{(2C)} \\ \vdots & \vdots & \ddots & \vdots \\ w^{(C1)} & w^{(C2)} & \ldots & w^{(CC)} \end{bmatrix}$$

where w^{ij}=the weight for a point with ground truth i and predicted label j. The main properties of a cost matrix M_C are

- Offseting the cost matrix by amount ϵ does not affect the loss.
- The cost of true prediction should be less than mean cost of all misclassifications[31, 32].

$$w^{ii} \leq \frac{1}{C^2 - C}\sum_{\substack{a,b=1 \\ a \neq b}}^{C} w^{ab}$$

.

- All the costs are non-negative ($w^{ij} \geq 0, \forall i, j = 1, 2, \ldots, C$).

B. IOULoss:
Intersection Over Union (IOU) loss directly optimizes the segmentation performance by taking into account false positives and negative error. IOU is defined by

$$IoU = \frac{TP}{TP + FP + FN}$$

where TP, FP, and FN are the number of true positives, false positives, and false negatives respectively. From the output scores of the model, these can be approximated by

$$L_{IoU} = 1 - \frac{\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c s_i^c}{\sum_{c=1}^{C}\sum_{i=1}^{N}(g_i^c + s_i^c - g_i^c s_i^c)}$$

C. Dice Loss:
Dice Loss is defined by

$$L_{Dice-Square} = 1 - \frac{2\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c s_i^c}{\sum_{c=1}^{C}\sum_{i=1}^{N}\{(g_i^c)^2 + (s_i^c)^2\}}$$

Unlike CE Loss, it does not require weights for class imbalanced tasks although it's sensitive to the shape of the objects.

4.4 Evaluation Metrics

Metrics evaluate the performance of a model. Several metrics are used to evaluate our segmentation network.

A. Recall: Recall is the true positive rate which calculates the proportions of actual positives that are predicted correctly i.e for any organ, what proportion of points that belong to the organ is predicted correctly. It is defined by

$$Recall = \frac{\#\text{true positives}}{\#\text{true positives} + \#\text{false negatives}}$$

For any class, true positives occur when model predicts the class correctly, False postives occur when model misclassifies a point as positive and false negatives when model misclassifies a positive point to other class.

B. Precision: Precision calculates what proportions of predicted positive examples are actually positive i.e for any organ what proportion of points that have been predicted to belong to the organ actually belongs to the organ. It is defined by

$$Precision = \frac{\#\text{true positives}}{\#\text{true positives} + \#\text{false positives}}$$

C. IoU: Intersection over union is the metric which evaluates the similarity of predicted object shape and the ground truth space. It is defined by

$$IoU = \frac{\#\text{true positives}}{\#\text{true positives} + \#\text{false negatives} + \#\text{false positives}}$$

.

D. Confusion Matrix: A confusion Matrix is defined by $M = (m_{ij}) \in \mathbb{R}^{C\times C}$, where m_{ij} denotes number of points belonging in class i that are predicted as class j. It shows the number of true positives, false negatives and false positives for every class.

5 Contributions

5.1 Modified RandLa-Net with Feature Extractor

While RandLa-Net inputs allow extra features such as intensity, gradient, etc, it does not learn good features even with receptive fields. So we propose to add a Feature

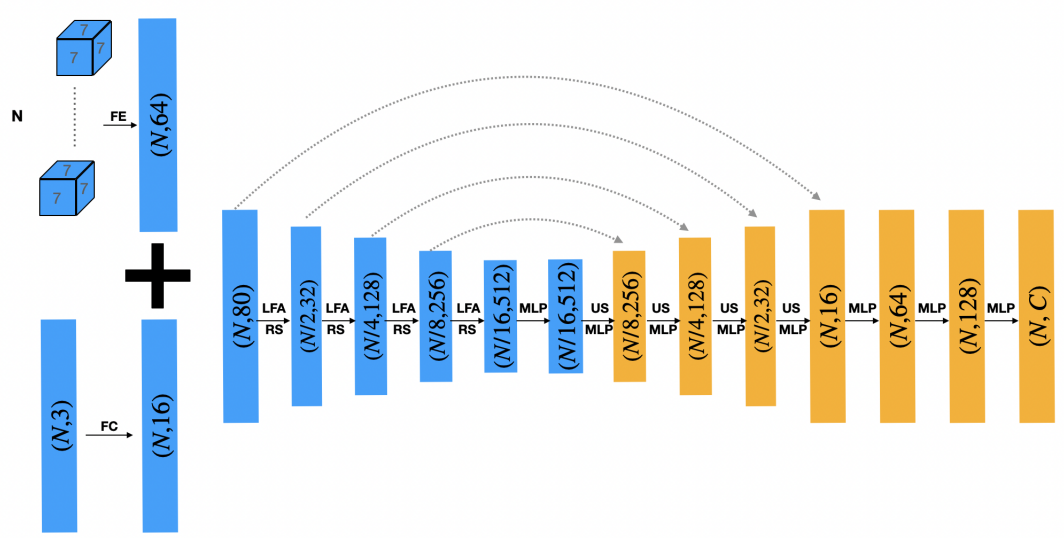

Figure 12: Modified RandLaNet with Feature Extractor.

Extractor to learn efficient features from neighboring intensity points. For every point p_i, we select the intensity of neighbor points in a $k \times k \times k$ grid ($k = 3, 5, 7$), which gives us an input size of $N \times k \times k \times k$. Since our original image is voxel-based 3D image and extracted point clouds are representative of the voxels, neighbor point clouds are the neighboring voxels in the original image.

After extracting the neighbor intensity points, the convolutional operation is per-

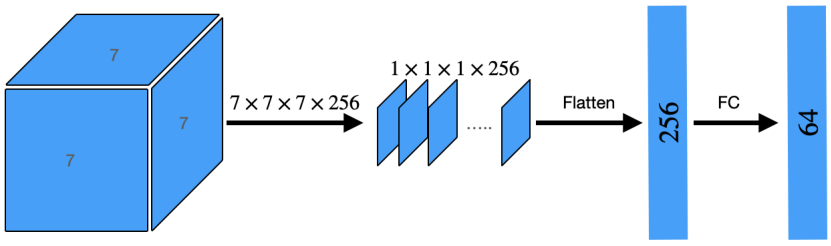

Figure 13: Proposed Feature Extractor. It is a convolutional operation followed by an FC layer. This operation is done for every N points.

formed with $k \times k \times k$ kernel size and 256 channels, which are flattened and passed through a liner layer ($(N, 256) \rightarrow (N, 64)$). Since N is huge, performing complex tasks at this stage is very computationally expensive. That is why the kernel size is equal to the neighborhood intensity shape. These features are concatenated with the output of the feature encoding layer and passed to the encoder (Figure 12). Also, the down-sampling factor has been changed to 2 instead of 4 to increase the receptive fields and allow more points to learn distinct features and also one more FC($(N, 64) \rightarrow (N, 128)$) layer has been added before the output layer to make the embedding size larger and learn more complex features.

5.2 ComboLoss:

We propose a combination of different weights in CE Loss when the frequency method can be extreme and it is defined by,

$$w_c^{\lambda} = \alpha_1 + \frac{\alpha_2}{r_c + \lambda} + \frac{\alpha_3}{r_c}, \sum_{i=1}^{3} \alpha_i = 1$$

For example as the minority class ratio gets smaller, the weight gets hyperbolic growths in frequency method, then as $x \to 0, \frac{1}{x} \to \infty$.So even if the larger weights improve the recall of the minority class, lower weight($x \to \infty, \frac{1}{1-x} \to 0$) of the majority class causes a lot misclassfication of the majority class reducing the precision of the minority class.

6 Experiments

This section presents all the experiments and the methods for our task.

6.1 Dataset

For our internship task, we have used Visceral(Visual Concept Extraction Challenge in Radiology) dataset. We used the gold corpus contrast-enhanced CT scan dataset which contains 20 volume images of dimension $512 \times 512 \times 450$ with annotation for kidneys, lungs, liver, urinary bladder, pancreas, adrenal glands, thyroid glands, aorta and some muscles. There are four modalities in the dataset which is listed in Table 1.

Identifier	Modality	Voxel Dimension(mm)
CT-Wb	CT	$0.8 - 0.9 \times 0.8 - 0.9 \times 1.5$
CTce-ThWb	contrast-enhanced CT	$0.6 - 0.7 \times 0.6 - 0.7 \times 1.2 - 1.5$
MRT1-Wb	MRI	$1.1 - 1.3 \times 1.1 - 1.3 \times 6 - 7$
MRT1cefs-Ab	contrast-enhanced MRI	$1.2 - 1.3 \times 1.2 - 1.3 \times 3$

Table 1: Different Modalities in Visceral Dataset. CTce-ThAb is used for our task.

We used CTce_ThAb because it is easy to perform edge detection on high contrast images to extract point clouds. In Visceral, Neuroimaging Informatics Technology Initiative (NIfTI)[1] format is used for the medical imaging data. To facilitate faster training, we reduced the size of the image to $128 \times 128 \times 112$. All the volume images and segmentation masks are transformed into RAS[2] reference format. RAS reference means that the axes in terms of the subject are left to right, posterior to anterior and inferior to superior(toe to head). Figure 14 shows different slices from various angles of the 3D volume image.

[1] http://nifti.nimh.nih.gov/
[2] https://nipy.org/nibabel/coordinate_systems.html

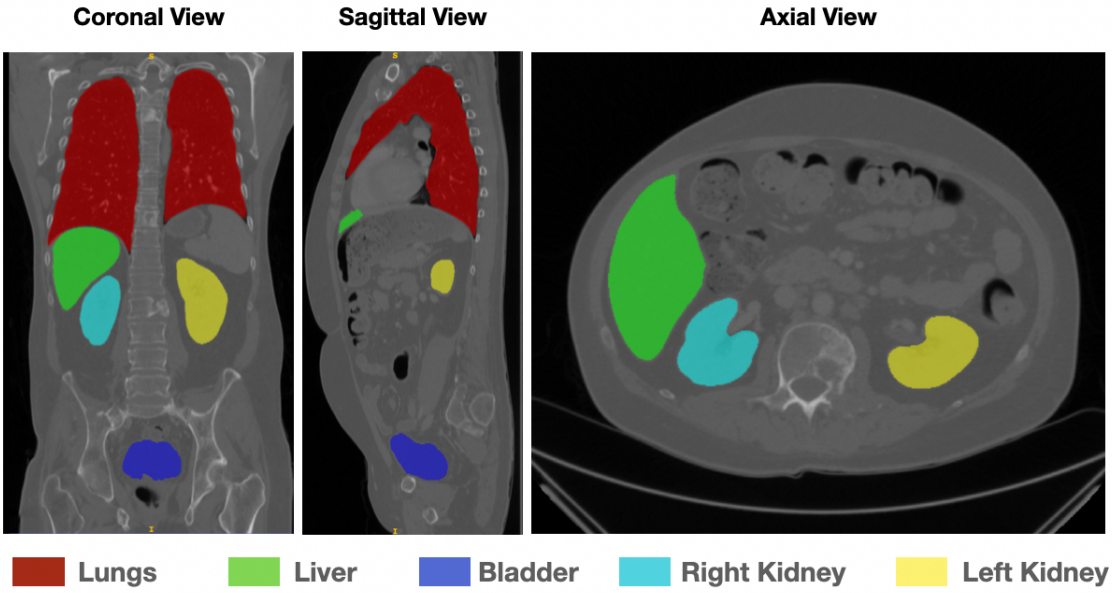

Figure 14: 2D slices of a visceral image from different view with ground truth segmentation task. The images are from contrast enhanced CT scans. For multi-class segmentation task five organs are chosen - lungs, liver, bladder, left and right kidney.

6.2 Point Cloud Extraction

Since every voxel can be represented as a point, the volumetric data itself can be transformed to point cloud data but it would take an enormous amount of space to process and train the network. We choose *lungs, liver, left and right kidney and bladder* for the segmentation task. There are several choices for sampling points from voxel-based images. Grid Sampling, Random Sampling, Contour detection, or SIFT-based keypoint detectors are some of it. Since the data has class imbalance with the background as dominating class, uniform, and random sampling will make the minority classes like bladder more scarce. Besides, the above-mentioned sampling methods will not guarantee the extraction of surface points. So we used canny edge detection to extract edges from all slices. After several trials, we choose 20 and 50 as the lower threshold and upper threshold respectively based on the maximum number of organ points that we can extract minimizing the number of background points. After extracting the edges, the edge locations in the array are considered as point coordinates(Figure 15). For every point in the point cloud, we also store the neighbor intensity values using $k \times k \times k$ grid centered around the point. We also transform the points from voxel space to scanner space (In scanner space, the magnetic isocenter is at the origin (0,0,0)). Let $P \in N \times \mathbb{R}^3$ be the positions of the voxels in the original image and R and T are the rotation and translation matrix. Let f be the mapping between voxel space to point cloud coordinate space. $f(P)$ denotes the point in point cloud space. Then

$$f(P) = PR + T \tag{9}$$

The kernel of the Feature Extractor in Figure 13 depends upon the number of intensity points we extract. We tried hyperparameter tuning for k values and chose $k = 7$.
The number of points extracted from the volume data is approximately $2 \times 10^5 \sim 3 \times 10^5$. The distribution of classes in point cloud dataset is plotted in Figure 16. So the point cloud dataset suffers from huge class imbalance as well as small training examples. So we first focused on tackling class imbalance problem with RandLa-Net.

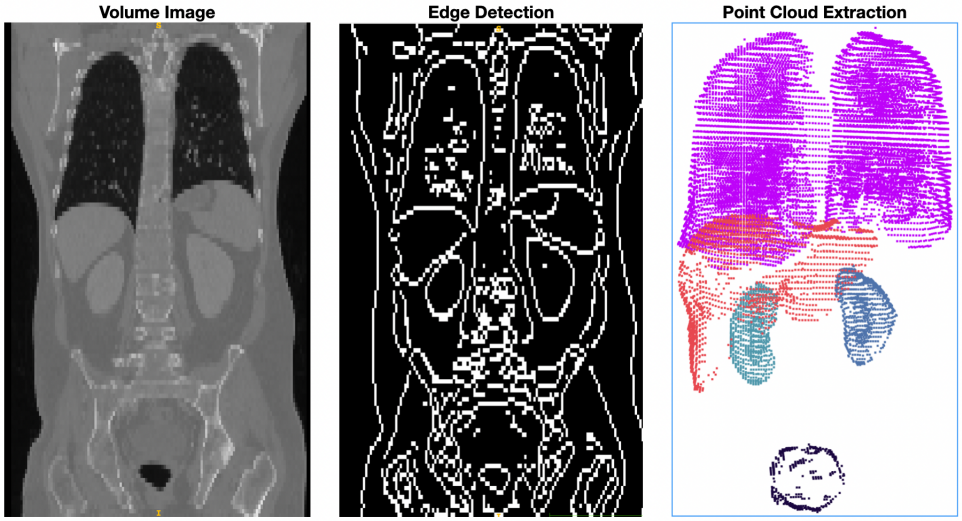

Figure 15: **Left**: Visualization of one slice of a volumetric image from visceral. **Middle**: Edge Detection of that slice. **Right**: Point cloud extracted from the volume image. The edges in the middle are considered as points representing the organ. For the ease of visualization, background points are excluded from the point cloud.

6.3 Point Cloud Segmentation

6.3.1 Train

We choose five organs for our test - lungs, liver, bladder, left and right kidney (Figure 15) for multiclass segmentation and lungs and liver separately for binary segmentation purpose. We used the PyTorch implementation of RandLa-Net[3] and for KNN, we used *pynanoflann* library[4] which uses faster KD-Tree KNN implementation. For GPU, we used Nvidia Quadro RTX 4000 8 GB and Nvidia Quadro RTX 8000 48 GB. For images of dimension $128\times128\times$, 8 GB GPU is sufficient but for $512\times512\times512$, we need 48 GB GPUs. The model takes approximately 30 hours to train. The intensity values ranges from -1024 to 1024 and are clipped within the range $[-250, 250]$ since any values outside the range refer to noise. All the input features are normalized. The parameters for training are below:

Training and Test Dataset	16 and 4	Optimizer	Adam
Downsampling Rate(DR)	2 or 4	Learning Rate	10^{-4}
Cross Validation	5 Fold	KNN k Value	8
Epochs	300 epochs per fold	Batch Size	1

Table 2: Parameters for training the RandLa-Net Model.

6.3.2 Results

1. Experiment on Loss Function

A. Multi-class Segmentation

We first started our experiments with multiclass segmentation with CE Loss. Input

[3]https://github.com/aRI0U/RandLA-Net-pytorch
[4]https://github.com/jlblancoc/nanoflann

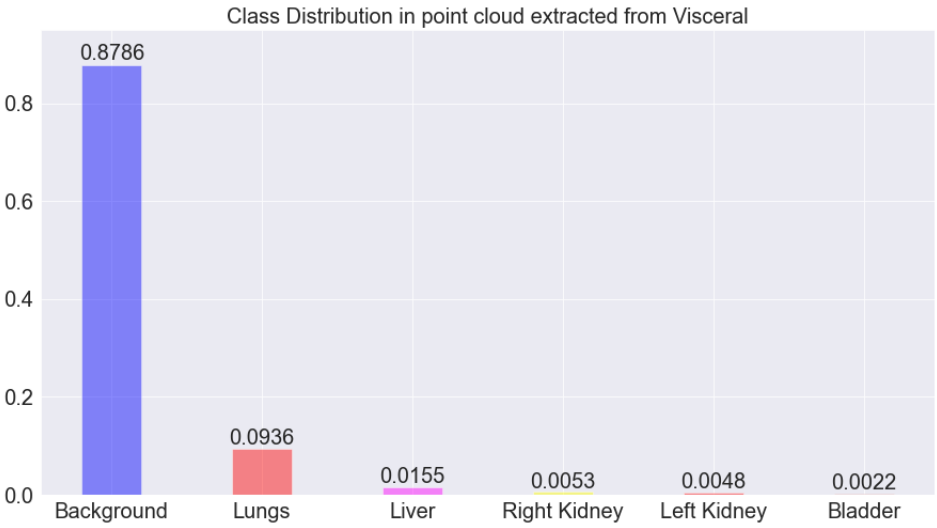

Figure 16: Class distribution of point cloud extracted from visceral dataset. The background is the most dominating class, covering almost 87% of the points.

shape is $(N, 3)$. Since we have a class imbalance problem, we first used CE loss with frequency based weighting. But since the ratio of background is very high and bladder is very low, the corresponding weights are 1.1 and 450 which underestimates the importance of background points, so we have vast number of false positives (from misclassified background points) for organs and also the model could not learn enough distinctive features for differentiating organs(Table 7 Index 1), so recall gets lower also. So we use ComboLoss which is a combination of different weights. The weights are $w_r = 0.3 + \frac{0.4}{r+0.02} + \frac{0.3}{r}$. This gives better results (Table 7 Index 2) than frequency based weighting. With IoULoss and DiceLoss seperately, the model performed very poorly, so we added ComboLoss with IoULoss and DiceLoss (Table 7 Index 8,9 and 10). But using only CL loss was the best option among the three. But there was still many false positives despite high recall.
The main reasons are the following. Considering the weight based confusion matrix where weights for all the organs are $W = \{w_1, w_2, w_3, w_4, w_5, w_6\}$ (organs in order with the columns of Table 7), we can create a weight matrix(Figure 17)

$$M_W = \begin{bmatrix} w_1 & w_1 & w_1 & w_1 & w_1 & w_1 \\ w_2 & w_2 & w_2 & w_2 & w_2 & w_2 \\ w_3 & w_3 & w_3 & w_3 & w_3 & w_3 \\ w_4 & w_4 & w_4 & w_4 & w_4 & w_4 \\ w_5 & w_5 & w_5 & w_5 & w_5 & w_5 \\ w_6 & w_6 & w_6 & w_6 & w_6 & w_6 \end{bmatrix}$$

For any background point, let the output score of the model be o_i, so in cross entropy loss, the loss for the point is $-w_1 log o_i$. This is irrespective of the predicted class for the point (if a background point is predicted correctly or incorrectly, then it is given the same weight) and the weight for background is very low and this is the main reasons for high recall for minority class and low IoU since the overwhelming number of false positives (affected by high number of misclassified background points) lower the IoU by dominating the denominator($FP >> TP + FN$). So we used Cost Matrix Cross Entropy (CMCE) for taking into account the large amount of misclassified background points which affects heavily the false positives for minority class, by giving them the same weights as the prediction class. We change the weights only

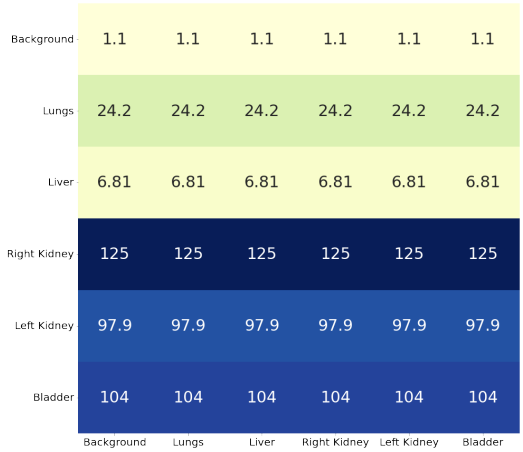

Figure 17: Weights for every class in CE **without taking into account** prediction cost.

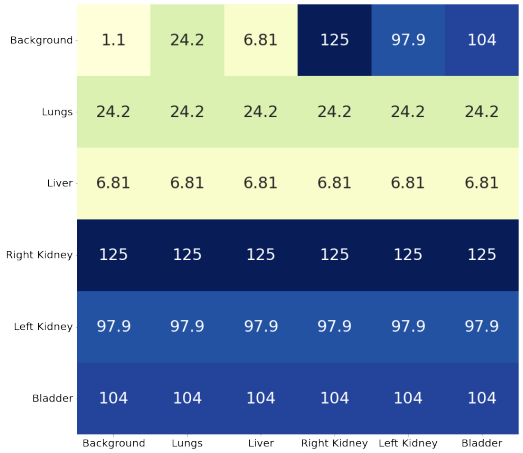

Figure 18: Weights for every class in CE **for taking into account** prediction cost.

for background class to analyse its effects (See Figure 18).
The confusion matrix in Figure 19 and Figure 20 are the results of a same model before and after applying CMCE. There has been a large reduction in false negatives for background points in the experiment after applying CMCE. It reduced almost 80% of the false negatives of background class predicted as bladder increasing the IoU of bladder by 43% despite a reduction of recall by 50%.

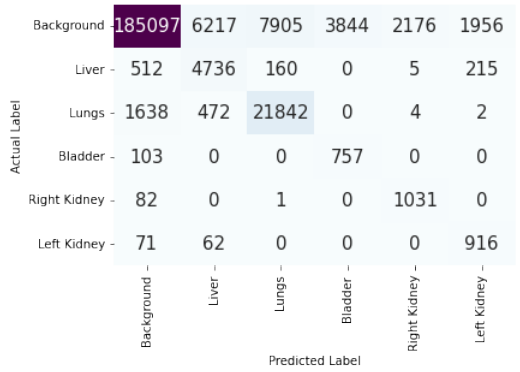

Figure 19: Confusion Matrix before applying CMCE.

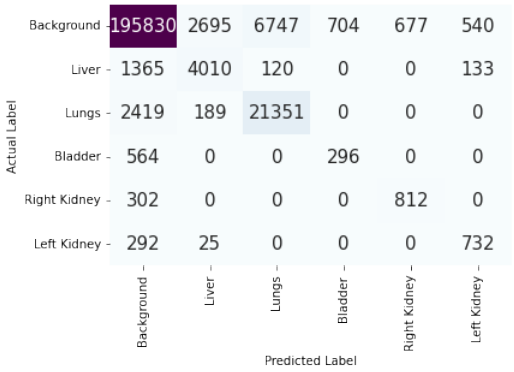

Figure 20: Confusion Matrix after applying CMCE.

Figure 21: Confusion Matrix for multiclass segmentation. RandLa-Net with Feature Extractor($7 \times 7 \times 7$) has been used with CMCE Loss with downsampling factor 2. We took the mean of the confusion matrix for all 20 images.

Using the same weights for background points misclassified as bladder points and ground truth bladder points makes the model put more emphasis on backgrounds misclassified as bladder points(due to its large quantity) which makes the recall of bladder lower which is the case here. Reducing the weights of the first row (from second column onwards) Figure 18 by 2,(i.e if a background is predicted as bladder, its weight during loss is half of the weights for the bladder class instead of being same) increases the recall of bladder by 57% but the IoU drops for the other class(Table 7 Index 12 and 13). So we conclude that using fixed weights will not be effective and we need learning based dynamic weights.

B. Binary Segmentation

We analysed the effect of weights on binary segmentation for liver and lungs with RandLa-Net+Feature Extractor($7 \times 7 \times 7$). Table 3 and 4 summarises the test results.

Experiment	Recall		IoU	
	Background	Liver	Background	Liver
RandLaNetFE + Cross Entropy (No Weights)	**0.9960** $\pm$ 2×10^{-6}	0.6918 $\pm$ 5.65×10^{-3}	**0.9878** $\pm$ 4.56×10^{-6}	**0.5748** $\pm$ 1.7×10^{-3}
RandLaNetFE + Cross Entropy (Frequency Weights)	0.9706 $\pm$ 1.8×10^{-5}	**0.9562** $\pm$ 3.1×10^{-4}	0.9692 $\pm$ 1.5×10^{-5}	0.4210 $\pm$ 1.8×10^{-3}
RandLaNetFE + Cross Entropy (CMCE)	0.9924 $\pm$ 3.84×10^{-6}	0.76$\pm$1.4$\times$ 10^{-3}	0.9862 $\pm$ 4.56×10^{-6}	0.5652 $\pm$ $6.5 \times 1o^{-4}$

Table 3: Results of Liver Segmentation for different weights in Cross Entropy Loss.

Experiment	Recall		IoU	
	Background	Lungs	Background	Lungs
RandLaNetFE + Cross Entropy(No Weights)	**0.9900** $\pm$ 6.6×10^{-7}	0.880 $\pm$ 1.1×10^{-5}	**0.9776** $\pm$ 1.1×10^{-6}	**0.806** $\pm$ 6.3×10^{-5}
RandLaNetFE + Cross Entropy(Frequency Weights)	0.9642 $\pm$ 1.1×10^{-5}	**0.979** $\pm$ 1.9×10^{-5}	0.9620 $\pm$ 1×10^{-5}	0.7372 $\pm$ 8.5×10^{-4}
RandLaNetFE + Cross Entropy(CMCE)	0.9844 $\pm$ 6.64×10^{-6}	0.8872 $\pm$ 4.1×10^{-4}	0.9722 $\pm$ 9.36×10^{-6}	0.7766 $\pm$ 5.3×10^{-4}

Table 4: Results of Lungs Segmentation for different weights in Cross Entropy Loss.

When frequency-based weights are used, the model performs better on recall for Liver and Lung but performs poorly on IoU since when we use frequency-based weights, the lower weights in the case of the background class do not give more importance to the false negatives of the background points than the true positives. Using CMCE(i.e for false negative cases of background, we use the weight of the organ) improves the IoU a bit but recall of the organ drops. When we use the same weights as the organ for misclassified background points, initially during training overwhelming number of misclassified background points dominate the loss and organ points get less importance in the loss, so we get a lot of misclassified organ points i.e recall gets lower.

2. Experiments on Feature Learning

Table 7 Index 2,3,4,5 show that using additional point features can definitely improve the IoU results. So we created a feature extractor which will automatically learn features from neighboring intensity points. These features will then be aggregated with attention module in RandLa-Net to learn distinctive features for points.
For the multi-class segmentation task, Table 7 Index 5,6,7 and 8 show the results of using feature extractors over manually creating features. This depends also on the receptive field (neighbour intensity dimension $k \times k \times k$) for the feature extractor. $3 \times 3 \times 3$ performs very poor compared to intensity and gradient features. We choose $k = 7$ for optimal performance in terms of computation and efficiency. Table 5 and 6 show the effect of Feature Extractor on Recall and IoU on binary segmentation task

Experiment	Recall		IoU	
	Background	Liver	Background	Liver
RandLaNet	$0.9830 \pm 2.68 \times 10^{-5}$	$0.6008 \pm 4.68 \times 10^{-3}$	$0.9730 \pm 7.2 \times 10^{-06}$	$0.3426 \pm 1.85 \times 10^{-4}$
RandLaNet + FE	$\mathbf{0.9924} \pm 3.84 \times 10^{-6}$	$\mathbf{0.76} \pm 1.4 \times 10^{-3}$	$\mathbf{0.9862} \pm 4.56 \times 10^{-6}$	$\mathbf{0.5652} \pm 6.5 \times 1o^{-4}$

Table 5: Results of Liver Segmentation with and without Feature Extractor.

Experiment	Recall		IoU	
	Background	Liver	Background	Liver
RandLaNet	$0.9568 \pm 5.9 \times 10^{-5}$	$0.7936 \pm 9.95 \times 10^{-4}$	$0.9348 \pm 4.14 \times 10^{-5}$	$0.5702 \pm 1.4 \times 10^{-4}$
RandLaNet + FE	$\mathbf{0.9844} \pm 6.64 \times 10^{-6}$	$\mathbf{0.8872} \pm 4.1 \times 10^{-4}$	$\mathbf{0.9722} \pm 9.36 \times 10^{-6}$	$\mathbf{0.7766} \pm 5.3 \times 10^{-4}$

Table 6: Results of Lungs Segmentation with and without Feature Extractor.

for liver and lungs.

3. Experiments on Downsampling Rate:
In every encoder block, RandLa-Net randomly removes some points which makes the learning faster but reduces receptive field and poorer feature quality in the latent vector. The original paper has downsampling rate 4. Let $N = \{n_1, n_2, n_3, n_4, n_5, n_6\}$ be the number of organ points for background, liver, lungs, bladder, left and right kidney. After four encoder steps every organs have almost n_i/d^4 where d is the downsampling factor. Since n_4 for bladder is very low, in the latent vector space only a small fraction of bladder points (typically number of bladder points $300 - 500$ for 200000 points in point cloud) are present. So we decreased the rate d to 2 and found significant improvements in IoU with almost 100% increase in bladder IoU, 7.6% for liver, 2.6% for lungs, 44% for right kidney and 22% for left kidney. (Table 7 Index 11,12,13)

4. Experiments on Latent Vector:
Since all the weighting based CE methods were ineffective to improve the segmentation and solve the class imbalance problem, we wanted to verify if the latent feature vectors in the network are as discriminative as they should be for points belonging to different classes. Since the downsampling steps reduce the number of points for minority class, the global features for organs are influenced only by a small number of points having higher embedding size but lower receptive fields to learn local spatial features. For effective segmentation, points belonging to the same class should have similar latent vector and points belonging to different class should have different latent vectors (i.e distance between latent vectors must be higher). We infer from Table 7, that the model is not learning enough distinctive features for every organs and this is the reason false negatives are more prevalent for cases with dominating classes like background and lungs.
So we used the embedding from Figure 12 with shape $(\frac{N}{16}, 512)$ after applying MLP.

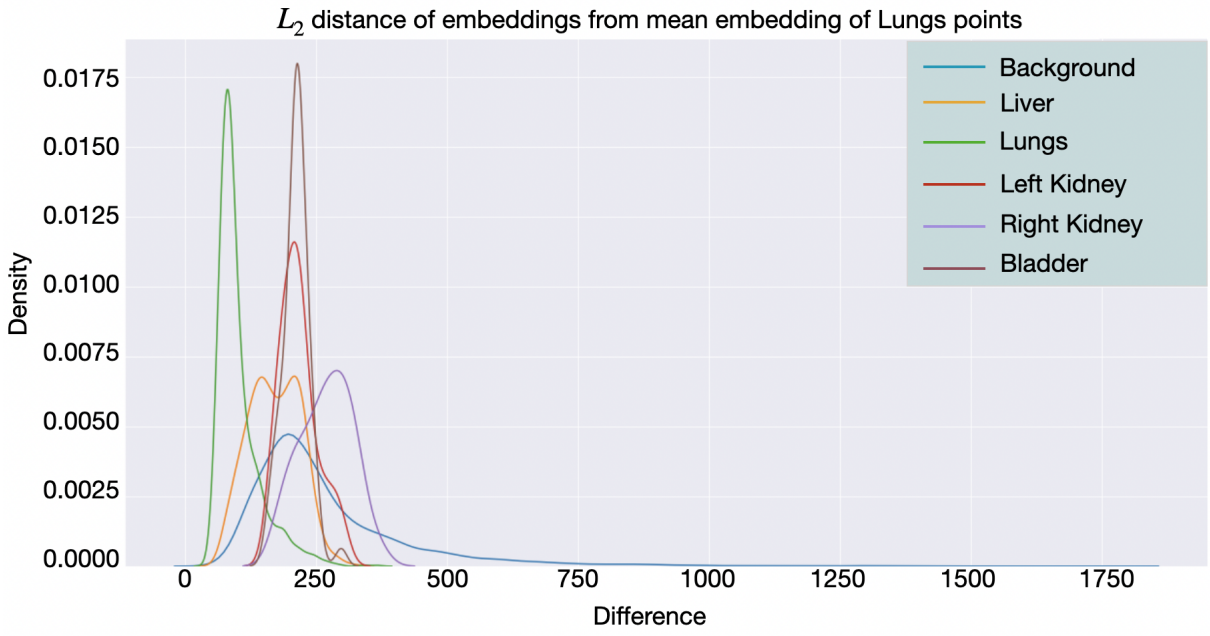

Figure 22: L2 Distance of latent vectors from mean lungs latent vector.

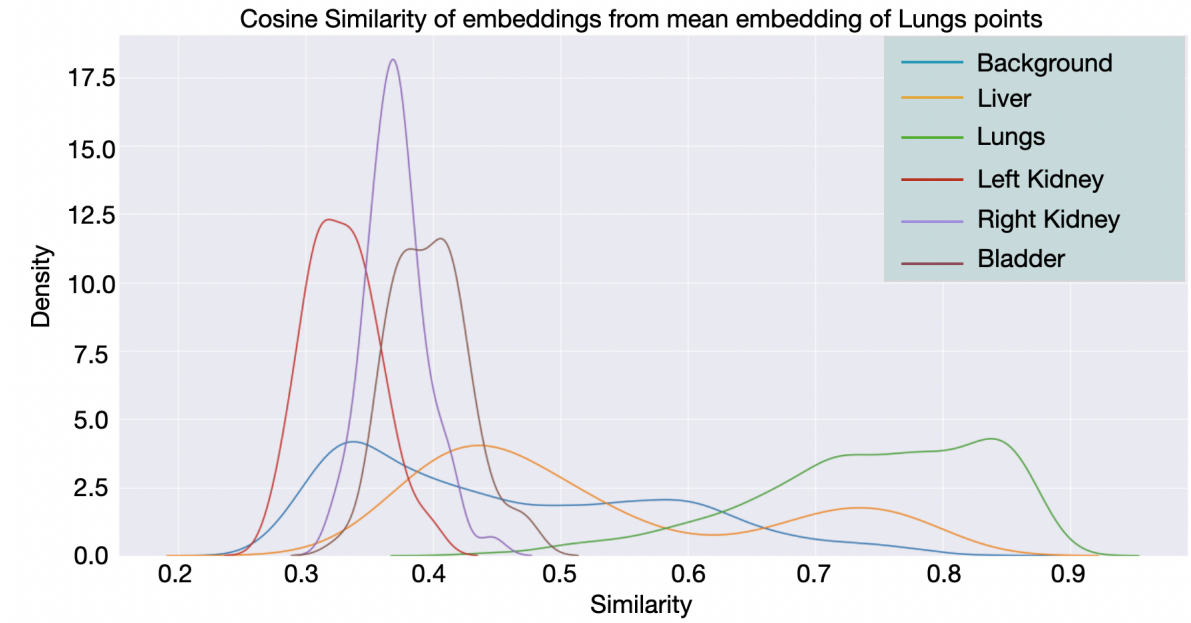

Figure 23: Cosine similarity of latent vectors from mean lungs latent vector.

Let $L^c = \{l_i^c\}_{i=1}^{N^c}$ be the set of latent vectors for every class c, N^c is the number of points of class c. For c=lungs, we calculate the mean latent vector, $M_c = \sum_{i=1}^{N^c} l_i^c$ and then we calculate the l_2 distance $\{||M_c - l_i^{\tilde{c}}||_2\}_{i=1}^{i=N^{\tilde{c}}}, \forall \tilde{c}$ and cosine similarity $\{\frac{M_c \cdot l_i^{\tilde{c}}}{||M_c|| \cdot ||l_i^{\tilde{c}}||}\}_{i=1}^{i=N^{\tilde{c}}}, \forall \tilde{c}$. We plot the l_2 distance in Figure 22 and cosine similarity in Figure 23. Some of the points in the liver have similar latent vectors as lungs as can be seen in the two figures, this is one of the reasons for liver points being misclassified as lungs in Figure 20. The similarity distribution for lungs has higher variance due to the difference in features in left and right lung which has been merged in our task. So for future tasks, we will add measure to make the latent vector for same organs

are similar and for different organs apart from each other.

Index	Experiment	Input Shape	Metric	Background	Liver	Lungs	Bladder	Right Kidney	Left Kidney	Mean
1	CE (w=$\frac{1}{r}$)+DR(4)	$(N,3)$	Recall	0.486	0.803	0.832	0.235	0.973	0.654	0.523
			IoU	0.482	0.12	0.271	0.024	0.035	0.104	0.173
2	ComboLoss(CL)+DR(4)	$(N,3)$	Recall	0.6274	0.828	0.8378	0.9336	0.978	0.9602	0.86
			IoU	0.6244	0.142	0.353	0.0398	0.124	0.1166	0.233
3	Intensity+CL+DR(4)	$(N,4)$	Recall	0.688	0.8219	0.875	0.968	0.968	0.951	0.878
			IoU	0.6846	0.146	0.4508	0.0492	0.1416	0.1052	0.263
4	Gradient+CL+DR(4)	$(N,4)$	Recall	0.63	0.835	0.8516	0.951	0.95	0.93	0.86
			IoU	0.627	0.16	0.38	0.032	0.107	0.1192	0.237
5	Gradient+Intensity+ CL+DR(4)	$(N,5)$	Recall	0.8366	0.7960	0.8300	0.9100	0.9300	0.8900	0.865
			IoU	0.8238	0.2954	0.5158	0.1350	0.2110	0.1916	0.362
6	FE(3)+CL+DR(4)	$(N,3)$,$(N,3,3,3)$	Recall	0.8134	0.9210	0.7982	0.9570	0.9730	0.9334	0.899
			IoU	0.8020	0.2628	0.6816	0.1092	0.1450	0.1166	0.3529
7	FE(9)+CL+DR(4)	$(N,3)$,$(N,9,9,9)$	Recall	0.823	0.9070	0.8206	0.9628	0.9706	0.9416	0.9043
			IoU	0.8114	0.2770	0.6946	0.1060	0.1440	0.1280	0.36
8	FE(7)+CL+DR(4)	$(N,3)$,$(N,7,7,7)$	Recall	0.826	0.8985	0.9295	0.9365	0.9485	0.882	0.903
			IoU	0.8205	0.336	0.624	0.1435	0.215	0.2015	0.39
9	FE(7)+IoULoss+ CL+DR(4)	$(N,3)$,$(N,7,7,7)$	Recall	0.646	0.9185	0.9375	0.946	0.9905	0.919	0.893
			IoU	0.6425	0.27	0.49	0.0875	0.1115	0.1105	0.285
10	FE(7)+Dice+ CL+DR(4)	$(N,3)$,$(N,7,7,7)$	Recall	0.9566	0.7324	0.7904	0.2064	0.6248	0.5018	0.6354
			IoU	0.9222	0.4912	0.6734	0.1246	0.2782	0.3258	0.4692
11	FE(7)+DR(4)+CMCE+ Comb-Weight	$(N,3)$,$(N,7,7,7)$	Recall	0.9616	0.676	0.8282	0.178	0.5636	0.589	0.633
			IoU	0.9294	0.4584	0.7296	0.1084	0.2738	0.328	0.471
12	FE(7)+DR(2)+CMCE+ Comb-Weight/2	$(N,3)$,$(N,7,7,7)$	Recall	0.9258	0.9290	0.7870	0.7164	0.9042	0.8654	0.8546
			IoU	0.9020	0.3522	0.7488	0.2490	0.3156	0.3446	0.4854
13	FE(7)+DR(2)+CMCE+ Comb-Weight	$(N,3)$,$(N,7,7,7)$	Recall	0.9446	0.8112	0.9118	0.4564	0.7924	0.852	0.795
			IoU	0.9282	0.4964	0.7488	0.2266	0.4006	0.3978	0.533

Table 7: Experiment Results for multi-class segmentation. CL is the Cross Entropy Loss with combination weights ($w_r = 0.3 + \frac{0.4}{r+0.02} + \frac{0.3}{r}$). FE(n) is RandLa-Net with Feature Extractor with $n \times n \times n$ intensity shape as explained Section 5.1. DR(n) is the random downsampling rate (In every encoder block $\frac{1}{n}$% points are randomly removed). Comb weight is the weight for CL. CMCE is the cost matrix cross entropy as explained in Section 4.3.A. The best results are in red.

6.4 Voxel Segmentation

Our final goal is to segment all the voxels in the original 3D image in the Visceral dataset. Since every voxel has a bijection mapping from voxel space to point cloud space (i.e every voxel can be mapped to a point in the point cloud using Equation 9), we can map a query voxel in the point cloud space and segment it using trained model. The shape of the organ will then be the decision boundary of the model. For faster approximation of the decision surface we have used the following steps. Let

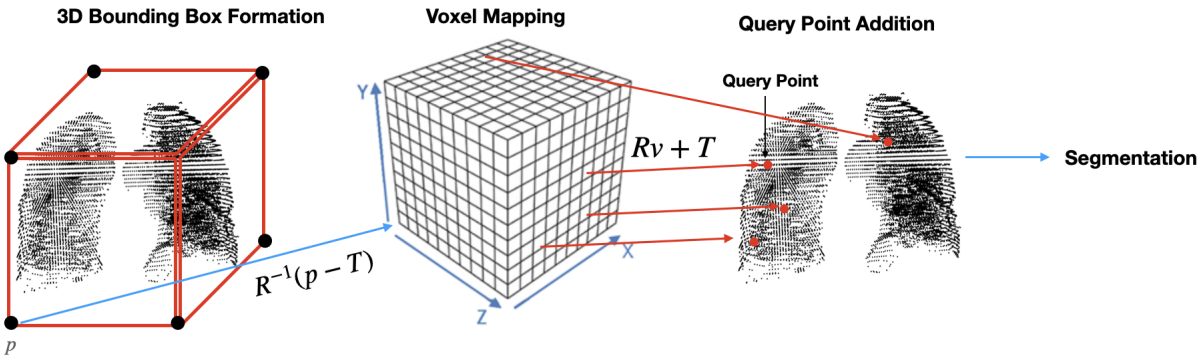

Figure 24: Illustration of voxel segmentation. 3D bounding box is formed using the min and max coordinate from the predicted label. These points are mapped to voxel space and all the voxels in between the bounding box are selected. These new query voxels are mapped to point cloud space and added to the input point cloud and passed to the trained model for predicting segmentation label.

P be the input point cloud, M be the trained RandLa-Net with Feature Extraction Layer and $L = M(P)$ be the predicted label of the input point cloud.

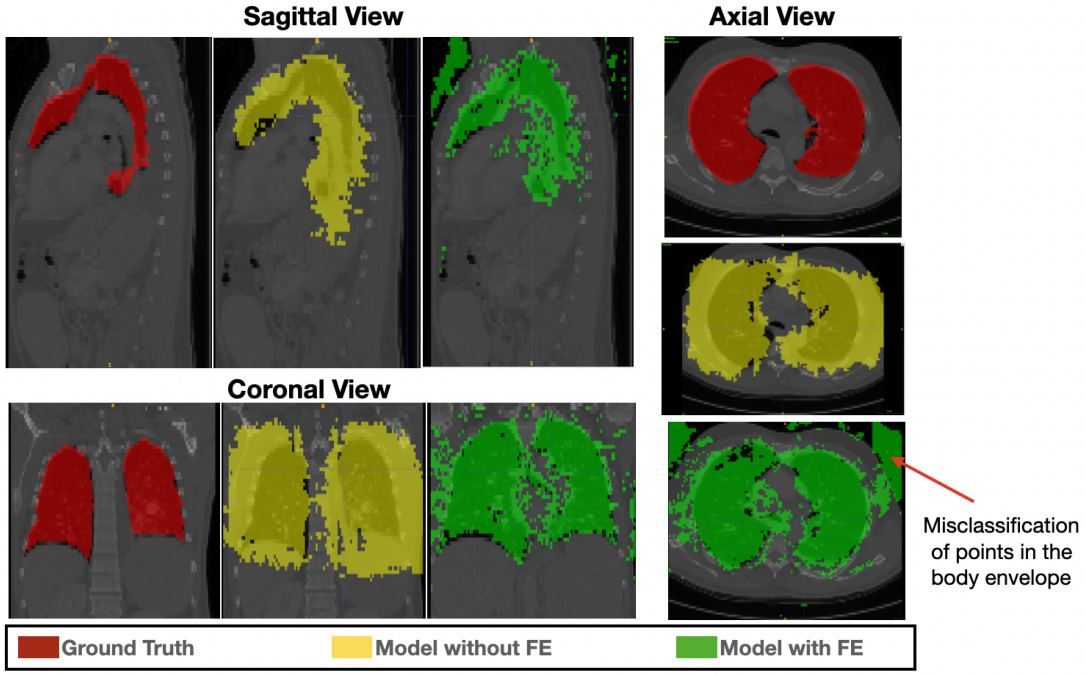

Figure 25: Voxel Segmentation of Lungs after mapping the voxels in the point cloud space and adding it to the input point cloud. RandLa-Net has been used for training the binary lungs segmentation task with CMCE Loss. FE is the Feature Extraction Layer.

1. We find the 3D bounding box for every organ using the min p_{min} and max p_{max} coordinates of the predicted segmentation label.

2. Once we find min and max coordinates, we map those two points using $f^{-1}(x) = R^{-1}(x - T)$ where R and T are the rotation and translation matrix for transformation from voxel space to scanner space(or in our case point cloud space). Let $v_{min} = f^{-1}(p_{min}), v_{max} = f^{-1}(p_{max})$.

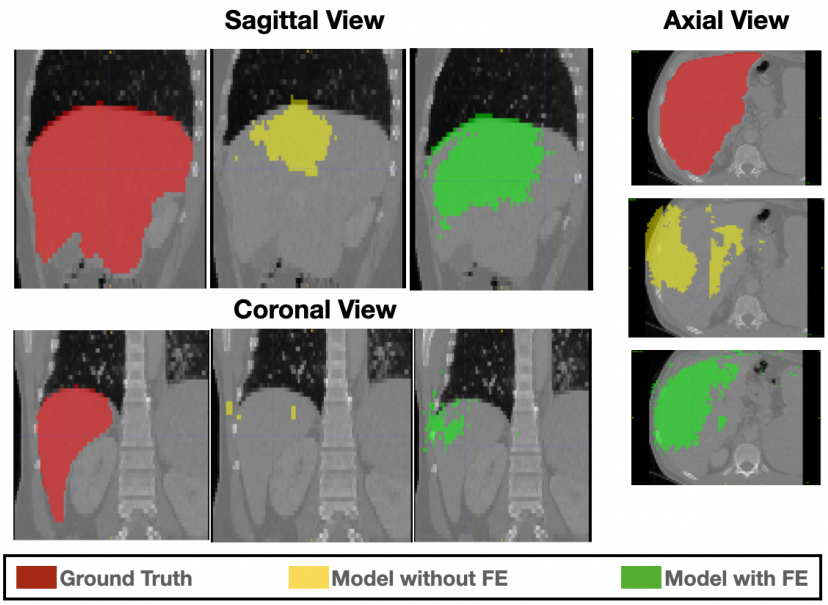

Figure 26: Voxel Segmentation of Liver after mapping the voxels in the point cloud space and adding it to the input point cloud. RandLa-Net has been used for training the binary liver segmentation task with CMCE Loss. FE is the Feature Extraction Layer.

3. Then we store all the voxels(D) in the 3d grid spanning from v_{min} to v_{max} .

4. We project all the voxels to point cloud space using $f(x) = Rx + T$ and add it to the input point cloud (i.e the extracted point cloud using edge detection), $N \rightarrow N + D$.

5. Finally we segment the new point cloud using the trained model. We pass the new point cloud as the input to the trained model to segment the labels for the new points.

The above steps are followed to approximate the decision boundary faster. We can segment all the $128 \times 128 \times 112$ voxels (Original Visceral image dimension is $512 \times 512 \times 450$, we downsized the image to $128 \times 128 \times 112$). This method of segmentation is model-independent, so any model can be used to segment the voxel by mapping them to the point cloud space. Although models having downsampling methods in intermediate feature learning layers can perform faster segmentation.

The extracted input point cloud (using Canny Edge Detection) contains point with high gradients and they are mostly surface points and the newly added query points are of low gradients and are mostly inside points of the organs. Our modified RandLa-Net with Feature Extraction performs better on the new query points than the original RandLa-Net. Figure 25 and 26 shows the voxel segmentation for lungs and liver respectively. The Feature Extraction layer learns the local spatial information using the neighbourhood intensity values to incorporate local features with the point features. This is why points outside the lungs in Figure 25 are less misclassified compared to the prediction of model without FE. But points in the body envelope in Figure 25 contain similar geometrical properties as points inside the lungs, which is why we see a lot of false positives on the body envelope which is absent in the case of model without feature extraction. We infer that using SDF learning, the model can

learn different features for points in the body envelope and inside the lungs points for effective segmentation.

7 Conclusions

In this report, we analysed a novel approach to segment 3d volumetric medical images using point cloud segmentation. We extracted surface points of the organs using canny edge detection and applied modified RandLa-Net with Feature Extraction Layer which performs significantly better than the original RandLa-Net[29]. We extracted the decision boundary of our trained model to accomplish the final task which is segmenting the voxels in the original volume image. Because of one-to-one mapping between voxel space and point cloud space on which the model is trained, we can map the voxel position to point cloud space and predict its label. Thus after training the model on high gradient points, we can query the label of any voxel after mapping to it the point cloud space and adding it to the original input point cloud. We also performed numerous experiments on the internal architectural problems related to RandLa-Net for its incapability to have good performance on minority classes. We conclude that it has been caused by not learning rich and distinctive features in the latent vector space for each class and the downsampling factor which decreases the receptive field causing loss of important features. Also we need to provide the suitable cost value in CMCE loss to solve the effect of class imbalance. Since our segmentation output in the final task contains many extraneous regions in Figure 25 and the decision boundaries are not smooth, the present model may not have preserved the anatomical structure of the organs. In our future work, we will use signed distance function to learn the decision boundary which separates the outside and the inside points of the organs depending on whether the SDF is positive or negative. For binary organ segmentation task, we can predict SDF only and the decision boundary will be the points whose predicted SDF value is zero. But for multi-organ segmentation, we will perform multi-label classification with two heads in the last layer of RandLa-Net with FE (Figure 12) to predict segmentation label as well as SDF value. We can extract the ground truth SDF value using the segmentation mask and use l_2 as distance metric. l_1 loss can be used as well as product loss[36] to penalize the wrong sign in predicted SDF value.

To counter the effect of class imbalance, we will learn the weights [32] by adding another layer in the model after output layer. These weights will be used to calculate the final softmax output. For example if an output score is $\{o_1, o_2, \cdots, o_c\}$ and the weight scores are $\{\xi_1, \xi_2, \cdots, \xi_c\}$, where c is the number of classes, then the softmax output of ith term is,

$$\hat{y}_i = \frac{\xi_i o_i}{\sum_{j=1}^{N} \xi_j o_j}$$

After this, we can use CE Loss without any weights.

Also, random sampling can be changed to learning-based sampling[28] for removing more irrelavant background points rather than random points in the organs. One other challenge is to learn distinctive features in latent space for learning shapes of different organs. In order to achieve that the embeddings of same organs should be of similar nature and different organs should have very dissimilar embeddings. So we will use a loss function[33] to minimize the variance of embeddings between same organ

points and increase the distance between different organ points. This will generate discriminative embeddings for points belonging to different organs.

References

[1] Eman Ahmed, Alexandre Saint, Abd El Rahman Shabayek, Kseniya Cherenkova, Rig Das, Gleb Gusev, Djamila Aouada, Bjorn Ottersten. *A survey on Deep Learning Advances on Different 3D Data Representations.* arXiv e-prints,2018

[2] S Niyasa, S J Pawana, M Anand Kumarb, Jeny Rajan. *Medical Image Segmentation with 3D Convolutional Neural Networks: A Survey.* Neurocomputing, Volume 493, 2022, Pages 397-413, ISSN 0925-2312,

[3] Saifullahi Aminu Bello , Shangshu Yu, Cheng Wang. *Review: deep learning on 3D point clouds*. Remote Sensing 12, No. 11:1729.

[4] Risheng Wang, Tao Lei, Ruixia Cui, Bingtao Zhang, Hongying Meng and Asoke K. Nandi. *Medical Image Segmentation Using Deep Learning: A Survey*. IET Image Process. 16, 1243– 1267 (2022).

[5] Anh Nguyen, Bac Le. *3D Point Cloud Segmentation - A Survey*. 2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM), 2013, pp. 225-230.

[6] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, Andreas Geiger. *Occupancy Networks: Learning 3D Reconstruction in Function Space*. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[7] Jeong Joon Park,Peter Florence, Julian Straub, Richard Newcombe, Steven Lovegrove. *DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation*. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[8] Ramesh N, Yoo JH, Sethi IK. *Thresholding based on histogram approximation* . IEE Proceedings-Vision, Image and Signal Processing 142.5 (1995): 271-279.

[9] Hancock ER, Kittler J. *Edge labeling using dictionary-based relaxation*. IEEE Transactions on Pattern Analysis and Machine Intelligence 12.2 (1990): 165-181.

[10] Olaf Ronneberger, Philipp Fischer, Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science(), vol 9351. Springer, Cham.

[11] Robert Popilock, Kumar Sandrasagaren, Lowell Harris, and Keith A. Kaser *CT Artifact Recognition for the Nuclear Technologist*. Journal of Nuclear Medicine Technology June 2008, 36 (2) 79-81.

[12] Neeraj Sharma and Lalit M. Aggarwal. *Automated medical image segmentation techniques*. Journal of medical physics vol. 35,1 (2010): 3-14.

[13] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, Olaf Ronneberger. *3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation*. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016.

[14] Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas. *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77-85.

[15] M. Fan. *Variants of Seeded Region Growing*. Image Processing, IET · June 2015

[16] Hang Su, Subhransu Maji ,Evangelos Kalogerakis, Erik Learned-Miller. *Multi-view Convolutional Neural Networks for 3D Shape Recognition*. 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 945-953

[17] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, Qian-Yi Zhou. *Tangent Convolutions for Dense Prediction in 3D*. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)

[18] Zhijian Liu, Haotian Tang, Yujun Lin, Song Han. *Point-Voxel CNN for Efficient 3D Deep Learning*. Proceedings of the 33rd International Conference on Neural Information Processing Systems 2019.

[19] Charles R. Qi, Li Yi, Hao Su, Leonidas J. Guibas. *PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space*. Proceedings of the 31st International Conference on Neural Information Processing Systems 2017.

[20] Hengshuang Zhao; Li Jiang; Chi-Wing Fu; Jiaya Jia. *PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing*. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5560-5568.

[21] Weijing Shi and Ragunathan (Raj) Rajkumar. *Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[22] Loic Landrieu, Martin Simonovsky. *Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs*. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4558-4567.

[23] Yanni Ma, Yulan Guo, Hao Liu, Yinjie Lei, Gongjian Wen. *Dynamic Graph CNN for Learning on Point Clouds.* 2020 IEEE Winter Conference on Applications of Computer Vision (WACV).

[24] Y. Ma et al. *Global Context Reasoning for Semantic Segmentation of 3D Point Clouds*. 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), 2020, pp. 2920-2929.

[25] Hengshuang Zhao, Jiaya Jia, Vladlen Koltun *Exploring Self-attention for Image Recognition*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10073-10082.

[26] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, V. Koltun. *Point Transformer.* 2021 IEEE/CVF International Conference on Computer Vision (ICCV).

[27] Jianhui Yu, Chaoyi Zhang, Heng Wang, Dingxin Zhang, Yang Song, Tiange Xiang, Dongnan Liu, Weidong (Tom) Cai. *3D Medical Point Transformer: Introducing Convolution to Attention Networks for Medical Point Cloud Analysis*. ArXiv 2021.

[28] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, Qi Tian. *Modeling Point Clouds with Self-Attention and Gumbel Subset Sampling*. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019 pp. 3318-3327.

[29] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, A. Trigoni, A. Markham. *RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[30] Ze Liu, Han Hu, Yue Cao, Zheng Zhang, Xin Tong. *A Closer Look at Local Aggregation Operators in Point Cloud Analysis*. ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII, Pages 326–342.

[31] Jun Ma, Jianan Chen, Matthew Ng, Rui Huang, Yu Li, Chen Li, Xiaoping Yang, Anne L. Martel. *Loss odyssey in medical image segmentation*. Medical Image Analysis, Volume 71, 2021, 102035, ISSN 1361-8415,

[32] Salman H. Khan, Munawar Hayat, Mohammed Bennamoun,Ferdous A. Sohel, Roberto Togneri *Cost-Sensitive Learning of Deep Feature Representations from Imbalanced Data.* IEEE Transactions on Neural Networks and Learning Systems (Volume: 29, Issue: 8, Aug. 2018)

[33] Zhidong Liang, Ming Yang, Hao Li and Chunxiang Wang *3D Instance Embedding Learning with a Structure-Aware Loss Function for Point Cloud Segmentation.* 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),October 25-29, 2020

[34] Evangelos Kalogerakis Aaron Hertzmann Karan Singh. *Learning 3D Mesh Segmentation and Labeling*. ACM Trans. Graph. 29, 4, Article 102 (July 2010), 12 pages.

[35] Razmig Kéchichian et al *Automatic Multiorgan Segmentation via Multiscale Registration and Graph Cut*. IEEE Transactions on Medical Imaging, Institute of Electrical and Electronics Engineers, 2018, 37 (12), pp.2739-2749.

[36] Yuan Xue, Hui Tang, Zhi Qiao, Guanzhong Gong, Yong Yin, Zhen Qian, Chao Huang, Wei Fan, and Xiaolei Huang. *Shape-Aware Organ Segmentation by Predicting Signed Distance Maps.* In Proceedings of the Thirty-Fourth AAAI Conference on Artifical Intelligence, 2020

[37] Igor Dydenko, Fadi Jamal, Olivier Bernard, Jan D'hooge, Isabelle E Magnin, Denis Friboulet *A level set framework with a shape and motion prior for segmentation and region tracking in echocardiography*. Medical image analysis, Volume 10, Issue 2, Pages 162-177.

[38] Fausto Milletari, Nassir Navab, Seyed-Ahmad Ahmadi. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. 2016 Fourth International Conference on 3D Vision (3DV), 2016, pp. 565-571.