SEMANTIC SEGMENTATION IN SEARCH AND RESCUE ENVIRONMENTS

by

151220162018   ÜMÜT BERKİN METİN

151220162049   DOĞUKAN KAAN BOZKURT

151220162050   BATUHAN KENDÜZ

A Graduation Project Report

Electrical Electronics Engineering Department

JUNE 2021

SEMANTIC SEGMENTATION IN SEARCH AND RESCUE ENVIRONMENTS

by

151220162018 ÜMÜT BERKİN METİN

151220162049 DOĞUKAN KAAN BOZKURT

151220162050 BATUHAN KENDÜZ

A Report Presented in Partial Fulfilment of the Requirements for the Degree Bachelor of Science in Electrical Electronics Engineering

ESKISEHIR OSMANGAZI UNIVERSITY

JUNE 2021

SEMANTIC SEGMENTATION IN SEARCH AND RESCUE ENVIRONMENTS

by

151220162018   ÜMÜT BERKİN METİN

151220162049   DOĞUKAN KAAN BOZKURT

151220162050   BATUHAN KENDÜZ

has been approved by

[illegible]

Asst. Prof. Dr. Burak Kaleci

[illegible]

Asst. Prof. Dr. Helin Dutağacı

[illegible]

Res. Asst. Kaya Turgut

[illegible]

Prof. Dr. Gökhan Çınar, Chairperson

ABSTRACT

In this thesis, we present implementation of point-based deep learning architectures that can be used for semantic segmentation of search and rescue environments. In recent years, semantic segmentation problem has been considered to interpret the scenes. The problem aims to segment a scene into semantic regions. For this reason, it is appropriate to determine walls, ramps, and terrain surfaces in a search and rescue environments. To achieve that we preferred to utilize point cloud data that is a set of points in 3D space. Besides, the points can have crucial features such as point normal and color information. In this way, the characteristic of a scene can be described in a proper representation. We used A-SCN, ELGS, Kd-Net, PointConv, SO-Net, and SpiderCNN architectures on the same scenes in training and testing. The ESOGU RAMPS dataset which includes scenes from a simulated environment was used in the experimental works. The test results show that the different architectures selected produce generally successful results on the same training and test scenes.

Keywords: *deep learning, semantic segmentation, search and rescue.*

ÖZET

Bu tezde, arama ve kurtarma ortamlarının anlamsal bölümlenmesinde kullanılabilecek nokta tabanlı derin öğrenme mimarilerinin uygulamasını sunuyoruz. Son yıllarda sahneleri yorumlamak için anlamsal bölütleme problemi gündeme gelmiştir. Problem, bir sahneyi anlamsal bölgelere ayırmayı amaçlar. Bu nedenle arama kurtarma ortamlarında duvarların, rampaların ve yer yüzeylerinin belirlenmesi yapılabilir. Bunu başarmak için 3B uzayda bir dizi nokta olan nokta bulutu verilerini kullanmayı tercih ettik. Ayrıca noktalar, nokta normali ve renk bilgisi gibi çok önemli özelliklere sahip olabilir. Bu şekilde, bir sahnenin karakteristiği uygun bir temsilde tanımlanabilir. Eğitim ve testlerde aynı sahnelerde A-SCN, ELGS, Kd-Net, PointConv, SO-Net ve SpiderCNN mimarilerini kullandık. Deneysel çalışmalarda simüle edilmiş bir ortamdan sahneleri içeren ESOGU RAMPS veri seti kullanılmıştır. Test sonuçları, seçilen farklı mimarilerin aynı eğitim ve test sahnelerinde genel olarak başarılı sonuçlar verdiğini göstermektedir.

Anahtar Kelimeler: *derin öğrenme, anlamsal sınıflandırma, arama ve kurtarma.*

ACKNOWLEDGEMENT

We would like to thank our supervisor, Asst. Prof. Dr. Burak Kaleci, for his useful comments and on-going support. We would like to thank our Research Assistant Kaya Turgut, for his suggestions regarding the collection of results and useful comments on early drafts of our thesis.

Res. Asst. Kaya Turgut
Electrical-Electronics Engineer, MSc,
Profession: Computer Vision, Robotics
Experience: 7 years,
Organization: Eskisehir Osmangazi University

Veysel Karani Öz
Computer Engineering Student (Senior),
Organization: Eskisehir Osmangazi University

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

<u>Symbols & Abbreviation</u>	**<u>Explanation</u>**
CPU:	Central Process Unit
GPU:	Graphical Process Unit
CNN or ConvNet:	Convolutional Neural Network
IoU:	Intersection over Union
mIoU:	Mean Intersection over Union
ROS:	Robot Operating System
SCN:	Shape Context Net
GPM:	Graph Pointnet Module
GAB:	Graph Attention Block
MLP:	Multi-Layer perception
ANN:	Artificial Neural Network
SOM:	Self-Organizing Map
SOFM:	Self-Organizing Feature Map
P:	Precision
R:	Recall
ACC:	Accuracy
TP:	True positive
FN:	True negative
FP:	False positive
CUDA:	Compute Unified Device Architecture
cuDNN:	CUDA Deep Neural Network
LTS:	Long Term Support
kNN:	k th Nearest Neighboor
tf:	TensorFlow

1. INTRODUCTION

In earlier research, point cloud data was commonly used for classification and part segmentation. It is, however, currently being employed in investigations involving the semantic segmentation of the domain. Robots may now separate and comprehend the parts and objects in their surroundings thanks to semantic segmentation. Robots can now be used for a variety of tasks, including guiding people through museums, discovering new planets like Mars, assisting disabled, elderly, or ill people, and cooperating with rescue crews in post-disaster search and rescue missions, thanks to semantic segmentation of the environment and semantic classification of robot locations [1]. The importance of scene comprehension is essential to the search and rescue robots' philosophy.

With today's technology many applications need the scene understanding. Driverless cars, human-machine interaction, picture search engines, and virtual reality are just a few examples [2]. These methods have kept up with the increasing data redundancy with the advancement of technology in hardware terms. Increasing capacity of processors (CPU and GPU), suitable sizes and ease of access have increased and accelerated the studies in this regard. Methods such as segmentation and classification are basically optimization methods. In line with the opportunities provided by the graphics cards, the orientation of the optimization methods to generic algorithms has become easier. In this sense, operations such as image processing, sound processing or problem solving, which are more suitable for the living nature, can be used more flexibly due to the success of generic algorithms in this regard. Therefore, as the number of architectures and datasets produced increased, it was possible to test many architectures on different datasets and compare the results.

Nature was used as a guide in the field of machine learning. Many different neural network methods are used in machine learning and therefore in the field of deep learning, and these neural network methods are inspired by the neuron structures of living brains. In deep learning, the convolutional neural network (CNN or ConvNet) is a deep neural network applied to analyze and classify visual images. Besides these improvement, deep learning methods has big jump on this development. Convolutional Neural Networks (CNNs), by far the leading method and surpassing other types, are a great margin in terms of accuracy and efficiency. ConvNets for 2D imagery has great success. But using deep learning in 3D data is difficult. 3D convolutional networks (3D ConvNets) are used to represent 3D data using voxel representation. However, owing to the lack of most 3D data, most computations are useless. Aside from that, pure 3D ConvNets have issues like significant resolution degradation and massively growing computing costs. Moreover, the fast expansions of sensing technology, as well as the large supply and demand from technologies such as autonomous vehicles, necessitate effective 3D computations [3].

On the other hand, the process of digitizing such visual data is difficult due to the high complexity of the analog structure. The method used to digitize such analog data is to express the data as a point cloud. The point cloud structure can be obtained with various RGB-D cameras, 3D laser range finders and LiDARs and real world objects can be digitized in terms of the position and features they occupy in space. These obtained point features, which constitute the input for the architectures we use in our experiments, are point cloud based. There are also many data storage types in addition to the point cloud. As an example of these, depth images, meshes, and volumetric grids can be given [4]. Using point cloud data is a more convenient way to visualize data and detect errors.

The main motivation of this paper is semantic segmentation of scenes that are placed in the ESOGU RAMPS [5] dataset with different point-based deep learning architectures. Selected architectures were modified for the ESOGU RAMPS dataset, pre-trained, trained, tested and visualized. The architectures used on the ESOGU RAMPS dataset during these processes can be listed as follows. A-SCN [6], ELGS [7], Kd-Net [8], PointConv [9], SO-Net [3] and SpiderCNN [10]. The architectures were evaluated with recall, precision, Intersection over Union (IoU), Mean Intersection over Union (MIoU), and accuracy metrics.

2. METHODOLOGY

In this section, we briefly mentioned on the point-based deep learning architectures we run with ESOGU RAMPS. These architectures are A-SCN, ELGS, Kd-Net, PointConv, SO-Net and SpiderCNN.

2.1 A-SCN

A-SCN (Attentional ShapeContextNet) uses shape context and inner distances to obtain shape matching. In other words, it classifies points hierarchically according to shape context structure to fit a model on point cloud data. SCN (ShapeContextNet) is an architecture that uses the shape context structure. Shape context structure has been used in many fields before, but it has not been used in the field of deep learning. In this structure, the input data is equipped using different numbers of disks. And these discs cover neighboring points, forming partitions of different sizes inside. These partitions create a priority between points [6]. This priority is provided SCN disks as shown in Figure 1.

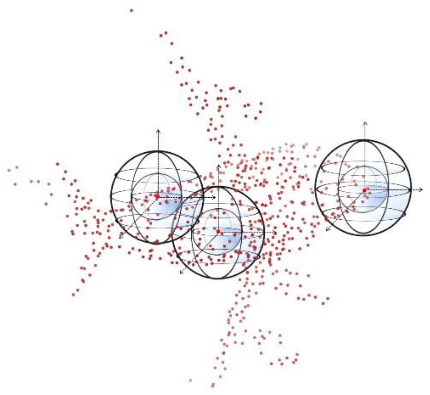


Figure 1. *An illustration of application of a shape context structure on an airplane sample [6]*

Although the Shape Context structure is a simple concept, it contains many different parameters. In this sense, this useful structure modifies the self-attention structure to itself to eliminate such situations. The Self-Attention structure produces very successful results in current deep learning technologies. The Self-Attention structure performs calculations for the input data containing a set of key-value structures with each other and provides logical results by multiplying the appropriate parameters for the set given as input and the output with appropriate vectors [11].

2.2 ELGS

The ELGS model creates a contextual representation for each point in the point cloud, considering neighboring points, to enrich semantic meaning of the points.

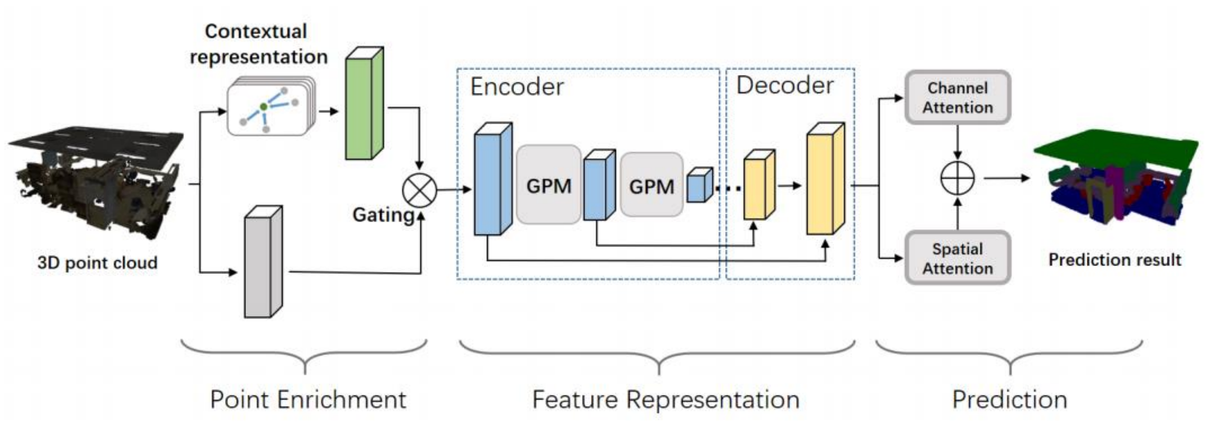


Figure 2. *Three coupled components of the ELGS model [7]*

Graph pointnet module (GPM) and graph attention block (GAB) are used to get characteristic representation of each point, due to the enriched semantic representations that result. Multiple GPMs can be used for output representation. As a result of this process, the point cloud structure is used to generate label information for points. In Figure 2, the proposed model for semantic segmentation that consists of three subcomponents is shown. The point strengthening aims to enrich not only the points but also the semantic representation of the respective point. The feature representation learns each point's feature representation using a decoder-encoder architecture with sideways connections. GPM generates and updates each point acting via a GAB module. Lastly, for the prediction section, channel and spatial attention are used for the final semantic label estimation for each point. This model ignores the general relationship in favor of focusing on point local relationships. Unlike so many other designs and models, the geometric context information within neighboring points receives greater attention. With the help of the graph pointnet module, context representation may reveal structural information in more depth (GPM).

2.3 Kd-Net

Kd-Network executes transformations and distributes transformation factors based on the subdivisions of the point clouds pushed onto them by Kd-trees. Kd-Net avoids weak scaling behavior by not operating on uniform 2D or 3D grids in any way. An example of Kd-tree structure created for 8 points on the left is shown in Figure 3. A classification structure associated with this Kd-tree is seen on the right.

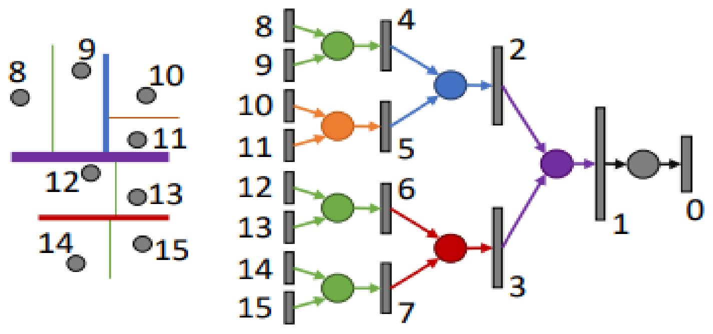


Figure 3. *Kd-tree logic [8]*

A Kd-tree structure splits the collection of points into two equal-sized subgroups and chooses the axis with the greatest gap between the two outermost points in the data cloud, applying it periodically in a top-down manner. As a consequence, Kd-tree with point numbers in point cloud N, N-1 non-leaf nodes is generated. The levels are another feature of a tree node. For tree leaves with a 3D point cloud, Levels equals Depth D. N = 2D in this case.

2.4 PointConv

3D point clouds are often disorganized and unordered geometric data structures, and convolutional neural networks are difficult to use directly. Therefore, a deep convolutional network method called PointConv was created. PointConv is a basic approach to 3D convolutional neural networks. PointConv is inspired by the Monte Carlo approach in 3D convolution. It uses MLP (multilayer perceptron) to approximate a weighting functions for the convolutional filter. MLP is a branch of feed forward neural network [12]. 3D point clouds are transformed into 2D images or 3D volumetric grids in the majority of 3D convolutional neural network research. Local Regions are created in previous architectures. PointConv creates a matrix for Local Regions. PointConv efficiently analyzes and calculates weight functions and creates a convolution where density is re-weighted. As you see in Figure 4, PointConv offers an efficient approach to memory.

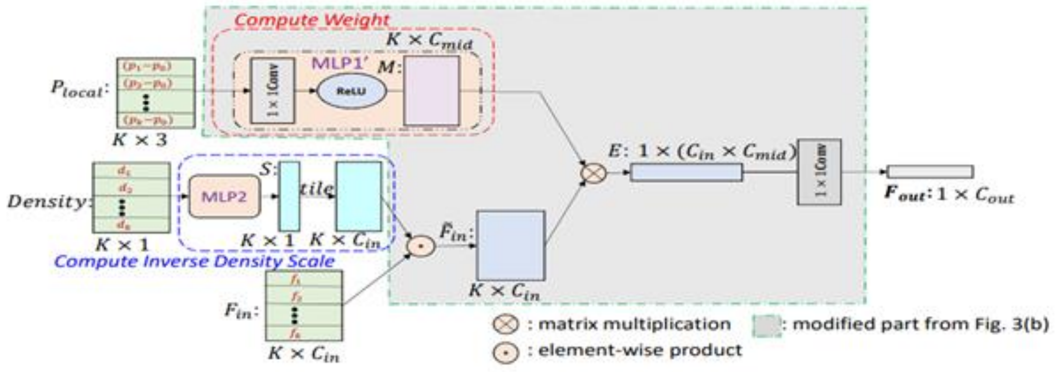


Figure 4. *Memory efficient on a local region with k points. [9]*

2.5 SO-Net

SO-Net is a deep learning architecture that can be used for part segmentation and classification using CNN (Convolutional Neural Networks) with self-organizing map kernels. SO-Net architecture as an artificial neural network (ANN), self- organizing map (SOM) or self-organizing feature map (SOFM), can be explain in two modes training and mapping. At

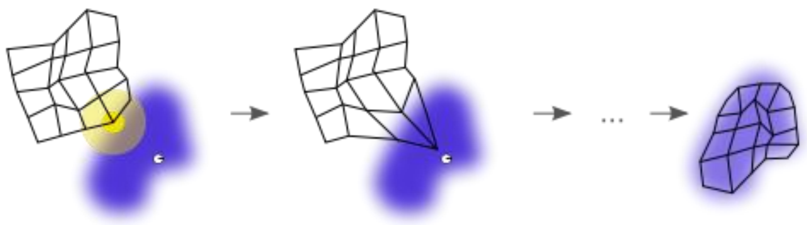


Figure 5*. So-Net, distribution of SOM nodes on the object [3]*

training process, SO-Net builds SOM nodes using input samples (vector quantization), where these nodes are visible part of SOM. Before training, SOM nodes set on to low-dimensional space (typically two-dimensional). On these finite regions where the object is located, nodes arrange in a regular hexagonal or rectangular grid. All nodes have a weight vector and during training process weight vectors are moved toward the input data. In a sense, SOM nodes are distributed according to training result on input data. An example for SOM distribution is visualized in Figure 5. SO-Net input sets must go through preprocessing. SOM nodes created in the preprocess will be used in the SO-Net training process.

2.6 SpiderCNN

Deep neural networks are generally designed according to the human brain and have achieved success in some areas. However, it is difficult to achieve success for irregular structured areas such as 3D point clouds. A point cloud is an irregular geometric data structure. Since point clouds are scattered in an irregular manner, it prevents direct use of convolutional neural networks.

Point clouds can be transformed into 3D voxels and this problem can be overcome by 3D convolution. That's why SpiderCNN has been developed. SpiderCNN is a new convolutional architecture. CNN (Convolutional neural networks) is a sub-branch of deep learning. CNN is mostly used in the analysis of visual information. It is mostly used in picture and video descriptions and can examine the input image and distinguish the objects in the image.

For example, when looking at a scene, it can be classified if it has definable features such as Inclined ramp, Wall, Flat ramp, Terrain. Nowadays convolution in conventional CNN is basically an integral formula obtained by discretization between F, which is a picture function on regular grids and a filter matrix. SpiderConv has taken g_w,
which is a special family of filters [10].

2.7 Tools

Tools and programs that used in this project:

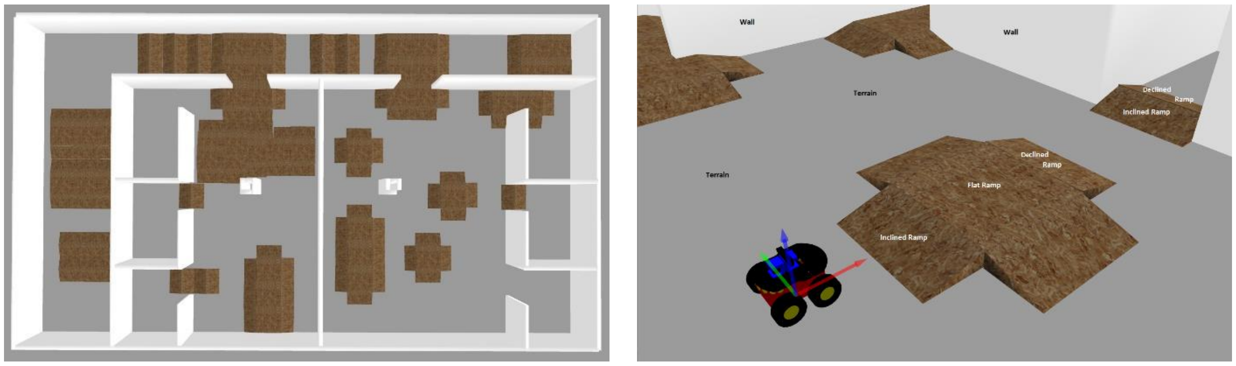
- CUDA
- Linux (Ubuntu)
- Python libraries (Numpy, Scipy, Panda, etc.) and tools.
- Anaconda
- Tensorflow and PyTorch frameworks
- Cloud Compare
- Visual Studio Code ,Sublime Text
- NVIDIA GTX 1070 Ti (GPU)

3. EXPERIMENTS

3.1 Experimental Setup

3.1.1 The ESOGU RAMPS dataset

The ESOGU RAMPS dataset was created in the Gazebo [13] simulation environment, which is given in Figure 6(a). A Pioneer 3-AT mobile robot with an ASUS Xtion Pro RGB-D camera was launched in that environment. To control the robot and capture the scenes Robot Operating System (ROS) framework [14] was utilized. The robot takes 681 different scenes while navigating through the environment. In these scenes, points belong to Inclined Ramp, Wall, Flat Ramp, and Terrain semantic classes as shown in Figure 6(b).



(a) Simulation environment *(b) Semantic classes*

***Figure 6**. ESOGU RAMPS environment [15]*

ESOGU RAMPS dataset consist of 681 scenes, and they represent a very large point cloud for an architecture. In the past, PointNet [16] blocking structure was often used in such large scene data. According to this structure, scene parts are divided at certain intervals in each scene. The blocks formed as a result of the splits are sent to the part segmentation and semantic segmentation methods in these ways. While creating the scene part, we created the blocks with one square meter areas and did not include the scenes with less than 100 points in the point cloud into the training and testing processes. Scene parts divided into blocks were sent after preprocessing required by the relevant architectures or by modifying their formats. In this sense, the architectures were trained for train testing and visualization.

3.1.2 Experimental Process and Parameters

Due to the use of many different libraries within the selected architectures, we implemented this work on the Ubuntu Operating System. Since the PyTorch and TensorFlow libraries working on the deep learning side work on the GPU, the necessary steps have been taken to make the drivers of the GPUs compatible with the relevant CUDA version.

First, the architectures were tested with their original datasets with the aim of checking whether they properly work. These datasets are Shapenet [17] and S3DIS [18] datasets. Shapenet dataset is prepared for part segmentation. Therefore, there are object structures, not scene structures. 16 different objects are used in the selected architectures. In the S3DIS dataset, there are 5 different large indoor areas, these areas are in 3 different buildings. These areas represent a total field of six thousand twenty m^2. Within these data clouds, there are 12 different semantic elements [5]. The Stanford Large-Scale 3D Indoor Spaces (S3DIS) collection is made up of 6 large indoor 3D point clouds. The interior scans span a total field of six thousand twenty m^2 and include a total of 215 million points. The data has been semantically divided into 272 rooms and tagged with 12 semantic components as well as a clutter label. Indoor semantic segmentation is usually done with S3DIS.

We focused on the ESOGU RAMPS dataset, which is our main goal, with the architectures that we achieved positive results with their own datasets. After the import of all the data, the relevant parts of the data that should be separated for the train and test phases. This process occurs as a result of reading the "test_scene_list.txt" file, which contains the names of the scenes to be used for testing, and separating the scenes corresponding to the scene names it contains from all data.

While obtaining the test results, the estimated label values are recorded for the respective blocks. These prediction values can be used to visualize relevant scenes. However, as we separate the scenes into blocks before sending the dataset to the architectures, the blocks must be put together so that the tested prediction values make a meaningful visual. While the blocks are being put together, we can shift each block to its previous places by using the reverse operation by recording the translation amounts in the operation where the blocks are separated. For each predicted value obtained, a different RGB value can be assigned for each label at the stage where the labels are saved. Together with these predictions, we sent the prediction scenes we obtained to the Cloud Compare [19] application. Points on related objects, visualized by the colors of different objects, can be read as the estimation error of the architectures.

3.2 Experimental Results

First of all, all architectures were trained with the same data and tested with the same dataset. In this case, comparing the results with each other is not a problem, and it prepares a suitable basis for comparison. In order to implement the selected architectures, the existence of scripts was investigated, and it was confirmed that these scripts were written by the authors of the paper. In addition, the feasibility of these architectures was determined, and the system requirements were checked. Selections were made in line with what was explained.

100 of the 681 scenes in the dataset were randomly determined and reserved for testing. These determined scenes were not included in the training process, however determined scenes were tested with trained models that were operated with scenes reserved for

training only. Recall (1), Precision (2), Intersection over Union (IoU) (3), mean Intersection over Union (mIoU) (4) and Accuracy (5) metric were considered. The metric and visual results are given in Table 1 and Figure 7, respectively.

$$recall = \frac{TP}{TP + FN} \tag{1}$$

$$precision = \frac{TP}{TP + FP} \tag{2}$$

$$IoU = \frac{TP}{TP + FP + FN} \tag{3}$$

$$mIoU = \frac{\sum_{i=1}^{n} IoU_i}{n} \tag{4}$$

$$Acc = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} (TP_i + FP_i)} \tag{5}$$

The successful prediction of the positive class by the architectural models is called True positive. The successful prediction of the non-positive class of the architectural models is called True negative. A false positive is that the architectural models cannot predict the positive class successfully.

Finally, the false negative is that the architectural models cannot successfully predict the non-positive class. Accuracy is the rate of number of corrects to entire predictions. Precision is the rate of positive identification onto actual correct, recall is the rate of actual positives onto identified corrects.

Table 1. *Metric results*

	Inclined Ramp			Wall			Flat Ramp			Terrain			Mean	
	P	R	IoU	P	R	Iou	P	R	Iou	P	R	IoU	IoU	Acc
A-SCN	74.4	38.4	33.9	**100.0**	59.2	59.2	49.2	98.2	48.8	92.6	**99.9**	92.6	75.9	58.6
ELGS	97.3	97.1	94.6	99.9	99.8	99.7	97.6	98.1	95.8	99.5	99.4	98.9	98.8	97.3
Kd-Net	X	X	X	X	X	X	X	X	X	X	X	X	X	X
PointConv	**99.6**	**99.5**	**99.2**	99.9	**99.9**	**99.8**	**99.6**	**99.6**	**99.3**	**99.8**	**99.9**	**99.8**	**99.5**	**99.8**
SO-NET	94.6	96.7	91.7	99.8	99.7	99.5	98.9	95.2	94.2	98.5	99.3	97.9	98.1	95.8
SpiderCNN	96.6	91.3	88.5	98.8	99.7	98.5	72.4	74.6	58.1	86.1	86.6	76.0	80.3	88.9

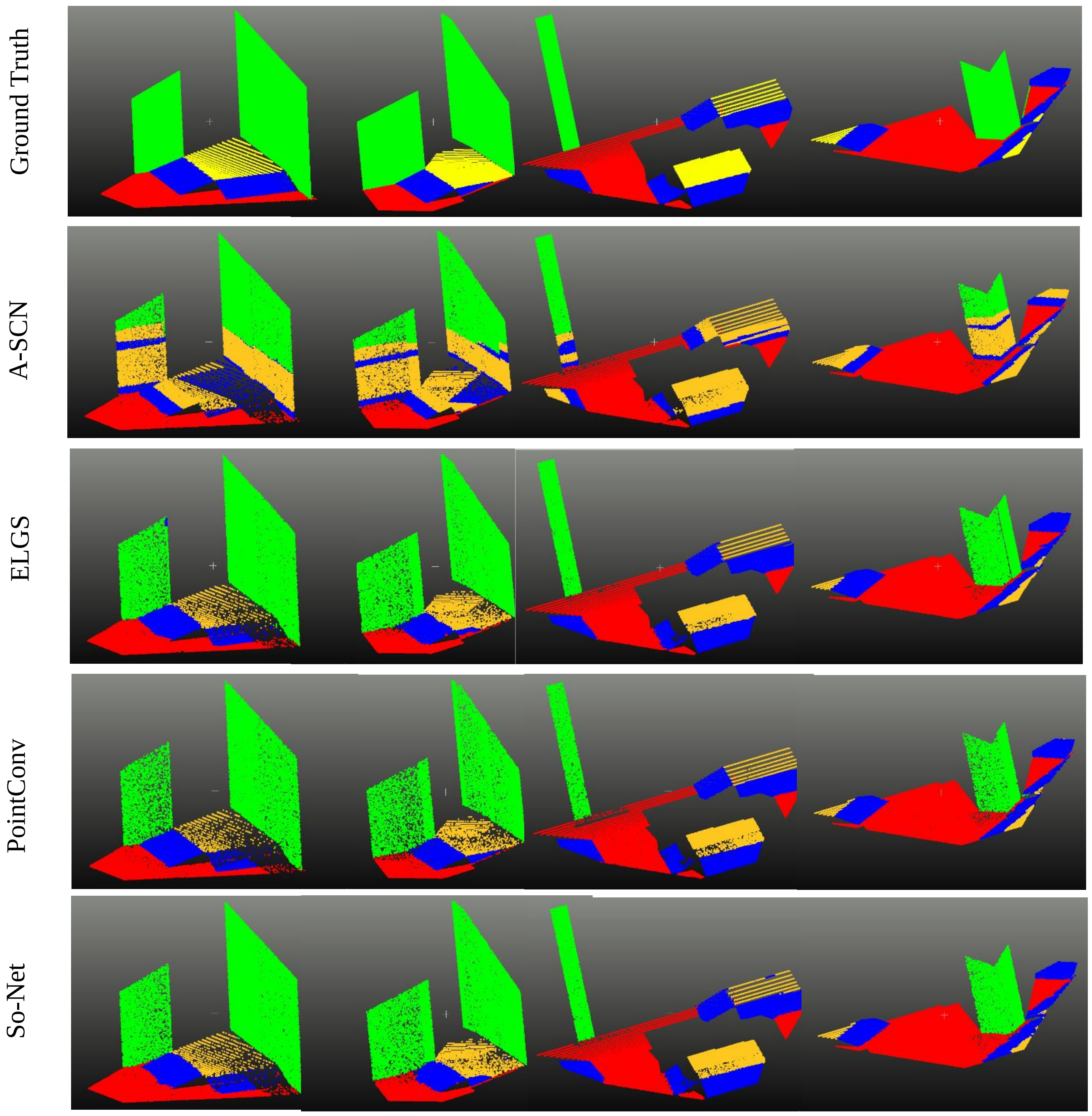


Figure 7. *Visual results*

3.2.1 A-SCN

The shape context structure contains parameters such as bins, r, φ, θ. The number of bins can be found by multiplying these three variables. These variables and disk structure can be found in Figure 8. Where r represents the radial distance of the bin structures, φ represents the bin number on polar coordinates, and θ represents the azimuthal angles of the bins. Basically, the process includes three main stages: selection, aggregation, and transformation. In selection, bins and their numbers are determined by the self-attention mechanism. In the aggregation process, the relationship of the formed bins with the points in the point cloud is created. In the transformation process, the aggregation outputs are processed, and the resulting features are sent to a kernel function. As a result, shape context blocks are created. Semantic segmentation application has been made with A-SCN is S3DIS data set and the results are presented in the paper as obtained.

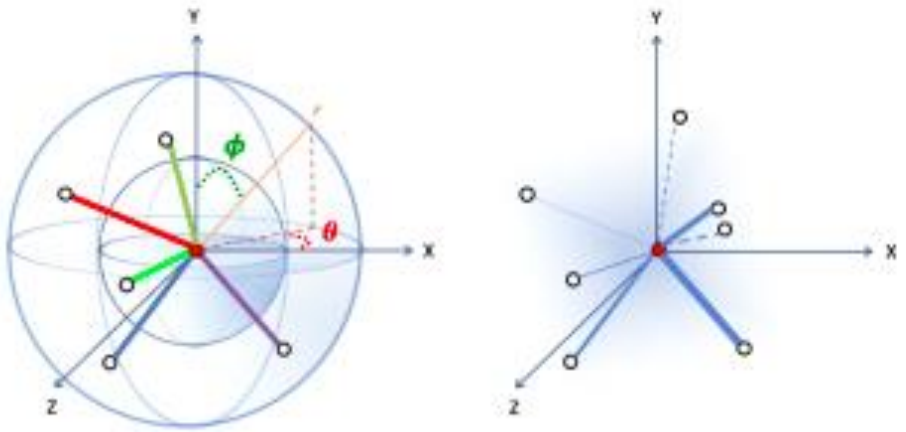


***Figure 8**. Disk structure of ShapeNetContext [6]*

During the application of the A-SCN method to the ESOGU RAMPS data set, the blocked data set was sent to the A-SCN architecture. 35 epochs were performed in the train process. The epoch that produced the best result was sent to the test process, and the test process was concluded with the evaluation technique applied to other architectures. Considering the visualization and test results, it has been shown that this architecture does not produce meaningful results with the ESOGU RAMPS dataset. When the results were obtained in this way, the train and test processes were repeated, but the results did not change. This situation can be attributed to the fact that the ESOGU RAMPS data set basically consists of two-dimensional planes. The 3-dimensional radial bins clusters created in the system can find themselves on only one plane. Therefore, as mentioned above, bins created in the selection phase cannot benefit sufficiently in the θ and φ features. The self-attention process is negatively affected by this and cannot produce meaningful results. It was observed that the Wall class was strongly incorrectly predicted, and this was only seen near the ramps as shown in the Figure 10. Here, we can say that the system searches for different radial extensions and therefore tends to ramp classes.

3.2.2 ELGS

ELGS is a model for point cloud semantic segmentation. The ELGS model constructs a contextual representation for each point by considering its neighbors to augment its semantic meaning using a new gated fusion method. A new graphic point network module (GPM) based on a graphic attention block (GAB) is used to create and feature highlighting of each point in the local structure. In this way, the layer depth may be raised in this way, and increasingly complicated structures can be segmented. Finally, the channel-wise and spatial-wise attention methods used to create the semantic label for each point.

After the ELGS model obtained successful results in the S3DIS data, it was transferred to work with the ESOGU RAMPS dataset to perform train and test operations. Since the ELGS model has a semantic segmentation structure, it was quickly adapted to the operations expected to be done with the ESOGU RAMPS data, and the data parameters of the ELGS model were changed accordingly. As a result of these processes, a great accuracy rate has been achieved, as can be seen in the visualized outputs. This is mostly due to the usage of the GPM module, as well as the fact that the ELGS modes prediction layer integrates global structure information between points, which improves representations. However, errors were encountered in some intersection areas where there are transitions between classes. However, errors were encountered in some intersection areas where there are transitions between classes. The solution to this is to find critical shape differences and detect edges more accurately.

3.2.3 Kd-Net

Many problems were encountered when trying to work with ESOGU RAMPS data in Kd-net. The basis of these problems was the size of the point cloud which is different from the original referred point cloud size. In addition to the many train files in the architecture, it is not clearly specified which train is working with on GitHub. Thereupon, it was decided to use ‘train_MG2.py' script, which includes skip connections in affine transformation used in segmentation. Since each block in ESOGU RAMPS data consists of 4096 points, the script content has been adapted to this mechanism. By adding a new encoding and decoding layer to the 'kdnet.py' script where the main operations of the architecture are done. Data size problem was tried to be avoided. Although the point set and class label information can be accessed correctly in the train file and the cut-dim required to create the kd-tree can be set, the predicted choice, required for the target to be created to check and compare the data could not be produced in the expected size. As a result of this problem, the article of architecture was re-examined, and solutions were sought. Although there is part segmentation information in the Kd-net article, it has been observed that this architecture can only make classification and work with determined data size.

3.2.4 PointConv

Since we did not have the Scannet dataset, we could not experiment on it. After adapting the ESOGU RAMPS data to PointConv, the training started. The training was stopped at epoch 33. After doing the test, the results we got were quite successful. Experiment on ESOGU RAMPS shows that PointConv is successful in semantic segmentation benchmarks in 3D point clouds compared to other architectures. PointConv learns the position of the point clouds in their local region via the multi-layer Perceptron the weight of the convolution kernel for convolutional filters and reweights the learned weighted functions. In this way, it makes the scene more understandable. Finally, we visualized our test results. When we look at the images, we observed that the PointConv architecture has a 99.8% accuracy rate between the predictions and the Ground Truth.

3.2.5 SO-Net

In SO-Net, the data is pre-processed before the training process. We performed this preprocessing for the ESOGU RAMPS dataset. After this pre-processing, SOM points are created, and this feature is combined with the label and point coordinates and sent to the architecture as input. SOM nodes are created in size nxn, resulting in size reduction. In another sense, we can classify the three-dimensional point cloud within the framework of 2D features. These created SOM nodes must be permutation variants in order to be associated with point coordinates. Here n can be selected in the range of [5-11] and accordingly SOM nodes have a size between 25 and 121.

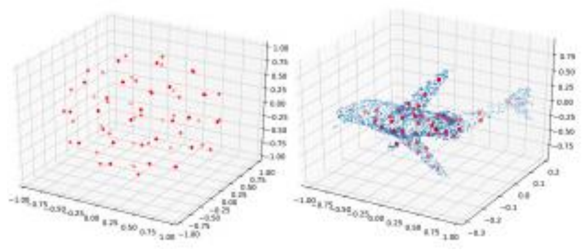


Figure 9. *SOM nodes [3]*

Nodes created after preprocessing affect the result and these nodes are updated and fit on the object during the training period. The fitting process of SOM nodes on the object can be observed in the Figure 9. Thus, SOM nodes created in the space occupied by the object produce efficient results using the k nearest neighbors(kNN) approach. After the pre-process, training and testing processes were applied. Since there is no semantic segmentation structure in SO-Net, the segment segmentation structure has been modified. There are some errors only at the points where different classes come into contact with each other. The reason for this situation can be shown as that after the sonnet has created the SOM nodes, these nodes perform their update with the kNN mechanism. With the visualization process, we prove our mistakes and success.

3.2.6 SpiderCNN

Shapenet model is used while SpiderCNN trains. There are surface normals in point clouds in Shapenet data. But there are no surface normals in ESOGU RAMPS data. For this reason, we have commented the surface normals in the ‘get_batch’ function in SpiderCNN ‘train.py.’ We defined our scene to the ‘seg.classes’ list. The 0,1,2,3 labels on the stage represent Inclined ramps, walls, flat ramps, and terrain. We stopped it when the SpiderCNN completed its 40th epoch. When we looked at the results, we observed that we could not achieve a very successful result for SpiderCNN. We can say that the parameters are important in Convolutional Filters. This can affect the poor results. When we look at the table, we see that SpiderCNN finds the wall without any problems. But we observed that SpiderCNN made errors in Flat Ramp and Terrain values.

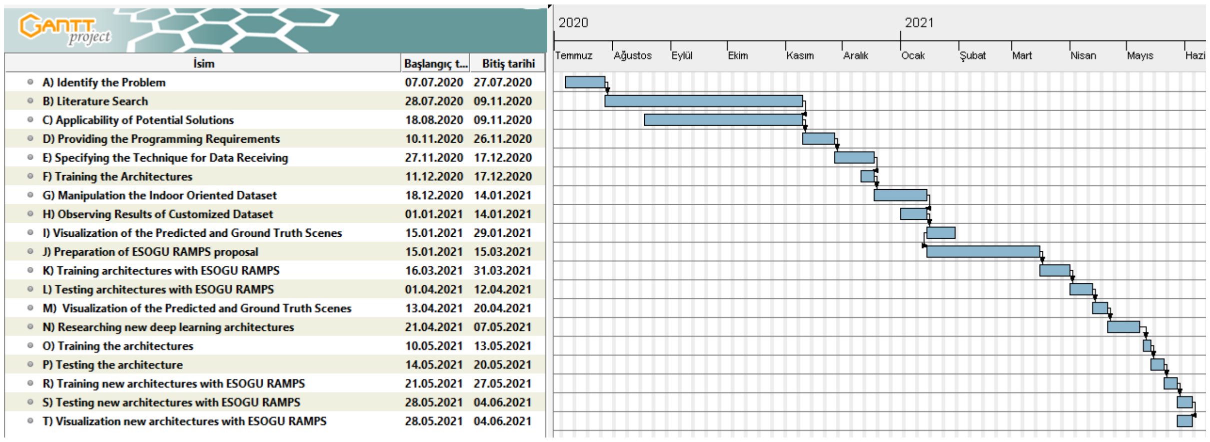
4. PROJECT PLAN

- **Work Package A – Identifying the Problem:** The definition of the problem and its solutions were discussed.
- **Work Package B – Literature Search:** Similar studies were reviewed on semantic segmentation based on deep learning.
- **Work Package C – Applicability of Potential Solutions:** Testing feasibility of various solution techniques on certain problem.
- **Work Package D – Providing the Programming Requirements:** Setting up the coding environments, tools and packages.
- **Work Package E – Specifying the Technique for Data Receiving:** Determining the steps that can be followed when loading data into architectures.
- **Work Package F – Training the Architectures:** To obtain the same experiment accuracies with corresponded methods.
- **Work Package G – Manipulation the Indoor Oriented Dataset:** with S3DIS dataset, we tried separating block process.
- **Work Package H – Observing Results of Customized Dataset:** Observing the training results of the pre-processed dataset and comparing between each method.
- **Work Package I – Visualization of the Predicted and Ground Truth Scenes**
- **Work Package J – Preparation of ESOGU RAMPS:** Implementation of blocking on ESOGU RAMPS data.
- **Work Package K – Training architectures with ESOGU RAMPS**

- **Work Package L – Testing architectures with ESOGU RAMPS**
- **Work Package M – Visualization of the Predicted and Ground Truth Scenes**
- **Work Package N – Researching new deep learning architectures:** Investigation of new deep learning architectures that can do segmental segmentation that can be suitable for ESOGU RAMPS data.
- **Work Package O - Training the architectures**
- **Work Package P - Testing the architectures**
- **Work Package R – Training new architectures with ESOGU RAMPS**
- **Work Package S – Testing new architectures with ESOGU RAMPS**
- **Work Package T – Visualization new architectures with ESOGU RAMPS**

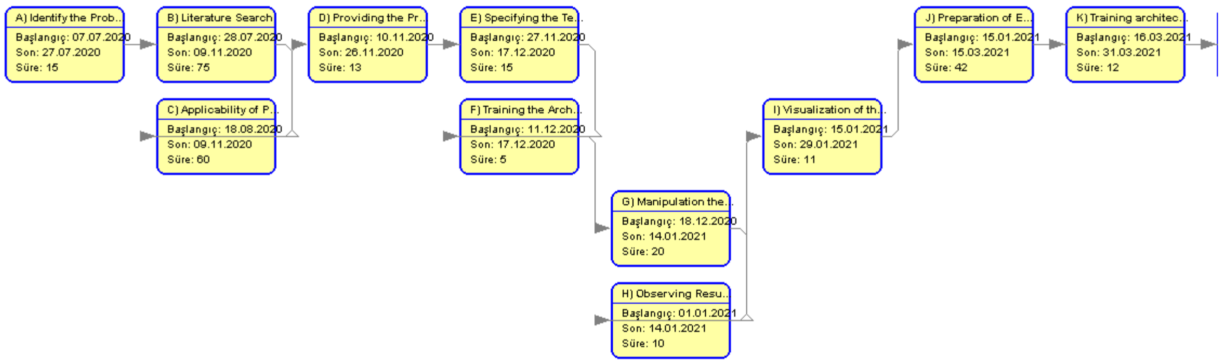
Table 2. Resource assignments for work packages.

Work Package	Resource	Duration (Days)
A	Berkin, Doğukan, Batuhan	20
B	Berkin, Doğukan, Batuhan	105
C	Berkin, Doğukan, Batuhan	85
D	Berkin, Doğukan, Batuhan	16
E	Berkin, Doğukan, Batuhan	20
F	Berkin, Doğukan, Batuhan	6
G	Berkin, Doğukan, Batuhan	27
H	Berkin, Doğukan, Batuhan	13
I	Berkin, Doğukan, Batuhan	14
J	Berkin, Doğukan, Batuhan	59
K	Berkin, Doğukan, Batuhan	15
L	Berkin, Doğukan, Batuhan	11
M	Berkin, Doğukan, Batuhan	7
N	Berkin, Doğukan, Batuhan	16
O	Berkin, Doğukan, Batuhan	3
P	Berkin, Doğukan, Batuhan	6
R	Berkin, Doğukan, Batuhan	6
S	Berkin, Doğukan, Batuhan	7
T	Berkin, Doğukan, Batuhan	7
PROJECT COMPLETITION TIME:		318

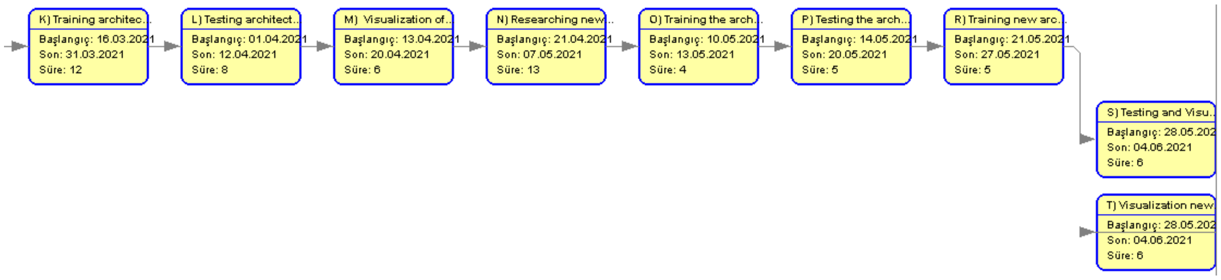


İsim	Başlangıç t...	Bitiş tarihi
A) Identify the Problem	07.07.2020	27.07.2020
B) Literature Search	28.07.2020	09.11.2020
C) Applicability of Potential Solutions	18.08.2020	09.11.2020
D) Providing the Programming Requirements	10.11.2020	26.11.2020
E) Specifying the Technique for Data Receiving	27.11.2020	17.12.2020
F) Training the Architectures	11.12.2020	17.12.2020
G) Manipulation the Indoor Oriented Dataset	18.12.2020	14.01.2021
H) Observing Results of Customized Dataset	01.01.2021	14.01.2021
I) Visualization of the Predicted and Ground Truth Scenes	15.01.2021	29.01.2021
J) Preparation of ESOGU RAMPS proposal	15.01.2021	15.03.2021
K) Training architectures with ESOGU RAMPS	16.03.2021	31.03.2021
L) Testing architectures with ESOGU RAMPS	01.04.2021	12.04.2021
M) Visualization of the Predicted and Ground Truth Scenes	13.04.2021	20.04.2021
N) Researching new deep learning architectures	21.04.2021	07.05.2021
O) Training the architectures	10.05.2021	13.05.2021
P) Testing the architecture	14.05.2021	20.05.2021
R) Training new architectures with ESOGU RAMPS	21.05.2021	27.05.2021
S) Testing new architectures with ESOGU RAMPS	28.05.2021	04.06.2021
T) Visualization new architectures with ESOGU RAMPS	28.05.2021	04.06.2021

Figure 10. *Gantt diagram of the project.*



(a) *PERT diagram part 1*



(b) *PERT diagram part 2*

Figure 11. *PERT diagram of the project*

5.CONCLUSION

As a conclusion, we adopted some point-based deep learning architectures for semantic segmentation of the scenes that is placed in ESOGU RAMPS dataset. These architectures were A-SCN, ELGS, Kd-Net, PointConv, SO-Net, and SpiderCNN. When the visual and metric results were considered ELGS, PoinConv, SO-Net, and SpiderCNN produce successful results with the dataset. On the other hand, we could not obtain successful accuracy and IoU values due to system problems in Kd-Net and manual input problems in A-SCN. Due to some anomalies in the dataset, precision-recall and IoU values between classes may show the same differences in different architectures. These results should be considered as a whole. Our simple structured dataset consisting of 2-dimensional planes has been put into training and testing processes with 6 different selected deep learning architectures. Some architectures include preprocessing by their own nature, and some have completely different data loading structures. The results obtained are generally motivating for studies on the ESOGU RAMPS dataset.

6.REFERENCES

[1] O. M. Mozos, ´ Semantic Labeling of Places with Mobile Robots, ser. Springer Tracts in Advanced Robotics. Springer-Verlag Berlin Heidelberg, 2010, vol. 61.

[2] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, Jose Garcia-Rodriguez “A Review on Deep Learning Techniques Applied to Semantic Segmentation” arXiv:1704.06857, Apr. 22, 2017.

[3] Jiaxin Li, Ben M. Chen, Gim Hee Lee “SO-Net: Self-Organizing Network for Point Cloud Analysis” arXiv:1803.04249 Mar 27 2018.

[4] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, Mohammed Bennamoun, "Deep Learning for 3D Point Clouds: A Survey" arXiv:1912.12033, Dec 27 2019.

[5] The ESOGU RAMPS dataset, ESOGU, https://ai-robotlab.ogu.edu.tr/Sayfa/Index/11, (June,2021).

[6] Saining Xie, Sainan Liu Zeyu, Chen Zhuowen Tu, "Attentional ShapeContextNet for Point Cloud Recognition" accessed 09 June 2021, https://pages.ucsd.edu/~ztu/publication/cvpr18_ascn.pdf.

[7] Xu Wang, Jingming He, Lin Ma, "Exploiting Local and Global Structure for Point Cloud Semantic Segmentation with Contextual Point Representations" arXiv:1911.05277, Nov 13 2019.

[8] Roman Klokov, Victor Lempitsky “Escape from Cells: Deep Kd-Networks for the Recognition of 3D Point Cloud Models” arXiv:1704.01222v2, Oct 26 2017

[9] Wenxuan Wu, Zhongang Qi, Li Fuxin, "PointConv: Deep Convolutional Networks on 3D Point Clouds" arXiv:1811.07246, Nov 17 2018.

[10] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, Yu Qiao “SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters” arXiv:1803.11527v3, Sep 12 2018

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkorei, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin, "Attention Is All You Need", accessed 09 June 2021, https://papers.nips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[12] “Multilayer Perceptron.” Wikipedia, Wikimedia Foundation, accessed 10 June 2021, https://en.wikipedia.org/wiki/Multilayer_perceptron

[13] Gazebo, Open source robotics foundations (OSRF), https://gazebosim.org/ , (June,2021).

[14] Robot Operating System (ROS), Open source robotics foundations (OSRF), https://www.ros.org/ , (June,2021).

[15] Kaya Turgut, Burak Kaleci, "A PointNet Application for Semantic Classification of Ramps in Search and Rescue Arenas" Sep 30 2019, accessed 09 June 2021, https://www.ijisae.org/IJISAE/article/view/1022 .

[16] Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation", arXiv:1612.00593, Apr 10 2017

[17] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li

Yi, Fisher Yu "ShapeNet: An Information-Rich 3D Model Repository" arXiv:1512.03012, Dec 9 2015.

[18] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, Silvio Savarese “3D Semantic Parsing of Large-Scale Indoor Spaces”, April 30 2016, accessed 24 December 2020, https://cvgl.stanford.edu/papers/iro_cvpr16.pdf

[19] CloudCompare (version 2.11) [GPL software]. (2021). Retrieved from http://www.cloudcompare.org/

[20] Charles R. Qi, Li Yi, Hao Su, Leonidas J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space", arXiv:1706.02413, Jun 07 2017