ASL Classification for 3D point cloud

Prajakta Survase
Electronics and Telecommunication Department
Vishwakarma Institute of Technology,Pune,India
prajakta.survase17@vit.edu

Sakshi Rathi
Electronics and Telecommunication Department
Vishwakarma Institute of Technology,Pune,India
sakshi.rathi17@vit.edu

Aniket Patil
IFM Engineering Private Limited,Pune,India
Aniket.Patil@ifm.com

Purushottam Ekande
IFM Engineering Private Limited,Pune,India
Purushottam.Ekande@ifm.com

Prof. Dr. Shripad Bhatlawande
Electronics and Telecommunication Department
Vishwakarma Institute of Technology,Pune,India
shripad.bhatlawande@vit.edu

***Abstract*—American Sign Language detection by technology is an upcoming field of interest as there is a large social group which can benefit from it. Recognizing sign language is one of the primary keys to help users to bridge the gap of communication with society. Image processing and deep learning can be used to assist recognition of sign language, which can then be comprehended by other people. PointNet architecture has been employed on 3D dataset to recognize American sign language alphabets. The 3D dataset used consists of 24 static sign language alphabets captured using a CamBoard pico flexx camera. PointNet provides a unified architecture for applications on such data and is highly efficient and effective. An extensive experimentation was carried out on the 3D dataset to train and test the system, in order to prove that the approach gives a good performance in terms of accuracy. After preprocessing the input, it predicts the alphabet of the point cloud. The paper demonstrates results obtained by training and testing the American Sign Language alphabets dataset on deep neural network using PointNet architecture and the model consists of multiple filter inputs that are processed on the same input. The validation accuracy obtained was above 96%. It also depicts various challenges faced in solving such a problem, and outlines future scope.**

Keywords—American Sign Language (ASL), Recognition and Classification, Deep Learning, 3D-data, Image Processing, PointNet.

I. INTRODUCTION

Spoken language is a medium of communication amongst a vast majority of the human population. However, a section of the population is not able to communicate with the majority of the population. This section of the population faces hindrance in the field of human computer interaction, especially when their only means of communication is through sign. Therefore, sign language assists this section of the community. Using facial expressions, static hand symbols, and hand gestures, sign language provides tools to communicate like spoken language. Here, American Sign Language (ASL) has been used, which has 24 static alphabets that are used for classification and recognition. This paper contributes towards the effort, by experimenting with such a methodology to verify its efficiency in recognizing sign language.

Several traditional 2D approaches are therefore, superseded by 3D-based ones. A few CNN-based systems have been proposed to use 3D data for classification. Several reliable sensors, e.g., CamBoard pico flexx camera, RGB-D cameras such as Microsoft Kinect etc., provide useful 3D data to feed the prediction systems with a new dimension of useful information.

Here, the dataset is acquired using a CamBoard pico flexx camera and is further used for classification purposes. The CamBoard pico flexx is the smallest 3D camera development kit based on Time-of-Flight (ToF) technology. A small form factor and the low power consumption enable a wide range of new use cases and make the CamBoard pico flexx the perfect development kit for 3D depth sensing applications. The CamBoard pico flexx comes with a powerful software suite Royale, containing all the logic to operate the 3D camera including a visualization tool, the Royale Viewer. Royale is cross platform compatible and runs on Windows, Linux, Ubuntu Linux, macOS and Android/ARM. In this work, we implemented a deep net architecture that consumes raw point clouds without voxelization or rendering. It is a

unified architecture that learns both global and local point features, providing a simple, efficient and effective approach for a number of 3D recognition tasks.

Point clouds are simple and unified structures that avoid the combinatorial irregularities and complexities of meshes, and thus are easier to learn from. A point cloud is just a set of points and therefore invariant to permutations of its members. PointNet is a unified architecture that directly takes point clouds as input and outputs either class labels for the entire input or per point segment labels for each point of the input. Under a unified architecture, PointNet is much faster in speed. We discuss an experimental evaluation of the approach.

II. RELATED WORK

Many existing studies have provided a detailed review and summary of the methods for point cloud classification. This section presents an overview of the traditional methods based on handcrafted features and modern methods based on 3D deep learning for point cloud classification. Our approach is closely related to other deep learning methods on point clouds.

However, the classification of the point cloud is challenging due to irregularity of point distributions, the enormous number of points, the non-uniform point density, and the complexity. A common solution is to find the local neighborhood of the points, extract specific geometric features according to the spatial context information implied in the neighborhood, and then classify the points according to these features. A general framework consists of: neighborhood selection, feature extraction, feature selection, and classification. In this framework, a critical concern would be how to extract features that would facilitate classification. To make the best of the features extracted, many researchers have conducted point cloud classification by means of machine learning algorithms, such as support vector machines (SVM), random forests (RF), boosting algorithms, and neural networks. In most cases, a larger number of features results in a higher capability of self-learning in machine learning, However, the larger the number of features was, the longer the calculation time required to extract and classify them.

Due to the unordered and irregular nature of the point cloud data, regular Deep Learning models, such as the convolution neural networks (CNNs) or the recursive neural networks (RNNs) cannot be directly applied in point cloud classification. With the advancement of deep learning, deep networks have been employed for point cloud classification. The PointNet uses deep neural networks to process point clouds with a spatial transform network and a symmetry function. PointNet stands out with its simple model, small number of parameters, and quick calculation.

Many models applicable to point cloud classification have recently been proposed, such as PointNet++, PointCNN, PointSIFT, and KPConv lack a common framework, and some have been deliberately expanded to enhance the classification efficacy, which increases the model complexity and the calculation time but also results in overfitting, rendering the trained model inapplicable to other scenes. The simplest approach is still to convert the point cloud to regularly distributed 2D images or 3D voxels and then use the model to classify the converted data. In order to avoid the problems that occur during data conversion, PointNet allows the direct input of the original point cloud data without having to convert the data to another form.

III. POINTNET

In this paper, PointNet network classifies 3D American Sign Language (ASL) point cloud data PointNet is a deep neural network that directly processes out-of-order point cloud data, as shown in Fig. 1. In PointNet, the max pooling layer has the primary symmetric function. The input for the network is the 3-D coordinates (N×3) of the 3-D point cloud containing N points. The original data is predicted by a 3-D spatial transformation matrix T-Net (3), and the transformation matrix T (3) of 3X3 is estimated and applied to the original data to achieve data alignment. The aligned data is extracted in points by a double-layer perceptron model MLP (64, 64) with shared parameters. The multi-layer perceptron is used for fusion, so that the feature dimension is reduced to 128. Finally, the training classifier outputs a score corresponding to M categories for each point to achieve point-by-point classification.

The 64-dimensional features after feature alignment are regarded as features of points, and the last 1024- dimensional features are regarded as global features of points. Therefore, the local and global features are combined to obtain a 1088-dimensional feature. Each point extracts 64-dimensional features, and through the feature space transformation matrix prediction network. T-Net (64) predicts the 64X64 transformation matrix, acting on the features to achieve alignment of features. Then continue to use the three-layer perceptron (64, 128, 1024) for feature extraction in feature points until the feature's dimension is changed to 1024, and then extract the global feature vector of the point cloud through the maximum pooling layer. T-Net in PointNet network is a sub-network of predictive feature space transformation matrices. It learns the transformation matrix consistent with the feature space dimension from the input data. Then use this transformation matrix to multiply the original data to realize the transformation of the input feature space so that each subsequent point has a relationship with each point in the input data. Through such data fusion, a step-by-step abstraction of the features of the original point cloud data is implemented.

IV. DATASET USED

The database for this project was majorly collected manually. There are 24 classes of alphabets used in this project. It is acquired by a CamBoard pico flexx. The dataset is created by 6 users where each user provides 1 BIN per alphabet.

Pico flexx has a resolution of 224x171 (38k) pixels and a framerate of 45-50 fps (3D frames). The point clouds obtained have 3 dimensions and are alphabets of American Sign Language. Here, point clouds of 24 static alphabets are used to form the dataset. The 24 static alphabets range from letters A to Z, excluding J and Z. J and Z are dynamic alphabets. There are approximately 300-point clouds for each alphabet.

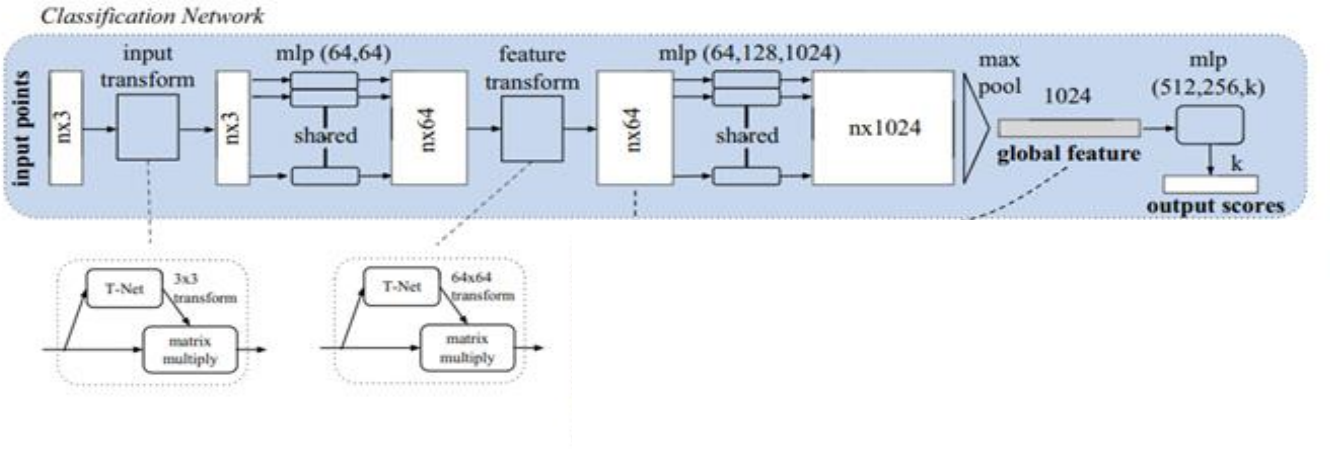

Fig. 1. PointNet network

V. FLOWCHART

The methodology is divided into 3 parts as shown in Fig. 2.:

- Pre-processing of point cloud
- Training the dataset on PointNet network
- Testing the model using GUI

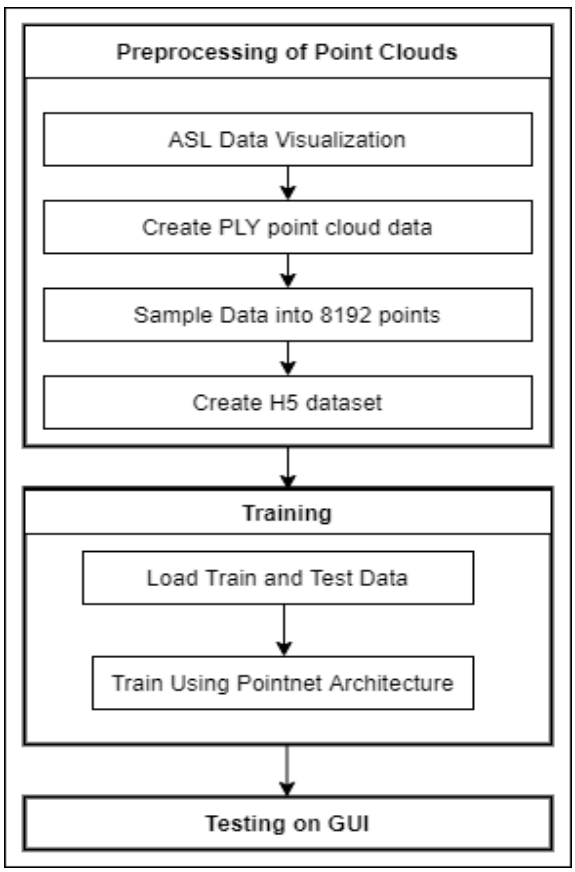

Fig. 2. Flowchart

VI. IMPLEMENTATION

The implementation is performed on the data of Polygon File Format (PLY) which includes point cloud in the form of vertices and faces. It also contains depth information. Datasets obtained after preprocessing are loaded to train the model. Various results are analyzed by varying the hyperparameters. The predictions made by the best model are done and visualized on a GUI.

A. Pre-processing of Point cloud

i. Data Visualization

In the given dataset, point clouds show static sign language alphabets. However, along with it in the point cloud, there are also portions of background and noise in it. The primary data is visualized in MATLAB as shown in Fig.3. The data frames are then separated and created in the PLY form. The PLY data is visualized in Fig. 4. MeshLab, which is an open-source system for processing and visualising 3D data.

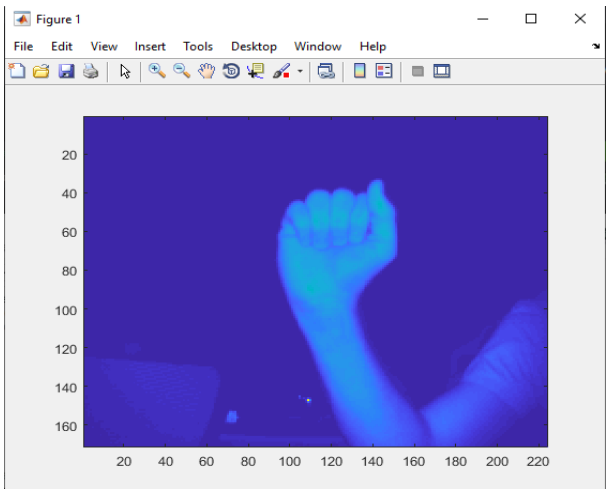

Fig 3. Visualization of data in MATLAB

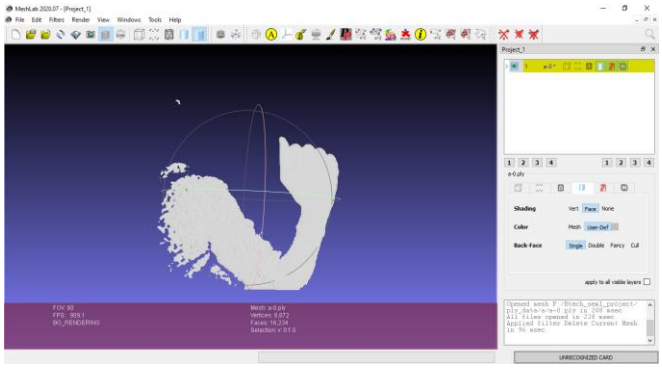

Fig. 4. Visualization of (PLY) in MeshLab

ii. Sampling of point clouds

Each input point cloud has 38k points, they are sampled to have equal number (8192) of vertices/triangles. Sampling is performed using weighted random selection and Barycentric coordinates algorithm. For this, n random triangles of the mesh are selected and one point is generated in a random position inside the corresponding triangle. For selecting triangles, a weighted random selection method is used. It is used to select the element from the list with a probability based on weights. So, it chooses the triangle based on the probability proportional to its area. Barycentric coordinates are used to generate a random point inside a triangle, thus, giving us sampled 8192 points.

iii. Creating h5 datasets

Input point cloud data is organized and compiled into a structured form of (H5) dataset. It stores data in a multidimensional array and in a hierarchical form. The dataset is compiled each for training and testing. 90% of data is used for training and 10% for testing.

B. Training

The resultant dataset is used for training. The dataset is trained on the PointNet network, which is a deep neural network. For each input point, we apply input and feature transformations, and then aggregate point features by max pooling.

The input for the network is of (8192×3). The model uses Convolution Neural Network and T-net Architecture with max pooling and batch normalization with respected layers. The data is predicted by a 3-D spatial transformation matrix T-Net (3) of 3X3 and estimated to the original data to achieve data alignment. T-Net (64) predicts the 64X64 transformation matrix, acting on the features to achieve alignment of features. The 64-dimensional features after feature alignment are regarded as features of points, and the last 1024-dimensional features are regarded as global features of points. Each point extracts 64-dimensional features, and through the feature space transformation matrix prediction network. It extracts the global feature vector of the point cloud through the maximum pooling layer. All the layers include ReLU and batch normalization. Dropout layers are used in classification net. Here, the classification is done by the softmax layer. In the network, the values of the hyperparameters which gave best results are initial learning rate = 0.001, batch size = 32 trained on 250 epochs. Finally, the training classifier outputs a score corresponding to 24 categories for each point to achieve point-by-point classification.

C. Testing

The final model is tested using GUI. A GUI is created on PyQt framework and the point clouds are visualized using a PPTK viewer. PyQt is a Python binding for Qt which has a set of libraries and development tools used for creating a GUI.The Point Processing Toolkit (PPTK) is a Python package used for visualization of 3D point clouds.

Hardware Requirements— The model is trained using Google Colab GPU with 16GB on Windows10 using TensorFlow version 2.3.0. The model is tested on an Intel(R) Core (TM) i3-6100U CPU @ 2.30GHz with 4GB on Windows 10.

VII. TRAINING & EXPERIMENTS

In order to evaluate the performance of the model in terms of accuracy, training experiments were performed by adjusting several hyperparameters. The experimentation and analysis discussed below. The experimentation results are shown in Table. 1.

A. Batch size

It has an impact on the resulting accuracy of models, as well as on the performance of the training process. First, the batch size is varied and different results are obtained. For batch size 8, the training accuracy does not go beyond 30%. Thus, smaller batch sizes are noisy and offer a regularizing effect and lower generalization error.When the batch size is set to 16, the training accuracy goes above 70% and validation accuracy goes beyond 90%. Thus, the performance improves significantly and gives a higher accuracy. Lastly, when batch size is set 32 both the training and validation accuracy go beyond 90%. Larger batch sizes result in faster progress in training, but don't always converge as fast.

TABLE I. EXPERIMENTATION TABLE

Sr No	Batch size	Initial Learning rate(lr)	Change in lr on given epochs	Epochs	Training accuracy	Validation accuracy
1	32	0.001	90 120 150	250	0.9235	0.9646
2	32	0.001	60 120 160	250	0.9144	0.9633
3	32	0.001	50 75	200	0.8792	0.9470
4	16	0.001	90 120 150	250	0.8374	0.9347
5	16	0.001	60 120 160	250	0.8148	0.9292
6	16	0.001	50 75	160	0.7601	0.9125
7	16	0.001	50 75	250	0.7383	0.9194
8	16	0.0001	50 75	180	0.68765	0.9250
9	8	0.001	50 75	250	0.3032	0.5347

B. Learning rate

Learning rate affects how quickly the model can converge to a local minimum. With respect to epochs and steps, learning rate is configured and therefore, leading to increase in accuracy for 16 and 32 batch size. Thus, the best results for batch 16 was obtained as 83% training accuracy and 93% validation accuracy. Similarly, for batch size 32. training accuracy and validation accuracy goes above 90%. Therefore, when the learning rate is too small, training is not only slower, but may become permanently stuck with a high training error. By using checkpoints to save the best performance model, by changing learning rate with epochs and adjusting the batch size the performance of the model can be improved.

VIII. RESULTS

A. Training results

The best model trained gives a training accuracy of 92.35% and validation accuracy of 96.46%. Training loss of 0.2159 and validation loss of 0.1258.
The below graph in Fig. 5. shows how the validation accuracy varies as the number of epochs increases. Total epochs plotted in the graph is 250.

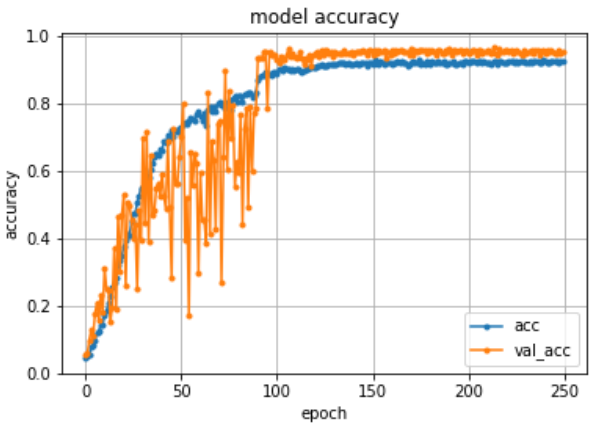

Fig. 5. Accuracy graph

The below graph in Fig. 6. shows how the validation and train loss decrease, as the number of epochs increases.

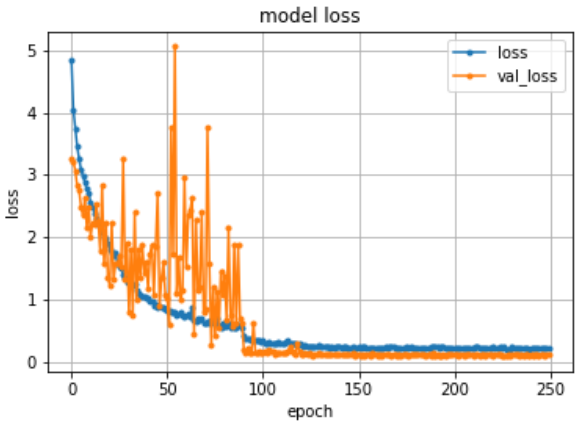

Fig. 6. Loss graph

B. Evaluate

For prediction, the system is designed to predict the result by loading the trained model and loading the data. The prediction is done on data of PLY format. First, the trained model is loaded for prediction. The point cloud data in 3-D point PLY form is read sequentially. The label for loaded point cloud is predicted as the result. The GUI model built using PyQt5 takes input as a trained model file. Then the respected point cloud file is loaded for prediction. The prediction along with probability will be displayed on the GUI window. The PPTK window displays the loaded point cloud file after prediction. In Fig. 7. below, the entire result is displayed.

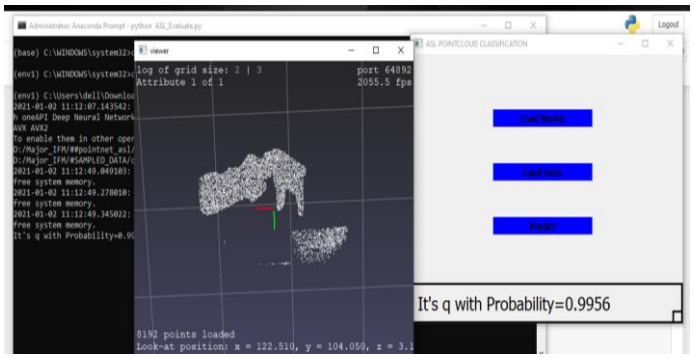

Fig. 7. Model prediction

In the Fig. 8., shown below the PLY of 'Q' alphabet is loaded along with the trained model. After predicting, it correctly shows 'Q' with a probability of over 99%. And the PPTK viewer in Fig. 9., visualizes input 3D point cloud

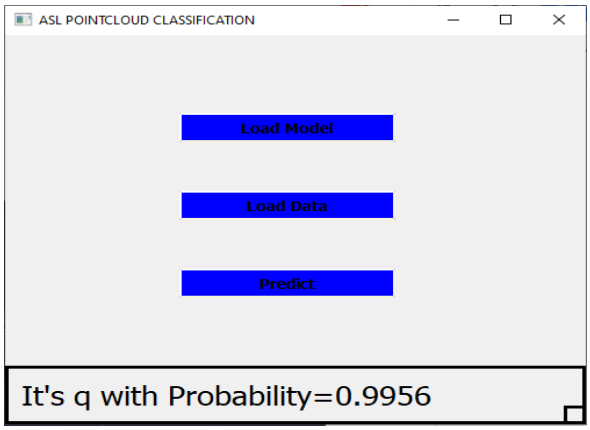

Fig. 8. GUI output

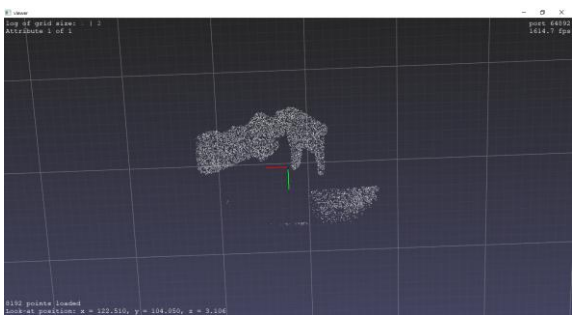

Fig. 9. PPTK viewer window

IX. FUTURE SCOPE

The model can be extended to further use of other sign languages like Indian Sign Language. Further segmentation can be applied to point clouds which classifies them into multiple homogeneous regions. It also analyses various aspects such as location and recognition of alphabets. The model can be trained further to recognize common words and expressions. Symbols with two hands can also be recognized by training further. Also, recognition of dynamic gestures can also be done.

X. CHALLENGES

The first challenge was to comprehend the output given by the CamBoard pico flexx camera and subsequently, choose appropriate 3-D point-cloud formats such as (OFF), (PLY), (PCD), (OBJ), etc. for preprocessing. Further, the challenge was to get uniformly distributed point clouds with the same number of points. Choosing an appropriate down-sampling method for the point cloud of large data points without losing much information.

XI. CONCLUSION

The model after training on GPU using PointNet Architecture and Deep Neural Network for 3D point cloud American Sign Language (ASL) data, using best parameters of training, results obtained showed a consistently high accuracy rate over 96%. Different experimental analyses were performed on training and results were discussed. The study helped to use the network with appropriate parameters. PointNet is a unified architecture that directly takes 3-D point clouds as input and outputs either class labels for the entire input or per point segment labels for each point of the input. Using this network, we were successfully able to use deep neural networks for correctly recognizing and classifying point clouds of static American sign language (ASL) alphabets.

Acknowledgment

We would like to thank our project mentors from IFM Engineering Private Limited. Aniket Patil and Purushottam Ekande. for their invaluable guidance and kind cooperation. We would like to thank our project guide Prof. Dr. Shripad Bhatlawande. Further, we all are also thankful for the support extended by the Management, Dr. R. M. Jalnekar, Honorable Director, Vishwakarma Institute of Technology, Pune.

References

[1] R. Charles, H. Su, M. Kaichun and L. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation", 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. Available: 10.1109/cvpr.2017.16

[2] A. Garcia-Garcia, F. Gomez-Donoso, J. Garcia-Rodriguez, S. Orts-Escolano, M. Cazorla and J. Azorin-Lopez, "PointNet: A 3D Convolutional Neural Network for real-time object class recognition", 2016 International Joint Conference on Neural Networks (IJCNN), 2016. Available: 10.1109/ijcnn.2016.7727386

[3] Z. Zhongyang, C. Yinglei, S. Xiaosong, Q. Xianxiang and S. Li, "Classification of LiDAR Point Cloud based on Multiscale Features and PointNet", 2018 Eighth International Conference on Image Processing Theory, Tools and Applications (IPTA), 2018. Available: 10.1109/ipta.2018.8608120

[4] Zhao, Y. and Wang, L., 2018. The Application of Convolution Neural Networks in Sign Language Recognition. 2018 Ninth International Conference on Intelligent Control and Information Processing (ICICIP)

[5] Das, A., Gawde, S., Suratwala, K. and Kalbande, D., 2018. Sign Language Recognition Using Deep Learning on Custom Processed Static Gesture Images. 2018 International Conference on Smart City and Emerging Technology (ICSCET)

[6] Islam, M., Mitu, U., Bhuiyan, R. and Shin, J., 2018. Hand Gesture Feature Extraction Using Deep Convolutional Neural Network for Recognizing American Sign Language. 2018 4th International Conference on Frontiers of Signal Processing (ICFSP)

[7] L. Zhu, W. Chen, X. Lin, L. He, Y. Guan and H. Zhang, "Random Walk Network for 3D Point Cloud Classification and Segmentation", *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2019. Available: 10.1109/robio49542.2019.8961535