3D Object Classification: PointNet++vs PointConv

Wen Xu

Abstract

This project comprehensively reviews the technical details of PointNet++ and PointConv, and compares the performance of their architectures for 3D Object Classification using the ModelNet40 dataset. The experiment involves preprocessing ModelNet40 dataset, implementing and training using standard deep learning methodologies for each architecture, with evaluation metrics such as classification accuracy, Recall, Precision and confusion matrices carefully recorded and analyzed. Results indicate that PointNet++ achieves slightly better and robust performance in capturing global context and handling diverse object shapes through its hierarchical feature aggregation.

1. Introduction

Point cloud data, often generated from sensors like LiDAR or RGB-D cameras, represents a significant modality in 3D data processing. Processing and understanding such data are crucial for various applications, including robotics, autonomous vehicles, augmented reality, and computer-aided design.

PointNet++ [1], introduced as an extension of the original PointNet architecture, revolutionized 3D deep learning by proposing a hierarchical feature learning framework. This approach enables PointNet++ to capture both local and global geometric features of 3D shapes by iteratively aggregating information from local point neighborhoods to construct comprehensive feature representations. In contrast, PointConv [2] introduces dynamic convolution operations directly on point clouds. By adaptively computing convolutional weights based on local point distributions and densities, PointConv aims to enhance feature extraction capabilities, particularly in scenarios where precise spatial relationships are critical.

This project aims to comprehensively compare the performance of PointNet++ and PointConv architectures on the challenging task of 3D Object Classification using the ModelNet40 dataset (https://modelnet.cs.princeton.edu), which contains 12,311 pre-aligned shapes from 40 categories (Figure 1). The primary objective is to evaluate and contrast how each architecture handles the complexities of 3D shape recognition, including variations in object scales, orientations, and spatial distributions. The approach involves preprocessing the ModelNet40 dataset to nomalize the point clouds, implementing both architectures, traning models and conducting rigorous experiments to measure classification accuracy. Through systematic evaluation and analysis, this study seeks to provide insights into the strengths and limitations of PointNet++ and PointConv, contributing to advancements in 3D deep learning methodologies and applications.

Figure 1. Examples of ModelNet40

2. Related Work

The increasing need for effective 3D point cloud processing has driven significant advancements in deep learning architectures, notably exemplified by PointNet++ and PointConv. PointNet++ extends the groundbreaking PointNet framework by introducing a hierarchical learning structure that captures local geometric details through a multi-scale approach. This methodology involves sampling and grouping local neighborhoods of points to build hierarchical features that progressively aggregate local information into global representations. The hierarchical nature of PointNet++ mimics convolutional layers in traditional CNNs, effectively learning features at different scales and improving robustness to variations in point densities and noise. Its empirical success on benchmarks

such as ModelNet40 and ShapeNet demonstrates its superiority in capturing complex geometric structures, offering significant improvements over PointNet and traditional voxel-based methods.

Following the advancements in hierarchical feature learning, PointConv introduces a novel convolutional operation tailored for point clouds, enabling the application of convolutional networks directly to irregular point sets. Unlike PointNet++, which relies on point-wise MLPs (multi-layer perceptrons) to aggregate features, PointConv implements convolution operations that directly respect the continuous nature of 3D space. This is achieved by dynamically learning the convolutional weights based on the local geometry of point neighborhoods, allowing PointConv to effectively capture local dependencies while maintaining invariance to point ordering. The approach is akin to traditional CNNs, where convolutional filters capture spatial relationships but adapted for non-grid point clouds. PointConv excels in tasks requiring detailed spatial understanding, such as semantic segmentation and object classification, showing improved performance over hierarchical and voxel-based counterparts.

Together, PointNet++ and PointConv represent significant milestones in the evolution of 3D deep learning. PointNet++'s hierarchical framework captures multi-scale features efficiently, while PointConv introduces convolution operations that adhere to the intrinsic properties of point clouds, ensuring better spatial feature representation. Both approaches highlight the ongoing trend towards developing architectures that can directly process irregular 3D data, paving the way for more accurate and efficient models in 3D computer vision.

3. Technical Approach

3.1. PointNet++

PointNet++ enhances the original PointNet architecture by incorporating a hierarchical learning strategy to effectively capture both local and global features in 3D point clouds. The architecture (as illustrated in Figure 2) is designed to process raw point cloud data directly, maintaining its unordered nature and invariance to input permutations. The key innovation in PointNet++ is the introduction of hierarchical feature learning that mimics the multi-scale receptive fields used in traditional convolutional neural networks (CNNs). This approach allows the model to aggregate local features progressively into higher-level representations, enabling it to understand complex geometric structures within the point cloud.

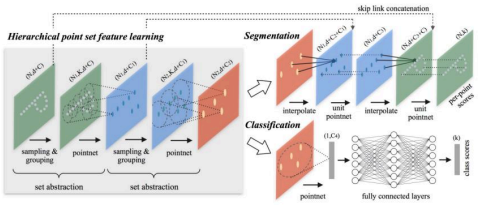


Figure 2. PointNet++ Architecture for Point Set Segmentation and Classification

The approach begins with a sampling layer that selects a representative subset of points from the input point cloud. This is typically done using a farthest point sampling (FPS) algorithm, which ensures a uniform coverage of the entire point cloud. The sampled points serve as the centroids for local neighborhoods, allowing the model to focus on specific regions of the point cloud without processing every point individually. Following sampling, a grouping layer constructs local neighborhoods around each sampled point by selecting points within a fixed radius or the nearest k neighbors. These neighborhoods capture local spatial structures and relationships among points, which are crucial for understanding fine-grained details and local variations in the 3D data.

For feature extraction, the PointNet architecture is applied to each local neighborhood to transform the local point coordinates and features into higher-dimensional representations. Each neighborhood's local features are encoded using a multi-layer perceptron (MLP) followed by max-pooling operations to aggregate these features into a fixed-length descriptor. This process is repeated at multiple levels, forming a hierarchical structure where local features are progressively aggregated into higher-level, more abstract features. Feature propagation layers are then used to interpolate the features from the sampled points back to the original points, ensuring that the final representation captures detailed information across the entire point cloud. This hierarchical structure allows PointNet++ to effectively capture both local and global contextual information, leading to improved performance in tasks such as point cloud classification and segmentation. By leveraging hierarchical learning and adaptive neighborhood search, PointNet++ achieves robustness to varying point densities and spatial distributions, making it a powerful tool for 3D point cloud analysis.

3.2. PointConv

PointConv introduces a convolutional operation specifically tailored for 3D point clouds, addressing the limitations

of earlier models such as PointNet and PointNet++ by directly applying convolutional networks to unordered point sets. Unlike traditional convolutions on regular grids (e.g., images), PointConv adapts the convolution operation to irregularly spaced points, leveraging a continuous convolution framework. This innovation allows PointConv to better capture local spatial relationships in 3D space, enhancing the ability to model geometric structures inherent in point clouds.

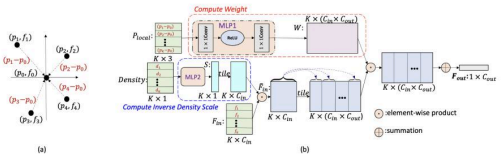


Figure 3. PointConv Architecture

As illustrated in the plot above (Figure 3), the core of PointConv's approach lies in its dynamic convolutional filter generation. Given an input point cloud, the model constructs local neighborhoods for each point based on spatial proximity. For each point and its neighbors, PointConv dynamically computes convolutional weights using learned MLPs that take into account the relative positions of the neighbors. These weights are used to perform a weighted sum of the features in the local neighborhood, akin to traditional convolutions but adapted to the irregular distribution of points. This allows the convolution operation to be sensitive to the local geometry, enabling the capture of complex spatial relationships that are crucial for understanding 3D structures.

To handle the varying density and distribution of points, PointConv incorporates a density normalization mechanism. This mechanism adjusts the convolutional weights based on the density of points in each neighborhood, ensuring that the influence of each point is normalized regardless of local variations in point density. This approach mitigates issues related to uneven sampling, where regions with high point density might otherwise dominate the feature aggregation process. By normalizing for density, PointConv maintains a consistent and unbiased feature representation across different regions of the point cloud, improving the robustness of the learned features.

PointConv also employs a hierarchical feature extraction strategy similar to PointNet++, where multiple layers of convolutions are applied to progressively larger receptive fields. Starting from small, local neighborhoods, the model captures fine-grained details and gradually aggregates these into higher-level, more abstract features through successive layers. This hierarchical approach enables the model to build a comprehensive understanding of the point cloud, capturing both local and global contextual information. The final feature representations are obtained by aggregating these hierarchical features, which can be used for various downstream tasks such as classification, segmentation, and object detection.

The architecture of PointConv is complemented by a feature propagation mechanism that interpolates the features back to the original points from the sampled subset used in the hierarchical process. This interpolation ensures that the final output features are densely distributed across the point cloud, allowing for detailed reconstructions and predictions at the original resolution. The combination of dynamic convolutional filters, density normalization, and hierarchical feature extraction makes PointConv particularly effective for tasks involving complex geometric shapes and varying spatial distributions, providing a more flexible and powerful alternative to earlier point cloud processing methods.

Overall, PointConv represents a significant advancement in point cloud processing by integrating convolutional operations directly into the irregular point domain. Its ability to dynamically adapt convolutional weights based on local geometry, coupled with density normalization and hierarchical feature learning, allows PointConv to capture intricate spatial relationships and handle diverse 3D data distributions effectively. This makes it a robust and versatile framework for various 3D computer vision applications, offering improvements over both voxel-based and earlier point-based approaches.

4. Experiments and Analysis

In my experiments, I employed the ModelNet40 dataset to train Pointnet++ and PointConv. Firstly, to enhance performance, I applied point cloud normalization as an augmentation technique. For PointNet++, I initialized the network with layers for point feature extraction and hierarchical feature aggregation, then trained the model using Adam optimizer with learning rate of 0.001 and batch size of 24 for 200 epochs, meanwhile monitoring the negative log likelihood loss. I repeated the similar process for PointConv, modifying the architecture to include dynamic convolution operations and density normalization to account for varying point densities. After finishing model training, I ran the best checkpoint on testing data and evaluated the models' performance based on accuracy metrics and robustness to variations in object geometry.

Metrics	PointConv	PointNet++
Overall Accuracy	0.92	0.93
Flower Pot Recall	0.15	0.4
Radio Recall	0.6	0.9
Wardrobe Recall	0.65	0.7
Night Stand Recall	0.71	0.7
Flower Pot Precision	0.2	0.26
Cup Precision	0.65	0.65
Xbox Precision	0.74	0.87
Radio Precision	0.1	0.72

Table 1. Performance Comparison between PointConv and PointNet++

The table above summarized the overall accuracy of these two models, along with a comparison of the top 3 worst Recall categories and the top 3 worst Precision categories from both models. Figures 4 and 5 display the Recall per category, while Figures 6 and 7 indicated the Precision per category. The results show that these two models perform very similarly, with PointNet++ slightly outperforms PointConv from overall accuracy as well as in the under performed categories, for example, PointNet achieved better accuracy on flower pot, wardrove and Xbox, etc, which demonstrated PointNet++ capabilities in recognizing 3D objects with diverse shapes and ambiguities.

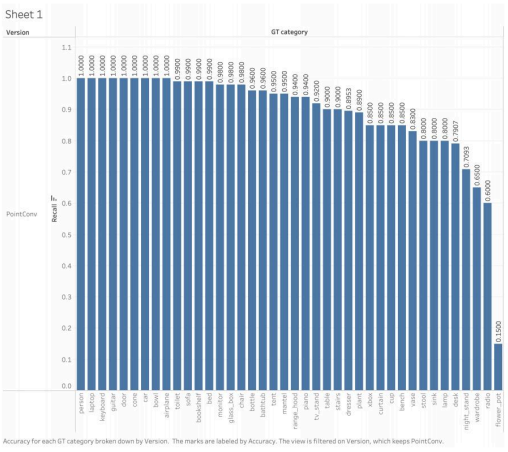


Figure 4. Per Category Recall of PointConv

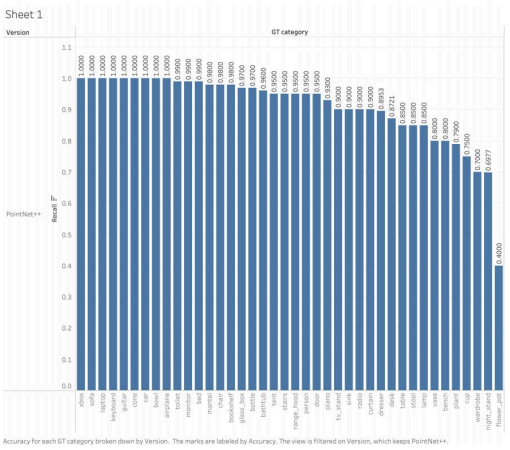


Figure 5. Per Category Recall of PointNet++

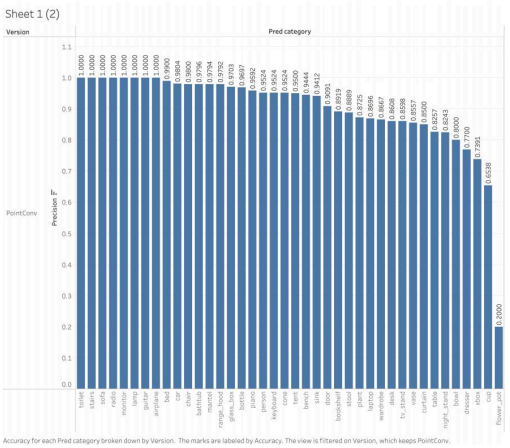


Figure 6. Per Category Precision of PointConv

To better understand the underlying causes of failures in certain categories, I generated confusion matrices for both models (Figures 8 and 9), where the ground truth categories are on the x-axis and the predicted categories are on the y-axis. This can clearly show case between which categories we have the confusion happening.

The matrices reveal a high rate of confusion between categories such as ”flower pot” and ”plant”, ”night stand” and ”dresser”, ”cup” and ”vase”, and ”wardrobe” and ”dresser”. Additionally, the ”radio” category shows confusion with multiple different categories. The potential reasons for these misclassifications fall into three main areas: 1) ambiguous data/label definitions, for example, many ”flower

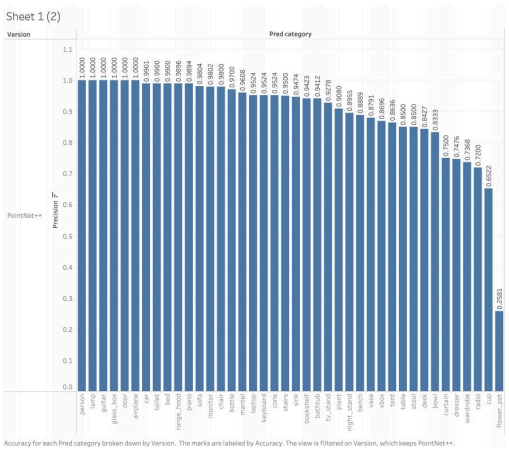


Figure 7. Per Category Recall of PointNet++

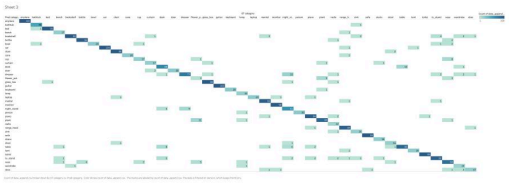


Figure 8. Confusion Matrices of PointConv

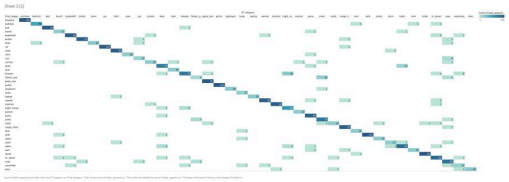
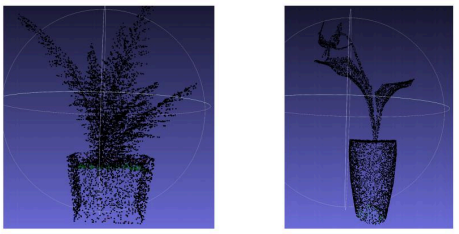


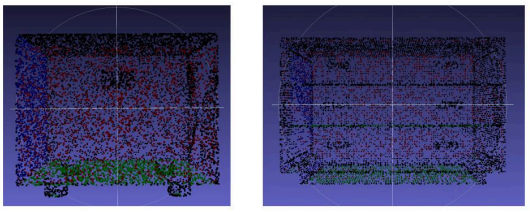
Figure 9. Confusion Matrices of PointNet++

pot" data are actually plants in flower pots cases (Figure 10); 2) similarities in shape, leading to confusion between items like "night stands" and "dressers" (Figure 11) or "cups" and "vases" (Figure 12); and 3) substantial variability within a category, making consistent predictions difficult, as seen with the diverse shapes of radios (Figure 13). As a contrast, the top performance categories such as airplane, car, guitar have much less ambiguities and less diverse shapes. Based on these failure analysis results, it is notable that PointNet++ and PointConv model can capture the 3D geometry well to some extend, but to further enhance accuracy we probably need to seek for some some label cleaning and data balancing work.



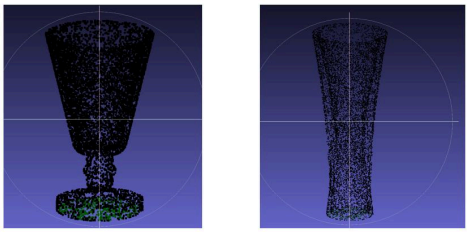
(a) Flower Pot 1 (b) Flower Pot 2

Figure 10. Flower Pot Data with Plant



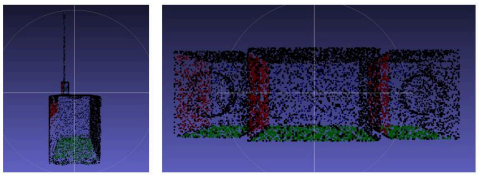
(a) Night Stand (b) Dresser

Figure 11. Confusion between Night Stand and Dresser



(a) Cup (b) Vase

Figure 12. Confusion between Cup and Vase



(a) Radio 1 (b) Radio 2

Figure 13. Different Examples in Radio Category

5. Conclusion

This project has provided a comprehensive comparison of PointNet++ and PointConv architectures for 3D Object Classification using the ModelNet40 dataset. Through rigorous experimentation and analysis, I have gained valuable insights into their respective strengths and weaknesses in handling complex 3D shapes represented as point clouds. PointNet++, leveraging hierarchical feature learning,

excelled in capturing global contextual information and achieving robust classification accuracies across various object categories. Its ability to aggregate local features into hierarchical representations proved effective in understanding and classifying diverse 3D shapes. Conversely, PointConv demonstrated superior performance in capturing fine-grained local features and adapting to varying point densities within the point clouds, showcasing its effectiveness in scenarios where precise spatial relationships are crucial. From my results, PointNet++ shows slightly better performance in overall accuracy and better robustness in handling challenging cases.

In addition, this project underscored the importance of data/label definition and data coverage, which caused the failures in both PointNet++ and PointConv. As next step, I am interested to explore the following areas to try to achieve further model performance improvement. Firstly, enhancing the labeling quality of existing data. For example, cases with plants in flower pots could be labeled as both "flower pot" and "plant." Additionally, increasing the representation of underrepresented data in the training set could help, such as cups that resemble vases or various types of radio.

6. Appendix

Git hub link: https://github.com/wenxu9696/3D-Object-Classification-PointNet-vs-PointConv

References

[1] Qi, Charles Ruizhongtai, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 1

[2] Wu, Wenxuan, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. *In Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 9621–9630, 2019. 1