Dynamic Inference Acceleration of 3D Point Cloud Deep Neural Networks Using Point Density and Entropy

Gyudo Park[1], SooHyeok Kang[1], Wencan Cheng[2], Jong Hwan Ko[3]
[1]AI Machine Research Lab, Hyundai Kefico Corp, Hyundai Motor Group
[2]Department of Artificial Intelligence, Sungkyunkwan University
[3]College of Information and Communication Engineering, Sungkyunkwan University
{Gyudo.Park, SooHyeok.Kang}@hyundai-kefico.com, {cwc1260, jhko}@skku.edu

Abstract

This paper introduces a density- and entropy-adaptive inference acceleration method for 3D point cloud based deep neural networks. Based on the entropy of each input frame, the method first determines the number of points to be inferred. Then we apply a novel density calculation method to sample the points in the order of density of each point. Experiments on two representative 3D scene flow estimation models with the KITTI dataset show that the proposed scheme reduces inference latency by 32% each within 0.01m of the estimation error.

1. Introduction

As the robot navigation and autonomous driving industries are rapidly developing, there is a lot of interest in the technology that recognizes 3D environments based on point cloud data. Recently, deep learning has been applied to this field and has demonstrated excellent cognitive abilities in tasks such as 3D object detection and semantic segmentation. However, most high-performance models require a significant amount of computational resources due to their complex structure, making it difficult to be inferred in real-time on mobile devices with limited resources. Therefore, a major challenge in implementing efficient deep learning models in limited environments such as embedded systems is reducing model size and increasing inference speed while minimizing performance loss.

One possible solution is applying dynamic inference, which is able to reduce the computational cost of deep neural networks by adaptively adjusting the inference process according to the input data. Dynamic inference has evolved in various forms for models that handle 2D image data. For example, there have been dynamic pruning methods [3, 7] that dynamically prune redundant weights when inferring and dynamic quantization methods [15, 20] that train with full precision and dynamically quantize during inference. In addition, other studies identify important parts of input data and use only those parts for inference to increase the efficiency of inference [14, 18].

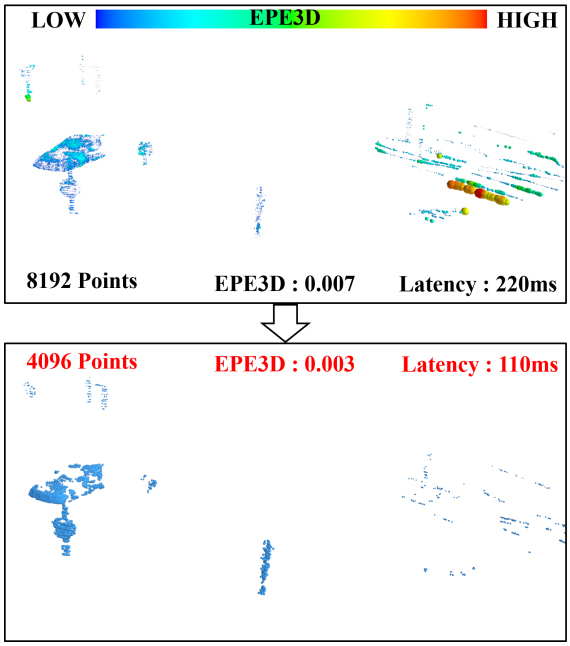

Figure 1. Comparison of Mean EPE3D and inference time by number of points. The color of the figure represents the value of EPE3D (End Point Error 3D) for each point. Blue has small errors and red has large errors.The above figure shows an inference of 8192 points with EPE3D 0.007 and latency of 220ms, while the below figure shows an inference of 4096 points with EPE3D 0.003 and latency twice as fast at 110ms.

However, dynamic inference has not been actively studied for 3D point cloud data. Furthermore, related researches [10, 11, 16, 17] only focus on simplifying 3D convolution operations or downsizing the model structure. Nevertheless, dynamic inference using density can be im-

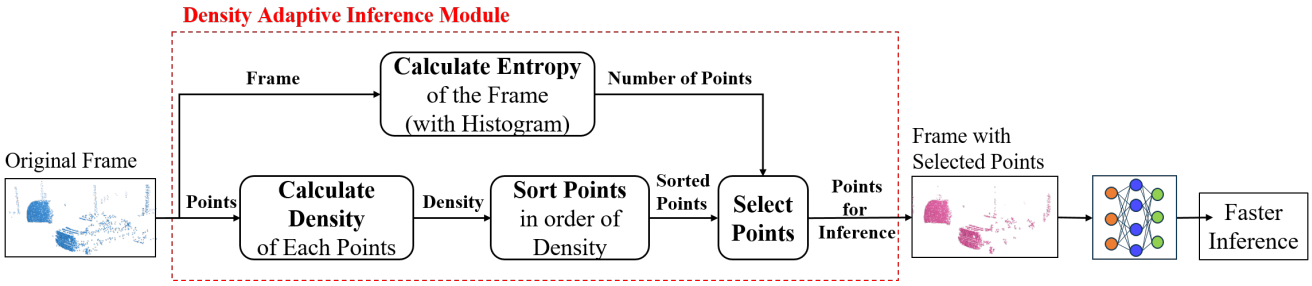

Figure 2. The Overall process of dynamic inference of point based 3D deep neural network. When the input frame passes through the density adaptive inference module, it infers faster with only selected points.

plemented while maintaining performance and reducing inference time. Density is used as a key factor to maintain performance in several studies of point cloud models. Efficient semantic segmentation of Lidar and photogrammetric reconstruction data is performed in [5] by clustering input point cloud data based on density, extracting local features of each cluster, and performing inference using these features. To improve classification performance, DanceNet [9] calculates the density of each point with a multilayer perceptron and applies different convolution kernel sizes adaptively according to the density.

However, these performance-oriented designs using density are complex in structure and require a lot of computational cost, but efficient point cloud down-sampling methods called point cloud compression can reduce computational costs. Effective encoding based on temporal and spatial redundancy of point cloud data was implemented in [2]. However, it is difficult to implement in real-time because it requires finding redundancy from multiple frames. In [4], data is compressed by improved octree through consideration of adjacent octree nodes. In [6], a compression method that learns point-wise features with an auto-encoder to preserve local information is implemented. However, structures that require multiple convolutional operations in [4,6] are difficult to shorten inference time.

Fig. 1 shows that faster and more effective inference is possible when efficient point cloud downsampling using density is performed. This figure is an example of the 3D scene flow estimation inference result of a single KITTI data. The figure above is the original inference result, and the figure below is the inference result of selecting 4096 points in order of small errors in the figure above. It shows that the mean EPE3D and latency are reduced by half compared to the original result, and the selected points are mainly concentrated on dense objects, indicating that density is an important factor.

In this study, we introduce a method of implementing real-time dynamic inference by performing efficient low-cost downsampling based on the density of point clouds. The proposed method determines the number of points to be inferred based on the sparsity of data using entropy that represents the distribution of data. This study also devises a new density calculation method reflecting the characteristics of point clouds where points are densely packed in objects. Subsequently, points are selected in density order for final inference. We evaluate the proposed method based on two representative 3D scene flow estimation models [1,19], which estimate point-wise motions from two consecutive frames. As a result, the inference time was reduced by 32% within the range of mean EPE3D value increase of 0.01 m for the two representative models.

Overall, the key contributions of this work are as follows:

- To reduce the computational cost and latency of the model, we propose a novel dynamic inference approach, which selects points from the input 3D point cloud that contributes more to the deep learning model inference.
- We propose a novel density calculation method that reflects the characteristics of a point cloud set.
- We evaluate the proposed method based on representative 3D scene flow estimation models and present the efficiency and effectiveness of the proposed dynamic inference.

2. Proposed Method

This work proposes a novel dynamic inference acceleration method of 3D point cloud-based deep neural networks that determines the number of inference points based on the entropy of the frame and determines the inference points based on density. Fig. 2 shows an overall process of proposed method. An input point cloud is represented as a set of 3D points $\mathbf{s} = \{s_i | i = 1, ..., n\}$, where each point contains points vector $s_i = (x_i, y_i, z_i)$. The number of points to be inferred is determined through the entropy of the point set $\mathbf{s}$, and the density is calculated for each point s_i to select points to be inferred in the order of density according to the number of points determined. This density adaptive dynamic inference method reduces computational costs by

reducing the points of input data, and minimizes the loss of prediction accuracy by selecting the points based on the density of each point.

2.1. Determining the Number of Sampled Points Using the Entropy

Entropy indicates how much different data are distributed within a given category, and the more diverse the data, the higher the entropy value. By leveraging these entropy characteristics, we are able to find the minimum number of points required for dynamic inference, and thus determine the distribution of point cloud data. Let $\mathbf{p} = (p_1, p_2, ..., p_n)$ be a finite discrete probability distribution and suppose $p_k \geqq 0 (k = 1, 2, ..., n)$ and $\sum_{k=1}^{n} p_k = 1$. The entropy $H(\mathbf{p}) = H(p_1, p_2, ..., p_n)$ is defined as (1).

$$H(p_1, p_2, ..., p_n) = \sum_{k=1}^{n} p_k \log_2 p_k. \quad (1)$$

To produce the probability distribution $\mathbf{p}$ in (1), we apply a multi-dimensional histogram of the input point cloud data. For a point set $s_{1i}(x_{1i}, y_{1i}, z_{1i})$, we create a multi-dimensional histogram of the x, y, z axes, and the data distribution is expressed as a probability by dividing the number of points belonging to each bin by the total number of histogram bin.

Using the entropy of the input data, the number of inference points is determined. We first calculate the minimum (Min(E)) and maximum (Max(E)) entropy of the datasets. Then we determine the minimum number of points (Min(N)) and the maximum number of points (Max(N)) required for inference. Finally, we match the minimum and maximum entropy values with the corresponding minimum and maximum number of inference points. The equation can be represented as

$$N = \frac{Max(N) - Min(N)}{Max(E) - Min(E)} (Input(E) - Min(E)) + Min(N), \quad (2)$$

where N denotes the number of points for inference and E is entropy of the input frame, Max, Min means maximum and minimum.

2.2. Point Sampling Based on the Density

When density is defined as the number of points included in a unit space, the density of each point in a point cloud can be relatively represented by taking the inverse of the volume of that space when the number of points included is the same. If we assume that the number of points included is the same and represent it as 1, and if we call the radius of that space r, which is expressed as

$$Density_{KNN}(\mathbf{s}) = \frac{1}{\frac{4}{3}\pi r_f^3}, \quad (3)$$

When we calculate the density with (3), radius r is the distance from the reference point to the k-th furthest point. However, since there is a possibility that the k-th point may be a noise signal that has been measured incorrectly and it does not reflect the distribution of other points included in that space, the density of that point cannot be expressed only by the distance from the reference point to the k-th point. Therefore, we define a new density calculation method by multiplying the inverse of the radius that reflects the distance for all other points in space by the sum of the volumes and then taking the inverse. This method is expressed as KNN density from obtaining the distance of k nearest points, and the equation is shown as (4).

$$Density_{KNN}(\mathbf{s}) = \sum_{m=1}^{k} \frac{1}{\frac{4}{3}\pi r_m^3 \times \frac{1}{r_m}} = \sum_{m=1}^{k} \frac{1}{\frac{4}{3}\pi r_m^2}. \quad (4)$$

3. Experiment

3.1. Settings

In the experiments, the proposed method is applied to two baselines, PointPWCNet [19] and BiPointFlowNet [1], which are representative 3D scene flow estimation models. Note that, the default number of input points to these two baselines are both 8192. These two models in [1, 19] are trained on FlyingThings3D [12] and validated on KITTI [13], which are two datasets commonly used for 3D scene flow estimation task. The experiments of this study for verifying dynamic inference performance were conducted only with the KITTI dataset. KITTI Scene Flow 2015 dataset provides actual Lidar scans and consists of 142 frame pairs for test. For a fair comparison, we applied the same metrics as those used in recent studies [1,8,19]. Moreover, we focus on the representative EPE3D metric, which measures the average distance error between the predicted pixel displacement vector and the actual pixel displacement vector in 3D space. Experiments were conducted on the Intel i7 9th generation CPU and NVIDIA RTX 2060 GPU with PyTorch.

3.2. Results

Fig. 3 shows the relationship between entropy and density for EPE3D. Fig. 3(a) is the scatterplot of entropy and mean EPE3D of KITTI data. On the right side of the plot, two examples are selected and visualized from the actual input data according to their entropy. These figures show that data with low entropy results in a low mean EPE3D value as points are concentrated on the object. Therefore, if we need to reduce the number of points to be inferred according to the input data, we can judge that it is effective to reduce the number of points in data with low entropy. Fig. 3(b) shows

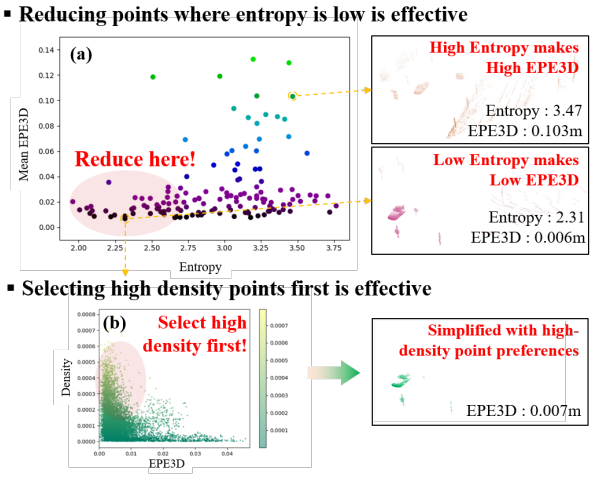

Figure 3. Relationship between Entropy, Density and EPE3D. The lower the entropy, the smaller the mean EPE3D value, and the points of the data with the lower entropy are selected first in the order of the high density.

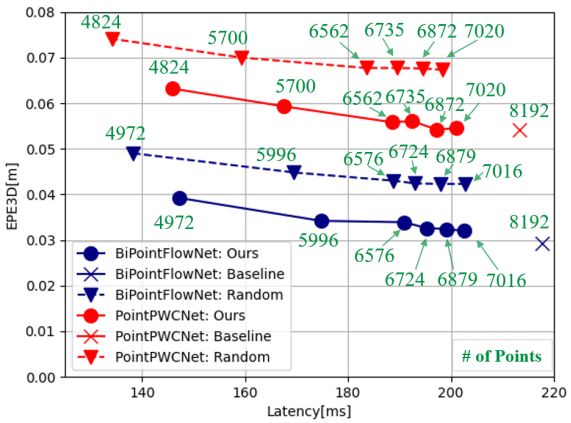

Figure 4. Comparison of baseline performance and proposed method performance for PointPWCNet (red) and BiPointFlowNet (blue). Solid lines represent the result of the proposed method, and dashed lines represent the result of random selection. The horizontal axis indicates latency, and the vertical axis indicates EPE3D. The latency means the time that includes not only the model inference time but also the entropy and density calculation time.

a scatterplot of EPE3D and density for each point of data with low entropy. According to the distribution of density and EPE3D, our method selects points with higher density in higher priority for inference, since they generally produce lower estimation error. As visualized in the lower right, the selected high-density points are well-presented with meaningful shapes and can provide the almost same EPE3D performance as using the full input.

Fig. 4 shows the results of applying the proposed method to PointPWCNet and BiPointFlowNet. For both models, compared to baseline, latency decreases as the number of points to infer decreases, and latency can be significantly reduced within EPE3D 0.01 m (1 cm). Overall, both models reduce latency when inferring with fewer than 8192 points. However, the proposed method avoids the significant EPE3D increase with negligible latency increase compared to the random selection. In specific, the proposed method can reduce the number of input points for PointPWCNet from 8192 (EPE3D=0.054) to 6562 (EPE3D=0.056) with little increase in EPE3D, which reduces inference time by about 25 ms (12%). Additionally, under the allowable EPE3D increase margin of 1cm (0.010 m), our method is able to reduce the number of inference points to 4824 and the model inference time by up to 68 ms (32%). When the inference points are reduced from 8192 to 6724 in BiPointFlowNet, EPE3D increases by only 0.003 m and the inference time is about 22 ms (about 10%) less. If the increase in EPE3D is constrained within 1cm, BiPointFlowNet can be inferred using only 4972 points (EPE3D = 0.039) while reducing the inference time by 70 ms (32%).

4. Conclusion

We proposed a dynamic inference acceleration method for 3D point cloud-based deep neural networks using densities and entropy that can be useful for the deployment of embedded systems. By calculating the entropy of each frame, the proposed method determines the optimal number of points to infer and employs a novel point cloud density calculation method for inferring scene flows. Experimental results reveal that the inference speed can be greatly reduced with little increase in error for 3D scene flow estimation deep learning models. In the future, we aim to improve the performance to the level where real-time control is possible by inferring within 100 ms in limited environments such as robots and vehicle controllers. We also plan to expand our research to other 3D tasks such as 3D object detection and 3D semantic segmentation by combining and improving various downsampling techniques such as point cloud compression.

Acknowledgments

This work was partly supported by the Institute of Information and Communication Technology Planning & Evaluation (IITP) grants (IITP-2019-0-00421, IITP-2020-0-00821) and National Research Foundation of Korea (NRF) grant (2022R1A4A3032913) funded by the Korea Government (MSIT).

References

[1] Wencan Cheng and Jong Hwan Ko. Bi-pointflownet: Bidirectional learning for point cloud based scene flow estimation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 108–124. Springer, 2022. 2, 3

[2] Yu Feng, Shaoshan Liu, and Yuhao Zhu. Real-time spatio-temporal lidar point cloud compression. In *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 10766–10773. IEEE, 2020. 2

[3] Xitong Gao, Yiren Zhao, Łukasz Dudziak, Robert Mullins, and Cheng-zhong Xu. Dynamic channel pruning: Feature boosting and suppression. *arXiv preprint arXiv:1810.05331*, 2018. 1

[4] Diogo C Garcia and Ricardo L de Queiroz. Context-based octree coding for point-cloud video. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 1412–1416. IEEE, 2017. 2

[5] Timo Hackel, Jan D Wegner, and Konrad Schindler. Fast semantic segmentation of 3d point clouds with strongly varying density. *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, 3:177–184, 2016. 2

[6] Yun He, Xinlin Ren, Danhang Tang, Yinda Zhang, Xiangyang Xue, and Yanwei Fu. Density-preserving deep point cloud compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2333–2342, 2022. 2

[7] Weizhe Hua, Yuan Zhou, Christopher M De Sa, Zhiru Zhang, and G Edward Suh. Channel gating neural networks. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[8] Yair Kittenplon, Yonina C Eldar, and Dan Raviv. Flowstep3d: Model unrolling for self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4114–4123, 2021. 3

[9] Xiang Li, Lingjing Wang, Mingyang Wang, Congcong Wen, and Yi Fang. Dance-net: Density-aware convolution networks with context encoding for airborne lidar point cloud classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 166:128–139, 2020. 2

[10] Yujun Lin, Zhekai Zhang, Haotian Tang, Hanrui Wang, and Song Han. Pointacc: Efficient point cloud accelerator. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 449–461, 2021. 1

[11] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[12] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 3

[13] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3061–3070, 2015. 3

[14] Mahyar Najibi, Bharat Singh, and Larry S Davis. Autofocus: Efficient multi-scale inference. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9745–9755, 2019. 1

[15] Jianghao Shen, Yue Wang, Pengfei Xu, Yonggan Fu, Zhangyang Wang, and Yingyan Lin. Fractional skipping: Towards finer-grained dynamic cnn inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5700–5708, 2020. 1

[16] Haotian Tang, Zhijian Liu, Xiuyu Li, Yujun Lin, and Song Han. Torchsparse: Efficient point cloud inference engine. *Proceedings of Machine Learning and Systems*, 4:302–315, 2022. 1

[17] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII*, pages 685–702. Springer, 2020. 1

[18] Wenhao Wu, Dongliang He, Xiao Tan, Shifeng Chen, Yi Yang, and Shilei Wen. Dynamic inference: A new approach toward efficient video action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 676–677, 2020. 1

[19] Wenxuan Wu, Zhiyuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. Pointpwc-net: A coarse-to-fine network for supervised and self-supervised scene flow estimation on 3d point clouds. *arXiv preprint arXiv:1911.12408*, 2019. 2, 3

[20] Haichao Yu, Haoxiang Li, Humphrey Shi, Thomas S Huang, and Gang Hua. Any-precision deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10763–10771, 2021. 1