NUMERICAL METHODS AND DATA ANALYSIS

Transformer point net: cost-efficient classification of on-road objects captured by light ranging sensors on low-resolution conditions

J. Pamplona [1], C. Madrigal [2], J. Herrera-Ramirez [3]
[1] GICEI, Institución Universitaria Pascual Bravo, Calle 73 No. 73a - 226, Medellín, Colombia;
[2] Grupodot S.A.S., Ak. 19 # 84-17 Oficina 701, Bogotá, Colombia;
[3] GAEYCC, Instituto Tecnológico Metropolitano, Calle 54a No. 30 - 01, Medellín, Colombia

Abstract

The three-dimensional perception applications have been growing since Light Detection and Ranging devices have become more affordable. On those applications, the navigation and collision avoidance systems stand out for their importance in autonomous vehicles, which are drawing an appreciable amount of attention these days. The on-road object classification task on three-dimensional information is a solid base for an autonomous vehicle perception system, where the analysis of the captured information has some factors that make this task challenging. On these applications, objects are represented only on one side, its shapes are highly variable and occlusions are commonly presented. But the highest challenge comes with the low resolution, which leads to a significant performance dropping on classification methods. While most of the classification architectures tend to get bigger to obtain deeper features, we explore the opposite side contributing to the implementation of low-cost mobile platforms that could use low-resolution detection and ranging devices. In this paper, we propose an approach for on-road objects classification on extremely low-resolution conditions. It uses directly three-dimensional point clouds as sequences on a transformer-convolutional architecture that could be useful on embedded devices. Our proposal shows an accuracy that reaches the 89.74 % tested on objects represented with only 16 points extracted from the Waymo, Lyft's level 5 and Kitti datasets. It reaches a real time implementation (22 Hz) in a single core processor of 2.3 Ghz.

Keywords: LiDAR, point cloud, deep learning, object classification, transformers, low resolution, autonomous vehicles, low specification computing.

Citation: Pamplona J, Madrigal C, Herrera-Ramirez J. Transformer point net: cost-efficient classification of on-road objects captured by light ranging sensors on low-resolution conditions. Computer Optics 2022; 46(2): 326-334. DOI: 10.18287/2412-6179-CO-1001.

Introduction

As LiDAR (Light Detection and Ranging) sensors become more affordable, computer vision systems based on 3D (three-dimensional) information become popular in applications from mapping and cartography [1], to robotics and navigation [2, 3]. In autonomous vehicles, LiDAR information has become one of the principal sources to ensure effective navigation. Commonly, autonomous driving systems use multi-sensor for their perception system [3]. But, due to some limitations of RGB cameras under certain conditions and the potential of LiDAR information, those methods that use computer vision using exclusively 3D information became an interesting field for the investigation community.

Processing of 3D information for classification, segmentation or detection are crucial tasks for collision avoidance systems applications. Those systems are inherent to auto manned vehicles, given their relevance to road safety [4]. These tasks have been addressed using different representations for the 3D information, where the point cloud is the native format of LiDAR. This representation has an extended use for visualizing 3D objects and scenes, but it has an intrinsic processing difficulty due to the default shape of the stored data [5]. This shape configures a challenge to establish geometrical or proximity relationships between points.

To face this challenge, the applications for 3D object classification mainly uses three different approaches for representations. Most of the typical representations have been based on 2D views images that take advantage of the acquired experience on RGB image analysis. This approach has been present in models where a significant number of 2D rendering extraction techniques have been developed. 2D point cloud slices [6, 7], object views [8], elevation images [9,10] and spherical representations [11, 12] are some of these, each with diverse versions. 2D views representations usually omit relevant 3D information, which can give a performance drop.

Another popular approach is the voxelization of the object space that maintains the 3D representation but giving order to the information [6, 13]. The voxelization

techniques have improved with the years, getting better and better descriptions of points inside each voxel [14]. These representations are usually processed using three-dimensional convolutions to exploit the 3D spatial relationship. That can be computationally expensive (compared with techniques for processing other representations) given the resolution needed and its sparse nature.

Even when getting organized information is the most intuitive way to process the point clouds, changing the representation is a process that generally skips information that is potentially relevant in the point cloud analysis. Therefore, there is an entire family of methods that works directly over the point cloud representation. Most of them establish correlations between nearby points as mesh analysis [15]to get geometrical descriptors [16, 17]. Nowadays, with the popularity of CNN (Convolutional Neural Networks) [18], several methods are presented where the CNNs have adapted to the irregularity of a point cloud. X-conv [19], spider [20], Annularly [21], spherical [22] and graph convolutions [23] are some examples of the extensive spectrum of possibilities to analyze point clouds. This kind of models face a bottleneck stablishing relationship among the nearest points to start the CNN processing. To solve this restriction, some models have used directly point clouds with approaches that can learn the spatial relationships between points. Examples of these approaches are PointNet [5] and Point Cloud Transformer (PCT) [24].

These days the deep learning methods have entirely dominated computer vision applications, including object classification and segmentation. The point cloud classification task is not the exception, where those that use CNNs are the most common applications. These deep learning methods tend to get bigger and bigger. An example is PointNet [5] that had 3.5M of parameters (M refers to million) in its first version, but its upgrade (PointNet++ [25]) has versions with 53M parameters. This upgrade obtains better local descriptors improving its performance in complex environments. This increase has been backed by the growth of the computational capabilities that are becoming accessible by cloud computing services that allow high-performance computations without owning the specialized hardware.

The point cloud classification tasks can be applied on a bunch of problems but as mentioned before, navigation and collision avoidance systems shine not only in autonomous vehicles but in mobile robotics platforms too. When the deep learning methods get to the implementation stage on these platforms, there are two main considerations:

1. Inference time: The real time is required on systems that should take decisions on its perception system.
2. Computation capability. It can be restricted in mobile platforms where real time processing is needed (the latency of cloud computing makes it unviable) and the power availability.

Aside from autonomous cars in development by the principal technological corporations and some high educational institutions, there are robotics applications that use constringed platforms, as home robotic assistants [26] and unmanned aerial vehicles (UAV) [27], which are very limited by the considerations above. On these applications, the methods on the state of the art are restricted by the computation capability-latency relationship. Additionally, those platforms commonly use low-resolution LiDAR sensors or RGB-D cameras (D for Depth) that could restrict the detected objects resolutions. Then, to overcome this limitation the used models should be competent to analyse low-resolution point clouds as in [28], but at the same time should be efficient enough to make the best use of available resources.

Motivated by these challenges, we propose an efficient method for 3D low-resolution on-road object classification optimized to run on mobile platforms. Our approach takes advantage of the low-resolution point clouds to generate constrained descriptor maps keeping a general shape description. As architecture, we adopt an end-to-end design that consumes directly point clouds as sequence of points and gets a transformation with a transformer module that encodes the correlations between points. We claim that this transformation allows our method to use the encoded sequence to get a spatial aware feature matrix suitable for compact CNN processing.

There are three main contributions in this work:

1. A point cloud transformation using it as a sequence into a transformer layer that learns the sequence-to-sequence transformation in order to generate CNN usable data.
2. A constrained model that is suitable for low computing capability hardware as single-board computers.
3. An efficient method that classifies effectively on-road objects in extremely low-resolution conditions (16 points).

Related work

Although there are diverse representations of the LiDAR data, as mentioned before, our preliminary analysis will only include those methods that directly process the point cloud, which will save some extra processing while change the original format. Those methods have based on deep learning architectures that have shown a better generalization on a wide range of applications.

A point cloud is usually stored as a matrix where the columns are the dimensions (x, y, z) and each row represent a single point. The points represented in contiguous rows are not necessarily the closest in the 3D space. In this representation, small changes in a point position can send it far from its original row. Some approaches exploit 3D information by using grouping points methods, but this can be computational expensive. Alternative approaches have overcome this bottleneck by learning computationally cheap transformations using directly point clouds.

PointNet appeared as a pioneer of raw point cloud processing method [5]. It solves the classification and segmentation task by proposing an architecture that learns

a symmetric point cloud transformation. To achieve this, it introduces a T-Net that can generate a point transformation matrix. The T-Net is a trainable CNN that uses the input matrix to generate a features vector that a neural network or MLP (Multi-Layer Perceptron) reshapes into the matrix elements (fig. 1). The same T-Net approach is used later in the model to get a feature transformation matrix which turns the transformed point cloud into a feature tensor. Those features are now spatial arranged in a squared matrix that are processed later by a classical CNN for final classification [5].

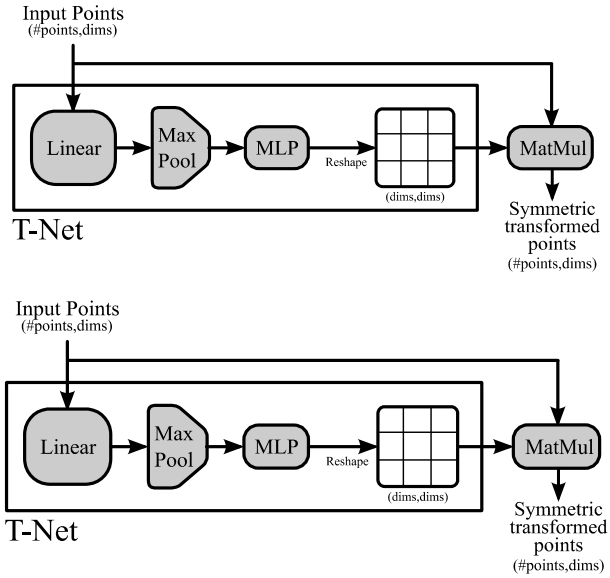

Fig. 1. Role of T-Net in symmetric point transformation

This initial work is not as robust as the newest ones because this is a global features analysis approach. But it establishes a base for subsequent methods like PointNet++ [25] which uses the same principle over grouped points, retrieving local descriptors and enhancing its robustness.

In the deep learning field, CNN has positioned as the standard for spatial correlations analysis. But in recent days, a famous natural language processing (NLP) approach is gaining attention on image processing (previously monopolized by CNNs). The transformers models proposed by google brain and research teams for NLP [29] are getting into fields owned by CNN spatial analysis. Now it is possible to use transformers in image analysis [30], and also, some methods start to appear on point cloud processing. Although the point cloud analyzed by attention-based models was explored successfully earlier by Point2Sequence [31], the new transformers approach has improved the performance with simpler architectures. The best example of this is PCT [32] that uses a point embedding layer to encode the point cloud into a higher dimensional feature space that is processed by four point-transformer blocks.

The task of these blocks is to make a sequence-to-sequence transformation that encodes de relationships between sequence positions by an attentional function. This function associates each input position with the other, done by the multi-head attention block (fig. 2*b*) which is constructed by parallel self-attention layers, schematized in fig. 2*a*., where the Q (Query), K (Key) and V (Value) vectors are used to imitate a search engine. The dot product of Q and K generates a scoring matrix that encodes the relationship of each point with the others. The score matrix is processed with the V vector to give attention to the relevant values on the sequence. Self-attention process is described in (1).

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T / (d_k)^{1/2}) \times V. \quad (1)$$

Multi-head attention block encodes how each position in the sequence is related to the others. This encoded positional representation is enriched with residual connections and a feed forward layer (fig. 8*b*) [29].

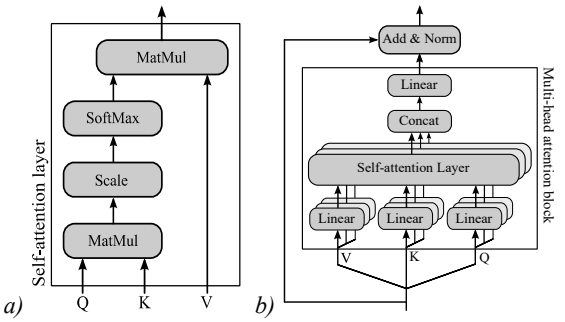

Fig. 2. Transformer schemes: a) self-attention layer, b) multi-head attention block with residual connection

PCT uses the point embedding layer followed by transformer blocks to get four sequential representations with attention information. Those are concatenated and processed by a linear convolutional layer that conditions the features for a classification MLP. In recent years, the PCT model was the first attention-based model that got excellent results on ModelNet40 benchmark [33] without using any convolutional layer [32].

Transformer point net details

In this section, we describe our proposal as follows: first, explaining the transformer-based point cloud transformation. Then we present how the sequence to features transformation matrix is achieved. Finally, our architecture for point cloud classification is presented, explaining how it is conditioned for low-resolution point clouds.

Order encoding transformation

As said before, the principal challenge for applying deep learning methods on raw point clouds is its disordered nature. To face this, we take advantage of sequence correlation encoder capability of the transformer model. This is used as follows. First, the rows of the matrix where the points are stored are used as a sequence. A positional encoding should be attached to this sequence showing an index base to the transformer. The original transformer paper used a sine-cosine representation. But

our model learns from the raw point cloud to obtain a specific positional encoding for our task. It should be considered that our point sequence is not of the same nature as those used by NLP. Both options have identical performance on NLP [29]. The positional encode is then learned by a feed forward block and bot. Positional encode and point cloud are used as inputs of a single transformer block. The transformer block sequence output encodes the input as a representation of the point cloud that contains point association information.

Sequence to features matrix transformation

As our work is focused on low-resolution point clouds, it is decided to adopt the first PointNet approach to generate general features instead of localized ones [5]. It is also used the transformation matrix approach from the same model. In our approximation, the T-Net (fig. 1) is used to learn a transformation matrix that turns the sequence into a spatial coherent feature tensor suitable for CNN processing.

Low resolution point cloud classification architecture

There are a couple of considerations to consider to architecture conditioning for low-resolution point cloud classification (as low as 16 points). The architecture configuration will differ from high-resolution 3D objects classification in a couple of points. First, there are some techniques of continuous subsampling or local feature extraction to get compact localized feature maps. Most of the actual methods use these techniques, but these are oversized for low-resolution objects. Thus, the high dimensional features that can be useful in describing high-resolution point clouds could be redundant and reducing it could potentially improve latency without penalizing the model performance. Finally, after proving that the performance of PointNet and PCT models are not punished by its dimensional reduction (experiments showed in the next section), the most simplistic possible configuration is proposed using the transformer-based point cloud transformation, and the sequence to features transformation blocks followed by a CNN and ending in a multi-layer perceptron (Fig. 3)

Experiments

In this section, we conducted a series of experiments that compare different architectures performance and validate the effectiveness of our proposal.

Dataset conditioning

As the model is designed as a tool for mobile platforms that takes real-world images with LiDAR like sensors, the typical benchmarks for 3D objects classification as ModelNet40 [33] are not an appropriate data source. The best source of the desired data is the autonomous vehicles datasets since most of these have LiDAR sensors that naturally capture distant objects as low-resolution point clouds.

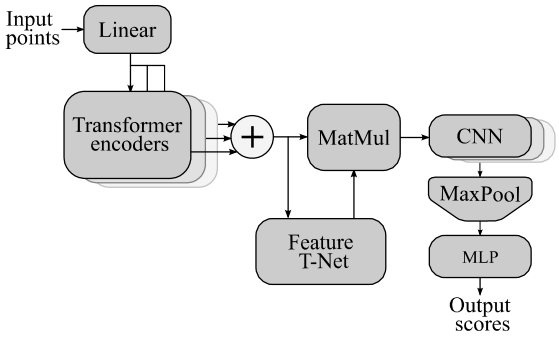

Fig. 3. Low-resolution point cloud classification scheme

Nowadays, there are enough LiDAR data available due to open datasets like Waymo [34], Lyft's level5 [35] and KITTI [36], which have been acquired by autonomous vehicles in development. These have been used on prediction and detection tasks, so these have labels that allow knowing the position and the spatial bounding box. So, it is possible to retrieve every point cloud that represents a single object in the scene. Each dataset has its own class types, but some could not be used because it could lead to a highly class imbalance. So, the classes used across the three datasets were vehicles, pedestrians, traffic signs and cyclists, which are the classes on the larger dataset (Waymo). As for the other two datasets, those include additional classes as sitting pedestrians, bicycles (without driver), miscellaneous objects like trash containers, and specific vehicles like vans or trucks. Those classes have low representation across the datasets, which will induce poor performance in a model where those classes are classified. If it is required, should be done an exhaustive labelling process to ensure good performance in a wider number of classes, but this is out of our scope.

The three datasets are composed of sequences of captures on-road environments, so randomly taken examples could lead to a non-objective evaluation due to most of the objects are almost identical in two consecutive captures. Then, some strategic sequences are selected in order to have enough quantity of each class for evaluation. This result in an unusual initial training-testing rate which is different for each class (due to class imbalance), but it ensures an objective model evaluation.

As the model focuses on low resolution, the objects with the least number of points were chosen (minimum 16 points). Those represent objects captured at a considerable distance (max 76 m for traffic signs, 88 m for pedestrians and 100m for vehicles and cyclists). The cyclists is the class with the least number of elements on the chosen ones across the datasets. To increase the number to almost 100,000 training examples, a simple $x-y$ axis swap was made. To get class balance, was randomly taken the same number of training examples from the other object classes. Finally, the training dataset is composed of 390,000 training examples, equitably divided into four classes. The same procedure was done over the validation dataset to reach the 24,000 examples fixing the training-testing rate (training 94 % – testing 6 %).

As deep learning models need a specific number of points to be trained, the 390,000 examples were subsampled in a structured way to 16 points. Our approach conserves points distributed on the extension of the object in a cheaper computationally scheme compared with the k-nearest neighborhood [37] based ones.

Models and configuration evaluations

To show the pertinence of our approach, we present an evaluation of the two reference models based on their sequence length and feature depth. The evaluation is made by analyzing the tradeoff between performance (accuracy) and computational complexity. Although there are many considerations in computation complexity for deep learning models, number of parameters (params) and FLOPs (number of floating points operations) are used because they could give a sufficient idea of computation needs. It is possible to choose the mobile device as Jetson platforms by knowing FLOPs and the number of parameters of a model [38].

The following tables, it is present the evaluation of the PCT [32] and PointNet [5] models. Given the small number of points, each model is trained until a clear stabilization tendency was observed.

Table 1 shows the configuration analysis of the PCT model for on-road 3D object classification after training it over the dataset defined in the previous subsection. Its columns show the results with model depth variation as transformer block number. The rows show the evaluation of the model with the reductions that follows:

1. In the first row the evaluation is made over the model only with transformers depth modification.
2. In the second row, the MLP was reduced to 2 layers.
3. In the third row, the model is reduced in the length features produced by the linear convolutional layer.
4. In the final row, the reduction is applied in the length of the sequence generated by the transformer blocks.
5. All the reductions were applied on the previous ones.

Tab. 1. PCT model configuration evaluations

Model reductions	Metrics	Transformer blocks		
		4	3	2
Original	Acc. (%)	83.18	84.85	**87.84**
	FLOPs (M)	241.55	193.61	**145.68**
	Params (M)	1.4	1.29	**1.17**
Smaller MLP	Acc. (%)	83.05	85.02	87.58
	FLOPs (M)	241.28	193.35	145.42
	Params (M)	1.28	1.16	1.04
Smaller features	Acc. (%)	82.42	84.63	86.41
	FLOPs (M)	229.84	184.04	138.23
	Params (M)	0.81	0.69	0.58
Sequence reduction	Acc. (%)	81.67	84.31	86.6
	FLOPs (M)	71.63	57.38	43.14
	Params (M)	0.35	0.31	0.28

Tab. 1 summarizes the results in all the configurations combinations explained before, where one of the most computational expensive configurations of PCT get the better results (cell in bold numbers). But the advantage of this configuration is only 1.24 % compared with the lighter one, which gives the results highlighted in the shaded cell. It is almost 3.4 times lighter in FLOPs and with 4.2 times fewer parameters. That small accuracy drop seems to be rational on the trade performance-computation cost. Another interesting fact is observed in the most expensive configuration (first row, first column), which is not getting the best results. This shows that deeper architectures are not the best election to analyze low-resolution point clouds.

Tab. 2 presents a similar analysis for PointNet architecture. Table columns define depth of generated features in the convolutional layers. In the first column, the model is analyzed without changes, but this is reduced in the following ones. In rows, some model reductions are evaluated following the rows analysis in tab. 1, except by the last row where a T-Net reduction is applied.

Tab. 2. PointNet model configuration evaluations

Model reduction	Metrics	Feature depth		
		64	48	32
Original	Acc. (%)	88.63	88.71	87.74
	FLOPs (M)	371.75	271.5	182.81
	Params (M)	36.51	30.14	24.3
Smaller MLP	Acc. (%)	86.9	87	87.58
	FLOPs (M)	116.67	73.11	41.1
	Params (M)	11.12	7.94	5.3
Smaller CNN	Acc. (%)	88.11	**88.82**	87.7
	FLOPs (M)	36.06	**26.41**	19.14
	Params (M)	4.78	**3.79**	2.93
Smaller T-Net	Acc. (%)	86.95	86.6	86.26
	FLOPs (M)	25.03	15.39	8.11
	Params (M)	3.26	2.27	1.41

In tab. 2 it is noticeable that the best performance (highlighted in bold numbers) is not achieved by the deepest configuration. Then it is proved that low-resolution point clouds can be effectively classified by constrained models, even with better results. This time, the accuracy of the original configuration (shaded cell) is improved by 0.19 % in a model with 14 times fewer FLOPs and 9.6 times fewer parameters.

The comparison of results in tab. 1 and 2 proves that there is not a significant loss in performance when the classification methods for extremely low-resolution point clouds get smaller. Even more, it could be possible to improve the performance with simpler architectures for this task. This information gives relevance to the proposed model, which is presented in terms of low computational needs.

Proposal evaluation

Our proposal is evaluated as follows: First, we do a general evaluation of the testing data in all the datasets. Then, the results are presented for each one. Finally, we perform a distance robustness analysis.

We configured our model (fig. 3) based on the results presented in tab. 1 and 2. Based on those, the hyperparameters for the model were determined by changing the following parameters:

1. Increasing the transformers blocks (1-4).
2. Increasing the Convolutional layers (2-5).
3. Increasing the features depth in each CNN layer (starting from the half of the initial model and scaling by 2 in 4 steps).
4. Alternating the Cost function between Categorical Cross Entropy and MSE
5. And using the Adam Optimizer by increasing its learning rate from 0.000001 to 0.01 in powers of 10.

This optimization in 5 dimensions is performed by exhaustive search giving the best performance on the model with two transformer and convolutional layers, the second scaling of feature depth, a cross-entropy cost function and a learning rate of 0.0001 in an Adam optimizer.

Once defined final shape of the model, it is ready for a training process. But before this, it is performed a couple of tests that will set a baseline of performance on denser point clouds. With 128 points resolution, is ensured enough information to describe an object shape without restricting the acquisition distance to a couple tens of meters and are far from extreme low resolution. The first test was done with the original configuration of PCT, where the accuracy achieved 93.63 %. Secondly, the original model of PointNet gets 93.47 % accuracy on the same dataset. On the other hand, the evaluation of our model shows an accuracy of 90.43 % under these conditions. But that reduction is expected because our model was designed for 16 points resolution.

As a result of this training (fig. 4), there we get 89.74 % in accuracy, improving the best results obtained in PoinNet and PCT for 16 points resolution dataset. Tab. 3 present the comparison of the three models results and characteristics.

Tab. 3. Model comparison

Model	FLOPs (M)	Params (K)	Accuracy (%)
PCT	145.68	1170	87.84
PointNet	26.41	3790	88.82
Ours	3.03	69.4	89.74

Those results are achieved by training and testing the architectures on the same computing system, over the same dataset described before. On this comparison, the improvement of our proposal is clear, enhancing the accuracy by 0.88 % with an architecture overwhelmingly compact.

In fig. 4 it is presented the training loss, the training and testing accuracy, and the learning rate decay are observables. It is noticeable that the training process was performed in a single session starting from scratch. Although some sections of the architecture were based on existing models, there is not possible to use transfer learning as the number of layers and feature depths are different.

On the other hand, comparing the model performance with PCT and PointNet results on the 128 points resolution, our model accuracy drop 3.89 %. This result can be explained by the lack of information on 16 points. Even under those conditions, the accuracy penalization is not as relevant as can be expected.

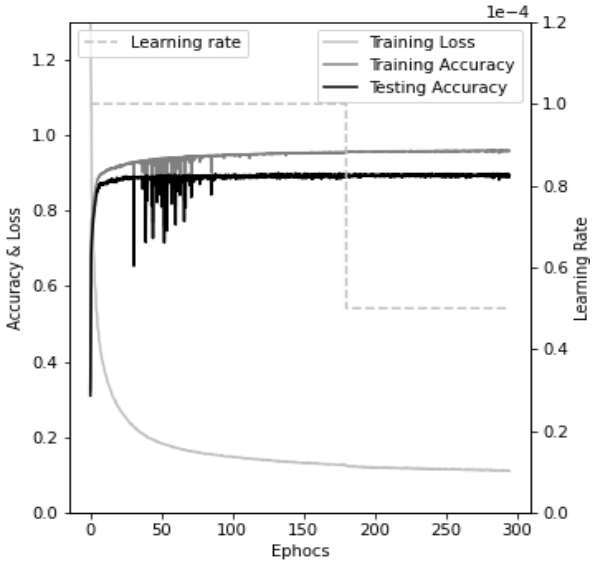

Fig. 4. Training process plot. On the left axes are presented training loss, training and testing accuracy. On the right axes is shown the optimizer learning rate

Our results allow making inference of a single point cloud in 45.4 ms using a single-core CPU at 2.3 Ghz. With that latency, it is suitable for most mobile computing platforms as Jetson (Nvidia) or even Raspberry (using multicore processing).

Besides the computational efficiency, it is relevant to the classification performance, which should be observed with full attention because although the accuracy metric is enough for comparison, it is not showing many details about the model performance. In order to give a better picture of the model performance we used the F1 score metric that take into account other aspects of the model performance (precision and recall) [39]. This metric allows to take all objects in the sequences selected for evaluation (unbalanced dataset), and to evaluate the performance of the proposal on each class. Tab. 4 summarize the performance (using the F1 score) of the model specified by class (columns) across each dataset (rows).

Tab. 4. F1 model score by class and dataset

Dataset	Car (%)	Pedestrian (%)	Trafic sign (%)	Cyclist (%)	Average (%)
Kitti	99.72	89.77	N/A	80.56	90.02
Lift	99.86	86.30	N/A	84.49	90.22
Waimo	98.47	92.47	96.79	44.47	83.05
All	98.84	92.25	96.78	53.61	85.37

This analysis shows that our method is highly efficient in the classification task over most classes, except on cyclists. But it is especially notable on the Waymo da-

taset, where the F1 score is under 50 %. On that dataset, two factors can contribute to this result. First, the cyclist is highly scarce compared with the other classes. Second, as the dataset is noticeably larger than the others, it contains many poses or representation which are not present in Kitti or Lyft level5 datasets. Those representations can be classified as cyclists class that does not affect accuracy significantly but is highly relevant on the F1 score.

The last column of tab. 4 is calculated as an average of the F1 score across classes without using weighted estimation. With an overall weighted F1 average score of 95.75 %, but this value is biased by the high number of non-cyclists objects.

The last analysis of the model is his ability to classify objects captured by LiDAR sensors at high distances. On the dataset construction, there was stored the capture distance among each object. That magnitude was calculated using the Euclidean distance over the (*x, y*) plane. Those distances were grouped by 20 meters intervals and analyzed by class type. Tab. 5 consolidate the classification accuracy for each class on five distance intervals.

Tab. 5. Model accuracies by capture distance

Distance	Car (%)	Pedestrian (%)	Traffic Sign (%)	Cyclist (%)
(0, 20]	90.02	91.76	96.65	80.88
(20, 40]	94.37	91.17	95.45	72.81
(40, 60]	98.93	92.60	95.47	69.57
(60, 80]	99.49	91.37	93.74	62.02
(80, 100]	99.82	0.00	N/A	100.00

The accuracies on Table 5 shows a consistency on pedestrian and car classes. A small decreasing is observed on traffic signs class, but the cyclist class shows a high influence of distance in its classification. The continues decreasing of cyclists classification performance shows that the distance should be added to the factors affecting the F1 score presented on tab. 4.

Conclusions

In this work, we proposed an extremely low-resolution 3D object classification method for on-road elements, which is suitable for real-time performance on portable computation platforms. We evaluated the ratio between performance and computation cost for two well-known state-of-the-art approaches for justifying and comparing the performance of our method. Our proposal took advantage of the transformer proficiency for sequential elements association, and we used it as a point cloud transformation approach to obtain a sequence that encode the position of points and the relation between them. The sequential representation is exploited with the PointNet alike tools obtaining impressive results. The Experiments over three datasets have proved the effectiveness for 3D on-road objects classification defined in 16 points. Those point clouds were processed in our model, at least 8.7 times smaller in FLOPs than the state-of-the-art methods on point cloud object classification. This architecture has an initial potential application in low-end mobile robotics, and we think it can be scalable to more demanding applications. Future work will investigate how could this approach be helpful in more complex tasks as the segmentation of point cloud scenes or object detection. Also focused on the computational complexity optimization.

References

[1] Debeunne C, Vivet D. A review of visual-LiDAR fusion based simultaneous localization and mapping. Sensors 2020; 20(7): 2068.

[2] Kolhatkar C, Wagle K. Review of SLAM algorithms for indoor mobile robot with LIDAR and RGB-D camera technology. In Book: Favorskaya MN, Mekhilef S, Pandey RK, Singh N, eds. Innovations in electrical and electronic engineering. Singapore: Springer; 2021: 397-409.

[3] Blokhinov YB, Andrienko EE, Kazakhmedov KK, Vishnyakov BV. Automatic calibration of multiple cameras and LIDARs for autonomous vehicles. Computer Optics 2021; 45(3): 382-393. DOI: 10.18287/2412-6179-CO-812.

[4] Michałowska M, Ogłoziński M. Autonomous vehicles and road safety. In Book: Mikulski J, ed. Smart solutions in today's transport. 17th Int Conf on Transport Systems Telematics (TST 2017), Katowice – Ustroń. Springer International Publishing AG; 2017: 191-202.

[5] Qi CR, Su H, Mo K, Guibas LJ. Pointnet: Deep learning on point sets for 3D classification and segmentation. IEEE Conf on Computer Vision and Pattern Recognition (CVPR2017), Honolulu 2017: 652-660.

[6] Tatebe Y, Deguchi D, Kawanishi Y, Ide I, Murase H, Sakai U. Pedestrian detection from sparse point-cloud using 3DCNN. International Workshop on Advanced Image Technology (IWAIT2018), Chiang Mai 2018: 1-4.

[7] Nagashima T, Nagasaki T, Matsubara H. Object classification integrating results of each scan line with low-resolution LIDAR. IEEJ Trans Electr Electron Eng 2019; 14(8): 1203-1208.

[8] Su H, Maji S, Kalogerakis E, Learned-Miller E. Multi-view convolutional neural networks for 3D shape recognition. IEEE Int Conf on Computer Vision (ICCV15), Santiago de Chile 2015: 945-953.

[9] Serna A, Marcotegui B. Detection, segmentation and classification of 3D urban objects using mathematical morphology and supervised learning. ISPRS J Photogramm Remote Sens 2014; 93: 243-255.

[10] Simony M, Milzy S, Amendey K, Gross HM. Complex-yolo: An euler-region-proposal for real-time 3D object detection on point clouds. European Conf on Computer Vision (ECCV2018), Munich 2018: 197-209.

[11] Xu F, Chen L, Lou J, Ren M. A real-time road detection method based on reorganized lidar data. PLoS One 2019; 14(4): e0215159.

[12] Wu B, Wan A, Yue X, Keutzer K. Squeezeseg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. IEEE Int Conf on Robotics and Automation (ICRA2018) 2018: 1887-1893.

[13] Maturana D, Scherer S. VoxNet: A 3D convolutional neural network for real-time object recognition. IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS2015), Hamburg 2015: 922-928.

[14] Aijazi AK, Checchin P, Trassoudaine L. Segmentation based classification of 3D urban point clouds: A super-

voxel based approach with evaluation. Remote Sens 2013; 5(4): 1624-1650.
[15] Zakani FR, Arhid K, Bouksim M, Aboulfatah M, Gadi T. Segmentation of 3D meshes combining the artificial neural network classifier and the spectral clustering. Computer Optics 2018; 42(2): 312-319. DOI: 10.18287/2412-6179-2018-42-2-312-319.
[16] Weinmann M, Jutzi B, Hinz S, Mallet C. Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. ISPRS J Photogramm Remote Sens 2015; 105: 286-304.
[17] Yang B, Dong Z. A shape-based segmentation method for mobile laser scanning point clouds. ISPRS J Photogramm Remote Sens 2013; 81: 19-30.
[18] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015; 521(7553): 436-44.
[19] Li Y, Bu R, Sun M, Wu W, Di X, Chen B. PointCNN: Convolution on x-transformed points. Adv Neural Inf Process Syst 2018; 31: 820-830.
[20] Xu Y, Fan T, Xu M, Zeng L, Qiao Y. SpiderCNN: Deep learning on point sets with parameterized convolutional filters. European Conf on Computer Vision (ECCV2018), Munich 2018: 87-102.
[21] Komarichev A, Zhong Z, Hua J. A-CNN: Annularly convolutional neural networks on point clouds. IEEE/CVF Conf on Computer Vision and Pattern Recognition (CVPR2019), Long Beach 2019: 7421-7430.
[22] Rao Y, Lu J, Zhou J. Spherical fractal convolutional neural networks for point cloud recognition. IEEE/CVF Conf on Computer Vision and Pattern Recognition (CVPR2019), Long Beach 2019: 452-460.
[23] Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM. Dynamic graph CNN for learning on point clouds. ACM Trans Graph 2019; 38(5): 146.
[24] Guo M, Cai J, Liu Z, Mu T, Martin RR, Hu S. PCT: Point cloud transformer. Comput Vis Media 2021; 7(2): 187-199.
[25] Qi CR, Yi L, Su H, Guibas LJ. PointNet++: Deep hierarchical feature learning on point sets in a metric space. Int Conf on Neural Information Processing Systems (31st NIPS), Long Beach 2017: 5105-5114.
[26] Rusu RB, Marton ZC, Blodow N, Dolha M, Beetz M. Towards 3D point cloud based object maps for household environments. Rob Auton Syst 2008; 56(11): 927-941.
[27] Lu Y, Xue Z, Xia GS, Zhang L. A survey on vision-based UAV navigation. Geo Spat Inf Sci 2018; 21(1): 21-32.
[28] Xiao C, Wachs J. Triangle-Net: Towards robustness in point cloud learning. IEEE/CVF Winter Conf on Applications of Computer Vision (WACV2021) 2021: 826-835.
[29] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. Int Conf on Neural Information Processing Systems (31st NIPS), Log Beach 2017: 6000-6010.
[30] Zhao H, Jia J, Koltun V. Exploring self-attention for image recognition. IEEE/CVF Conf on Computer Vision and Pattern Recognition (CVPR2020) 2020: 10076-10085.
[31] Liu X, Han Z, Liu YS, Zwicker M. Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. Conf on Artificial Intelligence (33th AAAI), Honolulu 2019: 8778-8785.
[32] Guo M-H, Cai J-X, Liu Z-N, Mu T-J, Martin RR, Hu S-M. PCT: Point cloud transformer. Comput Vis Media 2021; 7(2): 187-199.
[33] Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J. 3D shapenets: A deep representation for volumetric shapes. IEEE Conf on Computer Vision and Pattern Recognition (CVPR2015), Boston 2015: 1912-1920.
[34] Sun P, Kretzschmar H, Dotiwalla X, Chouard A, Patnaik V, Tsui P, Guo J, Zhou Y, Chai Y, Caine B, Vasudevan V. Scalability in perception for autonomous driving: Waymo open dataset. IEEE/CVF Conf on Computer Vision and Pattern Recognition (CVPR2020) 2020: 2446-2454.
[35] Kesten R, Usman M, Houston J, Pandya T, Nadhamuni K, Ferreira A, Yuan M, Low B, Jain A, Ondruska P, Omari S, Shah S, Kulkarni A, Kazakova A, Tao C, Platinsky L, Jiang W, Shet V. Lyft level 5 perception dataset 2020. 2021. Source: ⟨https://level5.lyft.com/dataset/⟩.
[36] Geiger A, Lenz P, Urtasun R. Are we ready for autonomous driving? The KITTI vision benchmark suite. IEEE Conf on Computer Vision and Pattern Recognition (25th CVPR), Providence 2012: 3354-3361.
[37] Fix E, Hodges JL. Discriminatory analysis. Nonparametric discrimination: Consistency properties. Int Stat Rev 1989; 57(3): 238-247.
[38] Ullah S, Kim DH. Benchmarking Jetson platform for 3D point-cloud and hyper-spectral image classification. IEEE Int Conf on Big Data and Smart Computing (BigComp2020), Busan 2020: 477-482.
[39] Lever J, Krzywinski M, Altman NS. Points of significance: Classification evaluation. Nature Methods 2016; 13(8): 603-604.

Authors' information

José Fernando Pamplona Zuluaga, (b. 1984) received his B.Sc. degree in Electronic Engineer (2017) and his M.Sc. degree in Automation and Industrial Control (2020) from Instituto Tecnológico Metropolitano de Medellin. He has worked as professor at Instituto Tecnológico Metropolitano de Medellin and Politécnico Colombiano Jaime Isaza Cadavid. Currently he works at GICEI research group in Institución Universitaria Pascual Bravo. Research interests are computer graphics processing, machine learning. E-mail: *jose.pamplona@pascualbravo.edu.co* .

Carlos Andrés Madrigal González, (b. 1981), graduated as Electronic Engineer from Universidad de Antioquia in 2005 and as a Doctor in Engineering from Universidad Nacional de Colombia sede Medellin in 2017. He was a professor and researcher in: Universisdad de Antioquia, Pontificia Universidad Católica de Chile, Pontificia Universidad Javeriana, Institución Universitaria Pascual Bravo, Uniciencia and Politécnico Colombiano Jaime Isaza Cadavid. Currently works in Instituto Tecnologico Metropolitano de Medellin and as data scientist in Grupodot S.A.S. Research interests are data science, artificial intelligence, electronics and communications, image processing and artificial vision. E-mail: *cmadrigal@grupodot.com* .

Jorge Alexis Herrera-Ramirez, (b. 1980) received his B.Sc. degree in Physics Engineering (2004) and his M.Sc. degree in Physics (2008) from the Universidad Nacional de Colombia, sede Medellín. He obtained his Ph.D. degree in Optical Engineering from Universitat Politecnica de Catalunya (Spain, 2014). During 2014, he was a Postdoctoral Fellow with the Centre for Sensors, Instruments and Systems Development of Universitat Politecnica de Catalunya. Since 2015, he became full-time Research-Professor at the Instituto Tecnologico Metropolitano de Medellin, Colombia. His current research interests involve digital holography and speckle, spectral imaging, and application of machine learning techniques on optical data. E-mail: *jorgeherrerar@itm.edu.co* .

Code of State Categories Scientific and Technical Information (in Russian – GRNTI)): 29.31.15, 29.33.43, 20.53.23.
Received June 28, 2021. The final version – November 22, 2021.