2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

FBPT: A Fully Binary Point Transformer

Zhixing Hou[1], Yuzhang Shang[2], Yan Yan[2†]

***Abstract*— This paper presents a novel Fully Binary Point Cloud Transformer (FBPT) model which has the potential to be widely applied and expanded in the fields of robotics and mobile devices. By compressing the weights and activations of a 32-bit full-precision network to 1-bit binary values, the proposed binary point cloud Transformer network significantly reduces the storage footprint and computational resource requirements of neural network models for point cloud processing tasks, compared to full-precision point cloud networks. However, achieving a fully binary point cloud Transformer network, where all parts except the modules specific to the task are binary, poses challenges and bottlenecks in quantizing the activations of Q, K, V and self-attention in the attention module, as they do not adhere to simple probability distributions and can vary with input data. Furthermore, in our network, the binary attention module undergoes a degradation of the self-attention module due to the uniform distribution that occurs after the softmax operation. The primary focus of this paper is on addressing the performance degradation issue caused by the use of binary point cloud Transformer modules. We propose a novel binarization mechanism called dynamic-static hybridization. Specifically, our approach combines static binarization of the overall network model with fine granularity dynamic binarization of data-sensitive components. Furthermore, we make use of a novel hierarchical training scheme to obtain the optimal model and binarization parameters. These above improvements allow the proposed binarization method to outperform binarization methods applied to convolution neural networks when used in point cloud Transformer structures. To demonstrate the superiority of our algorithm, we conducted experiments on two different tasks: point cloud classification and place recognition. In point cloud classification, our model achieved an accuracy of 90.9%, which is only a 2.3% decrease compared to the full precision network. For the place recognition task, we achieved 91.02% in the top @1% metric and 82.87% in the top @1% metric on the Oxford RobotCar dataset in terms of the average recall rate. Moreover, our model exhibits a significant reduction of over 80% in terms of model size and FLOPs (floating-point operations) compared to the baseline.**

***Index Terms*— Binary Transformer, Point Cloud, Dynamic-Static Hybridization, Hierarchical Training Scheme**

I. Introduction

In recent years, there have been significant advancements in the research fields of robot perception and artificial intelligence, and one of the key contributing factors is the discovery of the transformer [1] structure. The transformer model has greatly enhanced the performance of various tasks such as natural language processing [2], [3], image recognition [4], [5], and point cloud processing [6], [7], particularly in large-scale pre-trained tasks. However, a major limitation of applying these transformer models in real-world scenarios is their high memory and computational requirements, which hinders their widespread deployment in mobile or wearable device contexts.

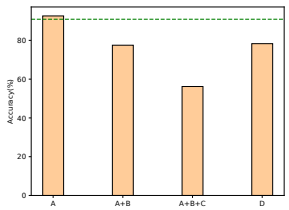


Fig. 1: degradation analysis (without local feature skip concatenation). A: binary local feature, B: binary weights in transformer module, C:binary activations in transformer module, and D: binary transformer module. Green dashed lines indicate the results of the full-precision PCT network.

To overcome these limitations, Transformer models used in various tasks need to be compressed while still maintaining good performance. Previous studies have proposed various Transformer compression methods, including pruning [8]–[10], knowledge distillation [11]–[13], low-rank approximation [14], [15], and adaptive dynamic networks [16], [17]. While these methods reduce the network size and increase efficiency, this paper focuses on quantization [18]–[24] as an attractive and alternative method that has achieved great success in practice. Among the quantization methods, extreme quantization [25]–[27] pushes the limits of transformer compression by converting the original 32-bit full-precision values into 1-bit binarized values. In binary quantization methods, the model size is substantially reduced, and the calculation speed is also accelerated using binary bitwise operations [28] such as $xnor$ and $bitcount$. This provides a significant advantage over non-binary quantization methods.

In addition to natural language and image processing, the analysis and processing of 3D point clouds are also crucial in the field of artificial intelligence research. Unlike natural language and images, 3D point clouds possess unique characteristics such as irregular formats, uneven density, and coordinate system dependency. As a result, the point cloud transformer model [6], [7], [29] differs from the classic transformer models used in NLP and computer vision domains. This distinction will be described in II-D. Given these differences, the binarization of point cloud transformers

† Corresponding author.
[1] Zhixing Hou is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. E-mail: hzx@njust.edu.cn
[2] Yuzhang Shang and Yan Yan are with the Department of Computer Science, Illinois Institute of Technology, Chicago, USA. E-mail: yshang4@hawk.iit.edu, yyan34@iit.edu (*Corresponding author)

necessitates a redesigned solution.

To our knowledge, no **fully binarized point cloud transformer** has been proposed thus far. As the **first work** to implement this research, we conducted binarization on different parts of the network to identify the modules most sensitive to binarization demonstrated in Fig. 1. re-organize: In order to avoid significant performance degradation, we propose a novel binarization mechanism that applies various granularities of binarization to different modules in a dynamic manner. Finally, based on the full-precision point cloud transformer network [7], we develop a fully binarized network replica that includes both weight and activation binarization.

However, in the fully binarized point cloud transformer network, the self-attention mechanism degenerates to uniformly distributed after softmax, which leads to a loss of weighted attention towards the $Value$ in Transformer. To prevent this degradation, we design a novel hierarchical training scheme to gradually determine model parameters and binarization parameters in three stages.

Moreover, we find that in the full-precision point cloud transformer network, vanilla self-attention has limited impact on network accuracy. However, after incorporating binarized self-attention, significant improvements in accuracy are observed, along with a certain ability to suppress overfitting.

We impose the proposed binary point cloud transformer model to the point cloud classification task and evaluate its performance on the ModelNet40 dataset [30]. The results show that our model achieve an overall accuracy of 90.9%, with only a 2.3% performance drop compared to the full-precision counterpart, while reducing the model size by 87.2% and the FLOPs (floating-point operations) by 80.2%. To further validate the superiority of our proposed binary point cloud transformer, we integrate it with metric learning and impose it on the place recognition task. Our model achieves an accuracy of 91.02% in the *top1%* metric and 82.87% in the *top1* metric on the Oxford RobotCar dataset [31].

In summary, the contributions and novelties of this paper can be summarized as follows:

- We introduce a novel, Fully Binary Point Cloud Transformer model called FBPT.
- Through experimental analysis, we identify performance bottlenecks in binarized point cloud Transformers and propose novel techniques to address these bottlenecks, including fine-grained binarization, dynamic-static hybridization binarization, and a hierarchical training scheme.
- We demonstrate the efficacy of our proposed FBPT model through experiments on point cloud classification and place recognition tasks. Our results reveal that the FBPT model exhibits excellent performance in terms of accuracy, model size, computational complexity, and generalization capabilities.

II. Related works

A. Model Binarization

Courbariaux et al. [32] first proposed the concept of the binary neural network, through two kinds of state quantities +1 and -1 to fit full-precision model parameters and activation values, achieved the effect of compressing the model and improving the operation efficiency. Subsequently, Rastegari et al. [28] introduced XNOR-Net, and demonstrated the effectiveness of the model through rigorous theoretical analysis and experiments on large classification datasets such as ImageNet1K. This approach can theoretically reduce the model size by $1/32$ and speed up inference by 58 times.

In the research on Transformer binarization, first, Bai et al. [25] proposed a transformer binarization work named BinaryBERT by equivalently splitting from a half-size ternary BERT network [19] that pushes the transformer quantization to the limit. Qin et al. [26] proposed BiBERT which introduces the Bi-Attention module and Direction-Matching distillation method from the perspective of information theory by maximizing the information entropy of the binarized representations. Liu et al. [27] also implemented a binarized BERT model called BiT that includes a two-set binarization scheme that utilizes different binarization methods to compact activations with and without non-linear activation layers. Furthermore, they made use of the multi-step distillation method, which first distills the full-precision model into an intermediate model and then distills the intermediate model into the binarized transformer model.

As for vision transformer research fields, Li et al. [21] compacted parameters and activations of the vision transformer model to 3 bit and Li et al. [22] further pushed the limit of vision transformer quantization to 2-bit. However, there has been no binarized vision transformer, i.e. 1-bit weights and activation representations proposed so far.

In the same situation as the vision transformer, few binarization works are applied to point cloud processing tasks. The first binary neural network for raw point cloud processing proposed by Qin et al. [33] introduced Entropy-Maximizing Aggregation and Layer-wise Scale Recovery to effectively prevent immense performance drop. Xu et al. [34] introduced a bi-modal distribution concept to mitigate the impact of small disturbances and exploited Expectation-Maximization to constrain the weights of the network into this distribution. Su et al. [35] designed a binarization and training method for the point cloud network with invariant scales and equivariant vectors simultaneously. But these three works are all based on the MLP structure originated from the classic work PointNet [36] without the participation of the transformer structure. So our work proposed in this paper is the first binary point cloud transformer model and it is also the first time to be applied to place recognition.

B. Dynamic and Static Quantization

In the research field of model quantization, post-training quantization refers to the process of quantizing the floating-point weights and activations in a trained model into fixed-point integers with a specified number of bits. This is

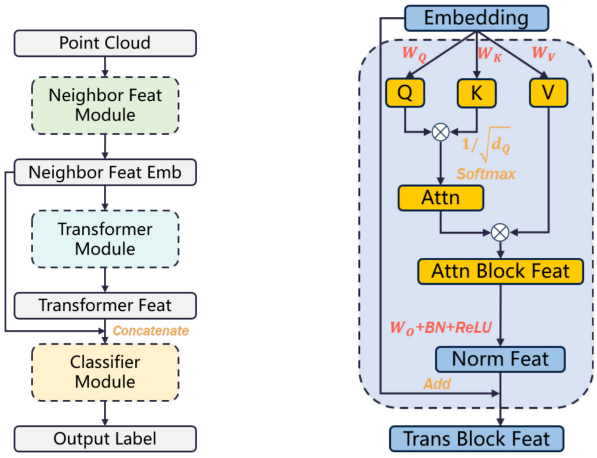


Fig. 2: (a) Point cloud transformer model used for classification. (b) Full-precision point cloud transformer block, which is the major structure in Transformer module.

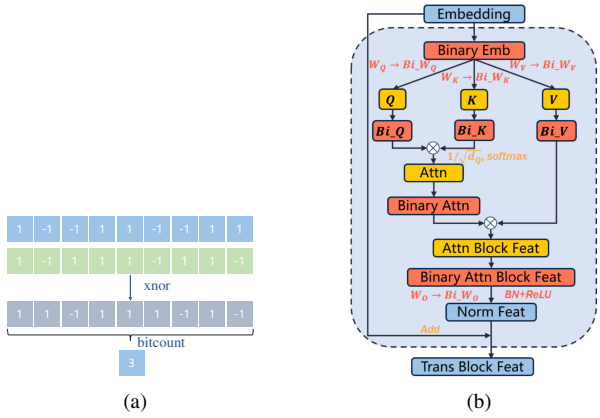


Fig. 3: (a) Bitwise binary operations: $xnor$ and $bitcount$. (b) Binary point cloud transformer block. The symbol $\otimes$ in binary transformer blocks indicates binary operation including $xnor$ and $bitcount$, equivalent to dot product operation in full-precision transformer blocks.

achieved by designing quantizers and utilizing a small amount of calibration data to compute quantization parameters. These parameters are then used to convert the model's weights and activations into their corresponding fixed-point representations. Depending on the different methods used to calculate quantization parameters for activations, the quantization process can be categorized into dynamic quantization and static quantization. In static quantization, quantization parameters of activations are determined offline using the activations of calibration samples. This approach allows for precise control over the quantization process based on a predefined set of activation values. On the other hand, dynamic quantization utilizes runtime statistics of activations to calculate quantization parameters. This method adapts the quantization parameters during runtime, taking into account the actual distribution of activations encountered during inference.

C. Quantization Granularity

Quantization can be done at different granularity levels. Per-tensor quantization applies a single step size to the entire matrix. For finer-grained quantization, per-token quantization assigns different step sizes to activations associated with each token, while per-channel quantization assigns different step sizes to each output channel of weights. Another option is group-wise quantization, which uses different step sizes for different channel groups.

D. Full-precision Point Cloud Transformer

There are several different types of transformer-related works to deal with point cloud tasks. Among them, the most typical works are Point Transformer [6], Point Transformer V2 [29], and PCT (Point Cloud Transformer) [7]. Due to its clear, concise, and effective structure shown in Fig. 2, the PCT model [7] is chosen as the full-precision model counterpart of our binary model. We divide the whole pipeline into three main components and name them the neighbor feature extraction module, the transformer module, and the classifier module.

III. Fully Binary Point Cloud Transformer

A. Revisit Binarization for Linear Layers

Linear transformation is a fundamental operation in transformer networks. In this section, we will focus on introducing binarization techniques that are specifically designed for linear layers. The definition of linear layers and their binarization in neural networks is as follows:

$$\begin{aligned}\mathbf{Y} &= \mathbf{W_r} * \mathbf{A_r} \\ &= \alpha\mathcal{B}(\mathbf{W_r^{'}}) * \beta\mathcal{B}(\mathbf{A_r^{'}}) \\ &= \alpha\beta(\mathcal{B}(\mathbf{W_r^{'}}) \otimes (\mathcal{B}(\mathbf{A_r^{'}}))) \\ &= \alpha\beta\mathbf{W_b} \otimes \mathbf{A_b},\end{aligned} \tag{1}$$

where the output of the linear layer is denoted as the matrix $\mathbf{Y}$, obtained by multiplying the weight matrix $\mathbf{W_r}$ with the input activation matrix $\mathbf{A_r}$. Here, $\mathbf{W_r}$ represents the weight of the linear layer, and $\mathbf{A_r}$ refers to the activation of the linear layer. The subscript $\mathbf{r}$ indicates that these matrices have real-valued floating-point precision. Before binarization, $\mathbf{W_r}$ and $\mathbf{A_r}$ are transformed into $\mathbf{W_r^{'}}$ and $\mathbf{A_r^{'}}$ respectively through a shift and scale operation. The binarization function $\mathcal{B}(\cdot)$ is applied to $\mathbf{W_r^{'}}$ and $\mathbf{A_r^{'}}$ based on the specific format of the input. Once the weights and activations are binarized, the original matrix multiplication (denoted as $*$) is replaced by binary bitwise operations (represented by $\otimes$). These operations include the $xnor$ operation and the $bitcount$ operation, as shown in Fig. 3(a).

The scalar factors α and β can be extracted from matrix multiplication and precomputed as a global coefficient for the overall result of binarized matrix operations, following the XNOR-Net [28]. The calculation of these scalar factors

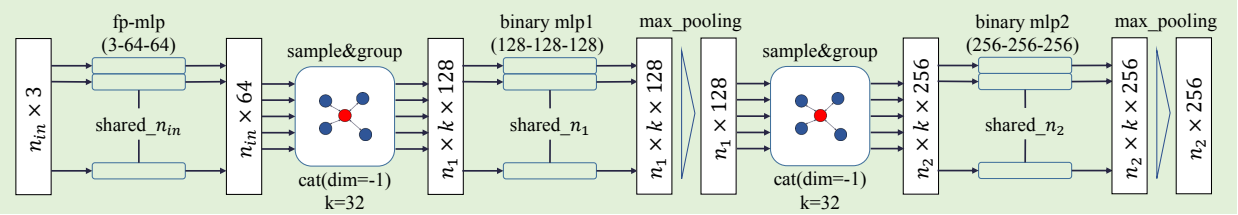


Fig. 4: The local feature module. The input point will be preserved as full precision and transformed to point-wise features with full precision MLP, while the point embedding to be fed into the transformer module generated with binary MLP following sampling and grouping.

is described as follows:

$$\begin{aligned} \alpha\beta &= \frac{\sum |W_i||A_i|}{n} \\ &\approx (\frac{1}{n}||W||_{l1})(\frac{1}{n}||A||_{l1}), \end{aligned} \tag{2}$$

where the subscript $l1$ denotes the $l1$-$norm$. To better fit real values with binarized values, various binarization schemes can be employed based on the format of activations, as described in the BiT paper [27], depicted in Eq. 3 and 4. For layers where activations are distributed from negative to positive values, the $sign$ function is used for binarization. On the other hand, for layers with activations confined to the range of positive values only (e.g., after $softmax$ function), the $round$ function is used. In such cases, the real values are binarized into the set $\{0, 1\}$:

$$\mathbf{A_b} = \begin{cases} 1 & \text{for} \quad x \geq 0 \\ -1 & \text{for} \quad otherwise \end{cases} \tag{3}$$

and for the layers only distributed on the range of positive values:

$$\mathbf{A_b} = \begin{cases} 1 & \text{for} \quad x \geq 0.5 \\ 0 & \text{for} \quad otherwise \end{cases} \tag{4}$$

B. Local Feature Module and Binarization

The local feature extraction module, illustrated in Fig. 4, is comprised of a series of blocks that are interconnected. Each block consists of several components, including shared multi-layer perceptron (MLP) networks, sampling and grouping operations, and pooling layers. The fundamental idea of this local features method can be attributed to the DGCNN [37].

Following conventional practices, certain elements in the module are retained as full-precision representations. Specifically, the input point cloud data and the MLP in the initial layer remain in their original form, without any quantization. However, for the remaining components within the module, such as activations and weights, a binarization process is applied resulting in a binary MLP representation. By employing this approach, we can effectively extract local features from the data while achieving a fully binarized module.

C. Bottleneck of FBPT

We independently binarized the corresponding parts in the network shown in Fig 1. we discovered that binarizing the transformer module leads to a significant decline in model performance. We can summarize the reasons for this performance drop into three aspects.

- Firstly, the self-attention part in the transformer essentially weights the values, and larger weights have a greater impact on the results. However, after passing through the softmax function, most of the values in the self-attention tend to cluster around zero, with fewer significant outliers. Conventional binarization methods would eliminate the influence of these outliers on the model.
- Secondly, the linear operations in the transformer not only involve computations between weights and activations but also computations between different activations, such as matrix multiplication between $Query$ and Key. Additionally, the distribution of activation values in the transformer varies significantly with many outliers, which change with the input. Therefore, fitting fixed binarization parameters to accurately represent the full precision activations is challenging.
- Finally, in the proposed fully binarized point cloud transformer network, the self-attention degenerates to uniformly distributed after the $softmax$ function, which leads to a loss of weighted attention towards the $Value$ in Transformer.

To address these issues, we proposed the following three improvements: fine-grained binarization (Section III-D), dynamic-static hybridization binarization (Section III-E), and hierarchical training scheme (Section III-F).

D. Fine-grained Binarization for Self-attention

We have increased the granularity of quantization for activations in the self-attention block. We believe that simply binarizing these activations, which are highly sensitive to data and exhibit complex distributions (such as query, key, value, and attention), would severely impact model performance. However, to effectively leverage the advantages of binary operations, we apply a higher granularity of binarization to these activations. Specifically, we first determine the range of activation values and then select appropriate partition points uniformly or non-uniformly across the entire range. This forms multiple groups of masks on the activation tensor, where each group shares the same set of binarization parameters. By using finer-grained binarization, the quantization bit-width of activations across the entire tensor is expanded from 1 bit to 2 bits or even 3 bits, while internally consisting of individual binary values, thereby ensuring smooth execution of binary operations.

E. Dynamic-static Hybridization Binarization

The proposed binarization method is inspired by the concepts of dynamic and static post-training quantization, which is introduce in II-B. We define static binarization as the utilization of binary activations during the training process, with optimal binarization parameters being learned through back propagation. In contrast, dynamic binarization entails retaining real-value activations during training while dynamically computing the parameters required for activation binarization during inference.

Specifically, for activations that are highly sensitive to inputs, such as $Query$, Key, and $Value$, we no longer used pre-trained binarization parameters. Instead, we calculated the binarization parameters in real-time during algorithm execution. This approach allows the binarization parameters to be dynamically adjusted according to changes in the input. Although calculating binarization parameters dynamically increases computational overhead, the proportion of activations requiring dynamic computation is relatively small. Binarized activations can be operated with binarized weights, thereby improving the execution speed of the algorithm.

F. Hierarchical Training Scheme

Training the FBPT model directly, similar to other network models, may result in self-attention being suppressed and becoming excessively small due to the presence of residual connections. Consequently, after passing through the softmax function, the self-attention weights degrade into a uniform distribution, rendering the weighting capacity of self-attention ineffective. To address this issue, a hierarchical training scheme is proposed which divides the training into three stages: non-transformer modules including local feature extraction module and parts of linear layer in the Transformer module, weights of the transformer module, and activations of the transformer module. Since self-attention is a crucial structure in the transformer, we perform static binarization on non-transformer modules in the first stage while freezing network parameters and binarization parameters. In the second stage, we apply static binarization to the weights of the transformer module. Subsequently, in the third stage, we employ dynamic binarization for the activations of the transformer module. The first two stages are completed during the training process, while the third stage involves real-time computation during inference. Due to the effect of these three stages on the determination of binarization parameters, we collectively refer to them as the hierarchical training scheme.

IV. Experiments

A. Experimental Settings

Classification dataset: ModelNet40 [30] dataset. This dataset is the most popular benchmark for point cloud classification. A total of 12,311 CAD models from the 40 categories are split into 9,843 for training and 2,468 for testing.

Place recognition datasets: We use the benchmark datasets created in [38] to train the FBPT model for the place recognition task. This frequently-used benchmark contains a modified Oxford RobotCar dataset [31] for training and testing, as well as three smaller in-house datasets: University Sector (U.S.), Residential Area (R.A.), and Business District (B.D.) only for testing. The dataset settings are also the same as the ones in [38].

TABLE I: Comparison results for point cloud classification.

Methods	Transformer	W-A	OA(%)	mAcc(%)
PointNet [36]	✗	32-32	89.2	86.0
PointNet++ [39]	✗	32-32	90.7	-
DGCNN [37]	✗	32-32	92.9	90.0
PCT [7]	✔	32-32	**93.2**	**90.2**
BiPointNet [33]	✗	1-1	86.4	-
FBPT(Ours)	✔	1-1	**90.9**	**87.8**

Evaluation metrics: The same as the previous works, overall accuracy (OA) and mean accuracy (mAcc) are used to evaluate the classification performance of the proposed model. While the top 1% (@1%) and top 1 (@1) average recall rate as the evaluation metrics are adopted to evaluate the place recognition performance of the algorithm.

Classification implementation details: The proposed FBPT model is trained from scratch without leveraging any pre-trained model. We use the stochastic gradient descent (SGD) optimizer and an initial learning rate of 0.01 with 0.9 momentum to train our network $400 + 100$ epochs (two stages) iteratively on the Pytorch platform. A cosine annealing schedule is adopted to adjust the learning rate. The mini-batch size is set to 32. All the evaluations are conducted on an NVIDIA RTX2080Ti GPU card.

Place recognition implementation details: The network structure for the place recognition follows the structure used for classification except for the classifier header. In our experiment, a simple max-pooling layer for aggregating a globally consistent descriptor after the last linear layers for class labels generation are removed. With respect to the loss function, the metric learning loss is required for the place recognition task. Specifically, the loss function we used is a role for minimizing the relative distance between descriptors extracted from point clouds sampled at the revisited place (defined as positive samples) and maximizing the relative distance between descriptors extracted from point clouds sampled at different places (defined as negative samples) in the metric space. The lazy quadruplet loss proposed in [38] is used in our work.

The dimension of the global descriptor produced by the last pooling layer is set to 256. We feed the final descriptors into the lazy quadruplet loss function and set the hyper-parameters γ and θ in the loss function to 0.5 and 0.2, respectively. We wrap 1 anchor sample, 2 positive samples, and 8 negative samples together as a mini-batch and synchronously input them into an NVIDIA A6000 GPU card. The optimizer Adam [46] with an initial learning rate of 5×10^{-5} is used to train our network $16+4$ epochs (two stages) iteratively on the Pytorch platform. The learning rate decays to 1×10^{-5} eventually.

TABLE II: Comparison results on the average recall at top 1% and top 1 of different baseline networks trained on Ox. and tested on Ox., U.S., R.A. and B.D., respectively.

Methods	Transformer	Binary Operation	Ave recall @ 1%				Ave recall @ 1			
			Ox.	U.S.	R.A.	B.D.	Ox.	U.S.	R.A.	B.D.
PN_VLAD [38]	✗	✗	80.33	72.63	60.27	65.30	62.76	65.96	55.31	58.87
PCAN [40]	✗	✗	83.81	79.05	71.18	66.82	69.05	62.50	57.00	58.14
DH3D-4096 [41]	✗	✗	84.26	-	-	-	73.28	-	-	-
DAGC [42]	✗	✗	87.49	83.49	75.68	71.21	73.34	-	-	-
LPD-Net* [43]	✗	✗	91.61	86.02	78.85	75.36	82.41	77.25	65.66	69.51
HiTPR [44]	✔	✗	93.71	90.21	87.16	79.79	86.63	80.86	78.16	74.26
EPC-Net [45]	✗	✗	94.74	96.52	88.58	84.92	86.23	-	-	-
PCT [7]	✔	✗	93.26	89.72	87.11	78.81	85.98	80.43	77.56	74.01
FBPT (ours)	✔	✔	91.02	86.33	82.76	76.42	82.87	78.31	73.01	71.95

B. Quantitative Results

Classification results: Experimental results on point cloud classification are shown in Tab. I. The bit number of the weights and activations is marked in the W-A column. In this table, PCT means full-precision point cloud transformer which is the counterpart of our binarized model. We compare our binary point cloud transformer with three kinds of methods, the first one is the full-precision method without transformer mechanism, such as PointNet [36], DGCNN [37], *etc.*, the second one is the full-precision transformer method (PCT [7]), and the last is the binary model (BiPointNet [33]) binarized from PointNet [36]. This experiment demonstrates the proposed FBPT model achieves only a 2.3% performance drop compared to the full-precision PCT model and even outperforms some full-precision models.

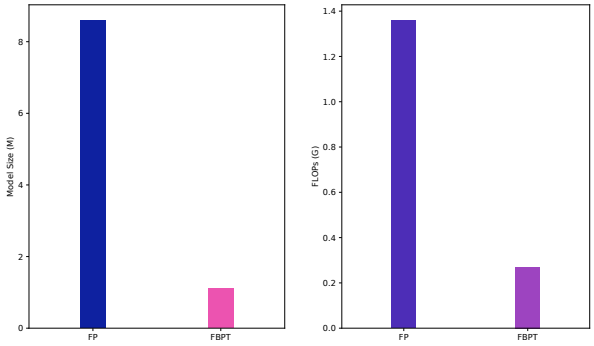
Place recognition results: The results of the compared methods are shown in Tab. II. It demonstrates our proposed FBPT model can achieve competitive results with the existing full-precision models with smaller memory space and lighter calculation load. It should be noted that the LPD-Net* indicates the handcrafted features are removed from the original LPD-Net [43] for a fair comparison. We remove the handcrafted features based on the fact that most previous works in LiDAR-based place recognition do not concatenate additional handcrafted features as their network input.

C. Computational Cost Analysis

We analyze the consumption of computing resources from two aspects: the model size and FLOPs (floating point operations). After binarization, the size of the same model structure reduces 87.2% from 8.6M to 1.1M, while the FLOPs reduce 80.2% from 1.36G to 0.27G shown in Fig. 5. Compared with the full-precision point cloud transformer, the proposed binary one is more suitable to be deployed on real-world robots or mobile devices.

D. Contribution of FBPT to Generalization

In the experimental analysis, we observed that the full-precision vanilla transformer had a limited contribution to the model. Therefore, we investigated the impact of removing the binarized transformer module (referred to as $FBPT\_B$) from the FBPT model to assess its significance (referred to as $FBPT\_WO$). We discovered that the binarized transformer module led to improvements in result accuracy shown in Table III. Moreover, more importantly, results on three small non-training datasets indicated that $FBPT\_B$ was able to mitigate overfitting and enhance the model's generalization capability



(a) model size (87.2%) (b) FLOPs (80.2%)

Fig. 5: Calculation resource consumption comparison.

TABLE III: Contribution to generalization.

Methods	Ave recall @ 1%			
	Ox.	U.S.	R.A.	B.D.
FBPT_WO	89.84	82.16	76.77	70.41
FBPT_B	91.02	86.33	82.76	76.42

V. Conclusion

This paper presents a novel Fully Binary Point Cloud Transformer model. To address the performance degradation issue caused by the use of binary point cloud Transformer modules. We propose fine-grained binarization, dynamic-static hybridization, and hierarchical training scheme. With these improvements, we obtained the excellent experiment results on point classification and place recognition tasks. Although there are currently limited devices that can support binary operations efficiently, research in this direction still holds scientific value and provides valuable guidance. Furthermore, the quantization experimental results further demonstrate the great potential and advantages of our model in real-world point cloud tasks, particularly on resource-constrained devices.

REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[2] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019, pp. 4171–4186.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *9th International Conference on Learning Representations*, 2021.

[5] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[6] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 259–16 268.

[7] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "Pct: Point cloud transformer," *Computational Visual Media*, vol. 7, no. 2, pp. 187–199, 2021.

[8] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," in *International Conference on Learning Representations*, 2019.

[9] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?" *Advances in neural information processing systems*, vol. 32, 2019.

[10] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5797–5808.

[11] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling bert for natural language understanding," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 4163–4174.

[12] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for bert model compression," in *Conference on Empirical Methods in Natural Language Processing*, 2019.

[13] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[14] X. Ma, P. Zhang, S. Zhang, N. Duan, Y. Hou, M. Zhou, and D. Song, "A tensorized transformer for language modeling," *Advances in neural information processing systems*, vol. 32, 2019.

[15] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2019.

[16] L. Hou, Z. Huang, L. Shang, X. Jiang, X. Chen, and Q. Liu, "Dynabert: Dynamic bert with adaptive width and depth," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9782–9793, 2020.

[17] W. Liu, P. Zhou, Z. Wang, Z. Zhao, H. Deng, and Q. Ju, "Fastbert: a self-distilling bert with adaptive inference time," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 6035–6044.

[18] O. Zafrir, G. Boudoukh, P. Izsak, and M. Wasserblat, "Q8bert: Quantized 8bit bert," in *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*. IEEE, 2019, pp. 36–39.

[19] W. Zhang, L. Hou, Y. Yin, L. Shang, X. Chen, X. Jiang, and Q. Liu, "Ternarybert: Distillation-aware ultra-low bit bert," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 509–521.

[20] S. Shen, Z. Dong, J. Ye, L. Ma, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer, "Q-bert: Hessian based ultra low precision quantization of bert," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 8815–8821.

[21] Z. Li, T. Yang, P. Wang, and J. Cheng, "Q-vit: Fully differentiable quantization for vision transformer," *arXiv preprint arXiv:2201.07703*, 2022.

[22] Y. Li, S. Xu, B. Zhang, X. Cao, P. Gao, and G. Guo, "Q-vit: Accurate and fully quantized low-bit vision transformer," *Advances in Neural Information Processing Systems*, vol. 35, pp. 34 451–34 463, 2022.

[23] Z. Yuan, L. Niu, J. Liu, W. Liu, X. Wang, Y. Shang, G. Sun, Q. Wu, J. Wu, and B. Wu, "Rptq: Reorder-based post-training quantization for large language models," *arXiv preprint arXiv:2304.01089*, 2023.

[24] Y. Shang, Z. Yuan, B. Xie, B. Wu, and Y. Yan, "Post-training quantization on diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1972–1981.

[25] H. Bai, W. Zhang, L. Hou, L. Shang, J. Jin, X. Jiang, Q. Liu, M. Lyu, and I. King, "Binarybert: Pushing the limit of bert quantization," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, pp. 4334–4348.

[26] H. Qin, Y. Ding, M. Zhang, Y. Qinghua, A. Liu, Q. Dang, Z. Liu, and X. Liu, "Bibert: Accurate fully binarized bert," in *International Conference on Learning Representations*, 2021.

[27] Z. Liu, B. Oguz, A. Pappu, L. Xiao, S. Yih, M. Li, R. Krishnamoorthi, and Y. Mehdad, "Bit: Robustly binarized multi-distilled transformer," *Advances in neural information processing systems*, vol. 35, pp. 14 303–14 316, 2022.

[28] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European conference on computer vision*, 2016, pp. 525–542.

[29] X. Wu, Y. Lao, L. Jiang, X. Liu, and H. Zhao, "Point transformer v2: Grouped vector attention and partition-based pooling," *Advances in Neural Information Processing Systems*, vol. 35, pp. 33 330–33 342, 2022.

[30] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[31] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[32] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.

[33] H. Qin, Z. Cai, M. Zhang, Y. Ding, H. Zhao, S. Yi, X. Liu, and H. Su, "Bipointnet: Binary neural network for point clouds," in *International Conference on Learning Representations*, 2020.

[34] S. Xu, Y. Li, J. Zhao, B. Zhang, and G. Guo, "Poem: 1-bit point-wise operations based on expectation-maximization for efficient point cloud processing," *arXiv preprint arXiv:2111.13386*, 2021.

[35] Z. Su, M. Welling, M. Pietikäinen, and L. Liu, "Svnet: Where so (3) equivariance meets binarization on point cloud representation," in *2022 International Conference on 3D Vision*, 2022, pp. 547–556.

[36] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[37] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[38] M. A. Uy and G. H. Lee, "Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4470–4479.

[39] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, 2017.

[40] W. Zhang and C. Xiao, "Pcan: 3d attention map learning using contextual information for point cloud based retrieval," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 436–12 445.

[41] J. Du, R. Wang, and D. Cremers, "Dh3d: Deep hierarchical 3d descriptors for robust large-scale 6dof relocalization," in *European Conference on Computer Vision*, 2020.
[42] Q. Sun, H. Liu, J. He, Z. Fan, and X. Du, "Dagc: Employing dual attention and graph convolution for point cloud based place recognition," in *Proceedings of the 2020 International Conference on Multimedia Retrieval*, 2020, pp. 224–232.
[43] Z. Liu, S. Zhou, C. Suo, P. Yin, W. Chen, H. Wang, H. Li, and Y.-H. Liu, "Lpd-net: 3d point cloud learning for large-scale place recognition and environment analysis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2831–2840.
[44] Z. Hou, Y. Yan, C. Xu, and H. Kong, "Hitpr: Hierarchical transformer for place recognition in point cloud," in *2022 International Conference on Robotics and Automation*, 2022, pp. 2612–2618.
[45] L. Hui, M. Cheng, J. Xie, J. Yang, and M.-M. Cheng, "Efficient 3d point cloud feature learning for large-scale place recognition," *IEEE Transactions on Image Processing*, vol. 31, pp. 1258–1270, 2022.
[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.