

Binarizing Sparse Convolutional Networks for Efficient Point Cloud Analysis

Xiuwei Xu^{1,2}, Ziwei Wang^{1,2}, Jie Zhou^{1,2}, Jiwen Lu^{1,2*}

¹Department of Automation, Tsinghua University, China

²Beijing National Research Center for Information Science and Technology, China

fxxw21, wang-zw18 g@mails.tsinghua.edu.cn;

fjzhou, lujiwen

g@tsinghua.edu.cn

Abstract

In this paper, we propose binary sparse convolutional networks called BSC-Net for efficient point cloud analysis. We empirically observe that sparse convolution operation causes larger quantization errors than standard convolution. However, conventional network quantization methods directly binarize the weights and activations in sparse convolution, resulting in performance drop due to the significant quantization loss. On the contrary, we search the optimal subset of convolution operation that activates the sparse convolution at various locations for quantization error alleviation, and the performance gap between real-valued and binary sparse convolutional networks is closed without complexity overhead. Specifically, we first present the shifted sparse convolution that fuses the information in the receptive field for the active sites that match the predefined positions. Then we employ the differentiable search strategies to discover the optimal positions for active site matching in the shifted sparse convolution, and the quantization errors are significantly alleviated for efficient point cloud analysis. For fair evaluation of the proposed method, we empirically select the recent advances that are beneficial for sparse convolution network binarization to construct a strong baseline. The experimental results on ScanNet and NYU Depth v2 show that our BSC-Net achieves significant improvement upon our strong baseline and outperforms the state-of-the-art network binarization methods by a remarkable margin without additional computation overhead for binarizing sparse convolutional networks.

1. Introduction

3D deep learning on point clouds [6, 12, 25, 27] has been widely adopted in a wide variety of downstream applications including autonomous driving, AR/VR and robotics due to its strong discriminative power and generalization ability. In these applications, real-time interaction and fast

Figure 1. Demonstration of sparse convolution and the proposed shifted sparse convolution. (a) Sparse convolution only operates when the center of kernel slides over the active sites. (b) Our shifted sparse convolution performs different operations for each group of output channels, which brings more information from the neighbor active sites.

response are required to guarantee safety and practicality.

Submanifold sparse convolution (we call it "sparse convolution" for short in the rest of this paper) [12] is one of the most popular and basic operator for point cloud analysis, which first voxelizes the point clouds and then applies 3D convolution on the voxels while keeping the same sparsity pattern throughout the layers of the network. Sparse convolution is widely adopted in most state-of-the-art architectures for point cloud analysis and so it is desirable to further improve its efficiency for more practical application. We opt for architecture-agnostic methods such as employing network binarization to achieve this goal. Binarized neural networks [19, 36] restrict the bitwidth of weights and activations to only one bit and substitute the multiplication-addition by xnor-bitcount operations, which decreases the

* Corresponding author.

storage and computational cost 32 and 64 respectively. We empirically find sparse convolution operation brings larger quantization errors compared to standard convolution, which leads to significant performance degradation when directly applying existing network binarization methods due to the large quantization errors.

In this paper, we present BSC-Net to learn binary sparsification. In efficient point cloud analysis in resource-exhaustive scenarios. Instead of directly binarizing the weights and activations in sparse convolutional networks, we search the optimal subset of convolution operations that activates the sparse convolution at various locations for binarization. The acquired convolution patterns significantly reduces the quantization errors in deployment, and achieves remarkable performance enhancement without extra computational cost. More specifically, we propose the shifted sparse convolutional networks whose convolution operations are activated for active sites consistent with the pre-defined locations, and the optimal positions for active site matching across various channels are obtained via differentiable search strategies. Therefore, the quantization errors in the fixed convolution operations are significantly alleviated by leveraging the shifted sparse convolution with the searched active site matching locations. Moreover, we empirically select the recent advances that are beneficial for sparse convolution network binarization to construct a strong baseline. Extensive experimental results on ScanNet and NYU Depth v2 for semantic segmentation of point clouds show that our BSC-Net reduces the operations per second (OPs) by 92.4% with only 3% mIOU degradation.

2. Related Work

Network quantization: Network quantization has been widely studied in computer vision due to the significant enhancement in storage and computation efficiency. Conventional methods can be divided into two categories including networks in one bit and multiple bits. For the first regard, weights and activations in networks are binarized with extremely high compression ratio. Hubara et al. [15] and Courbariaux et al. [8] substituted the add-multiplication (MAC) of real-valued models by xnor-bitcount operations, and employed the straight-through estimators (STE) to update the network parameters with back-propagation. Rastegari et al. [29] further presented scaling factors for weights and activations quantization error minimization. To recover the capacity degradation caused by aggressive quantization, Liu et al. [20] added extra shortcut connections between consecutive layers to diversify the feature maps. Benati et al. [1] modified the search space and strategy with stability regularization for the optimal architecture acquisition of binary networks. Qinet al. [28] maximized the information entropy in binary features and leveraged learnable scaling factors for information retention in point cloud analysis.

Since increasing the weight and activation bitwidths can significantly enhance the model capability, networks in multiple bits are proposed for better performance. Qatoui et al. [5] optimized the activation clipping threshold to find the right quantization scale. Zhang et al. [38] further searches the optimal quantizer basis and encoding for accurate quantization. Lee et al. [16] adaptively scaled the gradient element in STE to calibrate the direction of parameter update with minimal discretization errors. However, multi-bit networks still suffers from heavy computational and storage cost. Directly applying existing network binarization methods to submanifold sparse convolution destroys the geometric structure in the scene and degrades the feature informativeness significantly.

Sparse convolution networks: Increased attention has been paid to 3D deep learning on point cloud in recent years, which is important for autonomous driving, AR/VR and robotics. Due to the unordered property of point cloud data, voxelizing the points and applying convolution on 3D grids is a natural solution [4, 26, 40]. However, as point cloud only covers the surfaces of objects/scenes and the most space in 3D scans is empty, the dense volumetric representation is inherently inefficient. Moreover, the computational cost and memory requirement both increase cubically with voxel resolution, thus making it infeasible to train a voxel-based model with high-resolution inputs. To handle this problem, sparse convolution [10, 11] were proposed to restricts computation and storage to “active” sites (i.e. voxels which are not empty). However, as convolutional operator will increase the number of active sites with each layer, the feature sparsity is reduced accordingly. To further improve the efficiency of sparse CNNs, Graham et al. [12] introduced submanifold sparse convolution, which only conducts convolution when the center of kernel slides over active sites and keeps the same level of sparsity throughout the network. This made it practical to train networks with more convolution layers, such as UNet [30], FCN [21] and ResNets [14]. Choi et al. [6] proposed Minkowski Engine, which extended submanifold sparse convolution to higher dimensions. Tang et al. [33] further combined point-based model and sparse CNN to achieve both accurate and efficient 3D perception for large scale scenes. In particular, sparse convolutional networks are able to adopt common deep architectures from 2D vision, which can help standardize deep learning for point cloud, and they are widely utilized in state-of-the-art models for various tasks, such as semantic segmentation [24], object detection [31, 35] and instance segmentation [17, 32, 34].

Differentiable search: In order to reduce the search complexity during the exploration process, differentiable search has been widely used in network architecture search [18], mixed-precision quantization [3, 37] and continual learning [39]. During differentiable search, the superstruc-

ture containing all choices as different components is constructed, where the importance weights for each branch is optimized with gradient descent for optimal solution acquisition. Liu et al. [18] relaxed the space of network architectures to be continuous, and jointly optimized the branch importance weights and parameters of the hypernet for network architecture search. Cai et al. [3] assigned different bitwidths to various branches in the supernet for mixed-precision quantization, and chose the bitwidth in the component with the largest importance weight to be the quantization strategy during inference to achieve the optimal accuracy-complexity trade-off. Gu et al. [13] updated the feature weights through the presented bridge loss which enhanced the knowledge distillation between the students and teachers. In this paper, we extend differentiable search for the discovery of optimal position for active site matching in shifted sparse convolution, where the search cost is significantly reduced for exploration in large space.

3. Approach

In this section, we first briefly introduce the preliminary concept of sparse convolution and network binarization. Then we conduct experiments to show the quantization errors of network binarization methods in different convolution patterns, and introduce the shifted sparse convolution (SFSC) operation which is activated for sites in various locations of the receptive field. Finally, we demonstrate the differentiable search to discover the optimal position for active site matching in SFSC, and construct the BSC-Net with alleviated quantization errors and enhanced performance.

3.1. Preliminaries

Let x_u be an input feature vector of an active site, located at 3-dimensional coordinates $u \in \mathbb{R}^D$. As shown in Figure 1(a), the general sparse convolution [6, 7] by a kernel for x_u is formulated as:

$$F_0(W; x_u) = \sum_{i \in N^D(u)} W_i x_{u+i} \quad (1)$$

where $N^D(u)$ denotes the list of offsets in the 3-dimensional cube centered at origin u . The convolution kernel can be break down and assigned to each offset parameterized by W_i .

Sparse convolution is a practical substitution for vanilla 3D convolution, and skips the non-active regions that only operates when the center of convolutional kernel covers active voxels. Specifically, active voxels are stored as sparse tensors for the xed convolution operations, where all active synapses between input and output voxels are found to sites. As an alternative, we try to explore the subset of convolution. Therefore, the memory requirement and computational cost are significantly reduced in sparse convolutional networks. To further reduce the complexity during inference, network binarization can be leveraged for

Figure 2. Sign correspondence of activations for the first binary layer when binarizing convolutional network, sparse convolutional network and shifted sparse convolutional network for point cloud segmentation on ScanNet dataset. All networks share the same kernel weights. We sort x-axis (different patterns of sparse convolution) by their sign correspondence for better visualization.

weight and activation quantization. In a 1-bit sparse convolutional layer, both convolutional kernels and activations are binarized to -1 and $+1$. In this way, the time-consuming floating-point matrix multiplication can be replaced by bit-wise XNOR and popcount operations:

$$A_b^l = \text{sign}(\text{popcount}(\text{XNOR}(W_b^l; A_b^{l-1}))) \quad (2)$$

where A_b^l and W_b^l represent the binarized activations and weights in the l -th layer respectively, and A_b^l is defined as the binarized version of the real-valued latent weights via $W_b^l = \text{sign}(W_r^l)$.

3.2. Shifted Sparse Convolution

Since the xed operation in sparse convolution is only activated when the central input in the receptive field is active, the constrained exploration of the neighbor active sites makes sparse convolutional networks less robust to binarization. To show this, we calculate the sign correspondence (the proportion of activations in binary network that own same signs with the corresponding real-valued activations, which can measure the quantization error as proved in [29]) for convolutional network and sparse convolutional network with inputs from the ScanNet dataset. We choose the activations of the first binary layer to avoid the accumulation of quantization errors and adopt the same kernel weights for both networks. As shown in Figure 2, the sign correspondences for convolutional layer and sparse convolutional layer are 63.1% and 58.4% respectively, which confirms that sparse convolution will bring larger quantization errors than standard convolution.

However, it is infeasible to adopt convolutional layers in point cloud analysis networks for reducing quantization errors due to the large computational cost from growing active sites. For a single active site, a $3 \times 3 \times 3$ convolution kernel will operate 27 times while sparse convolution kernel only operates at the center. What if we keep the same number of operations with sparse convolution but operates at

other location? To answer it, we extend sparse convolution to enable it to active at different locations. Here we propose the shifted sparse convolution(SFSC) shown in Figure 1(b), which is defined as:

$$F_k(W; x_u) = \sum_{i \in N^D(u + s_k)} W_i x_{u+i} \quad (3)$$

$s_k \in \mathbb{R}^3; k \in \{1, 2, \dots, n_s\}$

where $u + s_k$ is the center of shifted cube instead of u . $N^D(u + s_k)$ is then comprised of the offsets in the shifted cube w.r.t. u . n_s is the number of all unique shifts. For example, for $3 \times 3 \times 3$ sparse convolution operation, there are up to $3^3 - 1 = 26$ possible shifts.

For a general sparse convolution operation, it conducts convolution only when the kernel center overlaps with active sites. While in our SFSC operation, the kernel center can shift to any other locations of the kernel. We use $F_{n_s} = \{F_0; F_1; F_2; \dots; F_{n_s}\}$ to represent the set of all SFSC operations. Note that we consider the general sparse convolution as a special case of SFSC. In a SFSC layer, instead of applying the same sparse convolution operation for all output channels as in a general sparse convolutional layer, we uniformly divide the output channels into several groups (namely channel group), each with a specific SFSC operation. It can be formulated as:

$$y = \text{concat}(f_1(W_1; x); \dots; f_{n_g}(W_{n_g}; x)); f_i \in F_{n_s} \quad (4)$$

where x and y are the input and output of this layer. n_g indicates the number of channel groups. W_i refers to the weights for the i -th SFSC operation. The outputs of all SFSC operations are concatenated along the channel dimension, resulting in a tensor with the same shape as the output of a general sparse convolutional layer.

We randomly sample 50 shift configurations for SFSC layers and compute the sign correspondence, which is shown in Figure 2. It can be seen that different SFSC layers vary a lot in quantization errors and a proportion of them are more robust to binarization compared to sparse convolutional layer. In another word, if we can find out the (near) optimal configurations for all SFSC layers in a network, the quantization error can be reduced without additional computational cost.

3.3. Efficient Search for Shift Operation

Due to the huge design space of shift operation, it is infeasible to decide an optimal configuration for the whole network: the shifted channels and shift directions may be different in each layer, and the total number of possible architectures will be $(8^4)^{13} = 9.1 \times 10^{46}$ for a network with 13 SFSC layers, each layer with 4 channel groups and 8 available shift directions. Although manually designed this way, the composition of SFSC operations are learned by BSC-Net, which shares the same shift strategy in all SFSC layers, is able to reduce the impact of binarization on the

Figure 3. Demonstration of our efficient search method for shift operation. For each SFSC layer and each channel group, we combine all the shift operations in the search space into a 5×5 sparse convolution and assign each direction with a soft selector indicating the importance of the corresponding shift operation, which enables us to directly search the best shift operations via end-to-end gradient descent.

network performance, we resort to automatic architecture search for a better performance. In this section, without further explanation, the default kernel size for original sparse convolution and SFSC is $3 \times 3 \times 3$.

In our BSC-Net, the optimal shift direction for each channel group and each layer may differ. Thus the problem is to search the optimal shift direction for each channel group in the SFSC layer. We formulate this by searching the optimal f_i in (4.4):

$$f_i = \sum_{j=1}^{J^s} \sigma_{ij}^a F_j; i \in \{1, 2, \dots, n_g\} \quad (5)$$

st: $\sigma_{ij}^a = 1; \sigma^a \in \{0, 1\}^g$

where σ^a is a binary selector of the shift direction. As searching in a discrete space makes it hard to optimize the choices, we reformulate the discrete search space as a continuous one by switching σ_{ij} to a composite function ϕ_{ij} :

$$f_i = \sum_{j=1}^{J^s} \phi_{ij}^a F_j; i \in \{1, 2, \dots, n_g\} \quad (6)$$

st: $a \in [0, 1]; \phi_{ij}^a = \frac{1}{1 + \exp(-\frac{a}{\phi_{ij}})}$

where the constraints on weight ϕ_{ij} are eliminated by introducing a set of real architecture parameters ϕ . This sigmoid is different in each layer, and the total number of possible architectures will be $(8^4)^{13} = 9.1 \times 10^{46}$ for a network with 13 SFSC layers, each layer with 4 channel groups and 8 available shift directions. Although manually designed this way, the composition of SFSC operations are learned by BSC-Net, which shares the same shift strategy in all SFSC layers, is able to reduce the impact of binarization on the ϕ_{ij} , which can be optimized end-to-end efficiently.

However, according to (6), the computation and memory increase linearly with the size of search space. All available SFSC operations need to be conducted in weighted summation $f_i = \sum_{j=1}^{n_s} a_{ij} F_j$. Moreover, each SFSC layer owns different parameters, increasing the difficulty of network optimization. To this end, we propose an efficient search method, which absorb all the operations in search space into a larger sparse convolution, as shown in Figure 3.

In this way, we convert the SFSC layer into a 5 × 5 composite sparse convolutional layer, which is used to construct a supernet. This enables us to efficiently search the optimal architecture parameters by end-to-end optimization, regardless of the search space. However, it should be clarified that although the size of search space will not affect the computational efficiency of the supernet, a large search space will make the optimization of architecture parameters hard to converge, thus deteriorate the final performance.

Once the supernet is converged, the optimal BSC-Net must be derived by discretizing the soft selector variables a of (6) into the binary selectors s required by (5). In order to make sure the performance of supernet can precisely reflect the capability of BSC-Net, we constrain a in each SFSC layer by a confidence loss:

$$L_c = \frac{1}{n_g \cdot n_s} \sum_i \sum_j \frac{\|X_i^g - X_j^s\|}{0.5} \quad (7)$$

which pushes a to discrete values.

Optimization approach: In order to decouple the weights and architecture parameters for robust learning [3], we adopt an alternating optimization approach: 1) fix the f_{ij} and optimize W_{ij} ; 2) fix W_{ij} and update f_{ij} .

When we derive the BSC-Net from a converged supernet, both weights and architecture parameters need to be considered. Here we find the following strategy works best: we first train the supernet with binary weight and activation to search for the optimal architecture parameters, from which we choose the shift directions with the highest architecture parameters. Then we initialize the searched BSC-Net with the weights from the supernet and follow the same training procedure as our baseline (introduced in Section 4).

4. Experiment

To investigate the performance of the proposed method, we conduct experiments on several indoor scene datasets including NYU Depth v2 (NYUDv2) [23] and ScanNet [9]. We first introduce the datasets, evaluation metrics and implementation details, which is followed by a strong baseline while maintains the exploration capability. We also evaluate designed for binarization of sparse convolution networks. Then we compare our BSC-Net with the state-of-the-art and BSC-Manual to demonstrate the effectiveness of the network binarization methods on sparse convolutional networks. Finally we design ablation studies to show the effectiveness and efficiency of the presented BSC-Net.

4.1. Experimental Settings

Datasets and metrics: We conduct experiments on two indoor datasets including NYU Depth v2 (NYUDv2) [23] and ScanNet [9]. NYUDv2 contains 1,449 RGB-D scene images, where 795 images are split for training and 654 images for testing. Following [12], we adopt 40-class setting where all pixels are labeled with 40 classes and convert the RGB-D images into 3D point clouds. As the horizontal and vertical directions of spatial dimensions in the RGB-D images are discrete, we voxelize the 3D point clouds to 1cm bins by only discretizing the depth dimension. ScanNet consists of 1513 reconstructed indoor scenes with 21 categories, which are split into 1201 and 312 scenes for training and validation respectively. We adopt two popular settings of ScanNet containing 2cm voxelization and 5cm voxelization as done in [6].

We report the mean intersection of union (mIoU), mean per-point classification accuracy (mAcc) and overall point-wise classification accuracy (Acc) for NYUDv2. For ScanNet, we report mIoU and mAcc. We use the same calculation method in [19] to count the binary operations (BOPs) and floating point operations (FLOPs), where the total operations for model computation complexity evaluation is counted by $OPs = BOPs/64 + FLOPs$. The storage cost are measured by summing the number of real-valued parameters and that of binary numbers divided by 8.

Implementation details: Following [12], we adopt different network architectures for the various datasets. For NYUDv2, we perform experiments with FCN [21] networks in different sizes, namely FCN-S (small) and FCN-H (huge). For ScanNet, we leverage the U-Net [30] architecture in small and huge sizes represented as UNET-S and UNET-H. The small and huge models differ in numbers of filters and sparse convolutional layers per level, which results in capacity variations of point cloud analysis. We use Adam with a stepwise scheduler to optimize the network parameters. The training hyperparameters are introduced in the supplementary materials in detail. We perform data augmentation by applying random affine transformers to the point cloud.

For our BSC-Net, the shift distance in SFSC operations is set to one and the number of channel groups which employ different shift directions is assigned to 8. The search space of directions contains shifting 8 operations represented by $(1; 1; 1)$ and staying still without shift. Limiting the search space of shift directions for channel groups in each layer significantly reduces the search difficulty. We evaluated two variations of our BSC-Net called BSC-Baseline presented techniques. BSC-Baseline represents the framework that binarizing the sparse convolutional networks with all beneficial recent advances combined (refer to Section

Table 1. The mIoU of binarized sparse convolutional networks on ScanNet of different baseline techniques, where the UNET architectures are leveraged. The methods from left to right indicate (1) removing all the skip connections; (2) replacing PReLU with Hardtanh; (3) calculating scaling factor for both activations and weights; (4) using STE to approximate the gradient; (5) removing the skip connections for downsampling/upsampling layers; (6) directly training network with binary weights and activations.

Method	Simplify BS	Simplify AF	Modify SF	Simplify GA	Simplify DS/US	Simplify Init.	Full baseline
mIoU (%)	37.4	50.5	46.1	49.9	47.3/48.7	34.1	51.7

4.2), and BSC-Manual stands for the network binarization for network consisted of SFSC layers with manually defined shift configurations instead of the searched ones. In BSC-Manual, We set $\frac{1}{2}$ of the channel groups unshifted, shift to $(+1; +1; 0)$ and $\frac{1}{4}$ shift to $(-1; -1; 0)$ for NYUDv2, and $\frac{1}{2}$ of the channel groups unshifted, shift to $(+1; +1; +1)$ and $\frac{1}{4}$ shift to $(-1; -1; -1)$ for ScanNet. BSC-Baseline and BSC-Manual are trained in the same way, while BSC-Net is trained with an additional searching stage.

4.2. Strong Baseline

Since network binarization degrades the performance significantly, techniques for accuracy improvements have been studied in recent works of model quantization. To show the performance improvement comes from our proposed method rather than other tricks, we build a strong baseline for binarizing sparse convolutional networks from the recently advances. Through the empirical study shown in Table 1, we discover the beneficial techniques for performance enhancement and list as below.

Block structure: We use the same block structure as ReActNet [19], where the operations are ordered as Binarization, SparseConv, BatchNorm, Activation in each basic block.

Activation function: PReLU [14, 19] considers the negative inputs with better convergence, and we substitute all ReLU activation layers with PReLU to strengthen the performance.

Scaling factor: We only calculate the layer-wise scaling factor for weights as demonstrated in [22], which is the mean absolute value of full-precision weights.

Gradient approximation: A piecewise polynomial function [20] is used to approximate the sign function, which acquires more accurate gradient during back propagation.

Downsampling/upsampling: Following [20], the skip connection for downsampling layer is composed of an average pooling and a real-valued convolutional layer with kernel size 1. We also verify that an unpooling layer with a full-precision convolution with kernel size 1 is beneficial in the skip connection for upsampling layer.

Initialization: We first pretrain the network with full-precision weights and activations for initialization. Then the model with binary weights and activations is trained for binarization.

4.3. Comparison with State-of-the-art

In this section, we compare our method with state-of-the-art binarization methods, including XNOR-Net [29], XNOR-Net++ [2], BiPointNet [28], Bi-Real-Net [20] and ReActNet [19]. We also provide the performance of the real valued models for reference. Experiments are conducted on NYUDv2 and ScanNet.

Results on NYUDv2: Table 3 illustrates the comparison of storage, operations per second (OPs) and semantic segmentation results across several popular network binarization methods and our BSC-Net. Bi-Real-Net performs best among previous methods, which shows the gradient approximation method and the skip connection structure are general and effective in sparse convolutional network binarization. Although BiPointNet is designed for 3D point cloud analysis, it fails to achieve satisfactory performance because the operations such as maxpooling and point-wise MLP used in PointNet are not adopted in sparse convolutional networks. BSC-Baseline outperforms the previous methods by a large margin and its performance is further boosted by the proposed SFSC module, i.e. BSC-Manual. When adopting the efficient differentiable search method, BSC-Net achieves the state-of-the-art performance in both architectures of FCN, while the extra computational overhead is negligible compared to previous methods. Observing the last row in Table 3, The performance gap between real valued FCN-H and our BSC-Net has even been narrowed to less than 3%, which shows the great application potentials of our method.

Results on ScanNet: Different from NYUDv2 in which the point clouds are generated from single RGB-D images, ScanNet provides larger and more complete point cloud scenes via 3D reconstruction. Therefore, we can evaluate our BSC-Net on ScanNet with different resolutions of the input point cloud after voxelization as shown in Table 2. Following [12], we evaluate all results three times to address the problem of the number of voxels being greatly smaller than that of points. Similar to NYUDv2, BSC-Baseline outperforms the previous state-of-the-art. We found the gap between BSC-Baseline and previous methods were larger because the upsampling layer in UNET is implemented by deconvolution, which is more sensitive to binarization than the interpolation used in FCN. In each setting, BSC-Manual gains consistent improvement over BSC-Baseline, and BSC-Net further achieves state-of-the-art performances

Table 2. Semantic segmentation results (%), model storage cost (M) and computation cost in OPs of different methods on ScanNet validation set. Param. means the model storage cost (M). 5cm voxel and 2cm voxel refer to different resolutions of the input point cloud after voxelization.

	Method	Param.	5cm voxel				2cm voxel			
			OPs		mIoU	mAcc	OPs		mIoU	mAcc
UNET-S	Real valued	4.335	1:21	10^9	65.2	73.3	5:32	10^9	68.7	78.5
	XNOR-Net	0.136	8:07	10^7	33.3	38.9	3:79	10^8	21.0	26.1
	XNOR-Net++	0.136	8:07	10^7	12.6	15.9	3:79	10^8	11.2	13.7
	BiPointNet	0.136	8:07	10^7	30.1	36.2	3:79	10^8	18.4	20.7
	Bi-Real-Net	0.138	8:12	10^7	48.3	56.6	3:82	10^8	51.2	63.3
	ReActNet	0.138	8:12	10^7	43.6	50.2	3:82	10^8	46.9	52.9
	BSC-Baseline	0.139	8:12	10^7	51.7	61.8	3:82	10^8	54.9	65.3
	BSC-Manual	0.139	8:12	10^7	53.2	63.7	3:82	10^8	57.8	66.6
	BSC-Net	0.139	8:12	10^7	54.4	65.2	3:82	10^8	61.4	70.4
UNET-H	Real valued	30.104	7:65	10^9	67.6	75.1	3:38	10^{10}	71.0	79.0
	XNOR-Net	0.939	3:61	10^8	46.6	53.5	1:75	10^9	34.9	40.1
	XNOR-Net++	0.939	3:61	10^8	13.3	16.4	1:75	10^9	12.8	16.2
	BiPointNet	0.939	3:61	10^8	45.2	52.4	1:75	10^9	34.3	39.8
	Bi-Real-Net	0.948	3:63	10^8	53.4	63.2	1:76	10^9	57.3	66.9
	ReActNet	0.949	3:63	10^8	52.2	59.0	1:76	10^9	57.1	67.0
	BSC-Baseline	0.952	3:63	10^8	56.0	65.9	1:76	10^9	59.3	68.3
	BSC-Manual	0.952	3:63	10^8	59.3	68.7	1:76	10^9	62.2	70.1
	BSC-Net	0.952	3:63	10^8	62.2	70.5	1:76	10^9	63.9	71.6

Table 3. Semantic segmentation results (%), model storage cost (M) and computation cost in OPs of different methods on NYUDv2 test set. Param. means the model storage cost (M).

	Method	Param.	OPs		mIoU	mAcc	Acc
FCN-S	Real valued	2.496	1:24	10^9	33.9	47.7	64.7
	XNOR-Net	0.108	1:72	10^8	22.1	32.7	57.3
	XNOR-Net++	0.108	1:72	10^8	8.5	13.5	43.9
	BiPointNet	0.108	1:72	10^8	24.9	35.7	59.3
	Bi-Real-Net	0.110	1:75	10^8	27.3	38.4	60.0
	ReActNet	0.110	1:75	10^8	25.4	36.6	58.9
	BSC-Baseline	0.110	1:75	10^8	27.8	39.9	60.1
	BSC-Manual	0.110	1:75	10^8	28.7	40.9	60.2
	BSC-Net	0.110	1:75	10^8	29.7	42.1	61.2
FCN-H	Real valued	10.025	4:82	10^9	36.9	50.4	67.2
	XNOR-Net	0.357	3:08	10^8	27.1	38.8	59.6
	XNOR-Net++	0.357	3:08	10^8	8.4	13.6	43.2
	BiPointNet	0.357	3:08	10^8	28.1	40.8	60.1
	Bi-Real-Net	0.361	3:09	10^8	30.4	41.8	61.1
	ReActNet	0.361	3:09	10^8	27.0	40.1	58.9
	BSC-Baseline	0.362	3:09	10^8	32.0	44.4	63.3
	BSC-Manual	0.362	3:09	10^8	32.5	45.1	63.5
	BSC-Net	0.362	3:09	10^8	33.9	46.2	64.5

which proves the effectiveness of differentiable search strategy. We also noticed that the improvement of BSC-Net over BSC-Baseline on ScanNet is larger than that on NYUDv2, that shows our method can exploit richer geometric information from 3D point clouds than 2.5D depth map.

4.4. Ablation Study

We conduct ablation studies to show how different hyperparameters and strategies influence the performance of the proposed BSC-Net. We study the effects of the search space and number of channel group as well as searching strategy in our differentiable search method on the ScanNet (voxelization) using UNET-S.

Performance w.r.t. searching hyperparameters: In order to reduce the search cost as well as the optimization difficulties, we search the optimal shift strategies for different layers in BSC-Net from a subset of the whole search space, and we also partition the channels into groups which share the same shift strategies. Table 4 demonstrates the performance variation for BSC-Net with different search space and group numbers of SFSC operations, where C_k^d and C_k^d represent the convolutions staying still and those shifted to the direction of the k-th vertex of a cube.

Observing the third, fourth and seventh rows, we conclude that the mIoU and mAcc of BSC-Net improves as the size of the search space increases. The improvement from search space in size 3 to that in size 5 is much higher than that from size 5 to size 9, which indicates that large search space causes optimization difficulties in differentiable search method and a subset of the whole search space contains the solution near to the optimal one. According to the last four rows in Table 4, the performance achieves the optimal for medium numbers of groups in the differentiable

Table 4. The effects of search space and group number in differentiable search method on the final performance.

Search space	Group number	mIoU(%)	mAcc(%)
Baseline: f Sg	—	51.7	61.8
f S; C ₁ ; C ₈ g	8	53.6	63.9
f S; C ₁ ; C ₄ ; C ₆ ; C ₇ g	8	54.2	64.9
	2	52.9	63.8
f S; C ₁ ; C ₂ ; C ₃ ;	4	53.4	64.6
C ₄ ; C ₅ ; C ₆ ; C ₇ ; C ₈ g	8	54.4	65.2
	16	54.0	64.5

Table 5. The effects of relaxation and derivation strategy in differentiable search method on the final performance. The architecture parameters are frozen.

Relaxation	Derivation			mIoU(%)	mAcc(%)
Random	D (32; 32) !	D (1; 1) !	D (1; 1)	51.5	61.2
	S (32; 32) !	S (1; 1) !	D (1; 1)	51.8	62.0
Softmax	S (32; 32) !	S (1; 1) !	D (1; 1)	52.3	63.2
	S (1; 1) !	D (32; 32) !	D (1; 1)	53.5	64.5
	S (32; 32) !	S (1; 1) !	D (1; 1)	52.6	64.0
Sigmoid	S (32; 32) !	S (1; 1) !	D (1; 1)	53.7	64.7
	S (1; 1) !	D (32; 32) !	D (1; 1)	54.4	65.2

search. Increasing the numbers causes the optimization difficulties due to the large search space, and decreasing the numbers excludes the promising solution due to the channel correlation.

Performance w.r.t. searching strategy: We further investigate the effects of searching strategy in Table 5, including relaxation method for the binary selector in (5) and strategy for deriving BSC-Net from the supernet. For softmax relaxation, f_{ij} and g are defined as $f_{ij}^a = \frac{\exp(-f_{ij})}{\sum_k \exp(-f_{ik})}$, and the confidence constraint in (7) is changed to:

$$L_c = \sum_i \sum_j \sum_k \sum_l \log f_{ij}^a; l_{ij} = \begin{cases} < 1; & j = \arg\max_k f_{ik}^a \\ 0; & \text{otherwise} \end{cases} \quad (8)$$

which pushes f^a to a one-hot tensor for better derivation. We use $S(W; A)$ and $D(W; A)$ to represent supernet and derived BSC-Net with W -bit weights and A -bit activations.

We first conduct random assignment on the shift directions for each layer and channel group. As shown in the first row, the performance of BSC-Net falls even behind of the baseline, which shows searching or manually designing a proper SFSC configuration is essential for realizing the potential of BSC-Net. The last six rows show that sigmoid relaxation is better than softmax, which is due to softmax will bring competition between different SFSC operations and hurts the performance of supernet. The results also testify the superiority of our derivation strategy.

Notably, we should emphasize that the SFSC operation is proposed for reducing quantization error, which does not (BAAI).

Figure 4. Visualization results of different methods.

work for real-valued networks. We do not observe an obvious change in performance when equipping the real-valued UNET or FCN with SFSC.

4.5. Visualization Results

We visualize the segmentation prediction for different methods in Figure 4. Black regions in the ground-truth refer to undefined categories. The predictions of previous methods are discontinuous and they misclassify the shelf as wall or window, while our BSC-Net outputs smooth and accurate predictions. The results in red boxes also provide an intuitive explanation on our method: when binarized, sparse convolutional networks fail to fully explore the neighbor active sites with increased quantization errors and thus predict discontinuous results. On the contrary, BSC-Net better exploits the neighborhood and reduces the quantization error.

5. Conclusion

In this paper, we have presented BSC-Net that learns binary sparse convolutional networks for efficient point cloud analysis. We present the shifted sparse convolution that is activated for receptive field whose pre-defined locations match active sites. By searching the optimal positions for active site matching in shifted sparse convolution, the quantization errors in binarized sparse convolutional networks are alleviated significantly without additional computational cost. For fair evaluation, we combine previous techniques to construct a strong baseline. Extensive experiments on semantic segmentation of point clouds demonstrate the superiority of BSC-Net.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFA0700802, in part by the National Natural Science Foundation of China under Grant 62125603, and in part by a grant from the Beijing Academy of Artificial Intelligence.

References

- [1] Adrian Bulat, Brais Martinez, and Georgios Tzimiropoulos. Bats: Binary architecture search. *ECCV*, pages 309–325, 2020. [2](#)
- [2] Adrian Bulat and Georgios Tzimiropoulos. Xnor-net++: Improved binary neural networks. *arXiv preprint arXiv:1909.13863* 2019. [6](#)
- [3] Zhaowei Cai and Nuno Vasconcelos. Rethinking differentiable search for mixed-precision neural networks. *CVPR*, pages 2349–2358, 2020. [2](#), [3](#), [4](#), [5](#)
- [4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* 2015. [2](#)
- [5] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085* 2018. [2](#)
- [6] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pages 3075–3084, 2019. [1](#), [2](#), [3](#), [5](#)
- [7] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair darts: Eliminating unfair advantages in differentiable architecture search. *ECCV*, pages 465–480. Springer, 2020. [4](#)
- [8] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830* 2016. [2](#)
- [9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, 2017. [5](#)
- [10] Martin Engelcke, Dushyant Rao, Dominic Zeng Wang, Chi Hay Tong, and Ingmar Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *ICRA*, pages 1355–1361, 2017. [2](#)
- [11] Benjamin Graham. Spatially-sparse convolutional neural networks. *arXiv preprint arXiv:1409.6070* 2014. [2](#)
- [12] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *CVPR*, pages 9224–9232, 2018. [1](#), [2](#), [3](#), [5](#), [6](#)
- [13] Yushuo Guan, Pengyu Zhao, Bingxuan Wang, Yuanxing Zhang, Cong Yao, Kaigui Bian, and Jian Tang. Differentiable feature aggregation search for knowledge distillation. In *ECCV*, pages 469–484, 2020. [3](#)
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, pages 770–778, 2016. [2](#), [6](#)
- [15] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NeurIPS*, pages 4114–4122, 2016. [2](#)
- [16] Junghyup Lee, Dohyung Kim, and Bumsub Ham. Network quantization with element-wise gradient scaling. *CVPR*, pages 6448–6457, 2021. [2](#)
- [17] Zhihao Liang, Zhihao Li, Songcen Xu, Mingkui Tan, and Kui Jia. Instance segmentation in 3d scenes using semantic superpoint tree networks. *ICCV*, pages 2783–2792, 2021. [2](#)
- [18] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055* 2018. [2](#), [3](#)
- [19] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. *ECCV*, page 143–159, 2020. [1](#), [5](#), [6](#)
- [20] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. *ECCV*, pages 722–737, 2018. [2](#), [6](#)
- [21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. [2](#), [5](#)
- [22] Brais Martinez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. *arXiv preprint arXiv:2003.11535* 2020. [6](#)
- [23] Pushmeet Kohli, Nathan Silberman, Derek Hoiem, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. [5](#)
- [24] Alexey Nekrasov, Jonas Schult, Or Litany, Bastian Leibe, and Francis Engelmann. Mix3d: Out-of-context data augmentation for 3d scenes. *ICDV*, pages 116–125. IEEE, 2021. [2](#)
- [25] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *ICVPR*, pages 652–660, 2017. [1](#)
- [26] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, pages 5648–5656, 2016. [2](#)
- [27] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS*, pages 5099–5108, 2017. [1](#)
- [28] Haotong Qin, Zhongang Cai, Mingyuan Zhang, Yifu Ding, Haiyu Zhao, Shuai Yi, Xianglong Liu, and Hao Su. Bipointnet: Binary neural network for point clouds. *arXiv preprint arXiv:2010.05501* 2020. [2](#), [6](#)
- [29] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: ImageNet classification using binary convolutional neural networks. *ECCV*, pages 525–542, 2016. [2](#), [3](#), [6](#)
- [30] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015. [2](#), [5](#)
- [31] Danila Rukhovich, Anna Vorontsova, and Anton Konushin. Fcaf3d: Fully convolutional anchor-free 3d object detection. *arXiv preprint arXiv:2112.00322* 2021. [2](#)

- [32] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d for 3d semantic instance segmentation. *arXiv preprint arXiv:2210.03105*, 2022. 2
- [33] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. *ECCV*, 2020. 2
- [34] Thang Vu, Kookhoi Kim, Tung M Luu, Thanh Nguyen, and Chang D Yoo. Softgroup for 3d instance segmentation on point clouds. In *CVPR*, pages 2708–2717, 2022. 2
- [35] Haiyang Wang, Lihe Ding, Shaocong Dong, Shaoshuai Shi, Aoxue Li, Jianan Li, Zhenguo Li, and Liwei Wang. Cagroup3d: Class-aware grouping for 3d object detection on point clouds. *arXiv preprint arXiv:2210.04264*, 2022. 2
- [36] Ziwei Wang, Jiwen Lu, Ziyi Wu, and Jie Zhou. Learning efficient binarized object detectors with information compression. *TPAMI*, 2021. 1
- [37] Ziwei Wang, Changyuan Wang, Xiuwei Xu, Jie Zhou, and Jiwen Lu. Quantformer: Learning extremely low-precision vision transformers. *TPAMI*, 2022. 2
- [38] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. *ECCV*, pages 365–382, 2018. 2
- [39] Yanzhe Zhang, Xuezhi Wang, and Diyi Yang. Continual sequence generation with adaptive compositional modules. *arXiv preprint arXiv:2203.10652*, 2022. 2
- [40] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. *CVPR*, pages 4490–4499, 2018. 2