



Zhiyuan Zhang<sup>1</sup> cszyzhang@gmail.com <sup>1</sup>Singapore University of Technology and Design

### Contributions

東京大学 THE UNIVERSITY OF TOKYO

- **ShellConv**, a simple yet effective convolution operator for orderless point cloud;
- **ShellNet**, an efficient neural network based on ShellConv.
- The state-of-the-art accuracy and efficiency are achieved on object classification, part segmentation, and semantic segmentation;

### **ShellConv**



### ShellNet



- Shell Size (number of points contained in each shell) is set to 16 for classification and 8 for segmentation • Network is implemented in TensorFlow and run on a NVIDIA GTX 1080 GPU for all experiments
- The optimization is done with an Adam optimizer with initial learning rate set to 0.001

# **ShellNet:** Efficient Point Cloud Convolutional Neural Networks using Concentric Shells Statistics

binhson.hua@gmail.com <sup>2</sup>The University of Tokyo

### Algorithm 1 ShellConv operator.

**Input:**  $p, \Omega_p, \{F_{prev}(q) : q \in \Omega_p\}$  \* Representative point, point set, and revious layer features of point set

\* Centralization with p as the center.  $\{q\} \leftarrow \{q - p : \forall q \in \Omega_p\}$ \* Lift each q to a higher dimensional space.  $\mathcal{F}_{local}(q)\} \leftarrow \{\mathrm{mlp}(q)\}$ \* Concatenate the local and  $F(q)\} \leftarrow \{[F_{prev}(q), F_{local}(q)\}$ evious layer features.

 $\{S\} \leftarrow \{S: q \in \Omega_S\}$  \* Determine which shell q belongs to according the distances from q to center p.

(S)  $\leftarrow$  {maxpool({ $F(q) : q \in \Omega_S$ }) :  $\forall S$ } \* Get fixed-size ature of each shell by a maxpool over all points in the shell.  $\leftarrow \operatorname{conv}(\{F(S)\}) * \operatorname{Perform} a 1D \operatorname{convolution} with all shell features$ 

from inner to outer. return  $F_p$ ;

Project Page with Codes

### **Quantitative Results**

#### **Classification results on ModelNet40:**

Method	Core Operator	input format	OA		
FPNN	1D Conv.	Point	87.5		
Vol. CNN	3D Conv.	Voxel	89.9		
O-CNN	Sparse 3D Conv.	Octree	90.6		
Pointwise	Point Conv.	Point	86.1		
PointNet	Point MLP	Point	89.2		
PointNet++	Multiscale Point MLP	Point+Normal	90.7		
PointCNN	X-Conv	Point	92.2		
ShellNet (ss=8)	ShellConv	Point	91.0		
ShellNet (ss=16)	ShellConv	Point	93.1		
ShellNet (ss=32)	ShellConv	Point	93.1		
ShellNet (ss=64)	ShellConv	Point	92.8		

#### **Segmentation results:**

Method	ShapeNet mpIoU	ScanNet OA	S3DIS mIoU	Semantic3D mIoU
SyncCNN	82.0	_	-	_
SpiderCNN	81.7	-	-	-
SplatNet	83.7	-	-	_
SO-Net	81.0	-	-	-
SGPN	82.8	-	50.4	-
PCNN	81.8	-	-	-
KCNet	82.2	-	-	_
3DmFV-Net	81.0	-	_	_
DGCNN	82.3	-	56.1	-
RSNet	81.4	-	56.5	_
PointNet	80.4	73.9	47.6	_
PointNet++	81.9	84.5	_	_
PointCNN	84.6	85.1	65.4	_
TMLC-MSR	_	-	-	54.2
DeePr3SS	_	-	-	58.5
SnapNet	-	-	-	59.1
SegCloud	-	-	-	61.3
SPG	-	-	62.1	73.2
ShellNet	82.8	85.2	66.8	69.4





The accuracy of point cloud classification of different methods over time and epochs.It can achieve over 80% accuracy within two minutes, and reach 90% on the test dataset after only 15 minutes of training.

Binh-Son Hua<sup>2</sup>

Sai-Kit Yeung<sup>3</sup>

saikit@ust.hk

<sup>3</sup>The Hong Kong University of Science and Technology

### **Qualitative Results**



**Outdoor Scene Semantic Segmentation on Semantic3D:** 



### **Network Efficiency**

Methods	Params	FLOPs	Time
		(Train/Infer)	(Train/Infer
PointNet	3.5M	44.0B / 14.7B	0.068s / 0.01
PointNet++	12.4M	67.9B /26.9B	0.091s / 0.02
3DmFV	45.77M	48.6B /16.9B	0.101s / 0.03
DGCNN	1.84M	131.4B /44.3B	0.171s / 0.06
PointCNN	0.6M	93.0B /25.3B	0.031s / 0.01
ShellNet	<b>0.48</b> M	15.8B /2.8B	0.066s / 0.02
with small RF	<b>0.48M</b>	9.51B /1.5B	0.025s / 0.01

Trainable parameters, FLOPs and running time comparisons. Compared to other methods, ShellNet is lightweight and fast while being accurate. Reducing the receptive field (small RF) by setting a smaller shell size can make the computation even faster as neighbor query becomes cheaper.

### Acknowledgement

from HKUST (R9429).

![](_page_0_Picture_49.jpeg)

## **Ablation Study**

	(A)	(B)	(C)	(D)
1. Sampling	Random	Farthest	Random	Random
2. Shell size	Fixed	Fixed	Dynamic	Fixed
3. KNN type	xyz	xyz	xyz	Features
Accuracy (%)	93.1	93.1	92.7	92.4
Train Time	0.066s	0.078s	0.118s	0.081s
Infer. Time	0.023s	0.024s	0.033s	0.029s

Experiments with neighbor point sampling. Setting (A) is the default strategy. Setting (B), (C), (D) are modified from (A) based on point sampling type, shell size, and neighbor query features. As can be seen, setting (B) – furthest point sampling, (C) – equidistant shells, (D) – latent features for neighborhood construction, produces similar accuracy but training and inference time becomes slower.