# FlatFormer: Flattened Window Attention for Efficient Point Cloud Transformer

Zhijian Liu<sup>1,#</sup>,Xinyu Yang<sup>2,#</sup>,Haotian Tang<sup>1</sup>,Shang Yang<sup>3</sup>,Song Han<sup>1</sup>1: MIT2: Shanghai Jiao Tong University3: Tsinghua University



# Background

### Point cloud deep learning is widely used in real-world applications



**Data:** 3D Point Cloud

FlatFormer: Flattened Window Attention for Efficient Point Cloud Transformer



### **Applications**





# Background

### State-of-the-art point cloud models are based on sparse convolution

	Method				Metrics										
	Date	Name	Modalities	Map data	External data	mAP	mATE (m)	mASE (1-IOU)	mAOE (rad)	mAVE (m/s)	mAAE (1-acc)	NDS	PKL *	FPS (Hz)	Stats
			Lidar	All	All										
>	2023-03-10	FocalFormer3D-TT/	Lidar	no	no	0.705	0.243	0.238	0.321	0.200	0.131	0.739	0.564	n/a	îîÎ
>	2023-03-06	Real-Aug	Lidar	no	no	0.689	0.237	0.227	0.320	0.184	0.131	0.734	0.597	n/a	îíÍ
>	2022-10-31	LinK	Lidar	no	no	0.698	0.238	0.229	0.312	0.234	0.136	0.734	0.575	n/a	îîÎ
>	2022-05-30	LargeKernel-L	Lidar	no	no	0.688	0.244	0.230	0.312	0.241	0.132	0.728	0.581	n/a	îîÎ
>	2022-11-11	SphereFormer	Lidar	no	no	0.685	0.242	0.233	0.353	0.191	0.131	0.728	0.616	n/a	îîÎ
>	2022-09-27	MDRNet-L	Lidar	no	no	0.684	0.242	0.229	0.319	0.227	0.129	0.728	0.581	n/a	îîÎ
>	2021-07-30	DAA_AVP	Lidar	no	no	0.697	0.237	0.229	0.376	0.248	0.124	0.727	0.937	n/a	îîÎ
>	2022-08-18	MGTANet	Lidar	no	no	0.675	0.250	0.233	0.308	0.193	0.124	0.727	0.538	n/a	îîÎ
>	2023-03-10	FocalFormer3D	Lidar	no	no	0.687	0.254	0.242	0.340	0.218	0.126	0.726	0.586	n/a	îîÎ
>	2021-04-19	CenterPoint-VID	Lidar	no	no	0.674	0.255	0.235	0.339	0.233	0.128	0.718	0.555	n/a	îîÎ





# Background

### Sparse convolution requires specialized system support to run on GPUs

→ Mit-han-lab / torchsparse   → Code    → Issues    → Pul	Public       Image: Security         Public       Image: Security         Il requests       4       Image: Discussions       Image: Security         Il requests       4       Image: Discussions       Image: Security	<pre>from torch import nn from torchsparse import nn as tsnn from torchsparse import SparseTensor</pre>			
Image: state state state       Image: state state state state         Image: state state state state       Image: state state state         Image: state state state       Image: state state         Image: state state       Image: state state         Image: state state       Image: state         Image: state       Image: state <th>4 tags       Go to file       Add file -       &lt;&gt; Code         tune function (#190)       ✓ 1a10fda on Jan 8 🕥 91 commits</th> <th rowspan="2">About   [MLSys'22] TorchSparse: Efficient Point   Cloud Inference Engine   ? torchsparse.mit.edu   acceleration   point-cloud   pytorch</th> <th rowspan="3"><pre># Model definition model = nn.Sequential(    tsnn.Conv3d(4, 16, 3, stride=2, padding=    tsnn.BatchNorm(16),    tsnn.ReLU(True), ).cuda() # Tensor definition x = SparseTensor(feats, coords).cuda()</pre></th>	4 tags       Go to file       Add file -       <> Code         tune function (#190)       ✓ 1a10fda on Jan 8 🕥 91 commits	About   [MLSys'22] TorchSparse: Efficient Point   Cloud Inference Engine   ? torchsparse.mit.edu   acceleration   point-cloud   pytorch	<pre># Model definition model = nn.Sequential(    tsnn.Conv3d(4, 16, 3, stride=2, padding=    tsnn.BatchNorm(16),    tsnn.ReLU(True), ).cuda() # Tensor definition x = SparseTensor(feats, coords).cuda()</pre>		
<ul> <li>.github</li> <li>docs</li> <li>examples</li> </ul>	Update pre-commit to avoid installing clang-format locally2 years agoUpdate FAQ.md (#103)2 years agoAdd optimizations introduced in MLSys paper (#150)8 months ago				
<ul> <li>torchsparse</li> <li>.gitignore</li> <li>.pre-commit-config.yaml</li> </ul>	Fix save_dir in the tune function (#190)3 months agoReformat codebase and add pre-commit (#81)2 years agoInclude paper, website information (#177)6 months ago	<ul> <li>▲ MIT license</li> <li>☆ 750 stars</li> <li>③ 18 watching</li> <li>♀ 98 forks</li> </ul>			
LICENSE          README.md         requirements.txt         setup.cfg	Reformat codebase and add pre-commit (#81)2 years agoInclude paper, website information (#177)6 months agoSupport cached_property for python version < 3.8 (#171)7 months agoReformat codebase and add pre-commit (#81)2 years ago	Report repository          Releases       4         Image: Sympletic state       1.4.0	# Forward and backward pass y = model(x)		
setup.py	Reformat codebase and add pre-commit (#81) 2 years ago	on Jun 24, 2021 + <b>3 releases</b>	loss = childenton(y,)		

TorchSparse: Efficient Point Cloud Inference Engine [Tang et al., MLSys 2022]

**FlatFormer:** Flattened Window Attention for Efficient Point Cloud Transformer

Installation: pip install --upgrade git+https://github.com/mit-han-lab/torchsparse.git

https://flatformer.mit.edu 4





## **Point Cloud Transformers** Achieve comparable accuracy but lag far in latency (3-4X slower)









# **Point Cloud Transformers**

### Achieve comparable accuracy but lag far in latency (3-4X slower)

**Global PCTs:** Apply MHSA globally across the entire point cloud.

# The model takes almost **1** second to run with **32k** input points.

Guo et al., "PCT: Point Cloud Transformer", CVM 2021



**FlatFormer:** Flattened Window Attention for Efficient Point Cloud Transformer



The runtime of global PCTs grows **quadratically** as the number of points grows.

**Window PCTs**: Apply MHSA to a set of non-overlapping windows.

- Window PCTs suffer from the **padding and** partitioning overhead due to the
- unbalanced workload across windows.

Fan et al., "Embracing Single Stride 3D Object Detector with Sparse Transformer", CVPR 2022





# **FlatFormer: Flattened Window Attention**

Equal-size grouping trades spatial proximity for computational regularity

Equal-window grouping maintains perfect **spatial** proximity but breaks the computational regularity.



FlatFormer: Flattened Window Attention for Efficient Point Cloud Transformer

Equal-size grouping ensures balanced **computation** workload but cannot guarantee the geometric locality.



https://flatformer.mit.edu 7



### **FlatFormer: Flattened Window Attention** Window-based sorting preserves geometric locality after flattening

	Α		
С	Е		
	н	В	
	G		F
		D	

Sort all points first by **window coordinates** and then by **local coordinates** within the window.







**Window Shift** 

Alternate Sorting Axis

Sorting Criterion	Window Shifting	Axis Alternation	Mean L mAPH
Random	—	—	57.8
Point	—	Υ	60.4
Window	Y	Ν	61.1
Window	Ν	Υ	61.2
Window	Υ	Y	61.7







### **Efficient Implementation** FlatFormer can benefit from existing system optimizations for transformers



![](_page_8_Picture_3.jpeg)

![](_page_8_Picture_6.jpeg)

![](_page_8_Picture_7.jpeg)

## **Results: 3D Object Detection on Waymo** FlatFormer closes the latency gap between PCTs and SpConv-based models

![](_page_9_Figure_1.jpeg)

![](_page_9_Picture_5.jpeg)

![](_page_9_Picture_6.jpeg)

![](_page_9_Picture_7.jpeg)

![](_page_9_Picture_8.jpeg)

# **Analysis: Grouping & Attention**

### Equal-size grouping is mostly spatially local.

![](_page_10_Figure_2.jpeg)

### Attention learns to **suppress outlier points**.

![](_page_10_Figure_4.jpeg)

# Attention

![](_page_10_Picture_9.jpeg)

![](_page_10_Picture_10.jpeg)

![](_page_10_Picture_11.jpeg)

# **Deployment on Edge GPUs**

![](_page_11_Figure_2.jpeg)

**FlatFormer:** Flattened Window Attention for Efficient Point Cloud Transformer

![](_page_11_Picture_4.jpeg)

### First point cloud transformer that achieves real-time performance on edge GPUs!

![](_page_11_Picture_9.jpeg)

![](_page_11_Picture_10.jpeg)