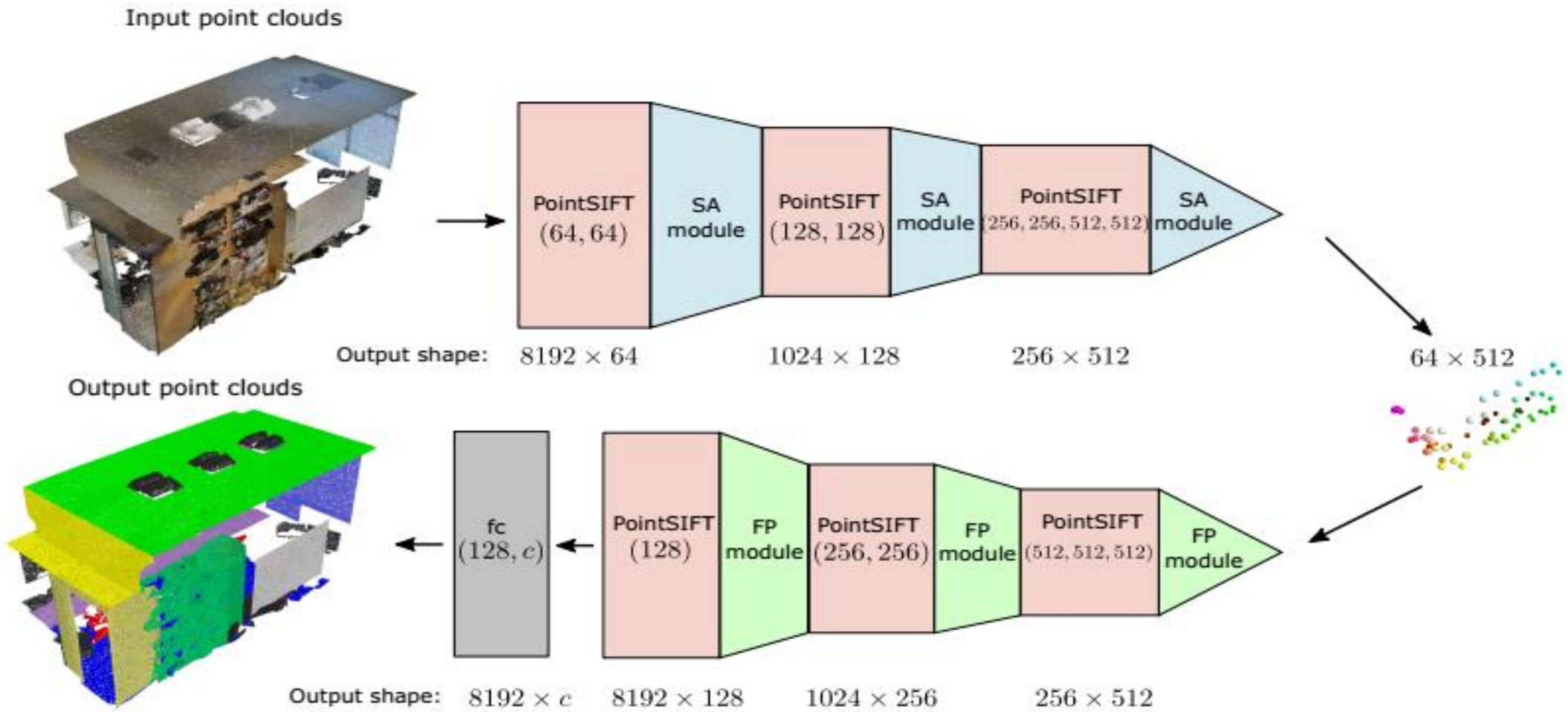
PointSIFT和PointCNN

报告人：李东威
时间：2019.04.19

PointSIFT

这篇文章提出了一个模块，输入点云，输出具有特好表达能力的点云特征，同时特征维度不变，能够嵌入不同的PointNet_based网络



PointSIFT： PointSIFT Module

1 OE Unit　输入一个点的特征，输出具有方向信息的特征(局部特征描述子)

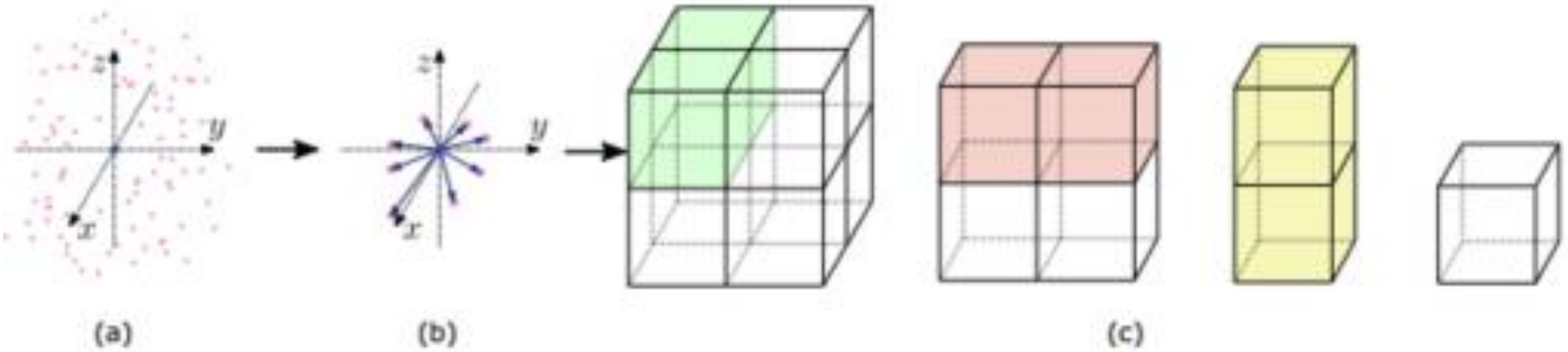
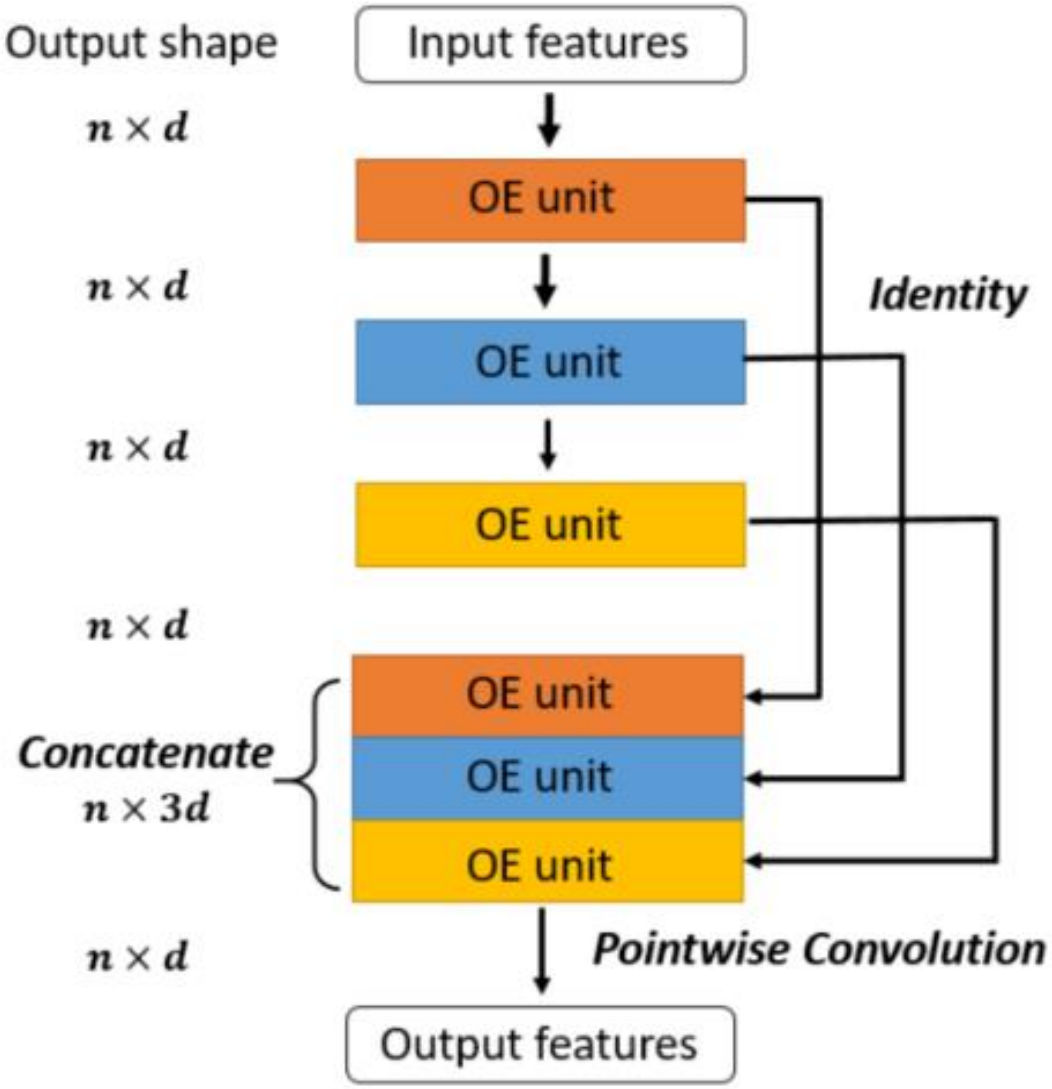


Figure 3. Illustration of Orientation-encoding(OE) Unit. (a): Point cloud in 3D space, the input point is at origin. (b) nearest neighbor search in eight octants. (c) convolution along X, Y, Z axis.

PointSIFT：PointSIFT Module

2 PointSIFT Module

亮点：自动选择合适的Scale
Multi-scale representation



PointSIFT： Insight

1 更好的选点方法（8方向）更好的表达能力（与KNN比较）

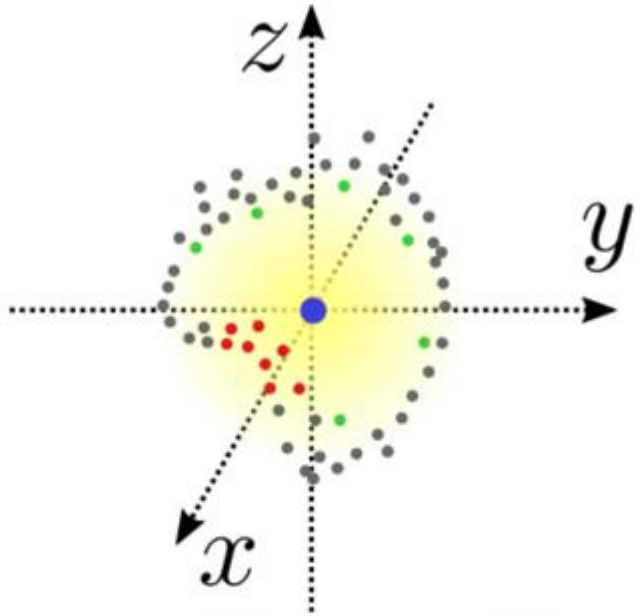


Figure 7. In this case, using K nearest neighbors, all chosen points are from one direction (red points). If we select points in different directions (green points), the representation ability will be better.

PointSIFT：Insight

2 不会丢失点（与PointNet++比较）？

Table 1. Effectiveness of PointSIFT Module.

downsampling step	first	second	third	fourth
point cloud size	8192	1024	256	64
captured point cloud size of Pointnet++[24]	6570	1010	255	64
captured point cloud size of PointSIFT framework	8192	1024	256	64

PointSIFT：实验结果和缺点

Table 4. ScanNet[6] label accuracy and mIoU

Method	Accuracy %	mean IoU
3DCNN[6]	73.0	-
PointNet[22]	73.9	-
PointNet++[24]	84.5	38.28
PointCNN[13]	85.1	-
Ours	**86.2**	**41.5**

Table 5. Overall accuracy and meaning intersection over union metric of S3DIS[1] dataset.

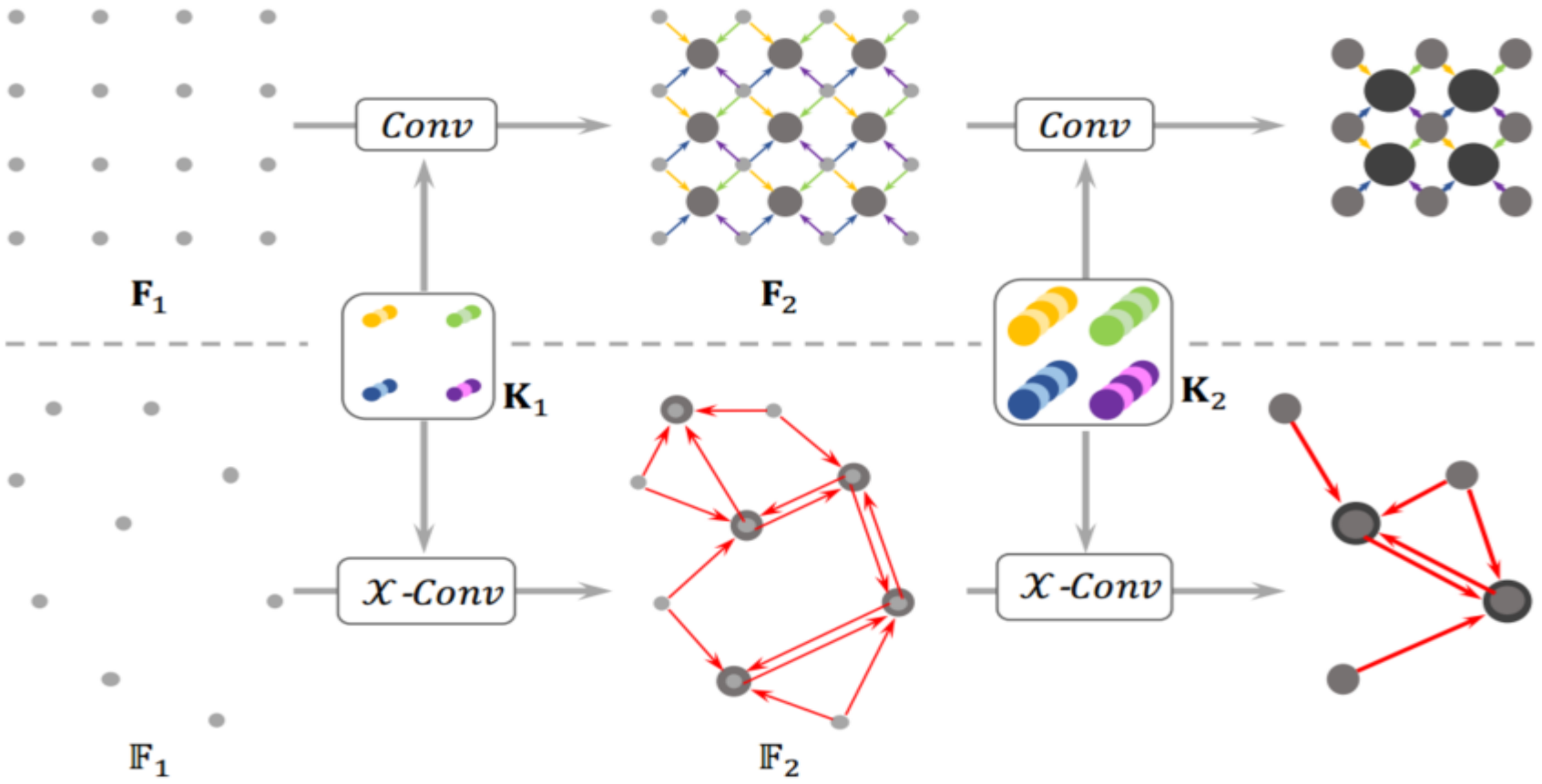
Method	Overall Accuracy (%)	mean IoU (%)
PointNet[22]	78.62	47.71
SegCloud[31]	-	48.92
SPGraph[12]	85.5	62.1
PointCNN[13]	-	62.74
Ours	**88.72**	**70.23**

1 计算量大

2 复现困难？

PointCNN

这篇文章提出了直接用于点云数据的卷积运算方法，使用了这种类似于二维的卷积算子，然后整个网络类似于Unet，达到了高于PointNet++的效果。



PointCNN：卷积直接使用的缺陷

1 结构信息，希望2和3应该是不同的特征矩阵
2 排序不变性，希望3和4是相同的特征矩阵

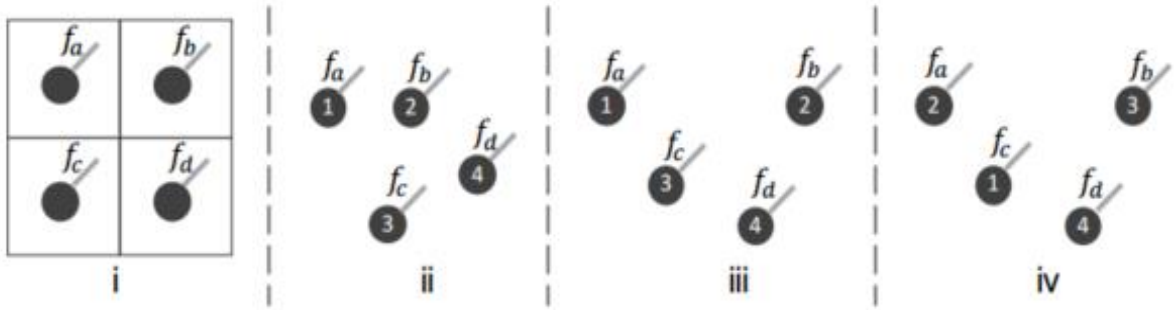


Figure 1: Convolution input from regular grids (i) and point clouds (ii-iv). In (i), each grid cell is associated with a feature. In (ii-iv), the points are sampled from local neighborhoods, in analogy to local patches in (i), and each point is associated with a feature, an order index, and coordinates.

$$
\begin{aligned}
f_{ii} &= Conv(\mathbf{K}, [f_a, f_b, f_c, f_d]^T), \\
f_{iii} &= Conv(\mathbf{K}, [f_a, f_b, f_c, f_d]^T), \\
f_{iv} &= Conv(\mathbf{K}, [f_c, f_a, f_b, f_d]^T).
\end{aligned} \tag{1a}
$$

$$
\begin{aligned}
f_{ii} &= Conv(\mathbf{K}, \mathcal{X}_{ii} \times [f_a, f_b, f_c, f_d]^T), \\
f_{iii} &= Conv(\mathbf{K}, \mathcal{X}_{iii} \times [f_a, f_b, f_c, f_d]^T), \\
f_{iv} &= Conv(\mathbf{K}, \mathcal{X}_{iv} \times [f_c, f_a, f_b, f_d]^T).
\end{aligned} \tag{1b}
$$

PointCNN：解决方法，学习变换矩阵

ALGORITHM 1: $\mathcal{X}$-Conv Operator

Input : $\mathbf{K}, p, \mathbf{P}, \mathbf{F}$
Output : $\mathbf{F}_p$

1: $\mathbf{P}' \leftarrow \mathbf{P} - p$
2: $\mathbf{F}_\delta \leftarrow MLP_\delta(\mathbf{P}')$
3: $\mathbf{F}_* \leftarrow [\mathbf{F}_\delta, \mathbf{F}]$
4: $\mathcal{X} \leftarrow MLP(\mathbf{P}')$
5: $\mathbf{F}_\mathcal{X} \leftarrow \mathcal{X} \times \mathbf{F}_*$
6: $\mathbf{F}_p \leftarrow \text{Conv}(\mathbf{K}, \mathbf{F}_\mathcal{X})$

输入就是找到代表点的邻域的点，然后对这些点进行处理，最终输出一个特征

首先搜索用的是随机搜索，但是对于分割任务用的是FPS，找出代表点，然后找出他的K个近邻

1 坐标变化

2 对坐标进行升维，进行更抽象的表示，来和特征向量进行匹配

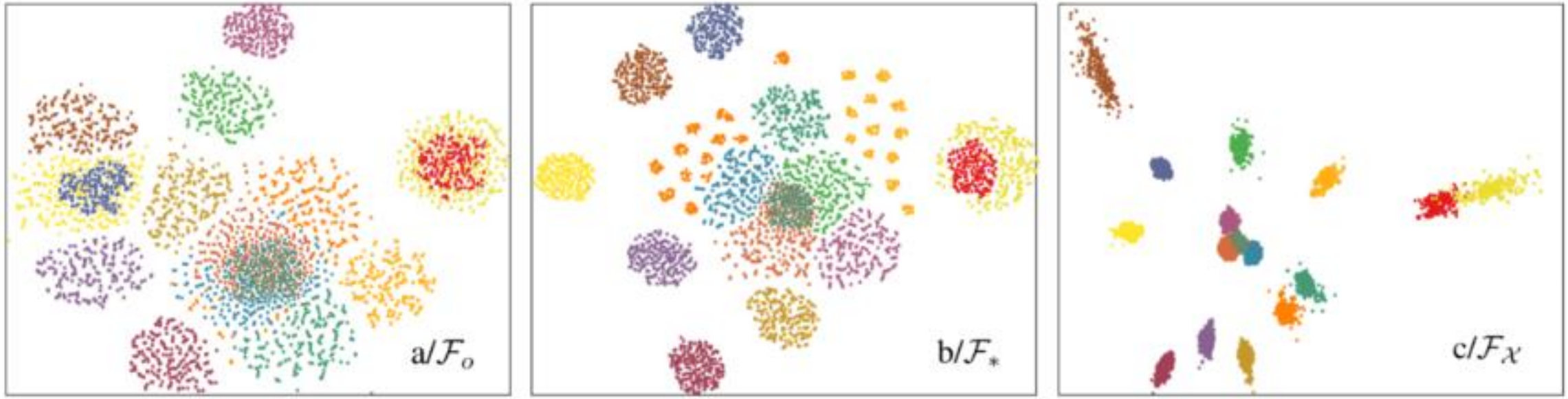
3 把升维之后的坐标矩阵和特征矩阵拼接

4 将坐标变化之后的坐标矩阵MLP,学到X变化矩阵

5 对拼接之后的特征矩阵进行X变换

6 常规的卷积，也就是加权和运算，最终生成一个特征

PointCNN：是否具有排序不变性？



1在理想情况下，对于同一个Patch，不同的排序所产生的特征矩阵应该是相同的，也就是对应图中的相同颜色的点聚集成一个

2第三张图代表经过了变换的特征，他们高度相似，距离很近，证明了排序不变性

PointCNN：亮点和问题

亮点：
1 这个和PointNet相比，提出了不同的解决排序不变性的方法，也就是学习一个矩阵，作者说Pooling的方式会丢掉一些信息，这样做更好，同时参数也会减少

2更高的精度， 更快的速度（对比PointNet++）

Methods		PointNet [33]	PointNet++ [35]	3DmFV-Net [4]	DGCNN [50]	SpecGCN [46]	PCNN [3]	PointCNN
Parameters		3.48M	1.48M	45.77M	1.84M	2.05M	8.2M	**0.6M**
FLOPs	Training	43.82B	67.94B	48.57B	131.37B	49.97B	**6.49B**	93.03B
	Inference	14.70B	26.94B	16.89B	44.27B	17.79B	**4.70B**	25.30B
Time	Training	0.068s	0.091s	0.101s	0.171s	14.640s	0.476s	**0.031s**
	Inference	0.015s	0.027s	0.039s	0.064s	11.254s	0.226s	**0.012s**

Table 6: Parameter number, FLOPs and running time comparisons.

问题：
1 作者也说，对于排序不变性没有解决的很好

2 搜索的时候也可能漏掉点？因为PointNet++用的搜索方式和这个差不多，所以这个也可能漏点

问题和展望

问题：

1没有跟上最新的研究进展

2实验做得少，代码写得少（难以深入了解）

展望：

1 按照LeaderBoard的顺序看新论文

2 看代码，发现问题并改进（e.g.网络每个部分是否达到了理想的效果，点云哪些地方分割的不好）