Point Cloud Classification with PointHop and PointHop++

C.-C. Jay Kuo
University of Southern California

Point Cloud Set

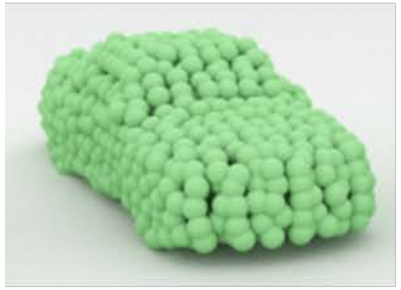
- Set of points in 3D, acquired by 3D scanning devices such as Lidar
- Applications: AR/VR, self-driving cars, robots, 3D CAD modelling etc.

Applications and Challenges

- 3D computer vision tasks: classification, segmentation and detection, etc.
- Challenges: points are irregular and unordered distributed in 3D space

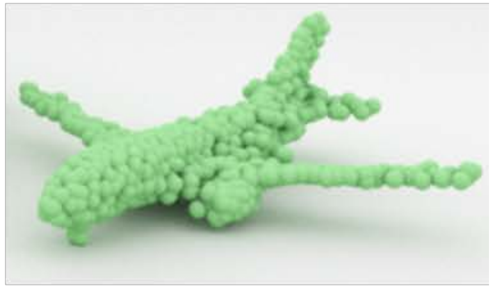
Point Cloud Processing

- Classification



CAR

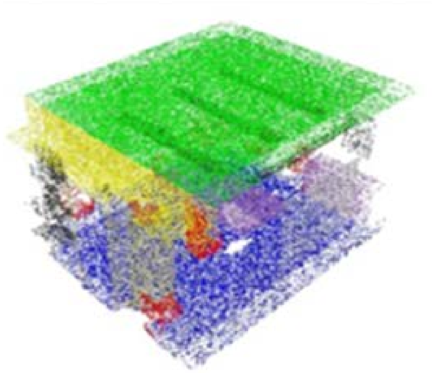
TABLE



AIRPLANE

- Segmentation

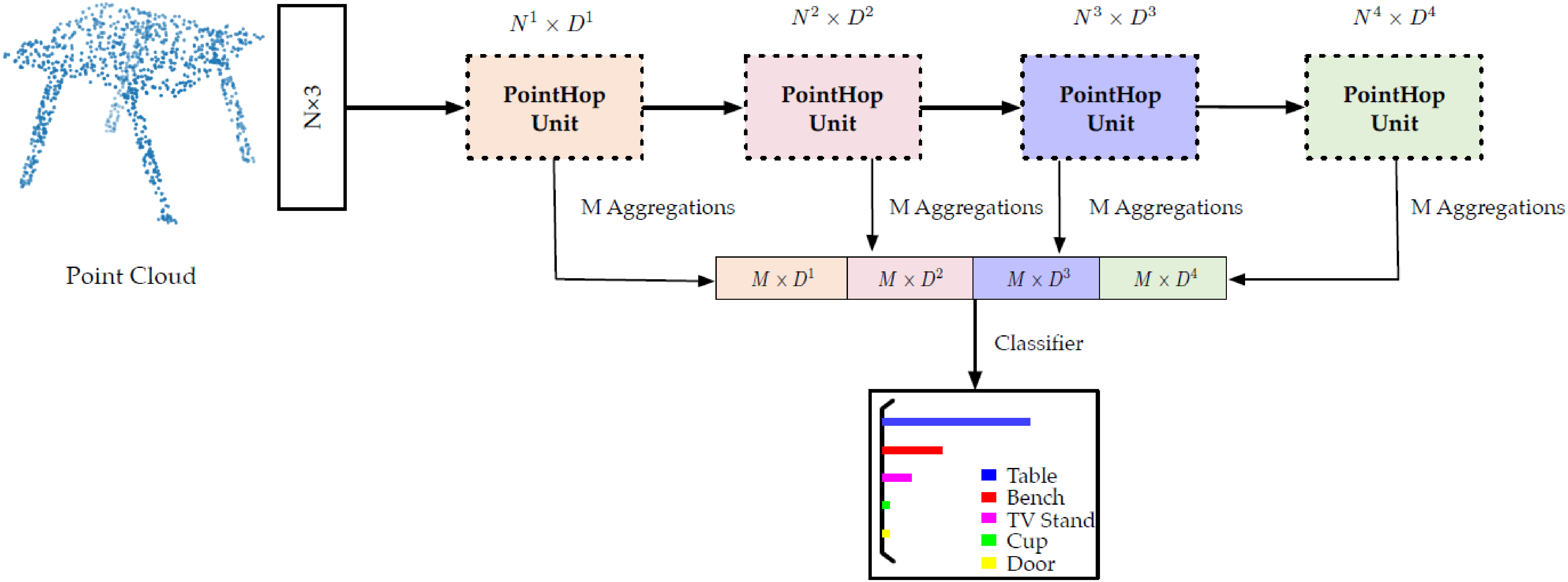
AIRPLANE



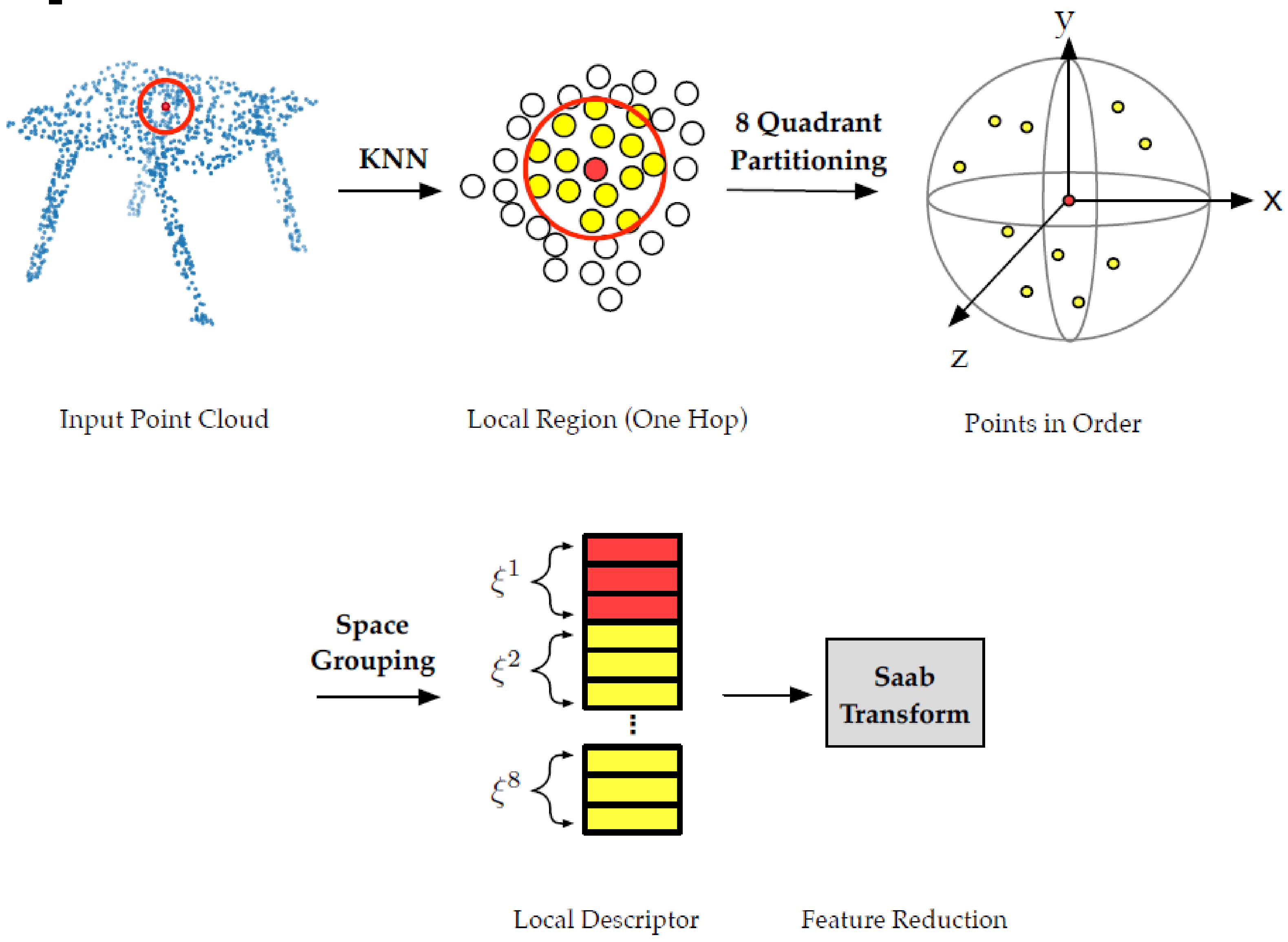
BUILDING

PointHop Method

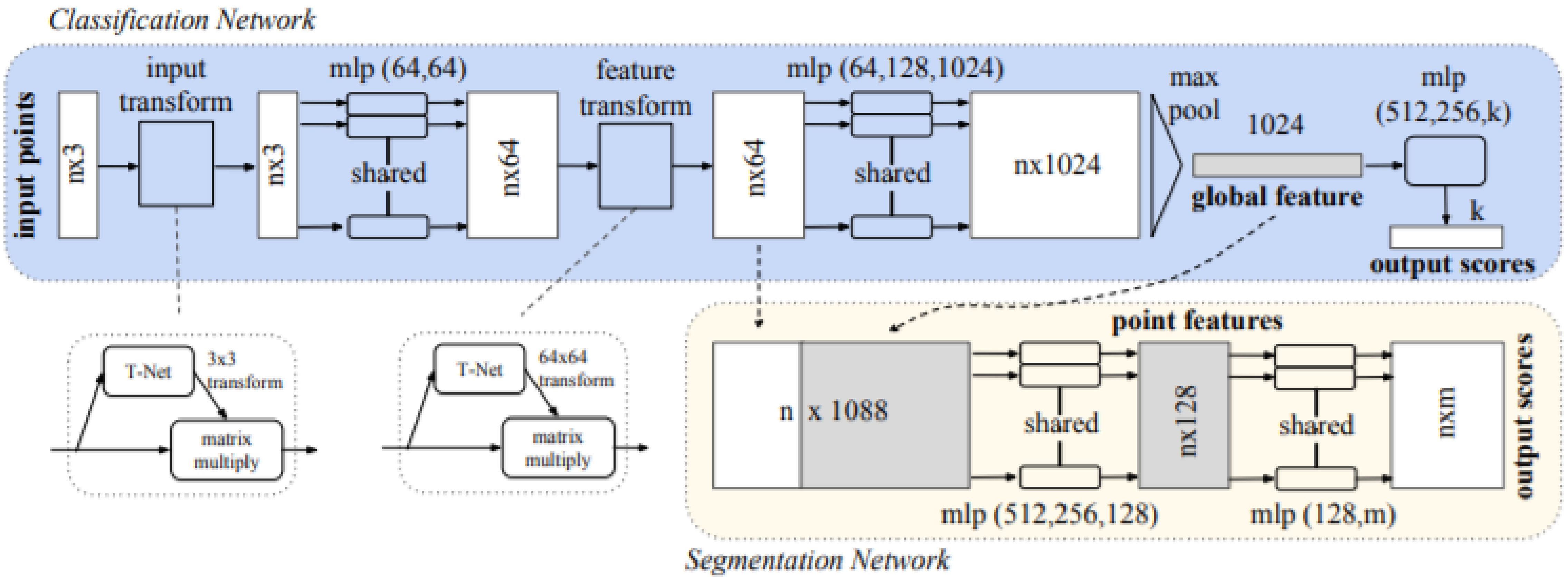
PointHop System



PointHop Unit

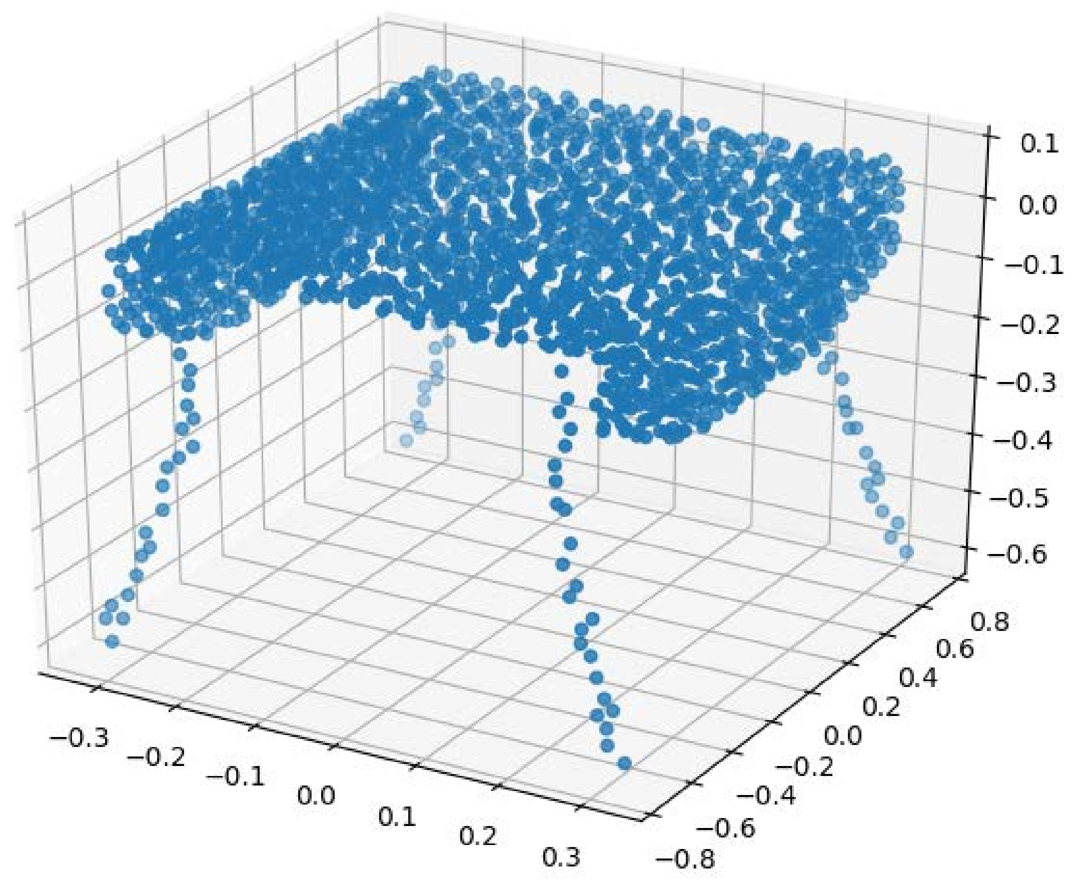


PointNet Architecture



Dataset – ModelNet40

- 40 categories of objects such as airplane, table, desk, sofa
- 9840 training samples, 2468 testing samples
- Every sample has 2468 points
- Every point has 3 coordinates



Performance Comparison

Dataset: ModelNet40

Method	Feature extraction	Average accuracy (%)	Overall accuracy (%)
PointNet [16]	Supervised	86.2	89.2
PointNet++ [17]		-	90.7
PointCNN [34]		88.1	92.2
DGCNN [18]		90.2	92.2
PointNet baseline (Handcraft, MLP)	Unsupervised	72.6	77.4
PointHop (baseline)		83.3	88.65
PointHop		84.4	**89.1**

Training Time Comparison

GPU platform: NVIDIA GeForce GTX 1080
CPU platform: Intel Xeon CPU E5-2620 v3 at 2.40GHz

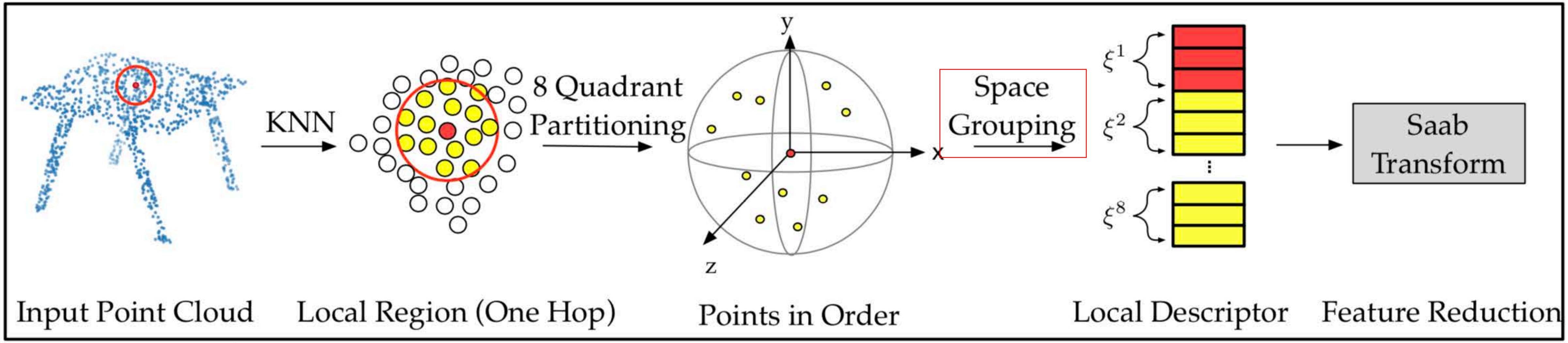
Method	Total training time	Device
PointNet (1,024 points)	$\sim$ 5 hours	GPU
PointHop (256 points)	$\sim$ 5 minutes	CPU
PointHop (1,024 points)	$\sim$ 20 minutes	CPU

PointHop++ Method

Problem Statement

- Aspects of development
  - reducing its model complexity
  - ordering discriminant features automatically based on the cross-entropy criterion

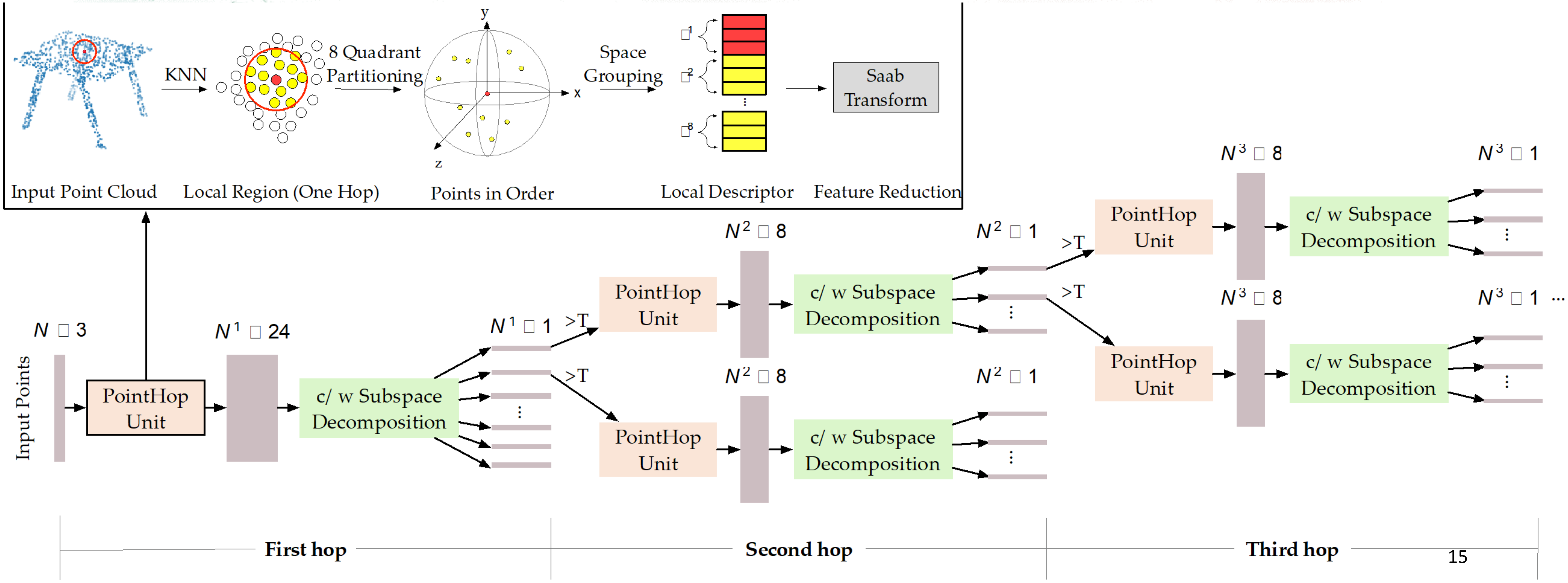
Initial Feature Space Construction



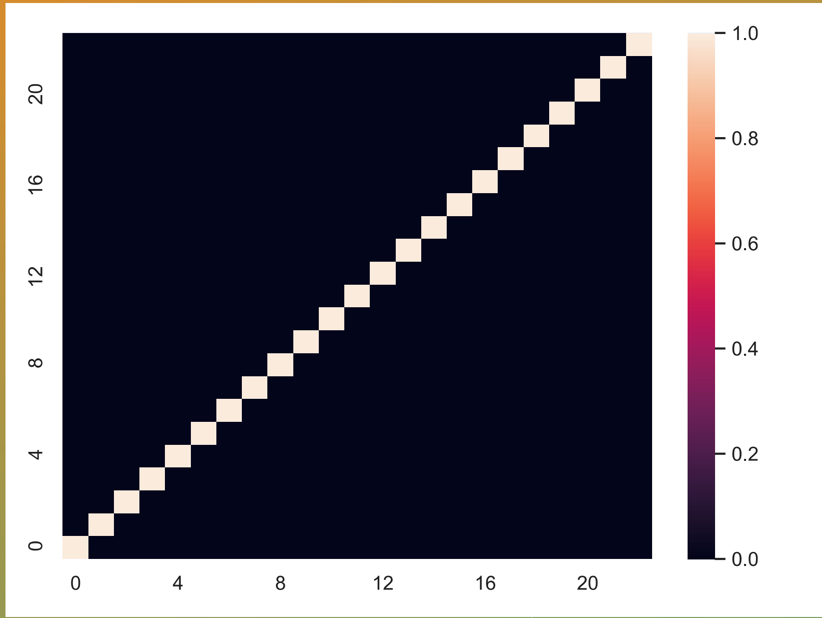
$$g : \underbrace{\mathbb{R}^D \times \cdots \mathbb{R}^D}_{k} \rightarrow \underbrace{\mathbb{R}^D \times \cdots \mathbb{R}^D}_{8}$$

Where D=3 for the first hop and D=1 for the remaining hops.

PointHop++ Architecture



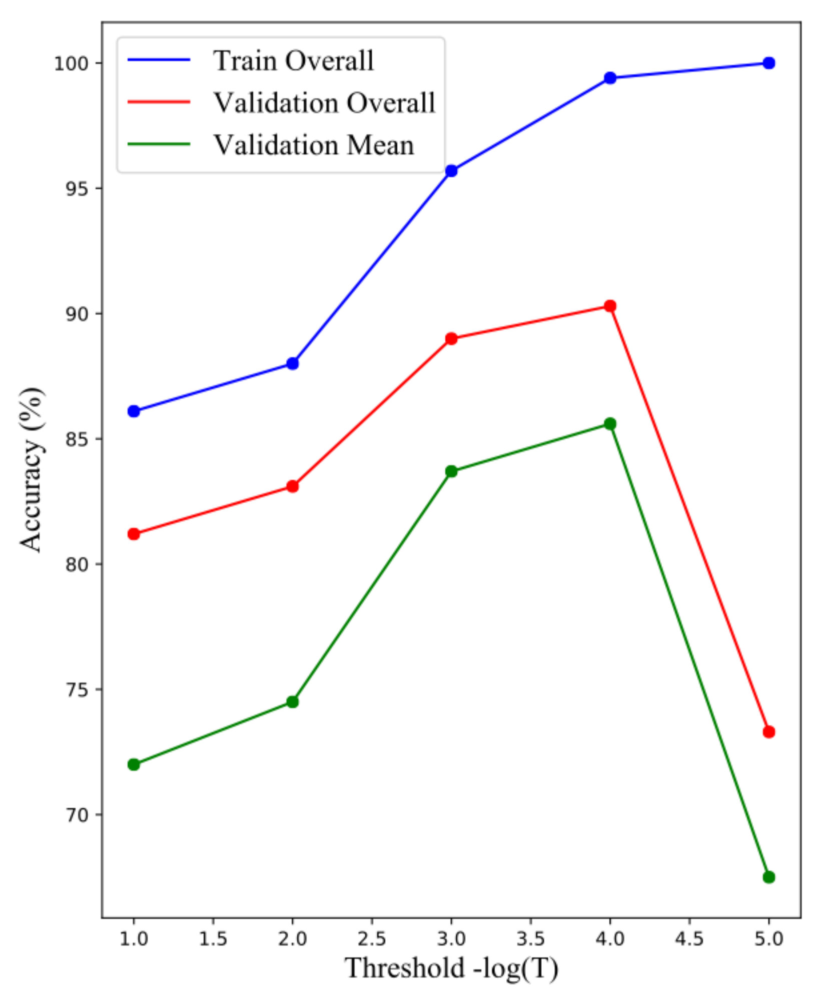
Channel-Wise (C/W) Subspace Decomposition



- The input to Saab transform[4] is $A = [a^1, \cdots, a^N]^T \in \mathbb{R}^{N\times 8D}$, where a^n is the 8D attribute vector of point p_n
- The filter weight is $W = [w_1, w_2, \cdots, w_{8D}] \in \mathbb{R}^{8D\times 8D}$, where $w_1 = \frac{1}{\sqrt{8D}}[1, 1, \cdots, 1]^T$, others are eigenvectors of covariance matrix A ranked by λ_i
- The output of Saab transform is $B = A \cdot W = [b_1, \cdots, b_{8D}]$, where $b_i \in \mathbb{R}^{N\times 1}, i = 1, \cdots, 8D$
- The correlation between Saab coefficients of different channels is:

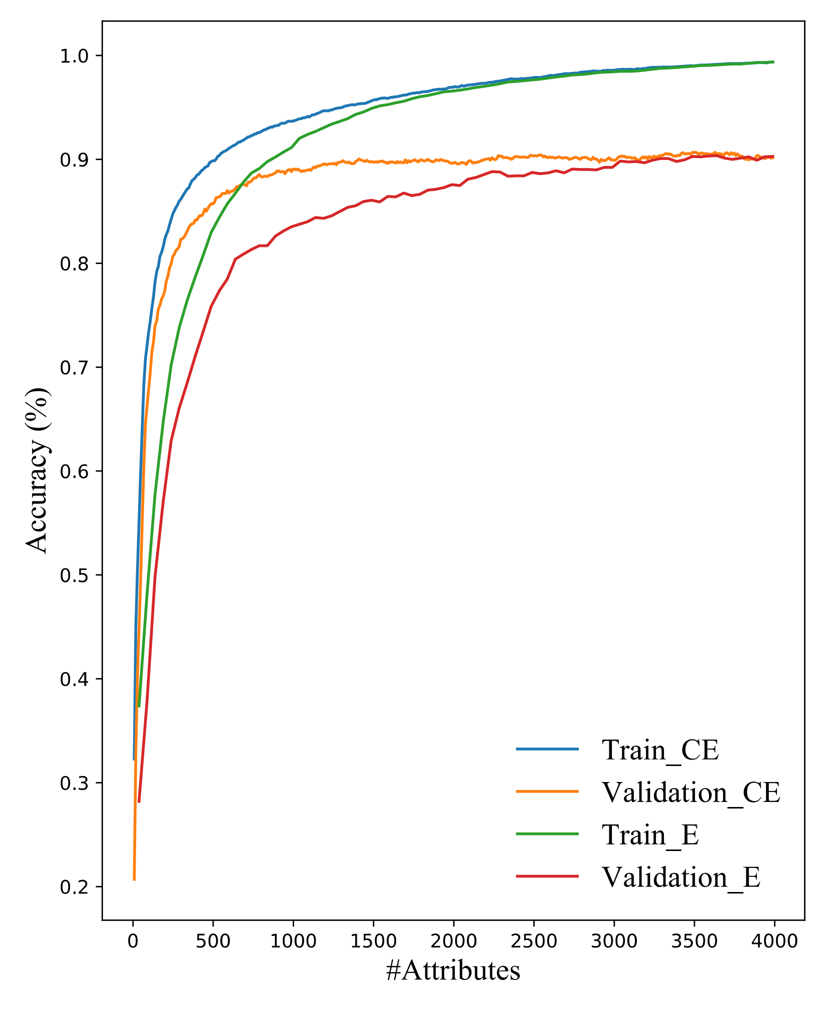
$$Cor(b_i, b_j) = \frac{1}{N}(A\cdot w_i)^T(A \cdot w_j) = \frac{1}{N}(\lambda_i w_i)^T(\lambda_j w_j) = 0, where\ i \neq j$$

4. C. –C. Jay Kuo, Min Zhang, etc, "Interpretable Convolutional Neural Networks via Feedforward Design." JVCIR (2019).



Channel Split Termination

- The energy of each subspace is $E_i = E_p \times \frac{\lambda_i}{\sum_{j=1}^{8D} \lambda_j}$, where $\mathrm{i} = 1, \cdots, 8D$ and E_p is the energy of its parent node
- If the energy of a node is less than a pre-set threshold, T, we terminate its further split and keep it as a leaf node of the feature tree at the current hop. Other nodes will proceed to the next hop.
- All leaf nodes are collected as the feature representation after the feature tree construction is completed.



Feature Priority Ordering

- A feature is more discriminant if its cross entropy is lower.
- Cross entropy calculation:
  1. partition the 1D subspace into J intervals
  2. Use majority vote to predict label for each interval
  3. Calculate the probability of each sample that belongs to a class
  4. Calculate the cross entropy, $L = \sum_{j=1}^{J} L_j$, $L_j = -\sum_{c=1}^{M} y_{j,c} \log(p_{j,c})$, where M is the class number, $y_{j,c}$ is binary indicator to show whether sample j is correctly classified, and $p_{j,c}$ is the probability that sample j belongs to class c.

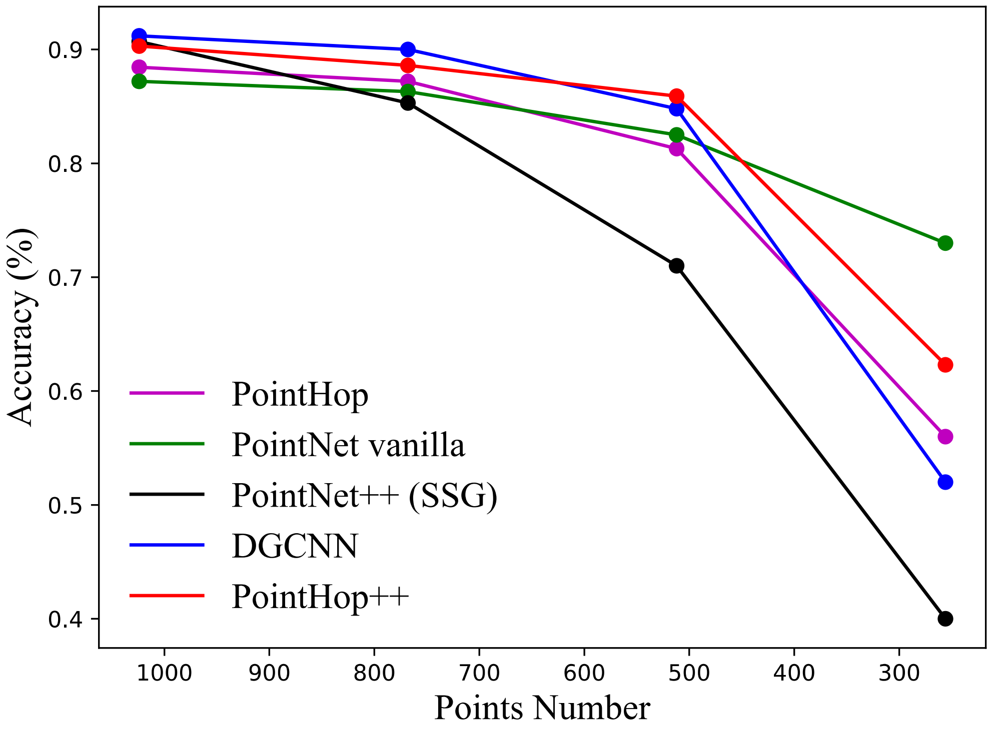
Experimental Results

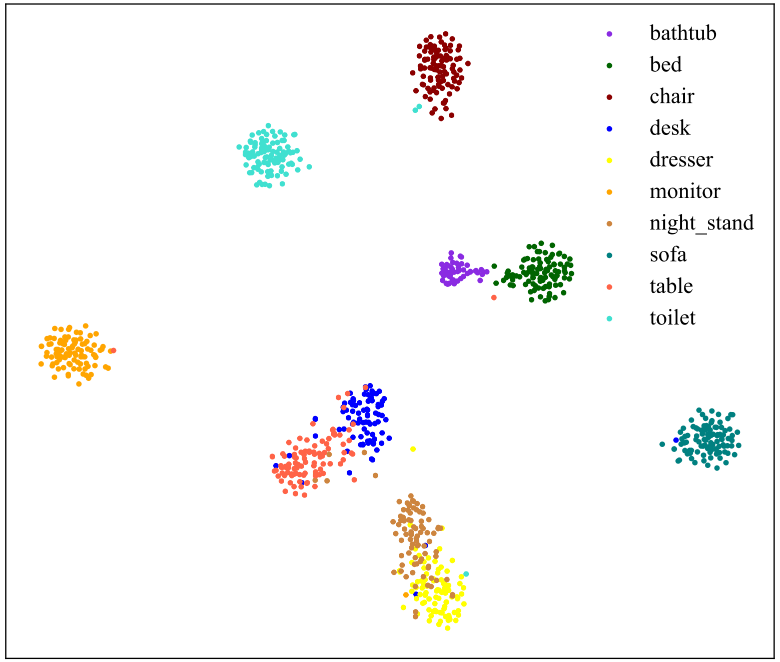
	Method	Accuracy (%)	
		class-avg	overall
Supervised	PointNet [10]	86.2	89.2
	PointNet++ [11]	-	90.7
	PointCNN [12]	88.1	**92.2**
	DGCNN [13]	**90.2**	**92.2**
Unsupervised	LFD-GAN [28]	-	85.7
	FoldingNet [29]	-	88.4
	PointHop [15]	84.4	89.1
	PointHop++ (baseline)	85.6	90.3
	PointHop++ (FS)	86.5	90.8
	PointHop++ (FS+ES)	**87**	**91.1**

Table 1. Comparison of classification results on ModelNet40, where the class-Avg accuracy is the mean of the per-class accuracy, and FS and ES mean "feature selection" and "ensemble", respectively.

Method	Time		Parameter No. (MB)		
	Training	Inference	Filter	Classifier	Total
PointNet [10]	7	10	-	-	3.48
PointNet++ [11]	7	14	-	-	1.48
DGCNN [13]	21	154	-	-	1.84
PointHop [15]	0.33	108	0.037	-	-
PointHop++	0.42	97	0.009	0.15	0.159

Table 2. Comparison of time and model complexity, where the training and inference time units are in hour and ms, respectively.





Experimental Results

- Robustness to sampling density variation
- Feature visualization

Conclusion

A tree-structured unsupervised feature learning system was proposed in this work, where one scalar feature is associated with each leaf node and features are ordered based on their discriminant power.

The resulting PointHop++ method achieves state-of-the-art classification performance while demanding a significantly small learning model which is ideal for mobile computing.