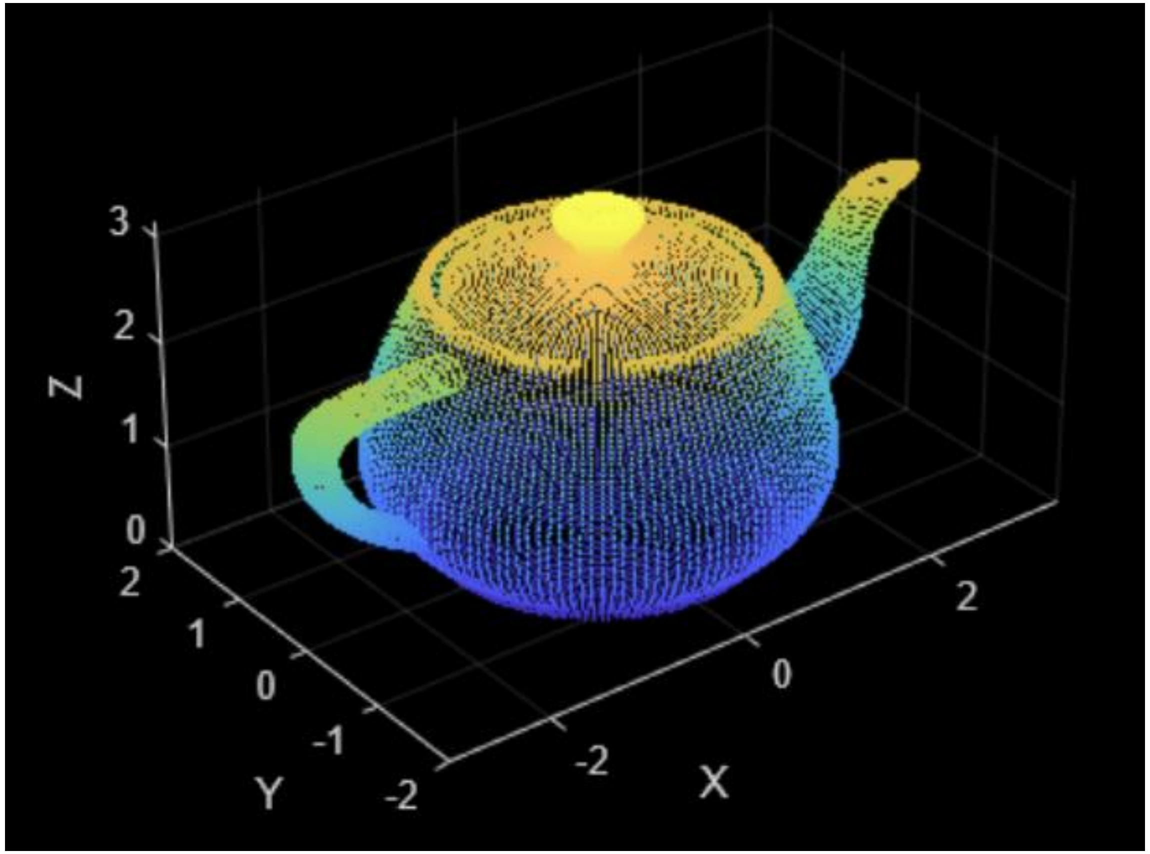
What is Point Cloud?



A point cloud:
A collection of data points defined in a three-dimensional coordinate system (X, Y, Z).

Irregular Format:
Point clouds are unordered, have variable point counts, and exhibit non-uniform distribution.

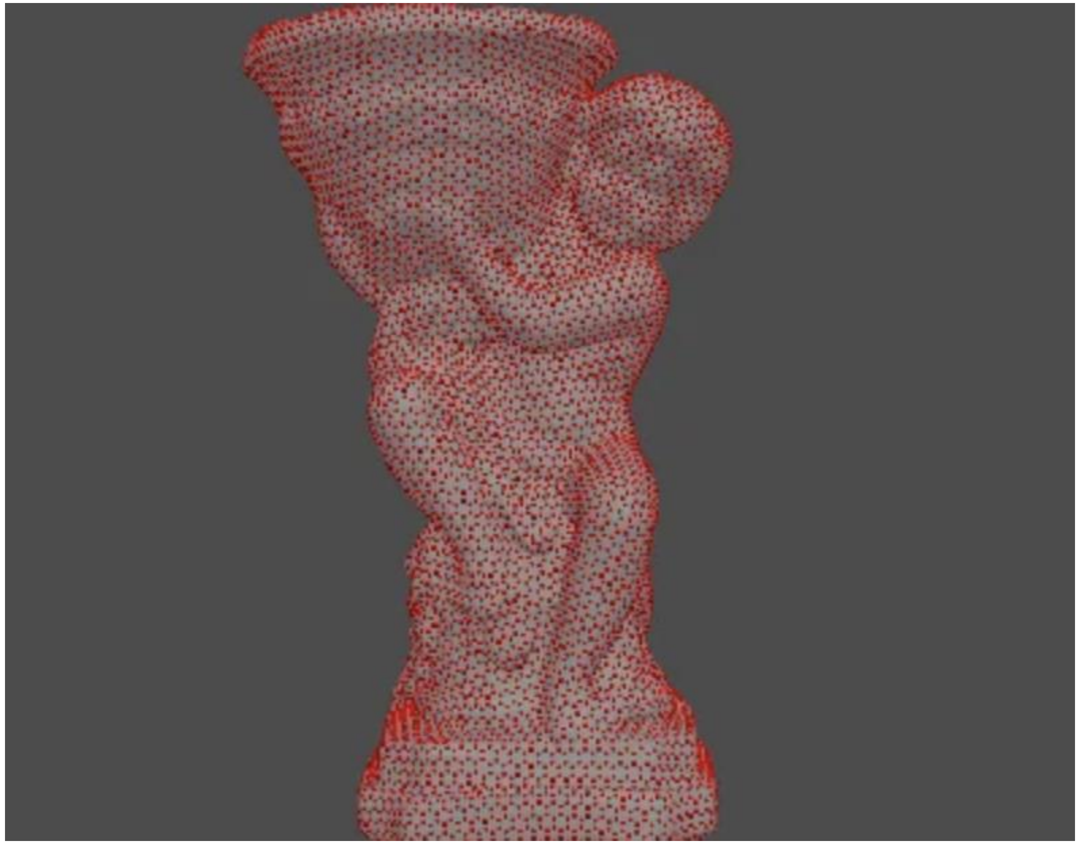
Traditional CNN Approach for Point Clouds:
Point Cloud → Voxelization/Image Projection

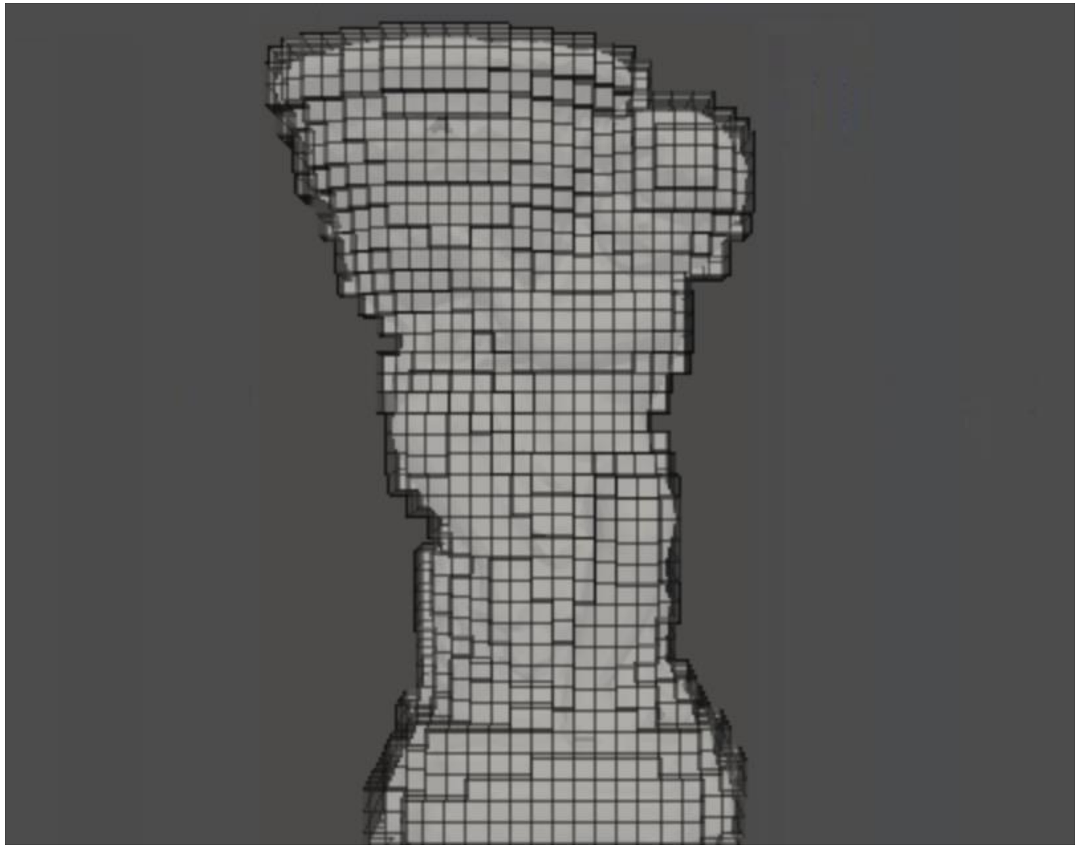
but with 2 major issues…

1. Data Explosion
- Massive memory
- 64^3 grid = 262K voxels
- Computational burden

2. Information Loss
- Quantization errors
- Loss of fine details
- Geometric invariance

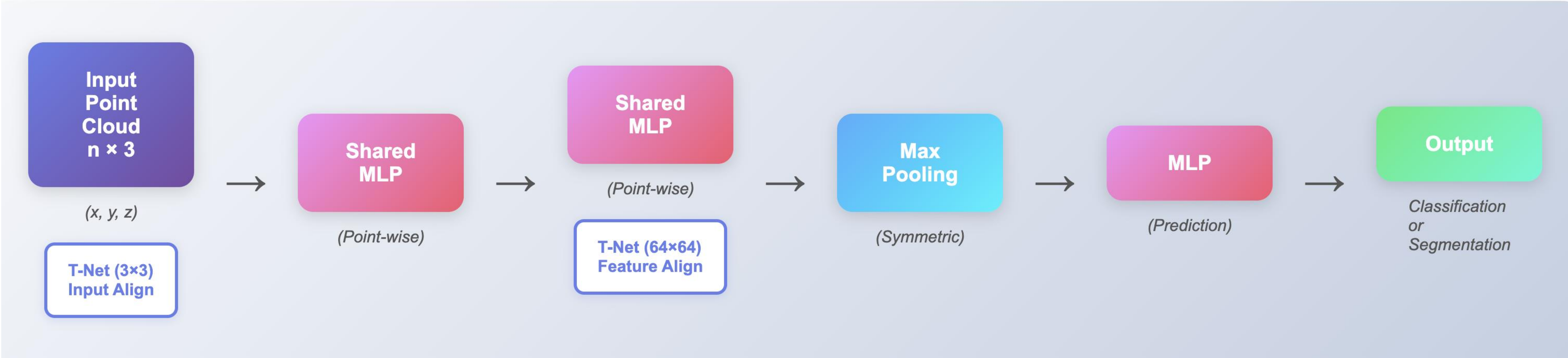
Point Cloud → **3D Voxel Grids**





Point Cloud → **Collections of images**

PointNet Architecture



Key ideas:

1. **Point-wise:** Shared MLP for each point
2. **Order-free:** Max pooling makes the model permutation invariant
3. **Geometry-aware:** T-Net learns point cloud alignment automatically

Point Cloud Features

Most features were **handcrafted** for specific tasks.

- Features are either:
  - **Intrinsic** (shape-based, rotation-invariant)
  - **Extrinsic** (coordinate-based, change with rotation)
- Also divided into:
  - **Local features** (neighborhood shape)
  - **Global features** (overall object shape)

So it's hard to find the **best feature combination** for each task.

PointNet learns features directly from raw points, without handcrafting.

Deep Learning on 3D Data

Approach	Key Papers	Method	Pros	Cons	Accuracy
Volumetric CNN	• 3DShapeNets • VoxNet • FPNN	Voxelize → 3D Conv	1. Mature 3D conv 2. Structured	1. Resolution limited 2. Memory intensive 3. Sparse but dense storage	77-86%
Multi-view CNN	• MVCNN • Vol+MV	Render → 2D Conv	1. Leverage 2D CNNs 2. Best performance	1. Rendering overhead 2. View selection 3. Hard to extend to segmentation	90.1%
Spectral CNN	• Spectral • Geodesic	Graph conv on mesh	Geometric operations	1. Only manifold meshes 2. Organic shapes only	N/A
Feature-based DNN	• 3D Deep Shape Descriptor • 3D Mesh Labeling	Features → MLP	Simple pipeline	1. Handcrafted features 2. Not end-to-end	Lower

Deep Learning on Unordered Sets

Point clouds = **unordered sets**, rarely studied in deep learning.

Prior work (Vinyals et al.) handles general sets but **no geometry**.

PointNet is the first to handle unordered geometric point sets.

Problem Statement

Point Cloud

$\{P_i \in (x, y, z) \mid (x, y, z) \in \mathbb{R}^3, \mathrm{i} = 1, \ldots, \mathrm{n}\}$

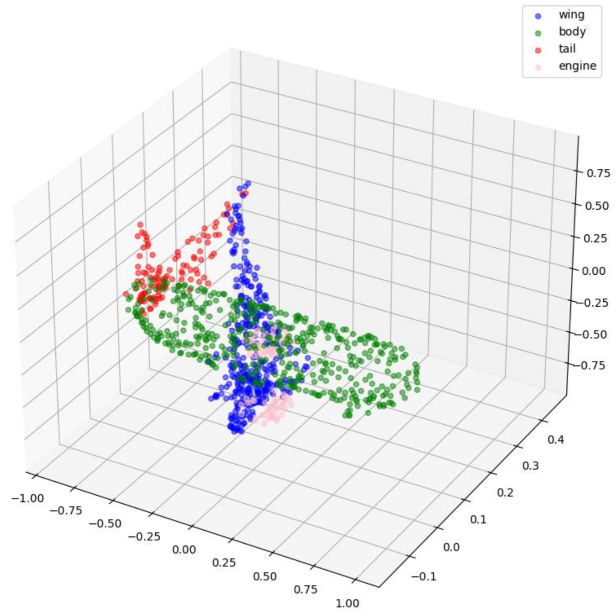
⇩

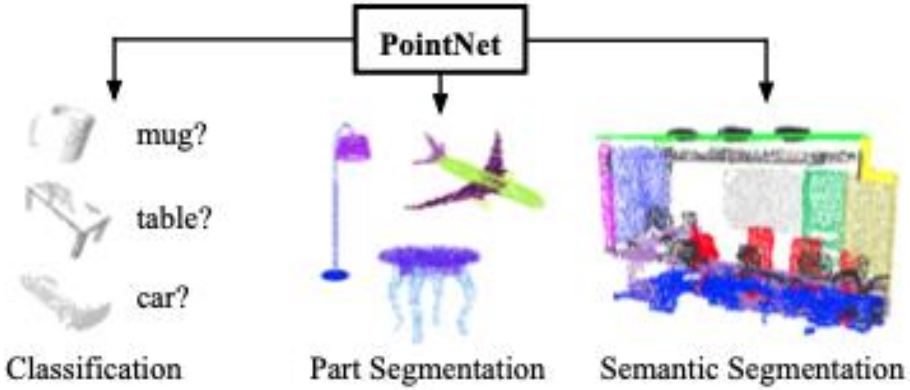
PointNet

⇩ ⇩

Classification (k x 1)

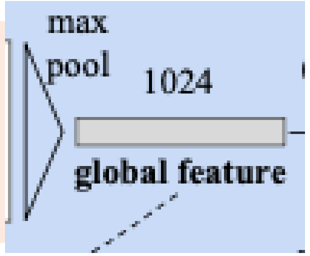
Part Segmentation /Semantic Segmentation (n x m)





Deep Learning on Point Sets

Unordered
(N! Permutation)



Symmetry Function

$$f(\{x_1, \ldots, x_n\}) \approx g\big(h(x_1), \ldots, h(x_n)\big)$$

$$g \coloneqq \prod_{k=1}^{n} \mathbb{R}^K \rightarrow \mathbb{R}$$

Locality
(Interaction among points)

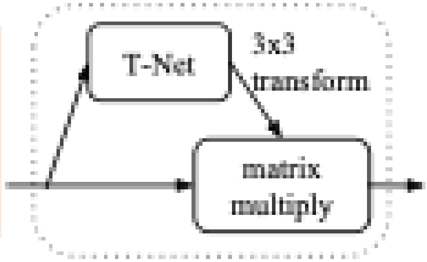
Local and Global Information Aggregation

$$f(G, L) \approx Concat(G, L, axis = 1)$$

$$G \in= \mathbb{R}^{(n\times K_1)}, L \in= \mathbb{R}^{(n\times K_2)}$$

$$f(G, L) \in= \mathbb{R}^{(n\times (K_1+K_2))}$$

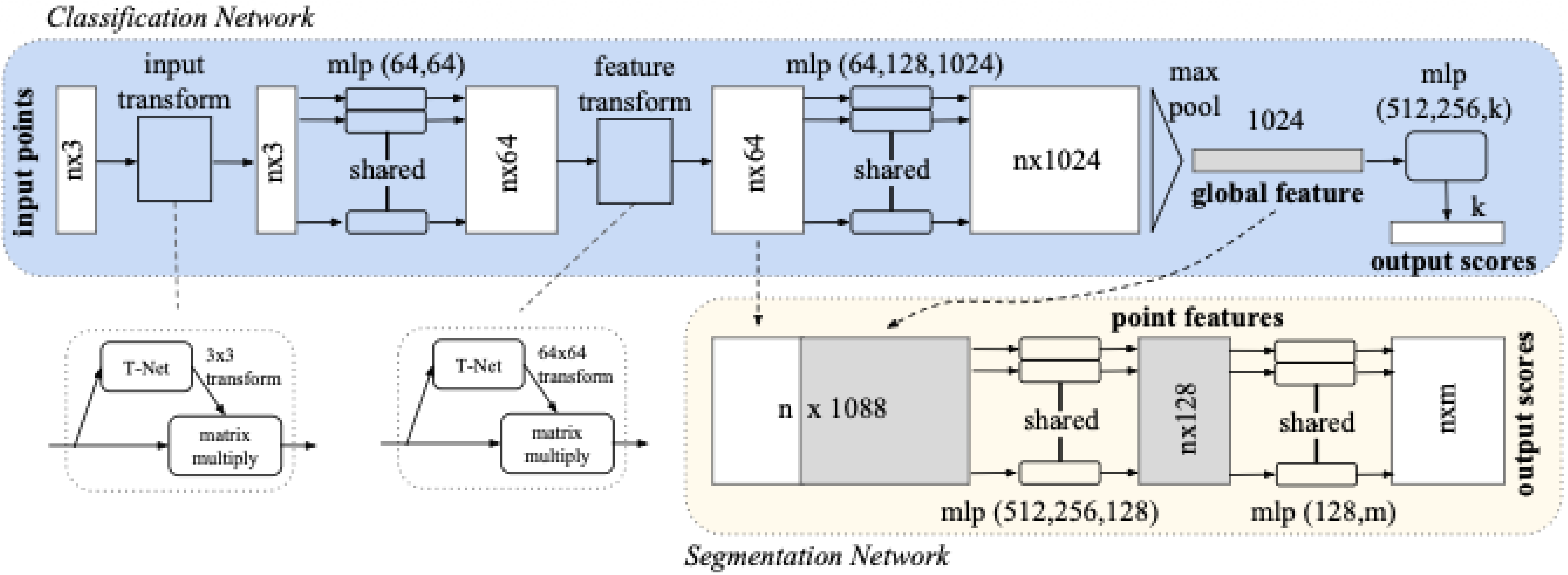
Invariance under transformations



Joint Alignment Network

$$\begin{matrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{matrix} \times T \qquad T \in= \mathbb{R}^{(3\times 3)}$$

Deep Learning on Point Sets



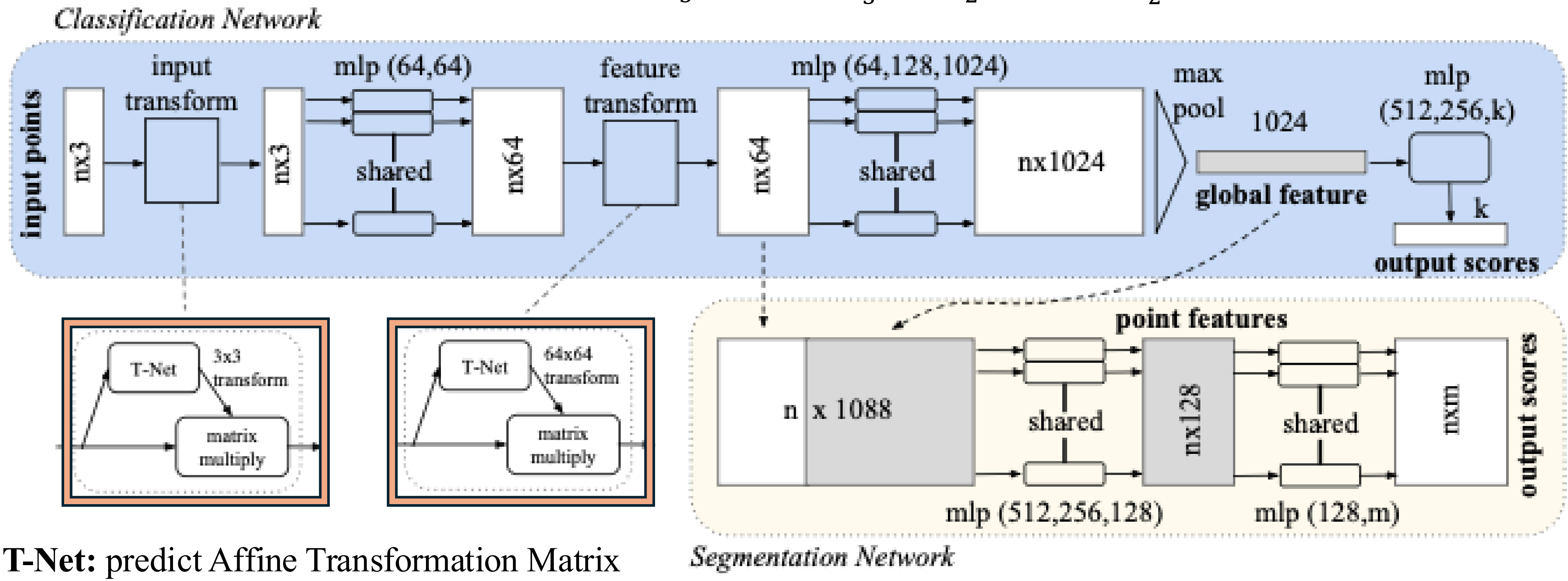
Deep Learning on Point Sets

***N*! *Permutation*: Row-wise Independent**

Invariance under transformations

Input/Feature Transform

$$\begin{matrix} P_1{}^T \\ P_2{}^T \\ P_3{}^T \end{matrix} \times A = \begin{matrix} h_1{}^T \\ h_2{}^T \\ h_3{}^T \end{matrix} \leftrightarrow \begin{matrix} P_3{}^T \\ P_1{}^T \\ P_2{}^T \end{matrix} \times A = \begin{matrix} h_3{}^T \\ h_1{}^T \\ h_2{}^T \end{matrix} \qquad L_{reg} = \|\mathbb{I} - AA^T\|_F^2$$



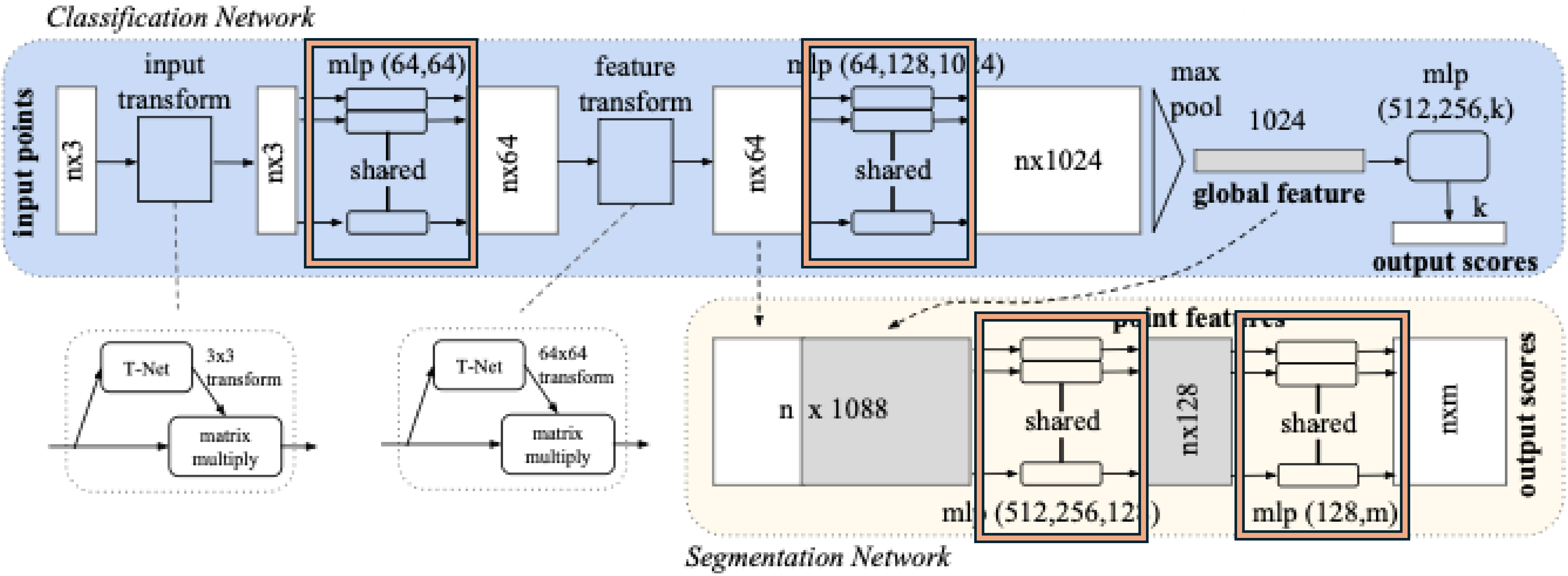
T-Net: predict Affine Transformation Matrix

Deep Learning on Point Sets

$N!$ *Permutation*: Row-wise Independent

Shared Weight MLP ~ 1D Convolution

$$\begin{matrix} P_1{}^T \\ P_2{}^T \\ P_3{}^T \end{matrix} \times \begin{matrix} w_{11} & \cdots & w_{k1} \\ w_{12} & \cdots & w_{k2} \\ w_{13} & \cdots & w_{k3} \end{matrix} = \begin{matrix} y_1{}^T \\ y_2{}^T \\ y_3{}^T \end{matrix} \qquad \begin{matrix} P_3{}^T \\ P_1{}^T \\ P_2{}^T \end{matrix} \times \begin{matrix} w_{11} & \cdots & w_{k1} \\ w_{12} & \cdots & w_{k2} \\ w_{13} & \cdots & w_{k3} \end{matrix} = \begin{matrix} y_3{}^T \\ y_1{}^T \\ y_2{}^T \end{matrix}$$



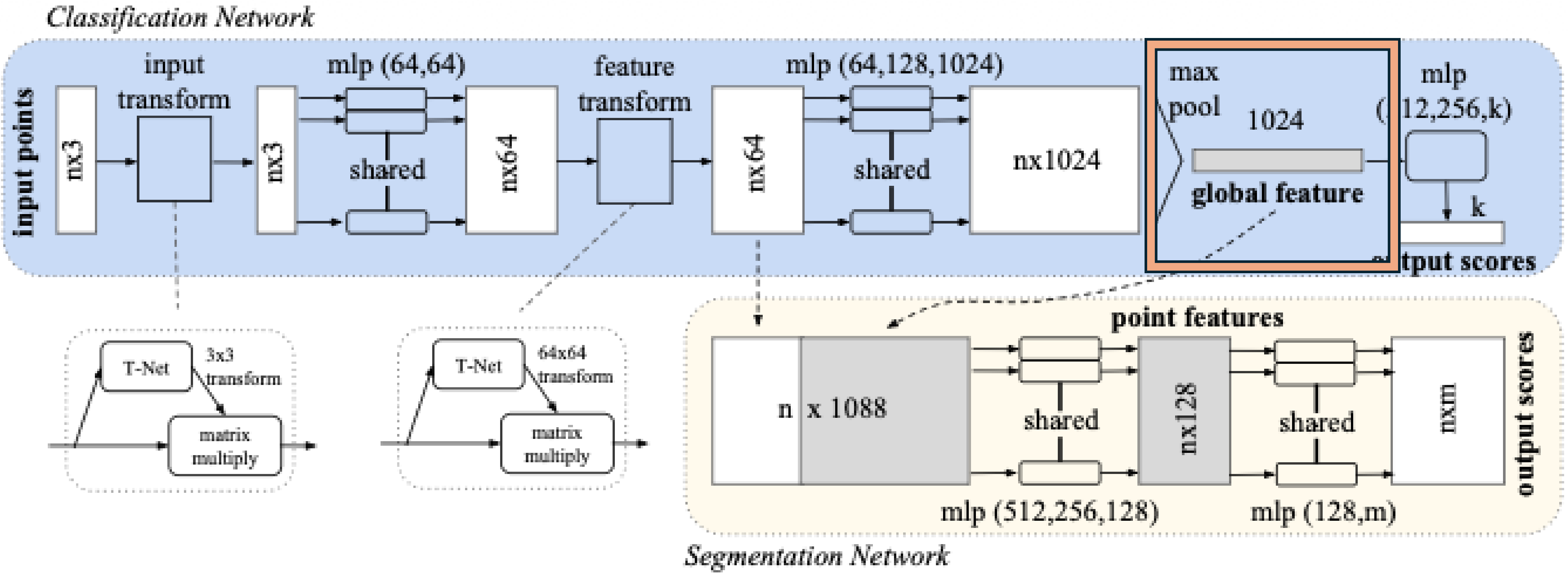
Deep Learning on Point Sets

***Classification*:**
Symmetry Function (Unorder)

MaxPooling (axis = 0)

$$maxpool(P_1, P_2, P_3) = maxpool(P_3, P_1, P_2)$$

Universal approximation & Stability

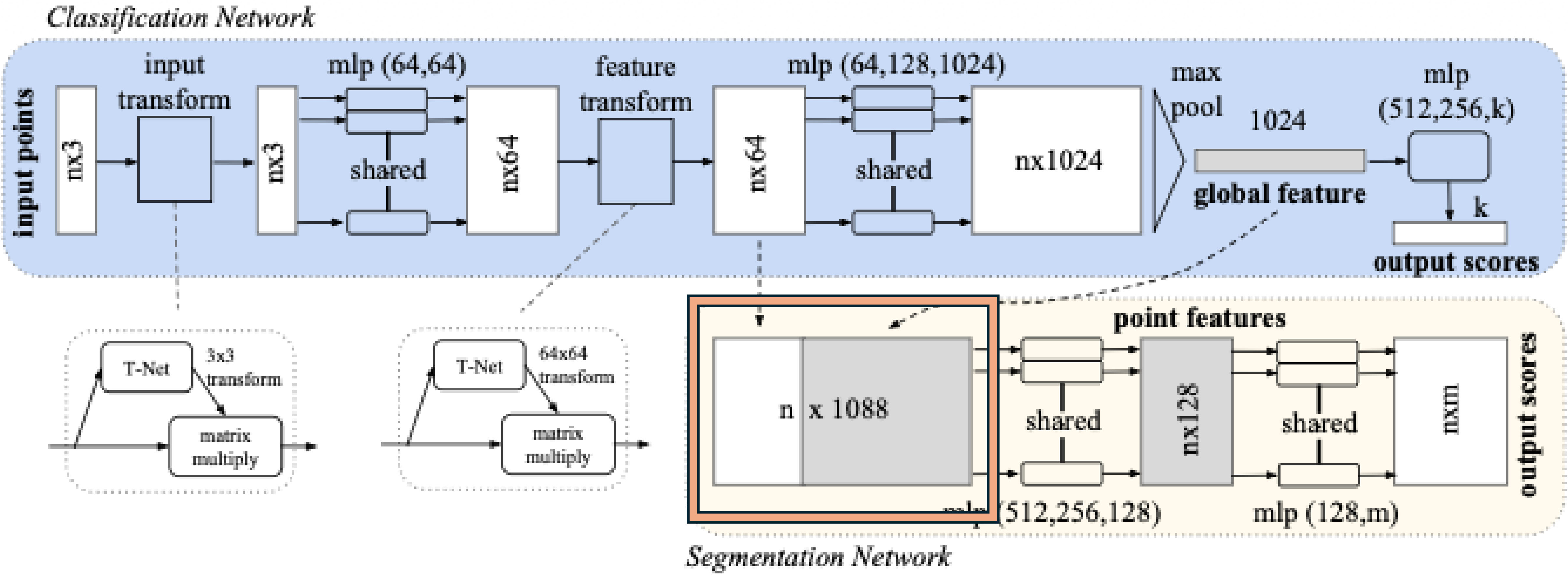


Deep Learning on Point Sets

***Segmentation*:**
Interaction among Points

Concatenate (axis = 1)

Combination of Local Geometry and Global Semantic

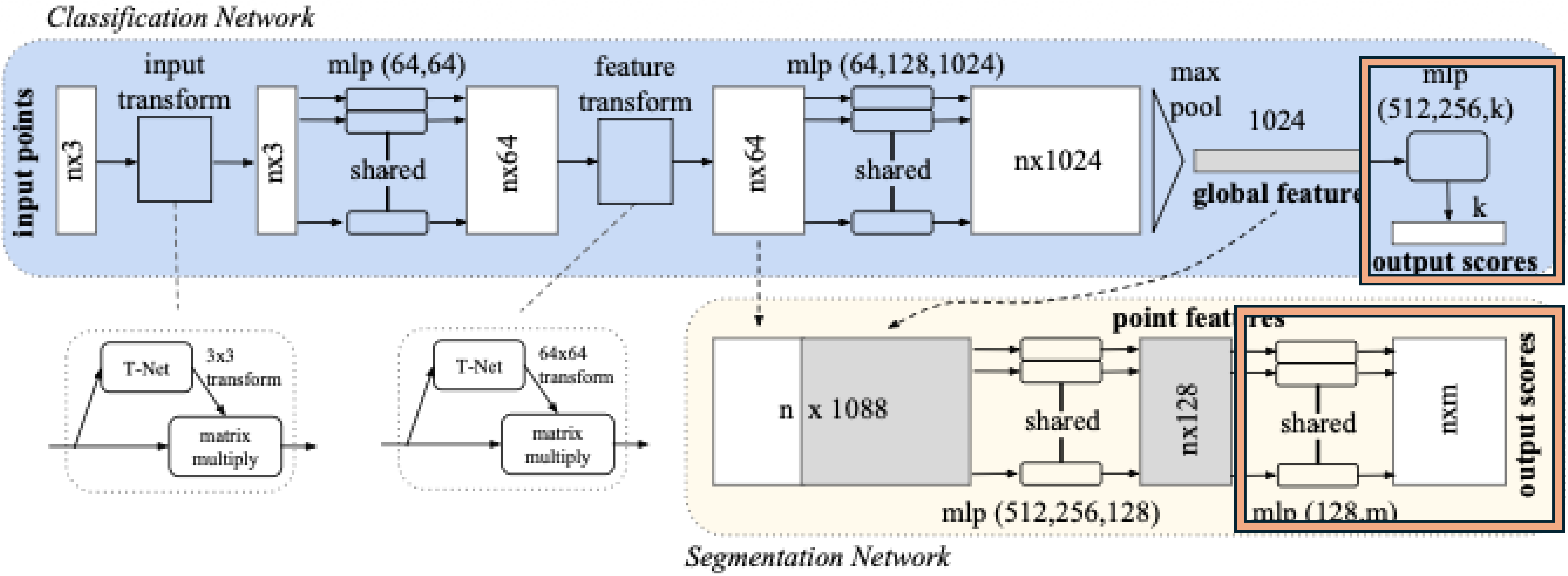


Deep Learning on Point Sets

Output of PointNet

Classification: (k x 1) k class label

Segmentation: (n x m) m class(part) label for every points



Supplementary

Universal Approximation

$$\gamma(\max_{x_i \in S}\{h(x_i)\})$$

Input/Feature Transform

Shared Weight MLP ~ 1D Convolution

MaxPooling (axis = 0)

MLP

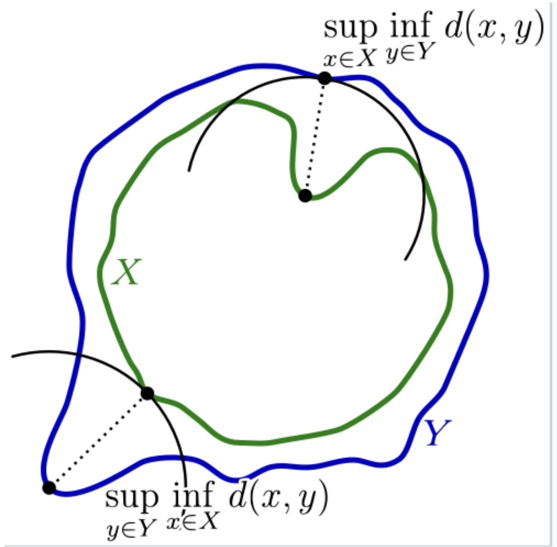
$$f: \chi \rightarrow \mathbb{R}$$
$$\chi = \{S | S \subseteq [0,1]^3, |S| = n\}$$

Hausdorff distance

$$d_{\mathrm{H}}(X,Y) := \max\left\{\sup_{x\in X} d(x,Y),\ \sup_{y\in Y} d(X,y)\right\},$$

$$d_H(S,S') < \delta \Rightarrow |f(S) - f(S')| < \epsilon$$
$$\forall \epsilon > 0, \exists \delta > 0, \quad S, S' \in \chi$$

Continuity



$$\bigcup_{S\in\chi} | f(S) - \gamma(\max_{x_i \in S}\{h(x_i)\}) | < \epsilon$$

Supplementary

Universal Approximation

$$\Phi_j(x) \approx \begin{cases} 1, & x \in C_j \\ 0, & \text{otherwise} \end{cases}$$

But continuous like Dirac-Delta Function

$$h(x) = \left(\alpha\phi_1(x), \alpha\phi_2(x), \ldots, \alpha\phi_{\mathrm{k}}(x)\right) \in R^{\mathrm{k}}$$

$$u(s) = \max_{x_i \in S}\{h(x_i)\} \approx \begin{cases} \alpha, & \mathrm{S} \cap \mathrm{C_j} \neq \emptyset \\ 0, & \mathrm{S} \cap \mathrm{C_j} = \emptyset \end{cases}$$

$$\gamma(x) : \mathbb{R}^{\mathrm{k}} \to \mathbb{R}$$

$$\bigcup_{S \in \chi} | f(S) - \gamma(\max_{x_i \in S}\{h(x_i)\}) | < \epsilon$$

$\therefore$ sufficiently large $k, \gamma(\max_{x_i \in S}\{h(x_i)\})$ can approximate **any** permutation invariance function.

Supplementary

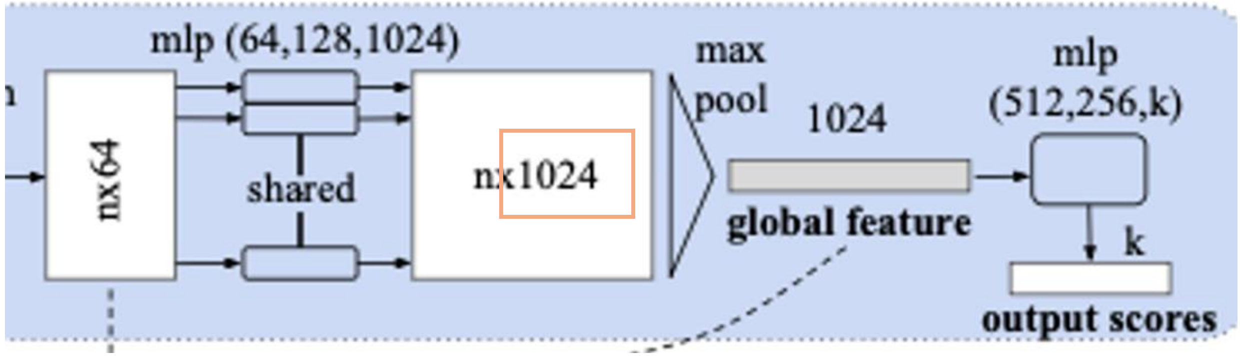
Bottleneck dimension and stability

MaxPooling (axis = 0)

$$u = \max_{x_i \in S}\{h(x_i)\}$$

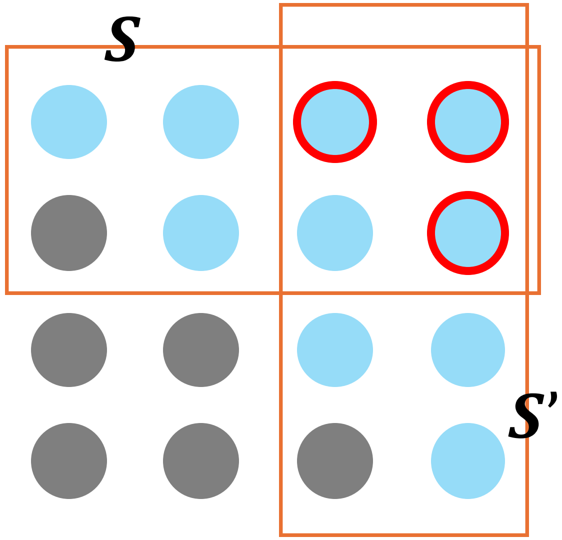
$$u : \chi \rightarrow \mathbb{R}^K$$

Bottleneck **k**



(a) $\forall S, \exists C_S, N_S \subseteq \chi, f(T) = f(S)\ \ if C_S \subseteq T \subseteq N_S$

(b) $|C_S| \leq K$



$$\boldsymbol{f(S) = f(S')}$$

Stability

Experiments

Task1: 3D Classification

Dataset:

- ModelNet40
- (12K shapes, 40 classes)

Results:

- MVCNN: 90.1% (80 views)
- VoxNet: 85.9% (12 views)
- PointNet: 89.2% (single view)

→ On par with SOTA
→ 141× faster than MVCNN

	input	#views	accuracy avg. class	accuracy overall
SPH [11]	mesh	-	68.2	-
3DShapeNets [28]	volume	1	77.3	84.7
VoxNet [17]	volume	12	83.0	85.9
Subvolume [18]	volume	20	86.0	**89.2**
LFD [28]	image	10	75.5	-
MVCNN [23]	image	80	**90.1**	-
Ours baseline	point	-	72.6	77.4
Ours PointNet	point	1	86.2	**89.2**

Table 1. **Classification results on ModelNet40.** Our net achieves state-of-the-art among deep nets on 3D input.

	#params	FLOPs/sample
PointNet (vanilla)	0.8M	148M
PointNet	3.5M	440M
Subvolume [18]	16.6M	3633M
MVCNN [23]	60.0M	62057M

Table 6. **Time and space complexity of deep architectures for 3D data classification.** PointNet (vanilla) is the classification PointNet without input and feature transformations. FLOP stands for floating-point operation. The "M" stands for million. Subvolume and MVCNN used pooling on input data from multiple rotations or views, without which they have much inferior performance.

Experiments

Task2: Part Segmentation

Dataset:

- ShapeNet Parts
- (16K shapes, 50 parts)

Results:

- Yi et al.: 81.4% mIoU
- 3D CNN: 79.4% mIoU
- PointNet: 83.7% mIoU

	mean	aero	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate board	table
# shapes		2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152	5271
Wu [27]	-	63.2	-	-	-	73.5	-	-	-	74.4	-	-	-	-	-	-	74.8
Yi [29]	81.4	81.0	78.4	77.7	**75.7**	87.6	61.9	**92.0**	85.4	**82.5**	**95.7**	**70.6**	91.9	**85.9**	53.1	69.8	75.3
3DCNN	79.4	75.1	72.8	73.3	70.0	87.2	63.5	88.4	79.6	74.4	93.9	58.7	91.8	76.4	51.2	65.3	77.1
Ours	**83.7**	**83.4**	**78.7**	**82.5**	74.9	**89.6**	**73.0**	91.5	**85.9**	80.8	95.3	65.2	**93.0**	81.2	**57.9**	**72.8**	**80.6**

Table 2. **Segmentation results on ShapeNet part dataset.** Metric is mIoU(%) on points. We compare with two traditional methods [27] and [29] and a 3D fully convolutional network baseline proposed by us. Our PointNet method achieved the state-of-the-art in mIoU.

→ +2.3% improvement
→ Robust to missing / partial points

Experiments

Task3: Scene Segmentation

Dataset:

- Stanford 3D Indoor Scenes
- (271 rooms, 13 classes)

Results:

- Baseline: 20.1% mIoU
- PointNet: 47.7% mIoU

→ 137% relative improvement

→ Unified architecture for all tasks

	mean IoU	overall accuracy
Ours baseline	20.12	53.19
Ours PointNet	**47.71**	**78.62**

Table 3. **Results on semantic segmentation in scenes.** Metric is average IoU over 13 classes (structural and furniture elements plus clutter) and classification accuracy calculated on points.