Spatial Transformer for 3D Point Clouds

Jiayun (Peter) Wang Rudrasis Chakraborty Stella X. Yu

INTERNATIONAL COMPUTER SCIENCE INSTITUTE

Motivations

✧ Convolutions on point clouds:

$$F_i^{(t)} = \sum_{j \in \mathcal{N}_i^{(t)}} w_{ij}^{(t)} F_j^{(t-1)}$$

Previous work (PointNet, PointNet++, PointCNN, ...):

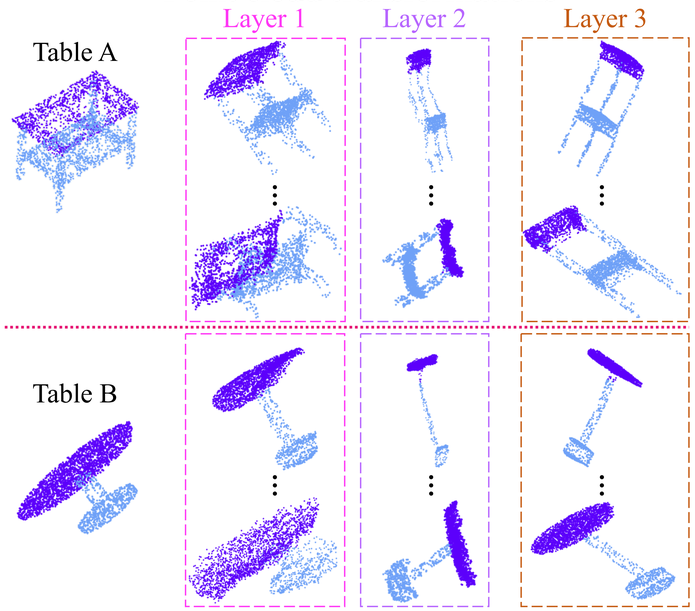
$$\mathcal{N}^{(1)} = \mathcal{N}^{(2)} = ... = \mathcal{N}^{(T)}$$

Single and **fixed** neighborhood.

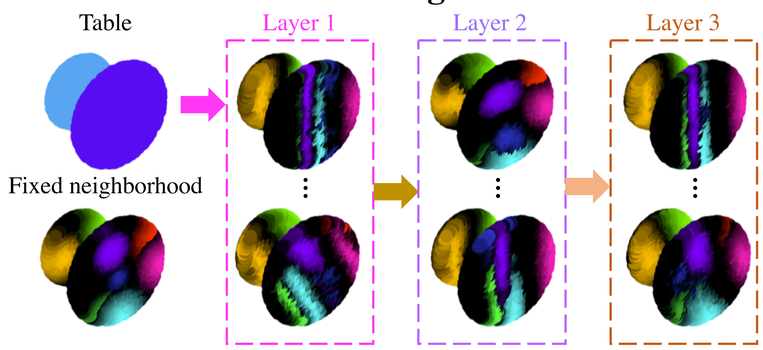
Ours: $\mathcal{N}^{(1)} \neq \mathcal{N}^{(2)} \neq ... \neq \mathcal{N}^{(T)}$

Multiple and **dynamic** neighborhoods, learned from point coordinates P and feature F.

Point cloud transformations



Point cloud neighborhood



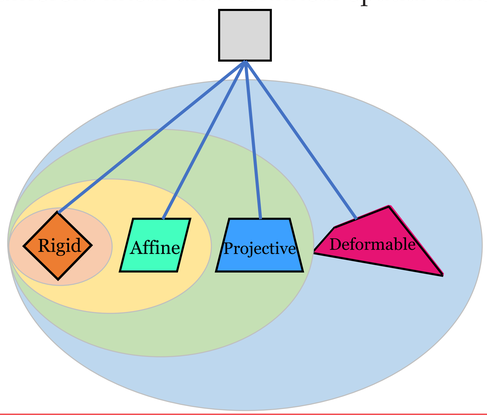
Our spatial transformers

✧ **Diverse** transformations at each layer enhance learning capacity

✧ Corresponding transformations at each layer capture **similar geometric shapes** regardless of in-class variations.

Spatial Transformers

We propose different linear and nonlinear spatial transformers.



✧ **Affine**: • Transform original point cloud $P \in \mathbf{R}^{3\times N}$:

$$G_i^{(t)} = A_i^{(t)} P$$

• Apply k-nearest neighbor search on transformed points $G_i^{(t)}$

• Obtain dynamic local neighborhood $\mathcal{N}_i^{(t)}$

• Perform specific point convolution on this graph:

$$F_i^{(t)} = \mathsf{CONV}_W(\mathcal{F}^{(t-1)}, \mathcal{N}_i^{(t)})$$

• Concatenate all the sub-features to get the output feature:

$$\mathcal{F}^{(t)} = \mathsf{CONCAT}(F_1^{(t)}, F_2^{(t)}, ..., F_{k^{(t)}}^{(t)}),$$

✧ **Projective**: Transform in homogeneous coordinates:

$$\widetilde{G}_i^{(t)} = B_i^{(t)} \widetilde{P}$$

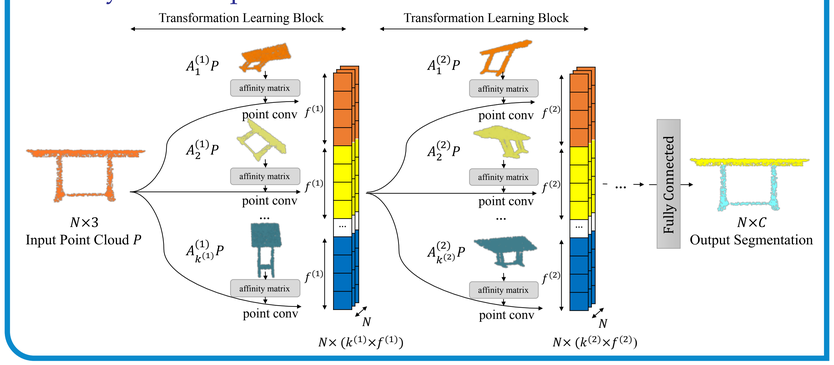
✧ **Deformable**: • Deformation matrix $D_i^{(t)} \in \mathbf{R}^{3\times N}$ allows every point the freedom to move:

$$P_i^{(t)} = A_i^{(t)} P + D_i^{(t)}$$

• Transformer is learned from both point location and feature:

$$G_i^{(t)} = \begin{bmatrix} A_i^{(t)} & C_i^{(t)} \end{bmatrix} \begin{bmatrix} P \\ \mathcal{F}^{(t-1)} \end{bmatrix} = C_i^{(t)} \begin{bmatrix} P \\ \mathcal{F}^{(t-1)} \end{bmatrix}$$

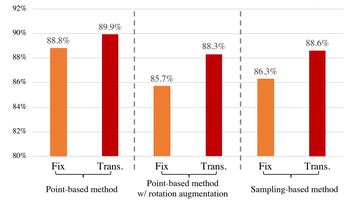
Each layer has k spatial transformers:



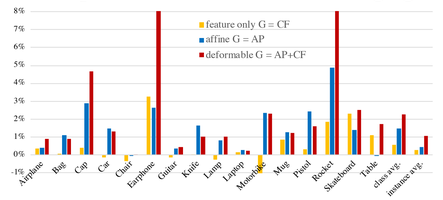
Experimental Results

✧ **Classification:** 2% average gain on ModelNet40.

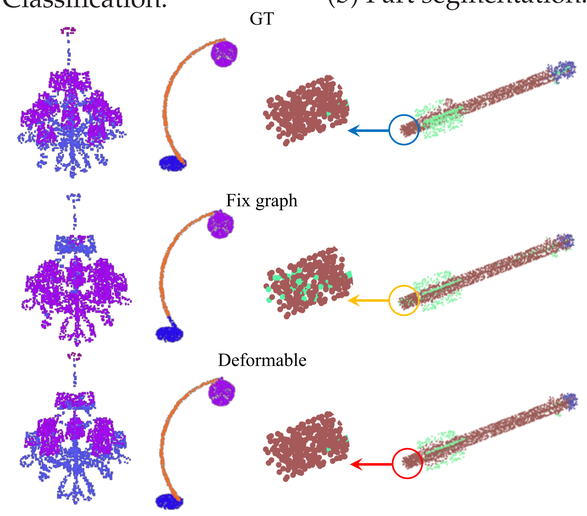
✧ **Part Segmentation:** 8% gain on earphone and rocket. 1% average gain.



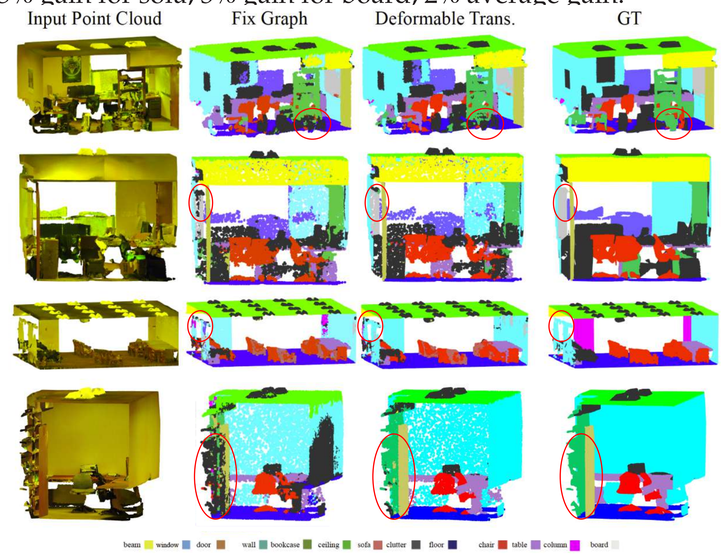
(a) Classification.



(b) Part segmentation.



✧ **3D Indoor Scenes Semantic Segmentation:**
5% gain for sofa, 3% gain for board, 2% average gain.



✧ **Object Detection:** Up to 9% gain on 3D KITTI LiDAR detection