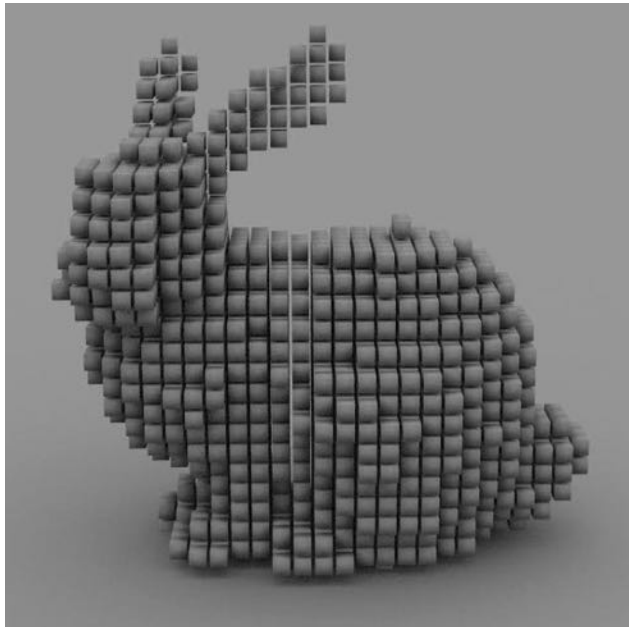
Learning Graph Structure for Point Cloud Processing

Zhuohan Li

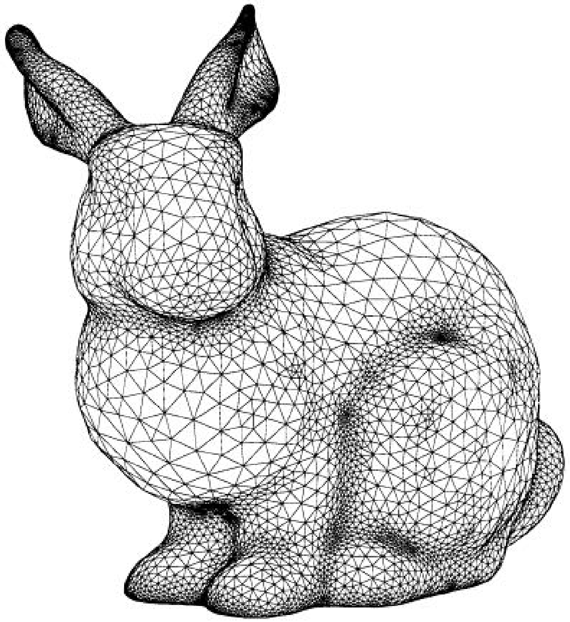
Peking University

2018/09/21

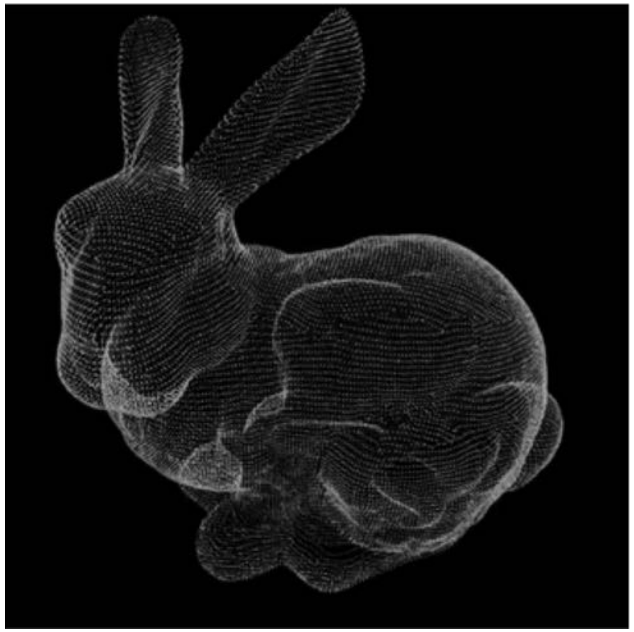
3D Data Representation



Volumetric



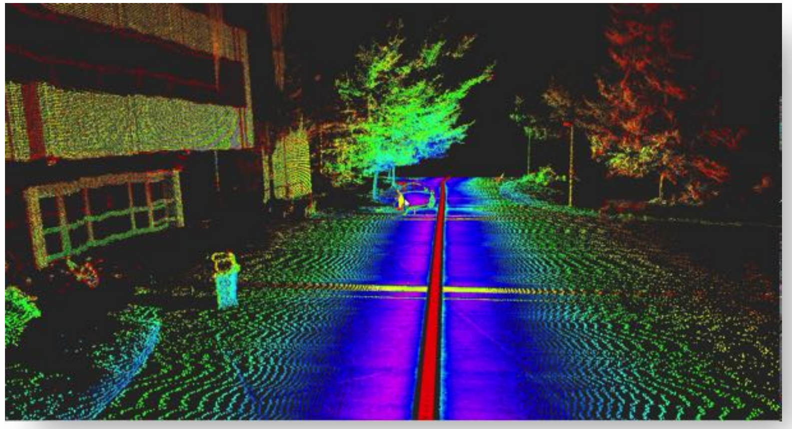
Mesh



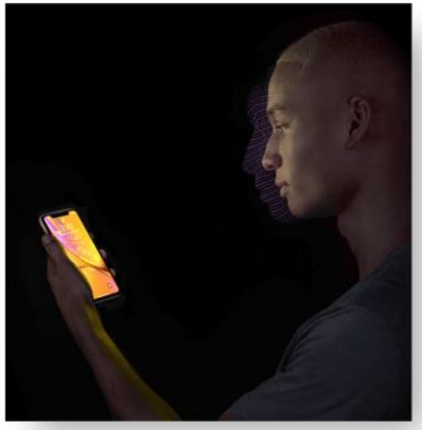
Point cloud

Point Cloud

- Unordered set $\{(x_i, y_i, z_i) \mid i = 1, \ldots, N\}$
- With underlying distance metric



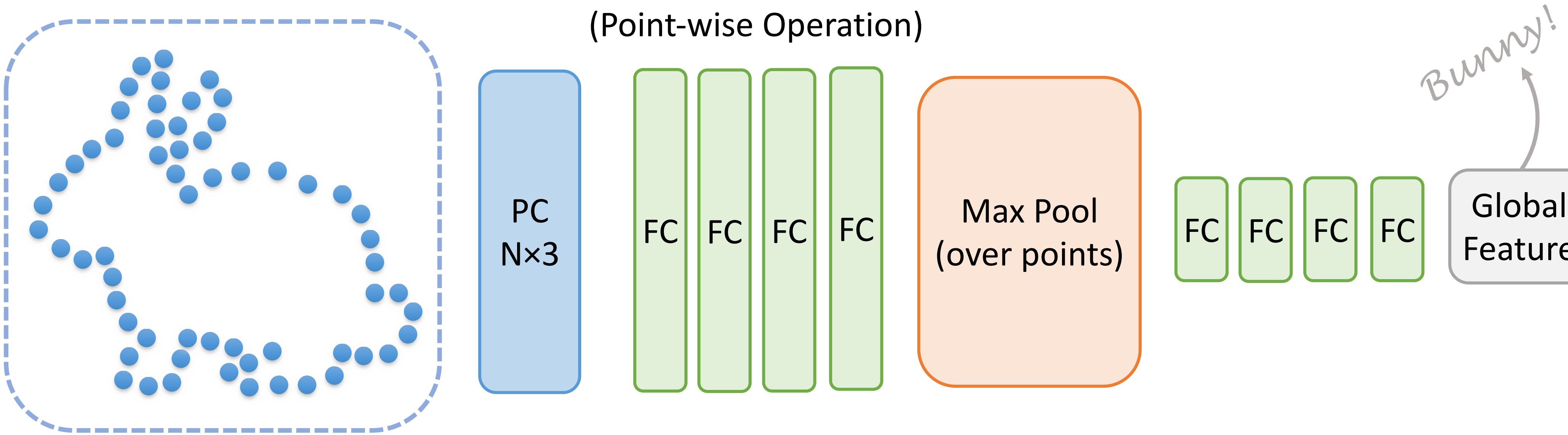
LiDAR



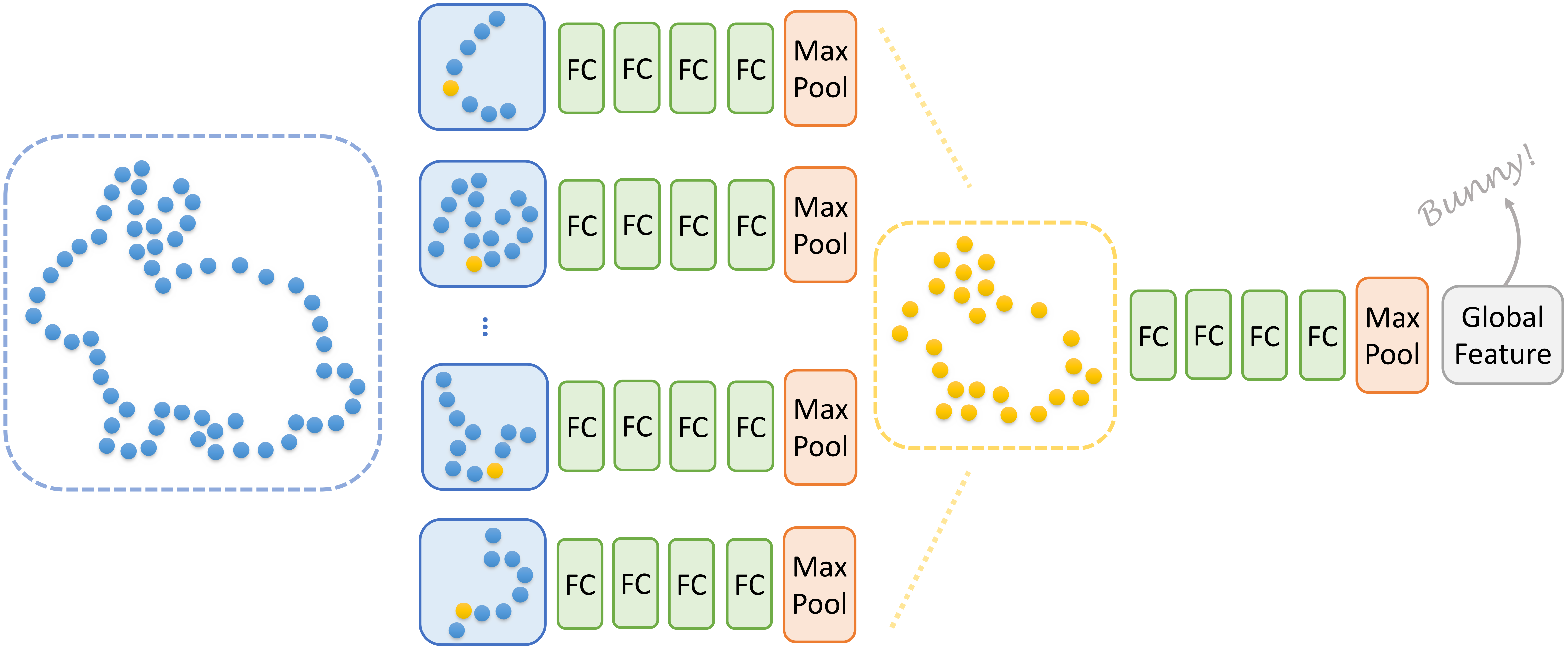
Face ID

Kinect

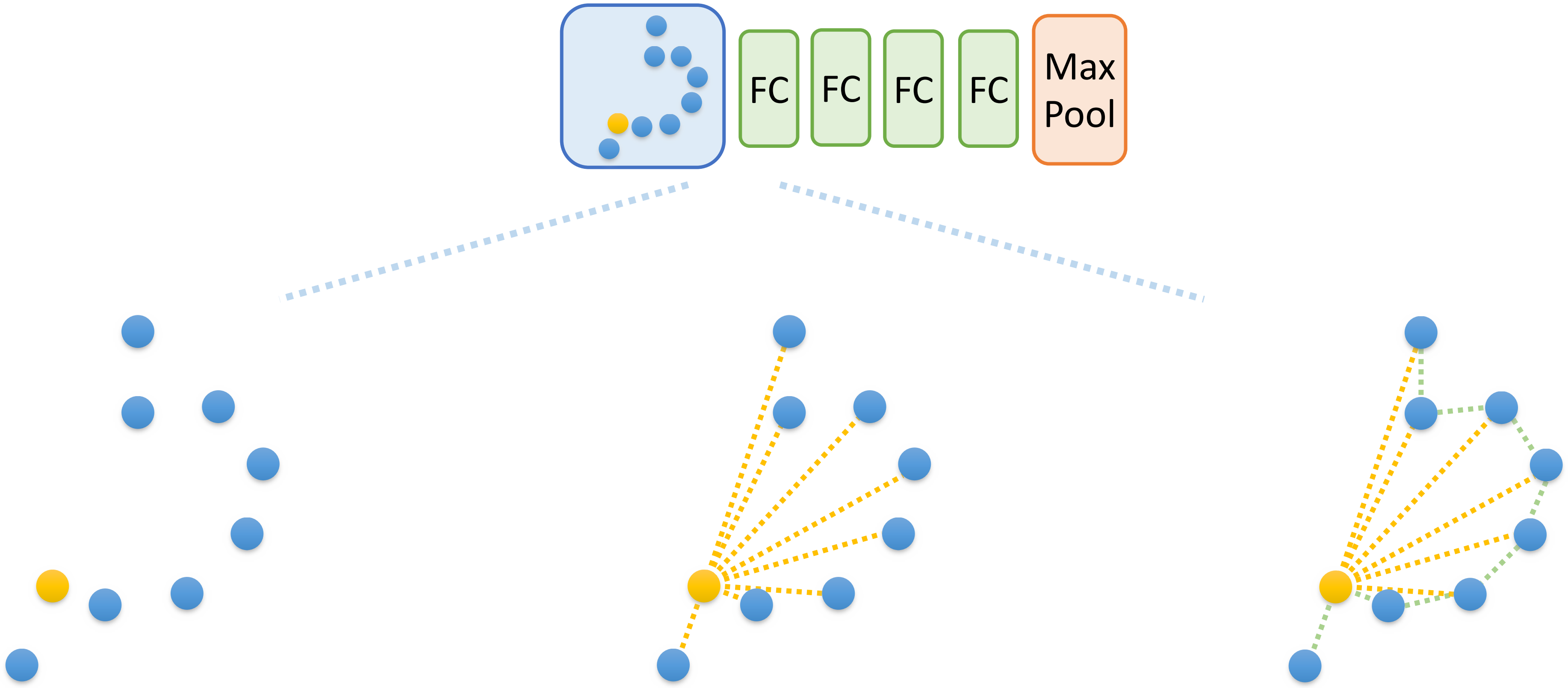
PointNet


(Point-wise Operation)
PC
N×3
FC
FC
FC
FC
Max Pool
(over points)
FC
FC
FC
FC
Global
Feature
Bunny!

PointNet++ (2 Layers)



Graph Structure in PointNet++



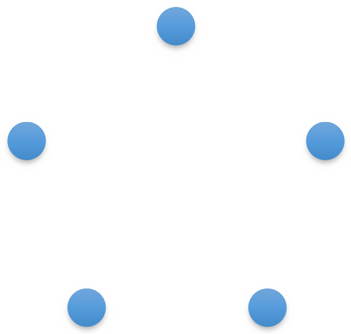
Original

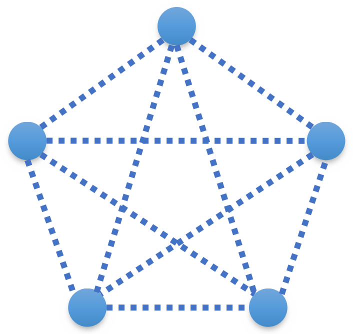
Subtract the sampling center (as in PointNet++)

Outline of the original shape

Points and Edges

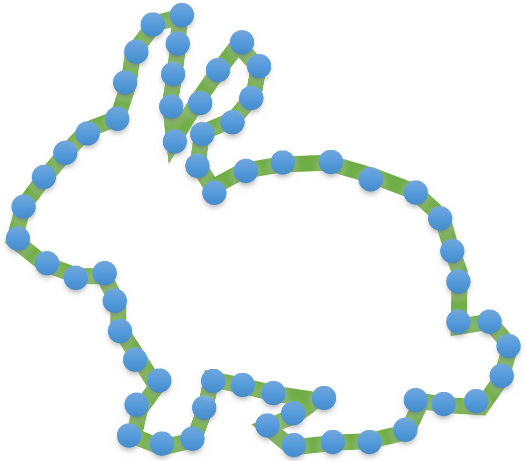
- For a point cloud with $O(n)$ points, there are $O(n^2)$ edges in total
- Most of the edges are *unimportant* for the understanding of the point cloud



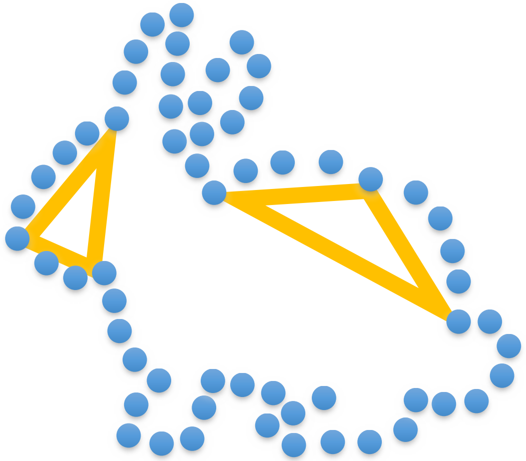


Task Dependent Graph Structure

- How to *decide* the importance of different edges?
- Different tasks need different graph structures



Perimeter of the bunny



Area of the bunny

Learning Task Dependent Graph Structure

- Embed a learnable graph structure into neural networks
- Let the network to choose the best graph for the tasks by *gradient-based optimization*

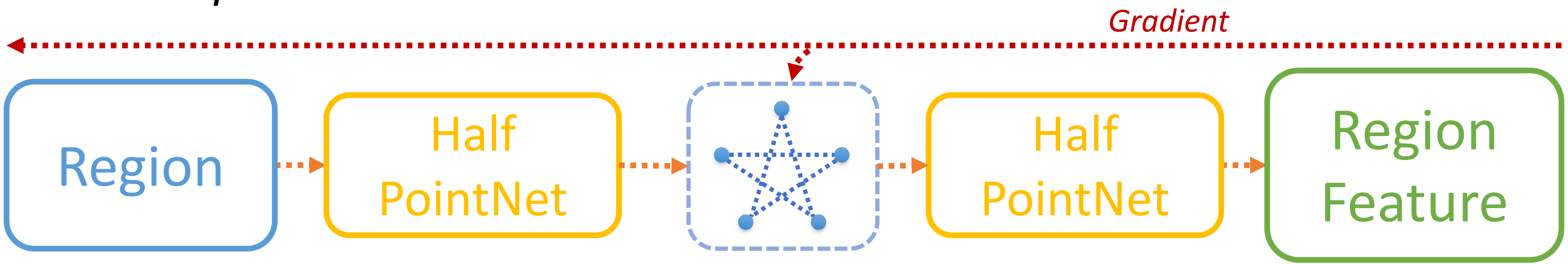
Region → PointNet Submodule → Region Feature

Better end-task performance

Deeper understanding of different tasks

Learning Task Dependent Graph Structure

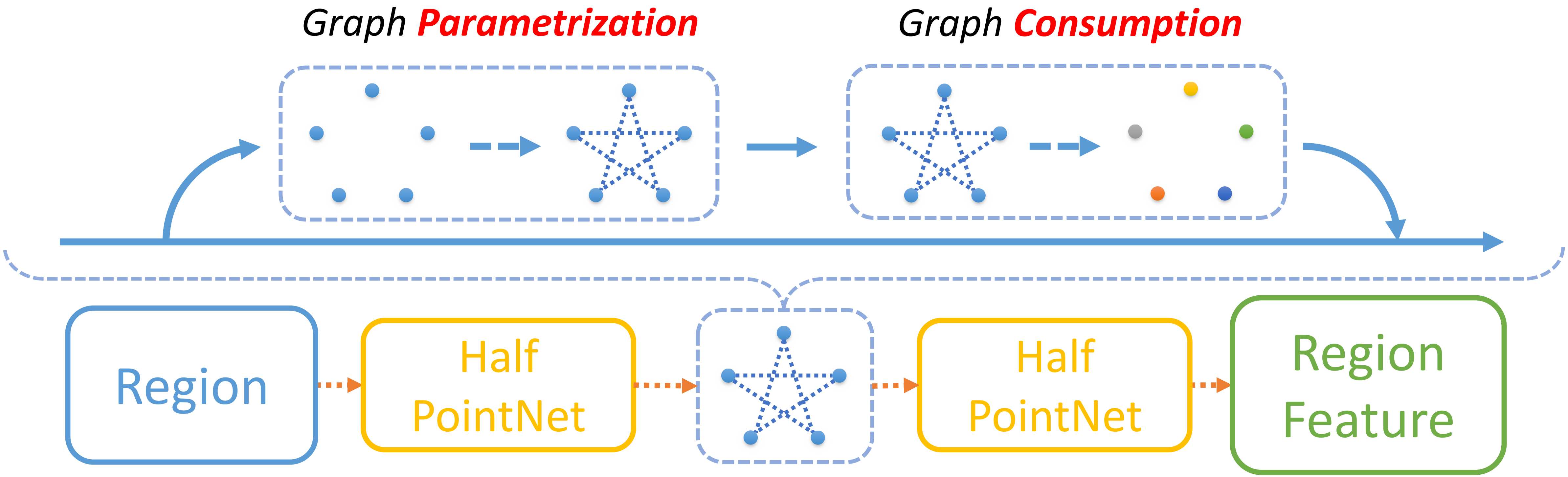
- Embed a learnable graph structure into neural networks
- Let the network to choose the best graph for the tasks by *gradient-based optimization*



Better end-task performance

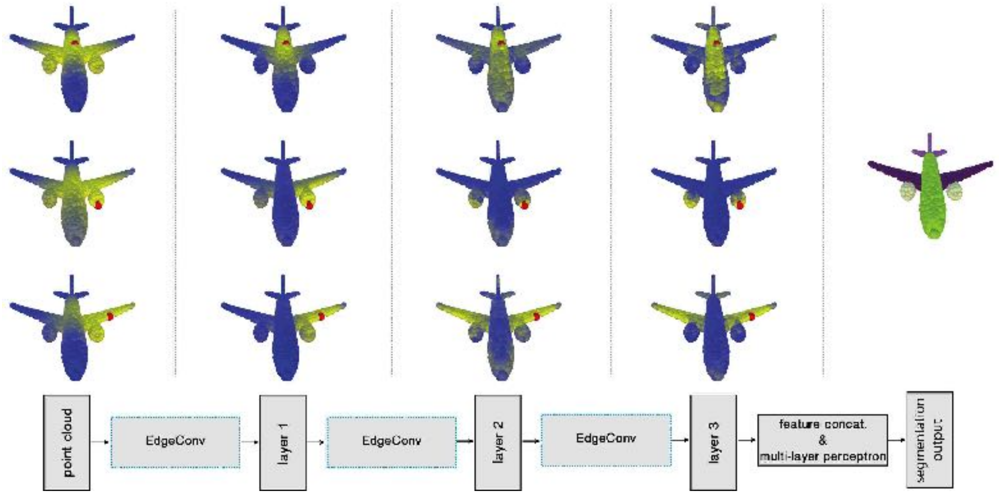
Deeper understanding of different tasks

Learning Task Dependent Graph Structure



Related Works (Graph Parametrization)

- Dynamic Graph CNN for Learning on Point Clouds (Edgeconv, DGCNN)
  - Nearest neighbors in the feature space
  - Edgeconv operation to consume the graph
  - No explicit gradient signal for graph editing



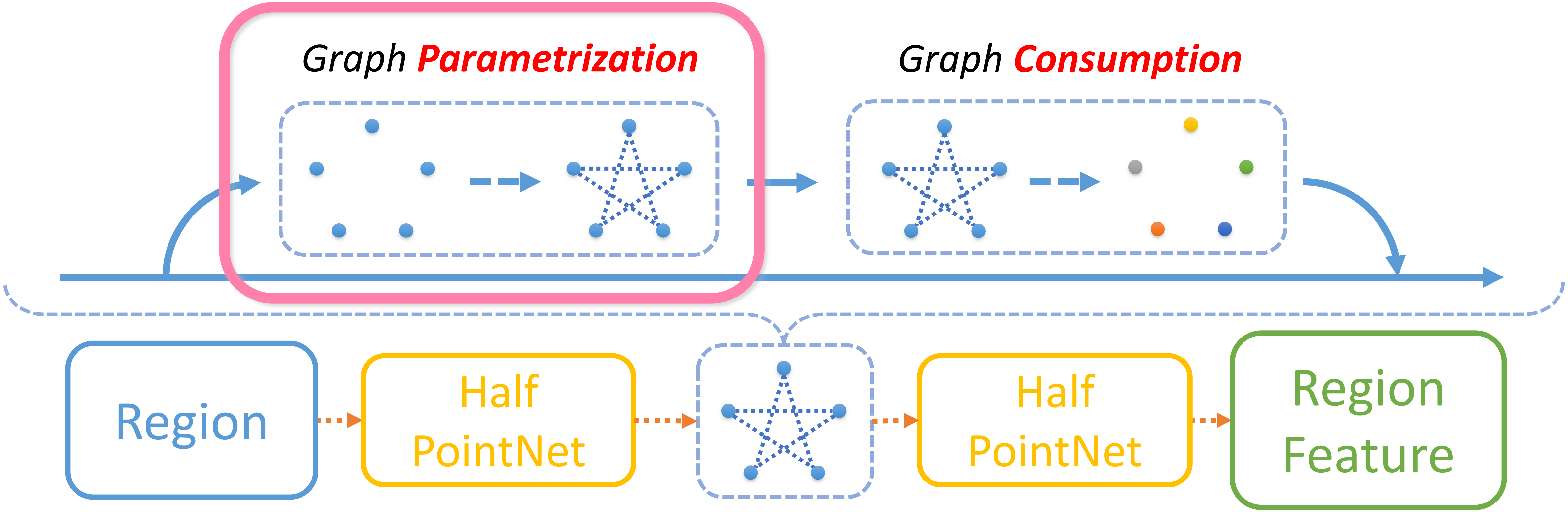
Related Works (Graph Consumption)

- There are many works on graph-structured data processing
- Most of the works assume graph is given

Spectral domain convolution

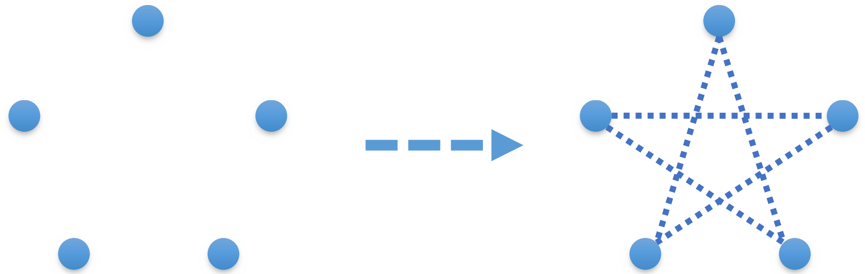
Spatial domain convolution

Learning Task Dependent Graph Structure



Graph Parametrization

- A point set $\{x_1, \ldots, x_n\}$ in a sampled neighborhood of PointNet++
- Point features $\{h_1, \ldots, h_n\}$
- Build a graph with edges $E = \{e_{ij}\}$ based on the point features



Attention Mechanism

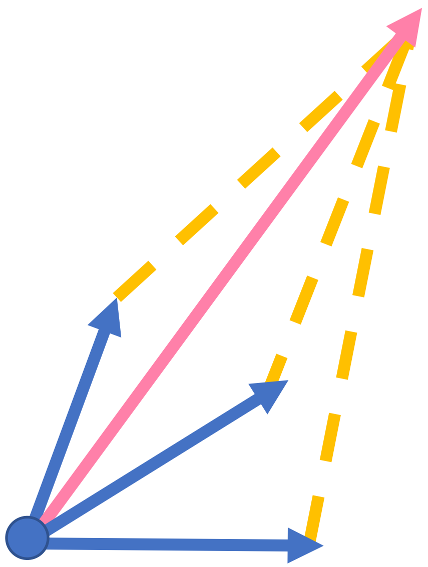
- One straightforward idea: *Attention Mechanism*
- Widely used in NLP domain to build relationship among the words
- Specifically, edge weights are defined as

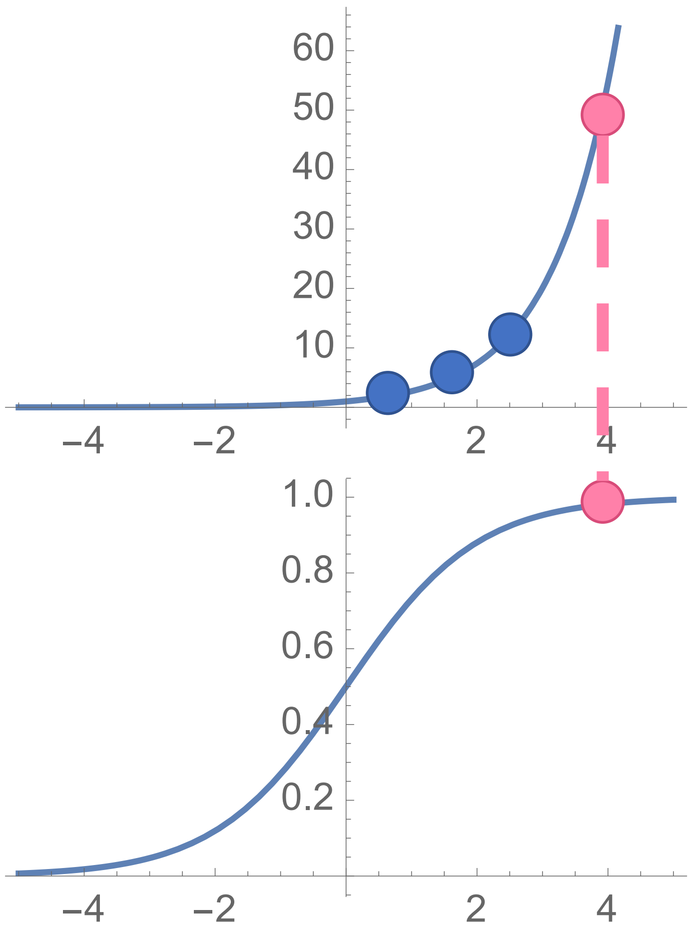
$$e_{ij} = \frac{\exp\left((Wh_i)^T (Wh_j)\right)}{\sum_k \exp\left((Wh_i)^T (Wh_k)\right)},$$

where W is a learned weight matrix

Why Attention not Work?

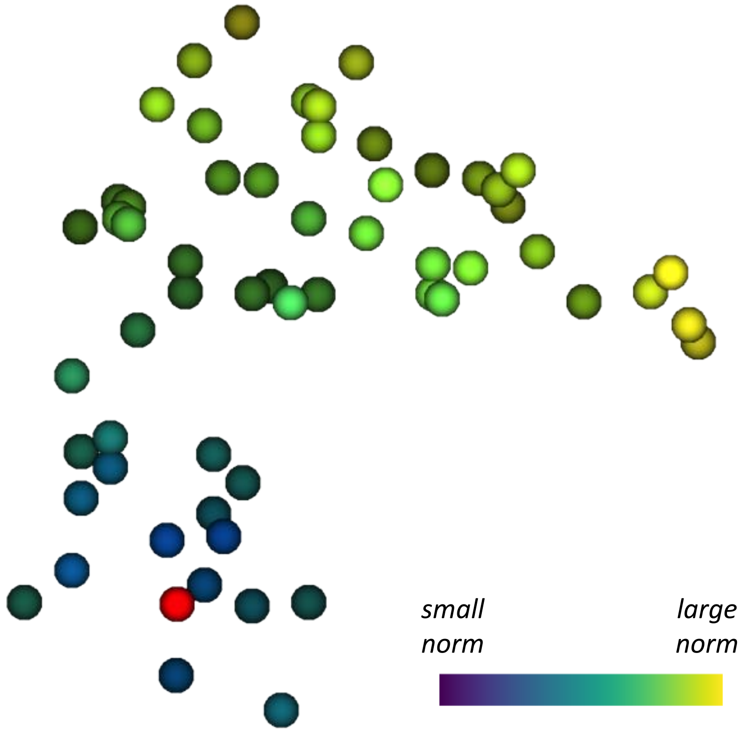
$$e_{ij} = \frac{\exp\left((Wh_i)^T(Wh_j)\right)}{\sum_k \exp\left((Wh_i)^T(Wh_k)\right)}$$





Point Clouds and Local Regions

- Points mostly lies on the surface of an object
- Extremely nonuniform sampled neighborhood
- Farthest point sampling samples a lot of corner points

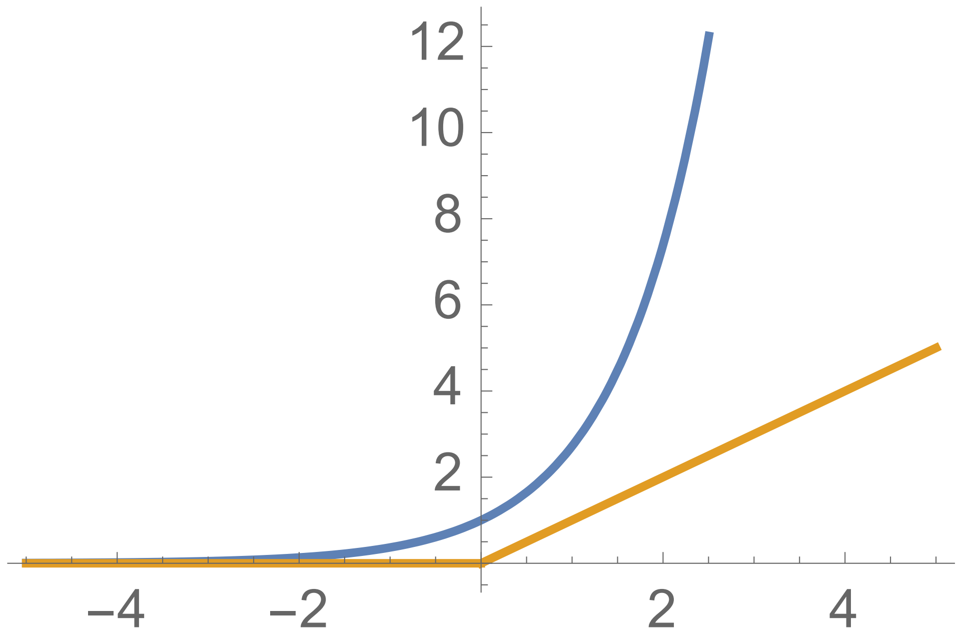


A sample neighborhood from the first layer of PointNet++

ReLU Dot-Product Graph Parametrization

- ReLU as First-order approximation of the SoftMax function as the graph parametrization:

$$e_{ij} = \frac{\mathrm{ReLU}\left((Wh_i)^T(Wh_j)\right)}{\sum_k \mathrm{ReLU}\left((Wh_i)^T(Wh_k)\right)}$$

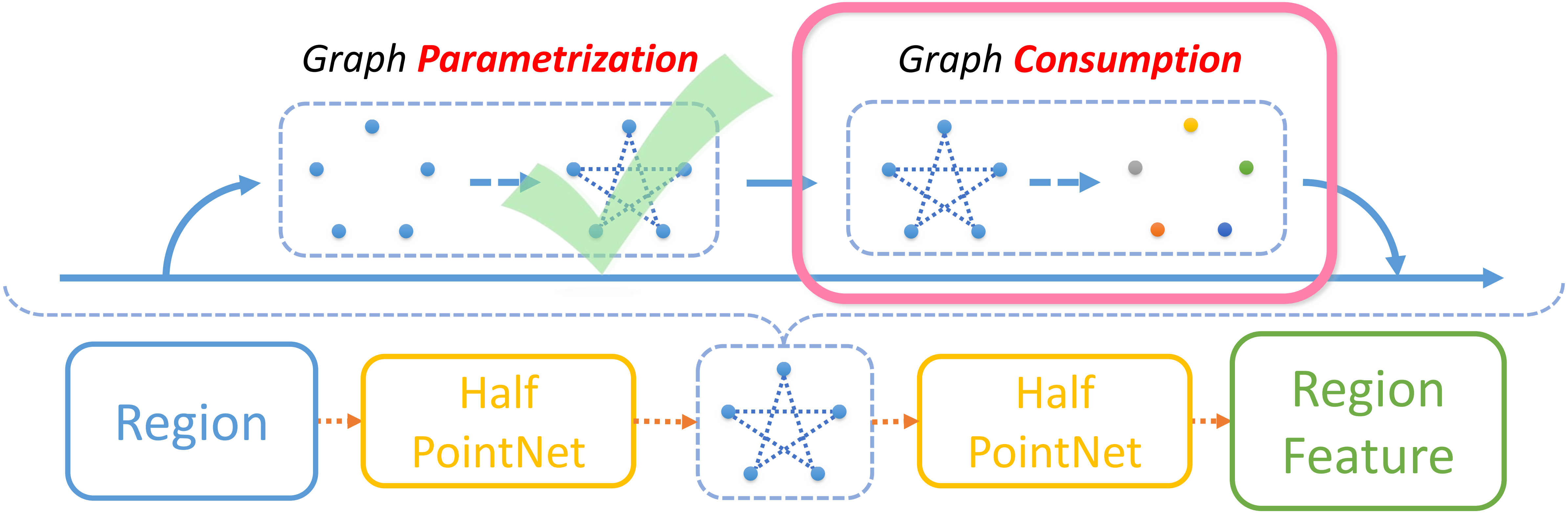


ReLU Dot-Product Graph Parametrization

$$e_{ij} = \frac{\mathrm{ReLU}\left((Wh_i)^T(Wh_j)\right)}{\sum_k \mathrm{ReLU}\left((Wh_i)^T(Wh_k)\right)}$$

- Denominator > 0, numerator ≥ 0
- Connection between points with similar point feature
- Smoother weights than attention mechanism

Learning Task Dependent Graph Structure



Diffusion Process

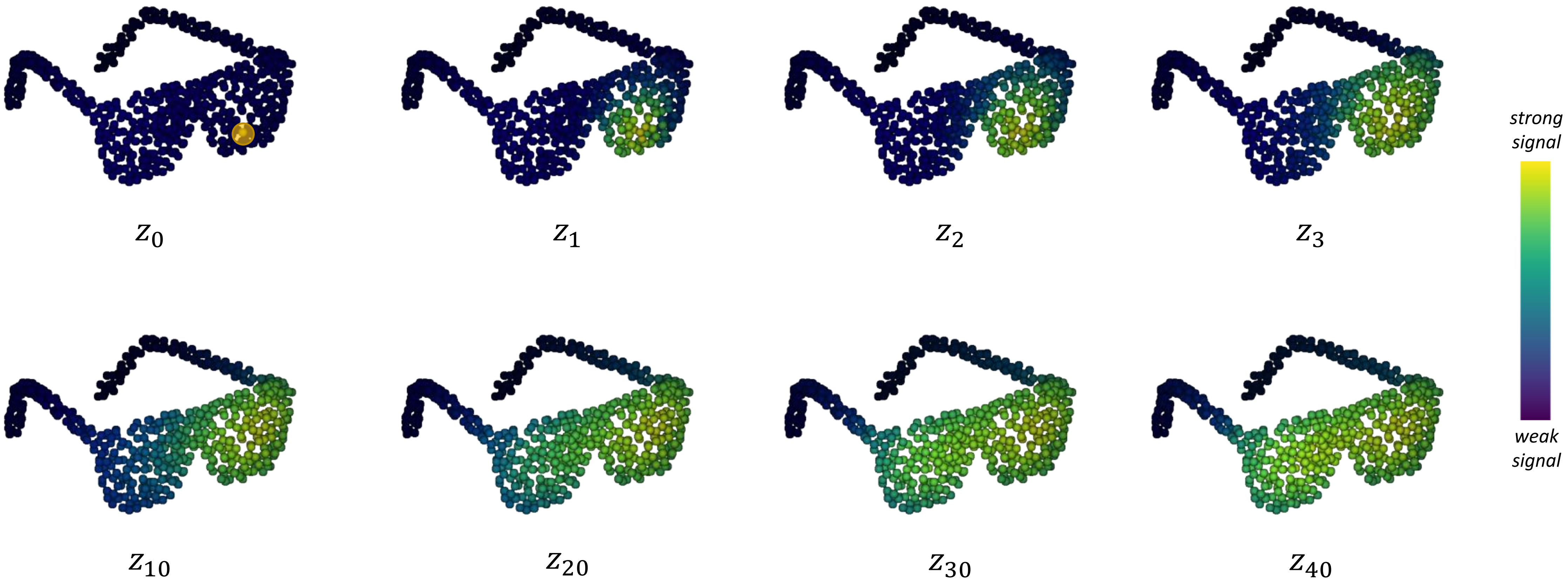
- Normalized adjacency matrix $A \in \mathbb{R}^{n \times n}$ of a graph,
- An initial signal vector $z_0 \in \mathbb{R}^n$
- The diffused signals would be

$$z_1 = Az_0, z_2 = Az_1, \dots, z_n = Az_{n-1}$$

- We can exactly recover adjacency matrix A by solving the linear system

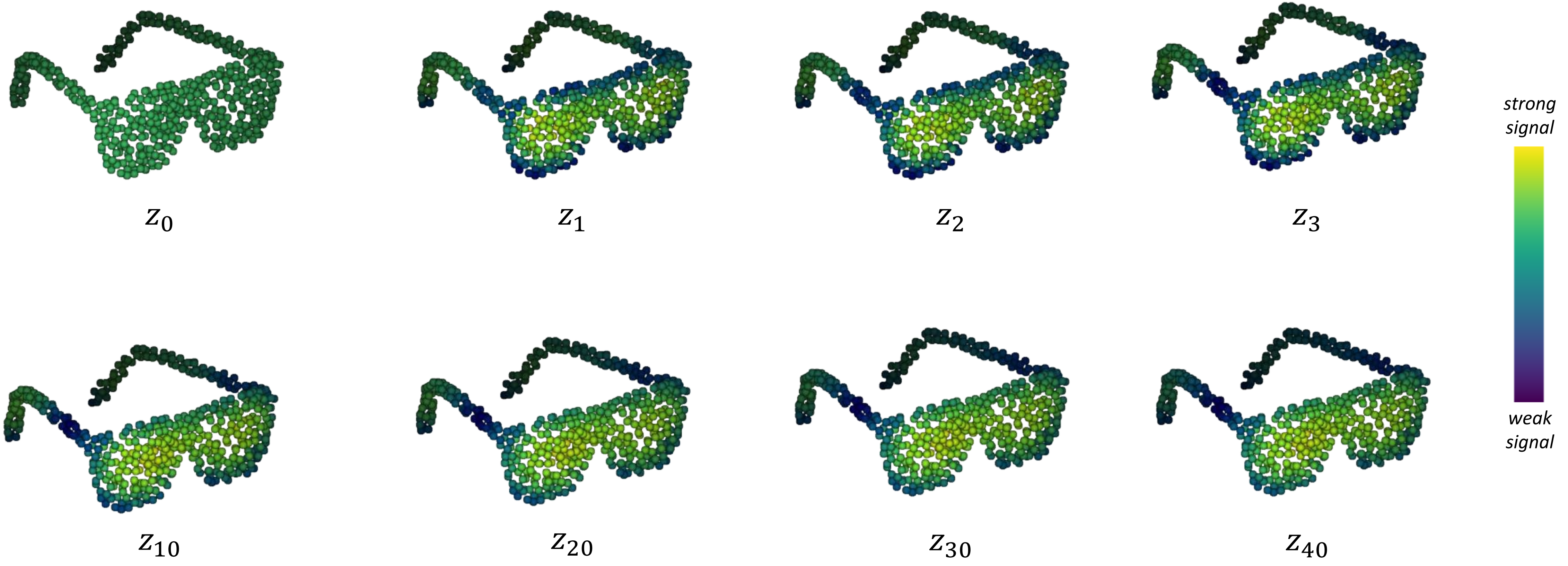
$$[z_1\ z_2\ \dots z_n] = A[z_0\ z_1\ \dots z_{n-1}]$$

Multiscale Resolution



Diffusion process example starting from a single point signal, using Gaussian Kernel as graph weights

Structure Emerging from Uniform Signal



Diffusion process example starting from uniform signal, using Gaussian Kernel as graph weights

Diffusing Multiple Signals

- With multiple initial signals $z_0^{(1)}, z_0^{(2)}, \dots, z_0^{(m)} \in \mathbb{R}^n$, the diffusion steps can be reduced to n/m to recover the matrix A

$$\left[z_1^{(0)}\ z_2^{(0)}\ \dots z_{n/m}^{(0)}\right] = A\left[z_0^{(0)} z_1^{(0)}\ \dots z_{n/m-1}^{(0)}\right]$$

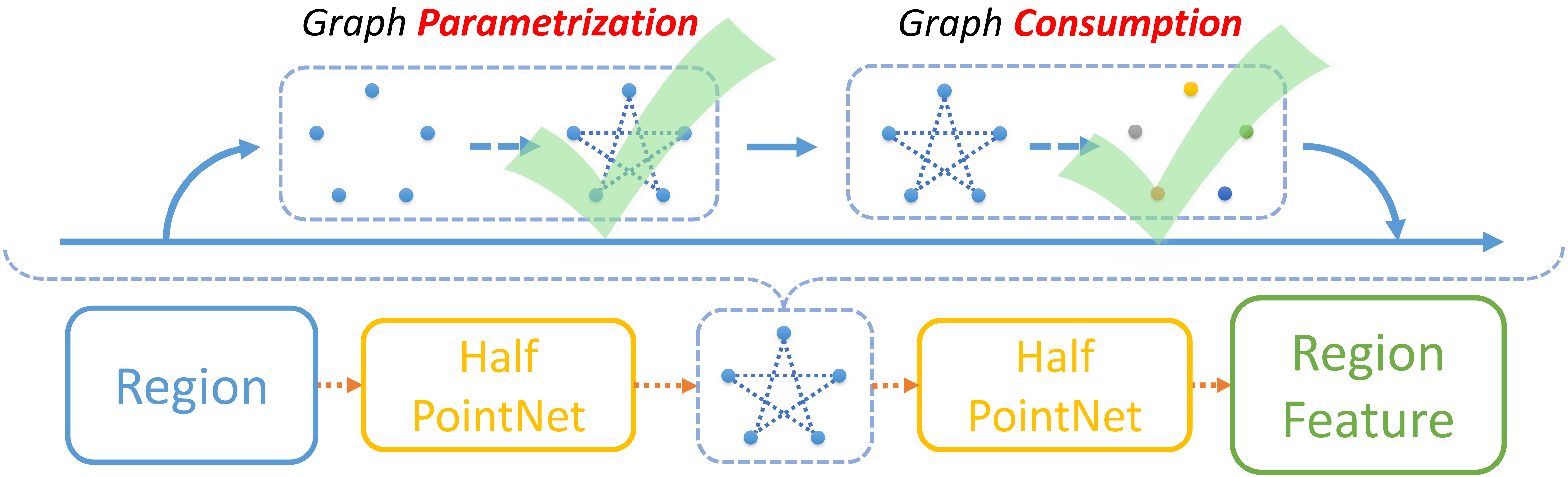
$$\left[z_1^{(1)}\ z_2^{(1)}\ \dots z_{n/m}^{(1)}\right] = A\left[z_0^{(1)} z_1^{(1)}\ \dots z_{n/m-1}^{(1)}\right]$$

$$\dots$$

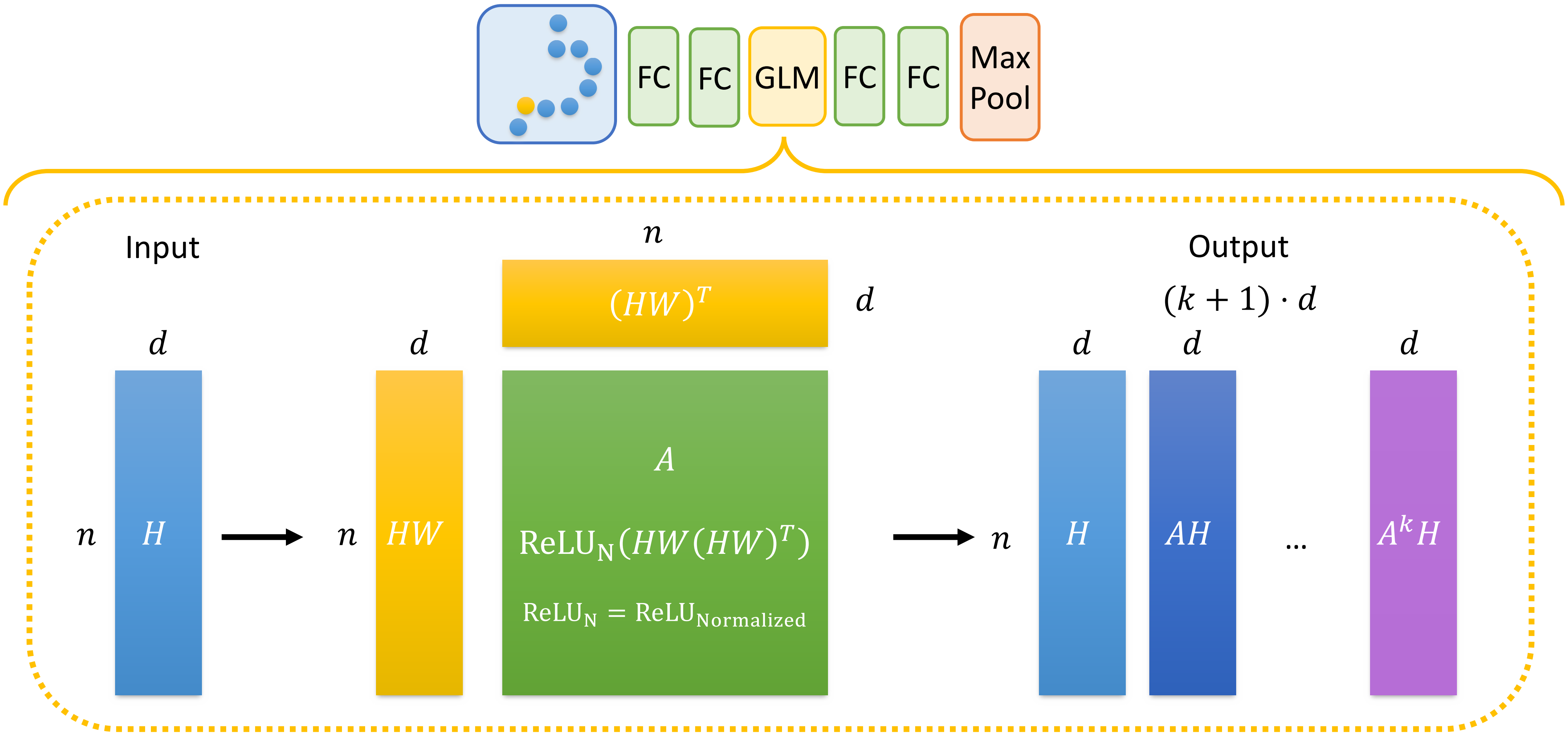
$$\left[z_1^{(m)}\ z_2^{(m)}\ \dots z_{n/m}^{(m)}\right] = A\left[z_0^{(m)} z_1^{(m)}\ \dots z_{n/m-1}^{(m)}\right]$$

- We use feature vectors ($m = d$) as initial signals for convenience

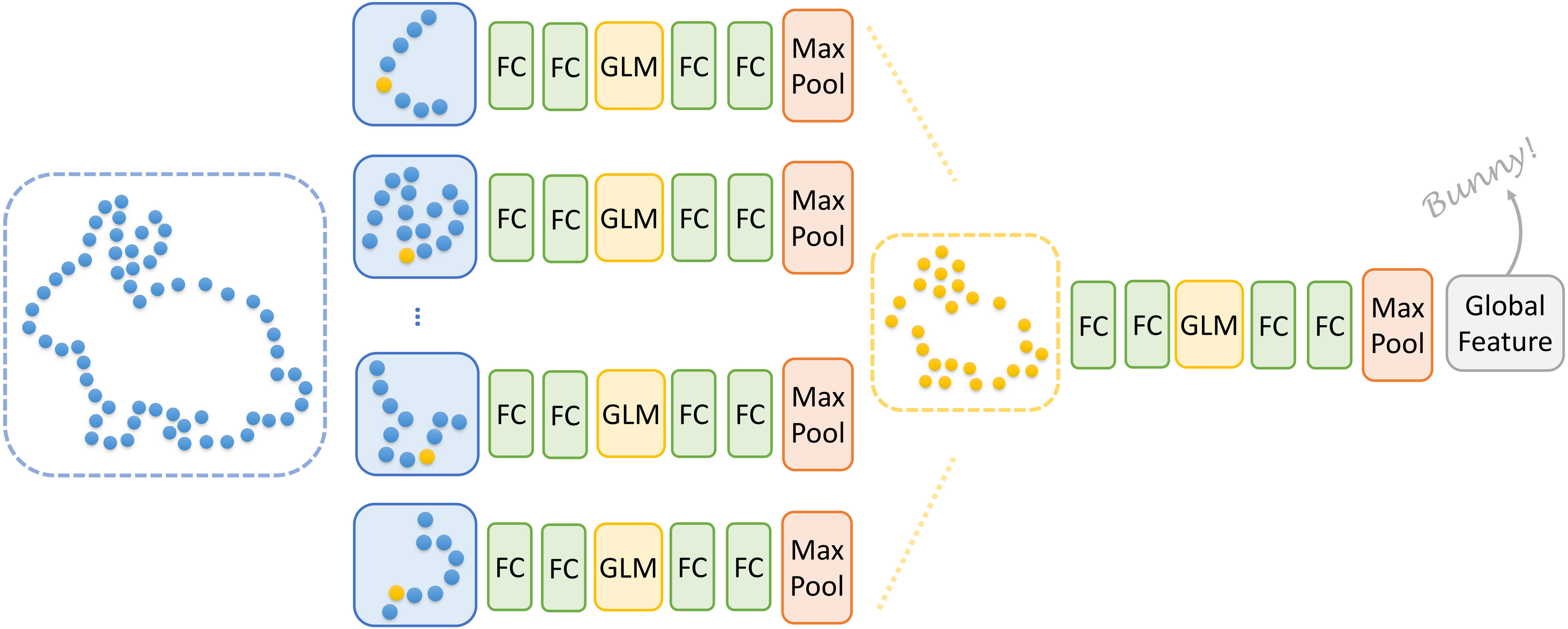
Learning Task Dependent Graph Structure



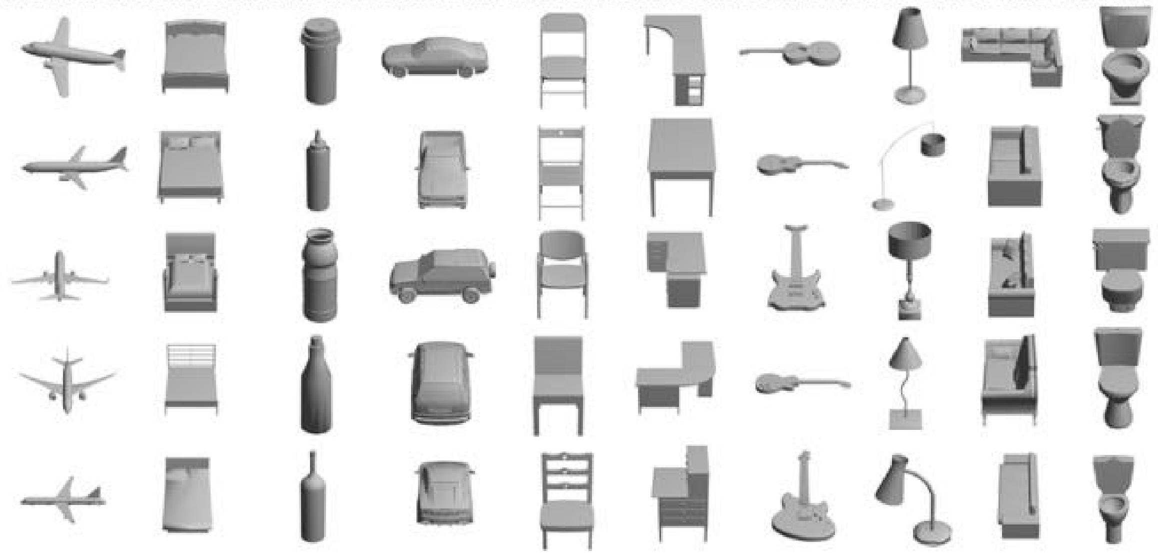
Graph Learning Module (GLM) Structure



PointNet++ with GLM (2 Layers)



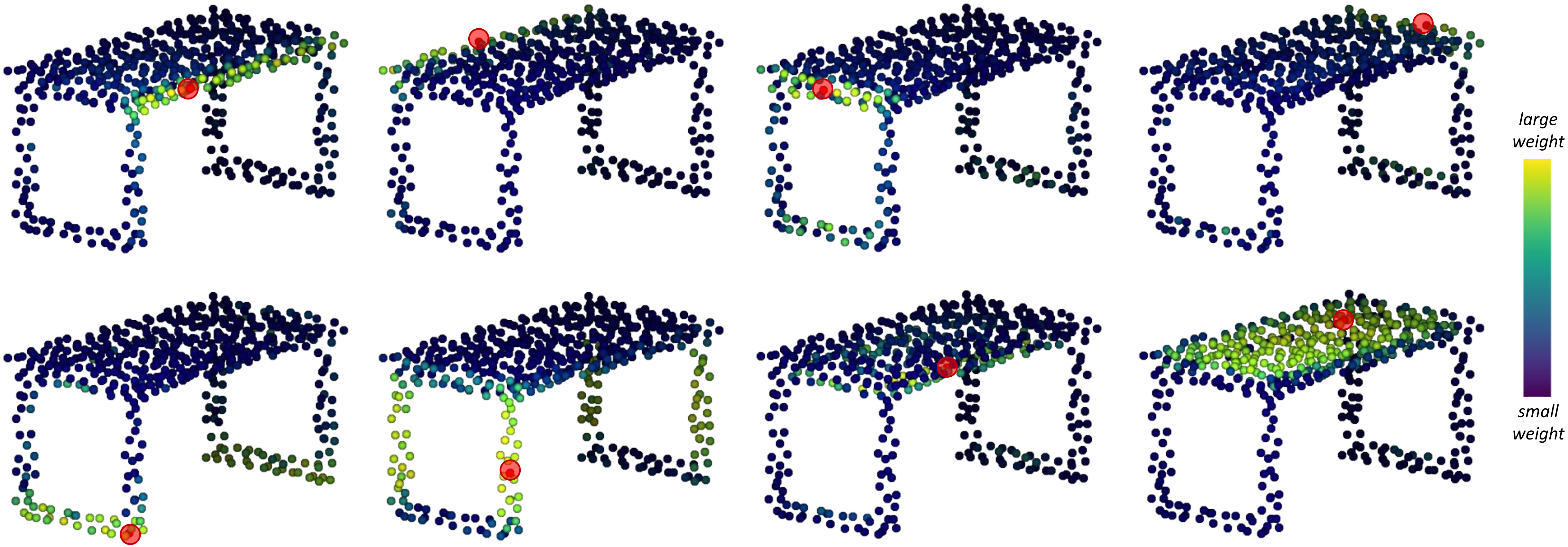
ModelNet40 Classification



Task	ModelNet40 Classification (Accuracy)
2-layer PointNet++ without GLM	90.3%
2-layer PointNet++ with GLM	**90.9%**

Two models share all the hyperparameters except for the GLM module

Learned Graph from Classification Task (Table)

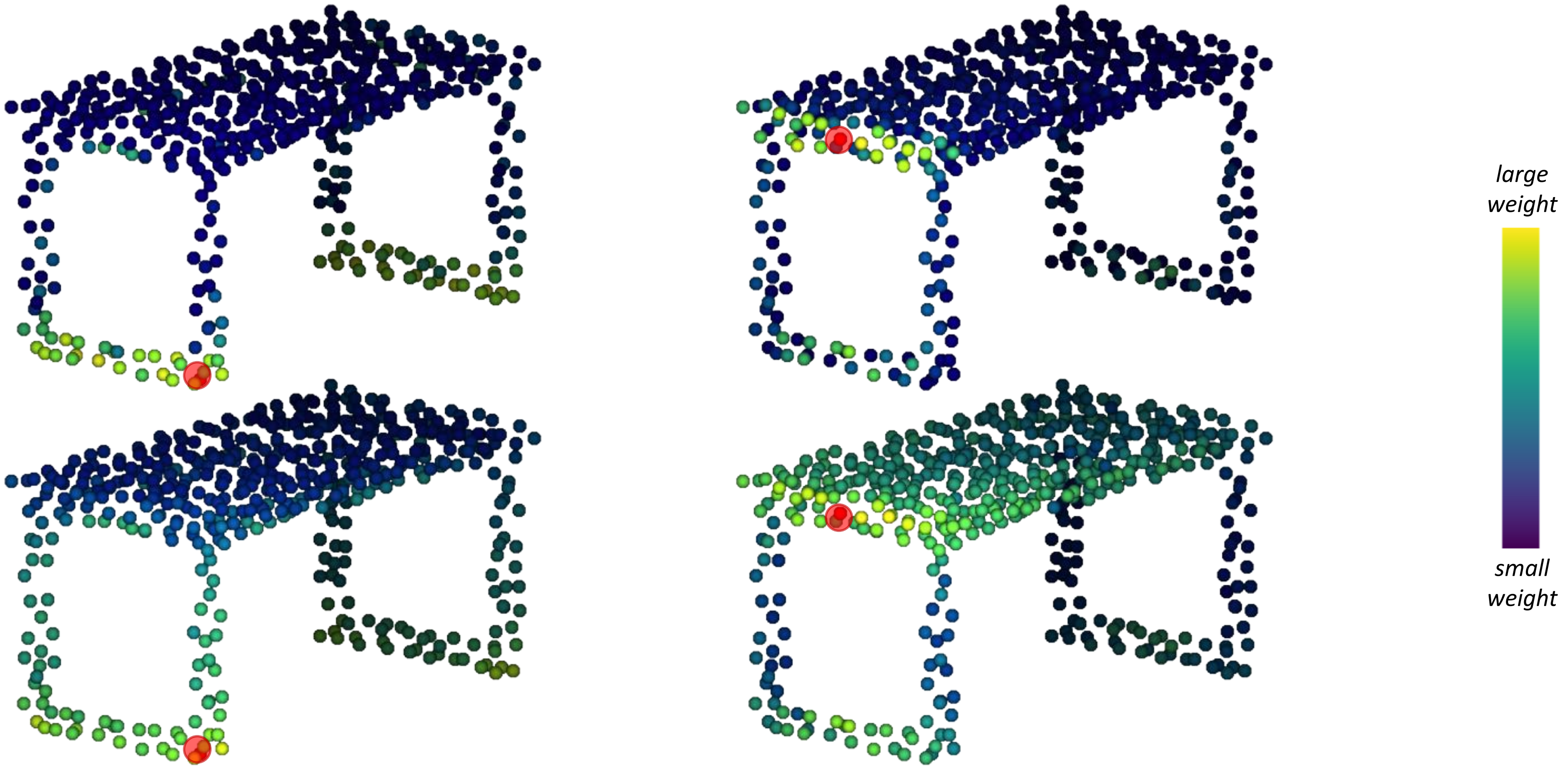


Edge weight distribution of the red point,
learned from ModelNet40 classification task

Graph Structure Comparison

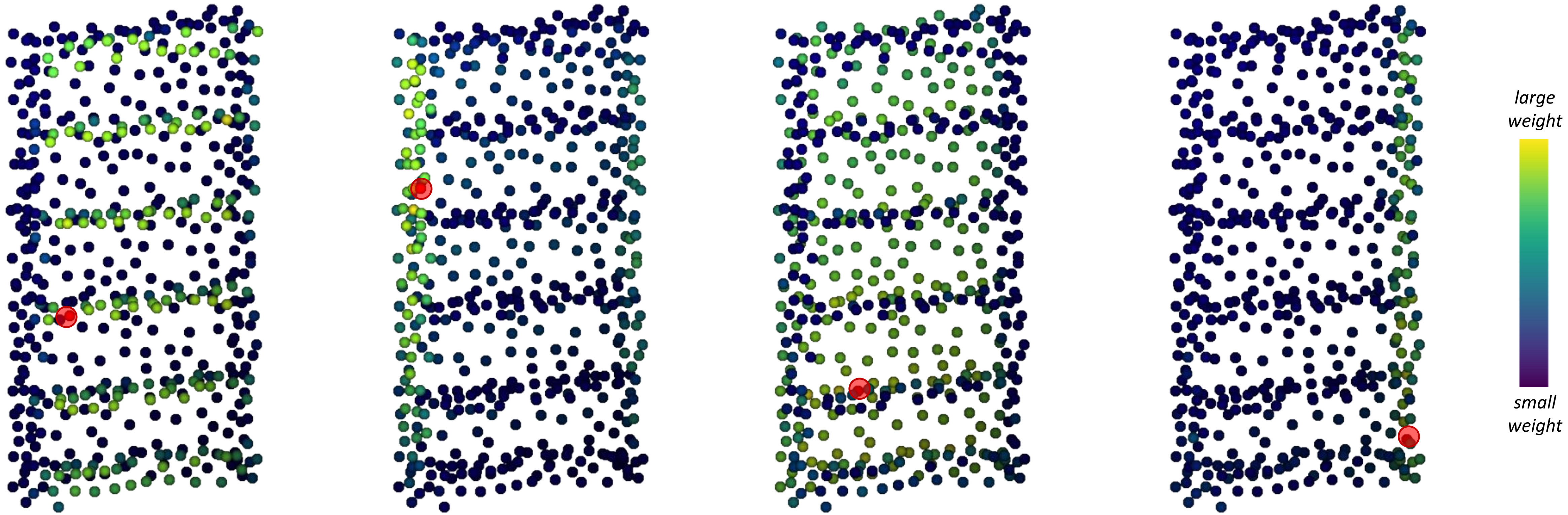
With GLM

Without GLM



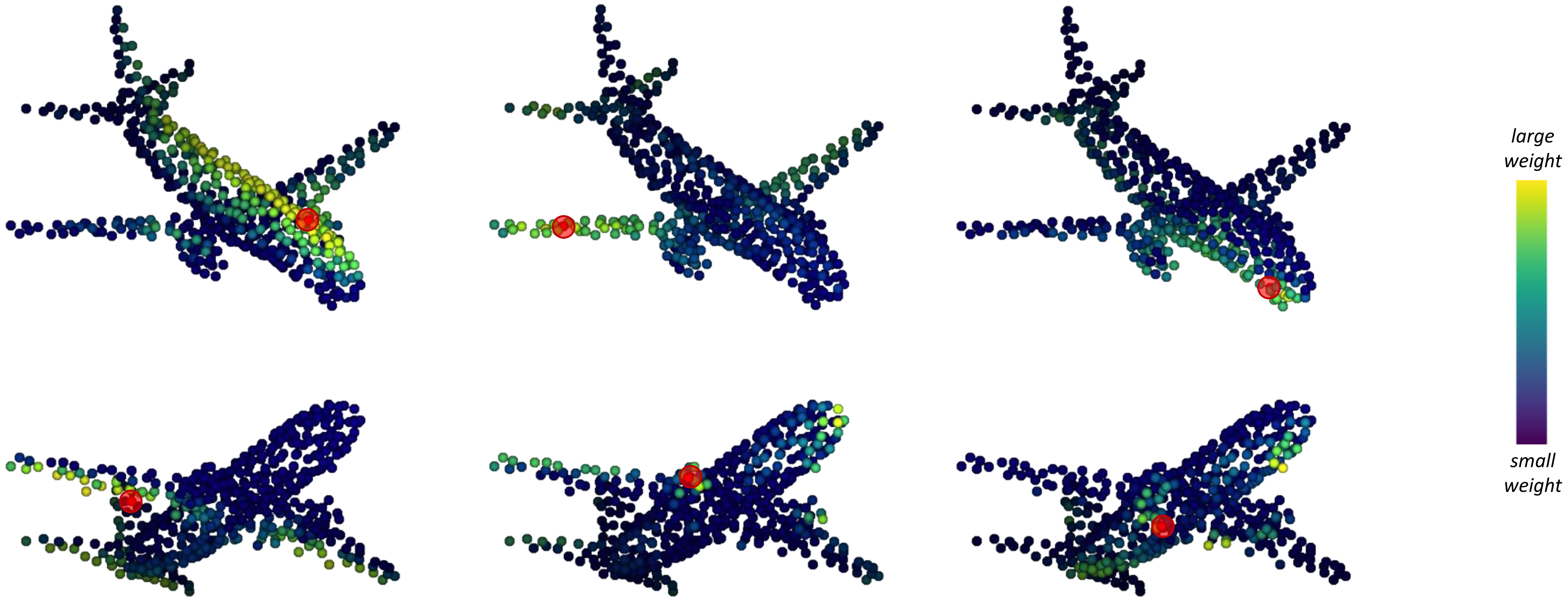
For PointNet without GLM, we use the inner-product of point features as the graph weights to show the function of GLM module

Learned Graph from Classification Task (Shelf)



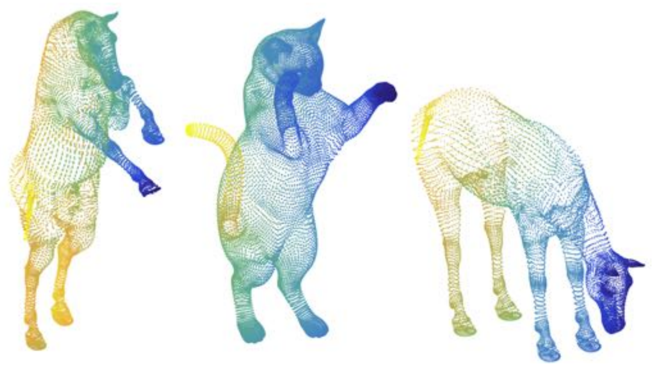
Edge weight distribution of the red point,
learned from ModelNet40 classification task

Learned Graph from Classification Task (Plane)



Edge weight distribution of the red point,
learned from ModelNet40 classification task

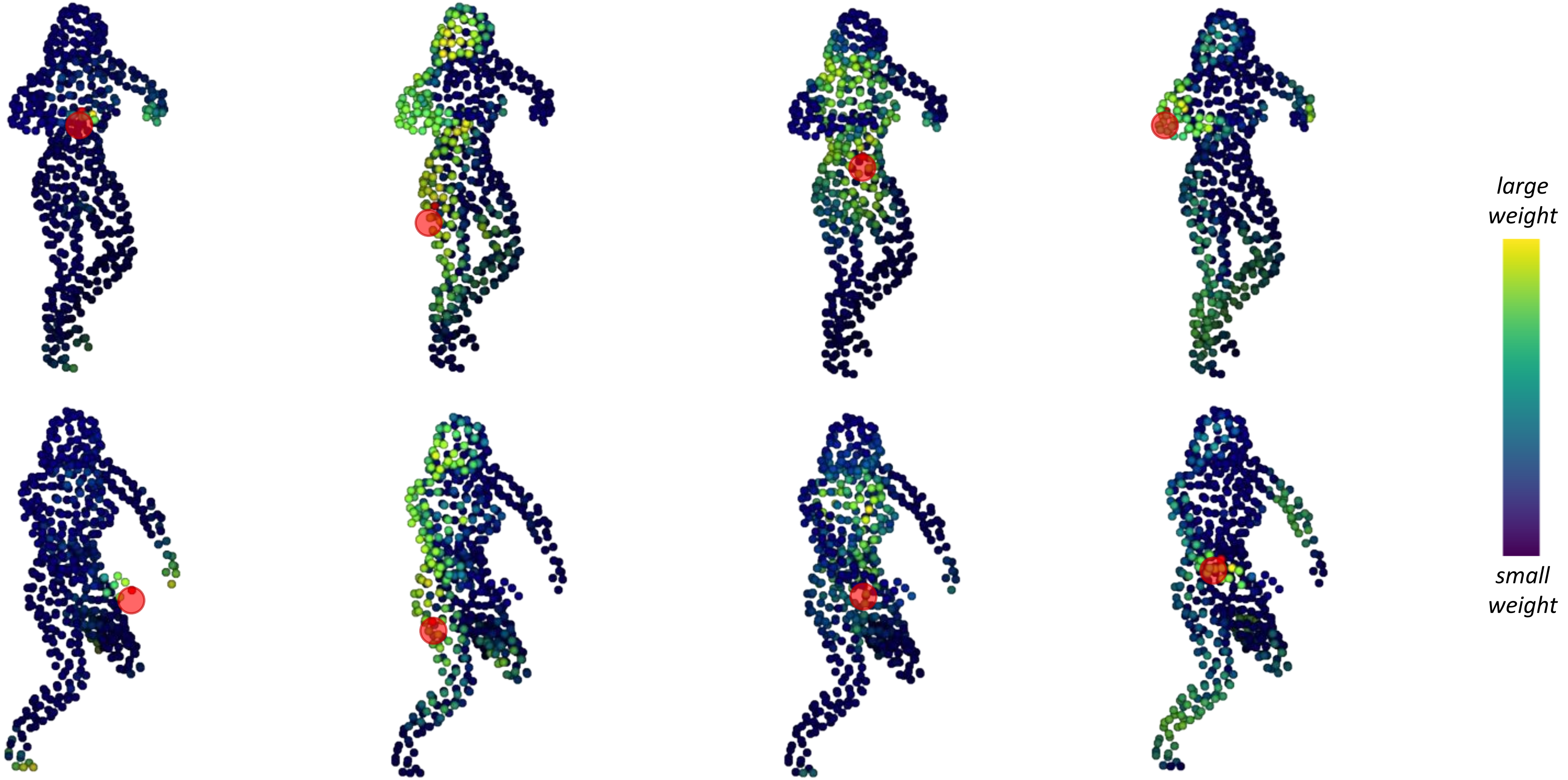
SHREC15 Non-Rigid Shape Classification



Task	SHREC15 Non-Rigid Shape Classification (Accuracy)
2-layer PointNet++ without GLM	96.8%
2-layer PointNet++ with GLM	**99.0%**

Two models share all the hyperparameters except for the GLM module

Non-rigid Graph Structure



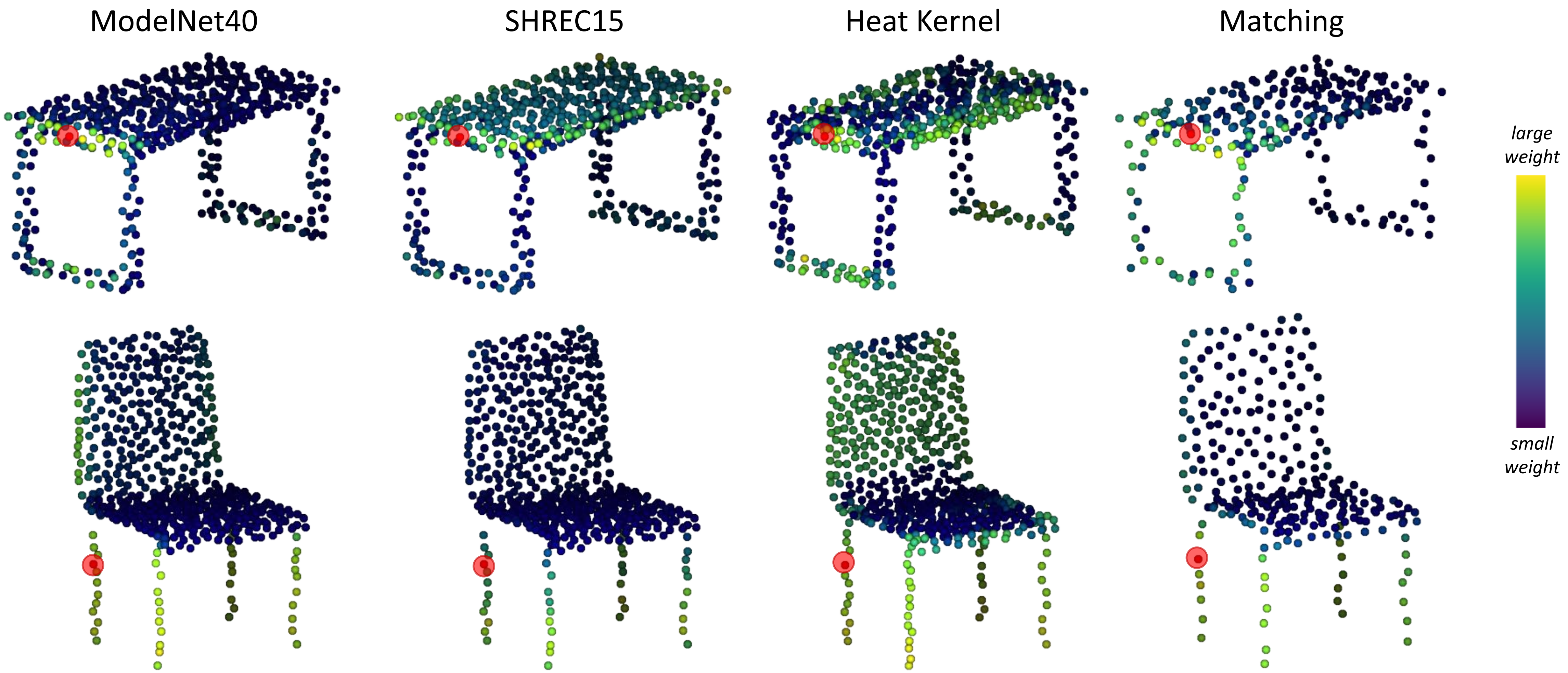
Edge weight distribution of the red point,
learned from SHREC15 *classification task*

Experimental Results

Task	ModelNet40 Classification (Accuracy)	SHREC15 Non-Rigid Shape Classification (Accuracy)	ShapeNet Part Segmentation (Mean IoU)	Heat Kernel Signature Regression (L2 Loss)	Matching (Contrastive Loss)
2-layer PointNet++ without GLM	90.3%	96.8%	84.8%	0.03744	0.1087
2-layer PointNet++ with GLM	**90.9%**	**99.0%**	**85.0%**	**0.02425**	**0.1044**

Two models share all the hyperparameters except for the GLM module

Graph Structure among Tasks



Ablation Study on Diffusion Steps

Task	Heat Kernel Signature Regression (L2 Loss)
2-layer PointNet++ without GLM	0.03744
2-layer PointNet++ with GLM (1 step)	0.03044
2-layer PointNet++ with GLM (3 steps)	**0.02425**

All models share all the hyperparameters
except for the GLM module

Thanks!

Zhuohan Li