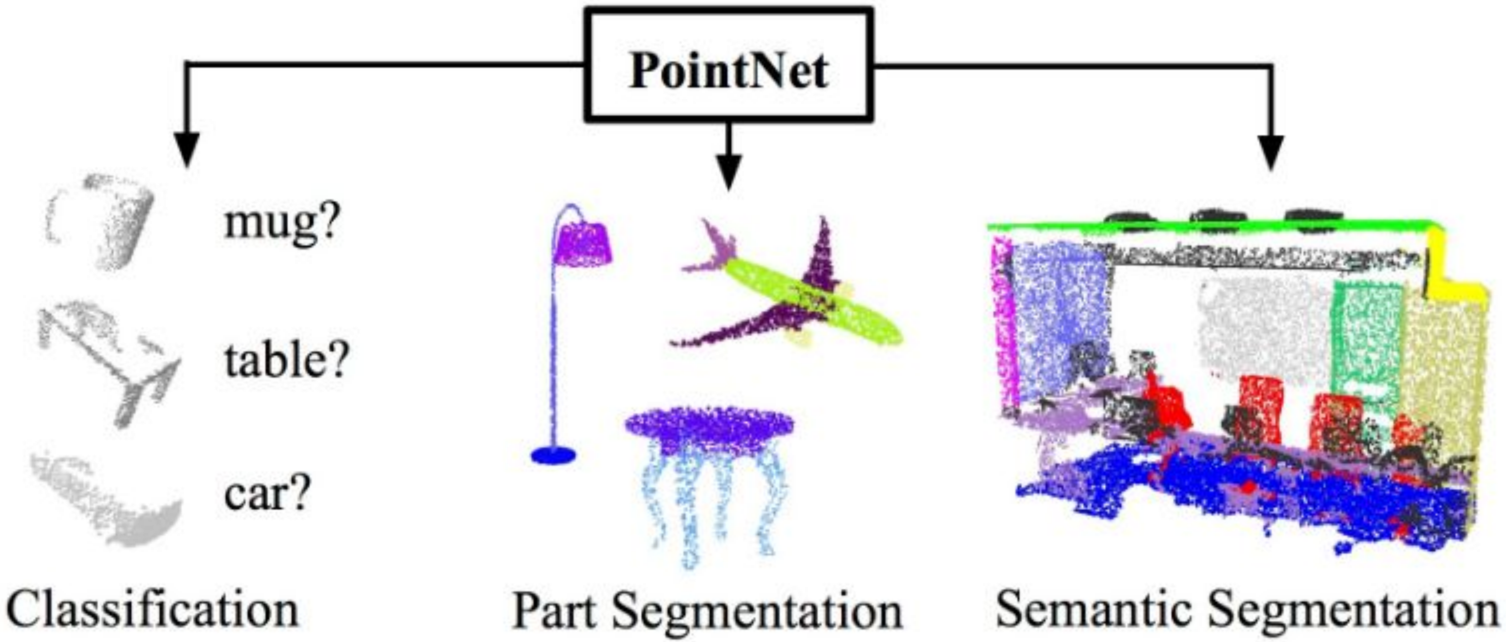
# PointNet (CVPR 2017)

Coming up: PointNet++ , TangentConv, SplatNet, FCGF

Kim Seung Wook \ CVLAB

# Abstract

- Point clouds have an irregular format
  - usually transformed to 3D voxel grids / image collections
  - ...which renders data unnecessarily voluminous and causes issues
- Introduces a network that **directly consumes** point clouds
  - well respects the permutation invariance of points
  - normalized into unit sphere
- Unified architecture for:
  - Object classification
  - Part segmentation
  - Semantic segmentation


PointNet
mug?
table?
car?
Classification
Part Segmentation
Semantic Segmentation

# Key contributions

1. A novel deep architecture suitable for consuming unordered point sets in 3D
2. Show how such a net can be trained to perform 3D shape classification, shape part segmentation and scene semantic parsing tasks
3. Empirical and **theoretical** analysis on the stability and efficiency of our method
4. Illustrate 3D features computed by selected neurons in the net and develop intuitive explanations for its performance (visualization, proof for (3))

# Per-task specifics

**Object Classification**

Input cloud is either:

1. Directly sampled from a shape
2. Pre-segmented from a scene point cloud

PointNet outputs K scores for all the K candidate classes

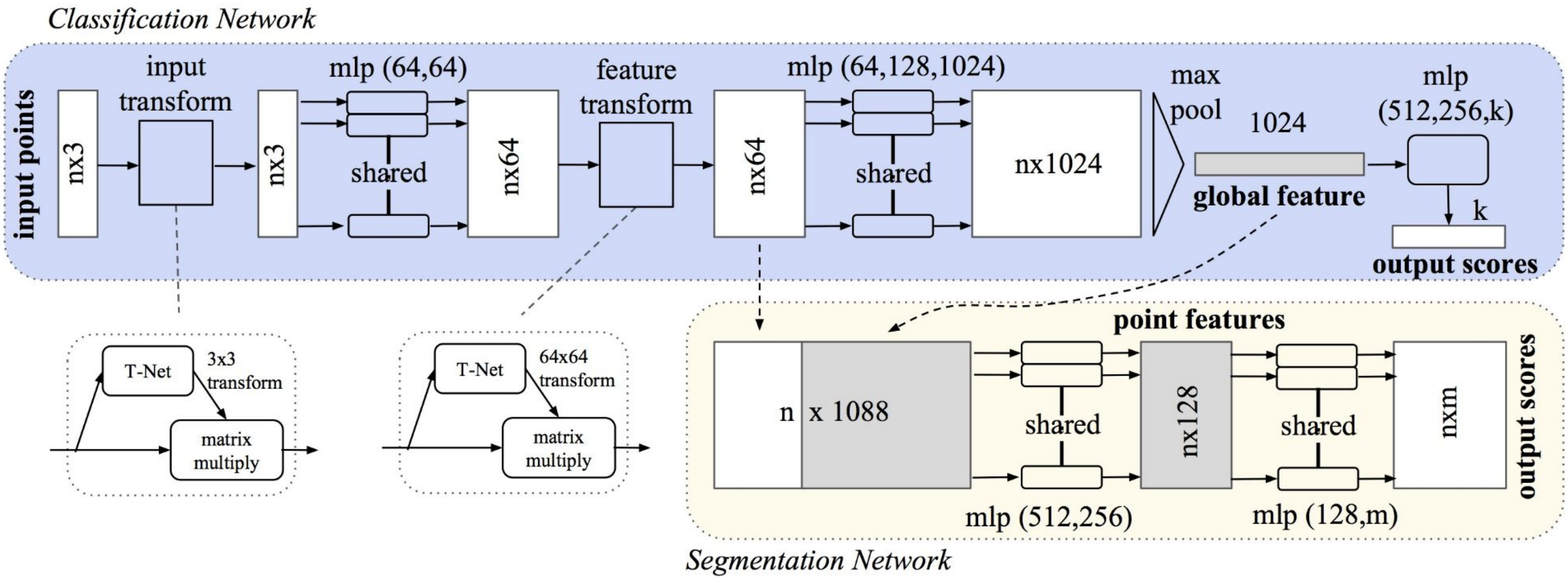
**Semantic segmentation**

Input can be either:

1. A single object for *part region segmentation*
2. A sub-volume from a 3D scene for object region segmentation

Outputs n X m scores for each of the n points, and each of the m semantic subcategories.

# PointNet Architecture




Batchnorm used in all layers with ReLU. Dropout layers are used for the last mlp in classification net

# Properties of point sets (pointclouds)

1. **Unordered**
   - Unlike pixel / voxels, there is no specific order
   - A network that consumes **N 3D point sets** needs to be invariant to **N!** permutations of input set in data feeding order
2. **Interaction among points**
   - Points are not isolated, neighboring points form a meaningful subset
   - Model needs to be able ot capture local structures
3. **Invariance under transformations**
   - Learned representation should be invariant to certain transformations

# Architecture explained

**3 key modules**

1. Max pooling layer
   - as a SYMMETRIC FUNCTION to aggregate information from all the points
2. Local and Global information combination structure
3. Two joint alignment networks
   - to align both input points and point features

# Symmetry function for unordered input

**Three strategies** to make a model invariant to input permutation:
- sorting into canonical (normalized) order
- treating input as a sequence to train RNN (and augmenting data by permutations
- using a simple symmetric function to aggregate info from each point

sorting and RNN are “plausible”, but not the best choices
- proved through theory and experiments

**IDEA:** approximate a general function defined on a point set by applying a symmetric function on transformed elements in a set.

# Symmetry function for unordered input

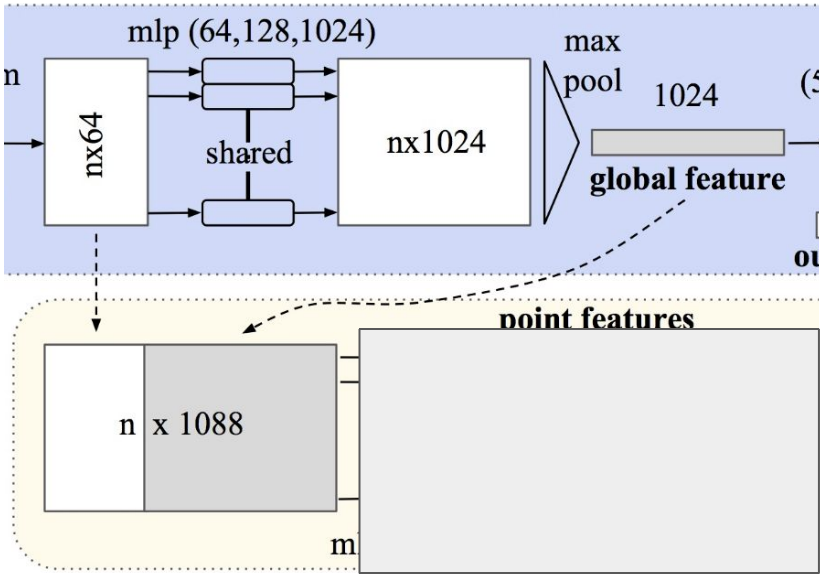
$$f(\{x_1, \ldots, x_n\}) \approx g(h(x_1), \ldots, h(x_n)), \qquad (1)$$

where $f : 2^{\mathbb{R}^N} \rightarrow \mathbb{R}$, $h : \mathbb{R}^N \rightarrow \mathbb{R}^K$ and $g : \underbrace{\mathbb{R}^K \times \cdots \times \mathbb{R}^K}_{n} \rightarrow \mathbb{R}$ is a symmetric function.

approximate *h* by MLP
approximate *g* by composition of single variable function and max pooling function

Through collection of h, we can learn a number of *f*s to capture different properties of point set

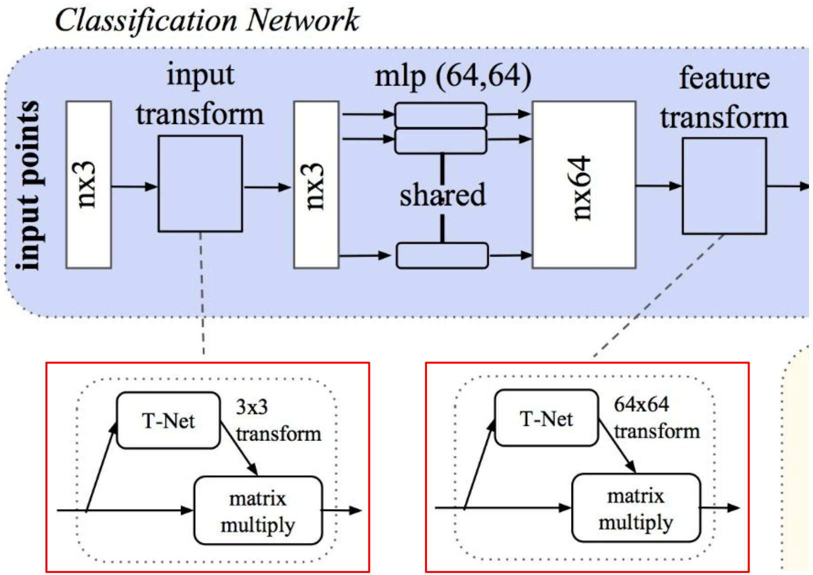
# Local and Global Information Aggregation




Concat the global feature with EACH of the point features

Aware of both local and global information
-> Can predict based on both *local geometry* and *global sementics*

# Joint Alignment Network




Align all inputs sets to a canonical space before feature extraction

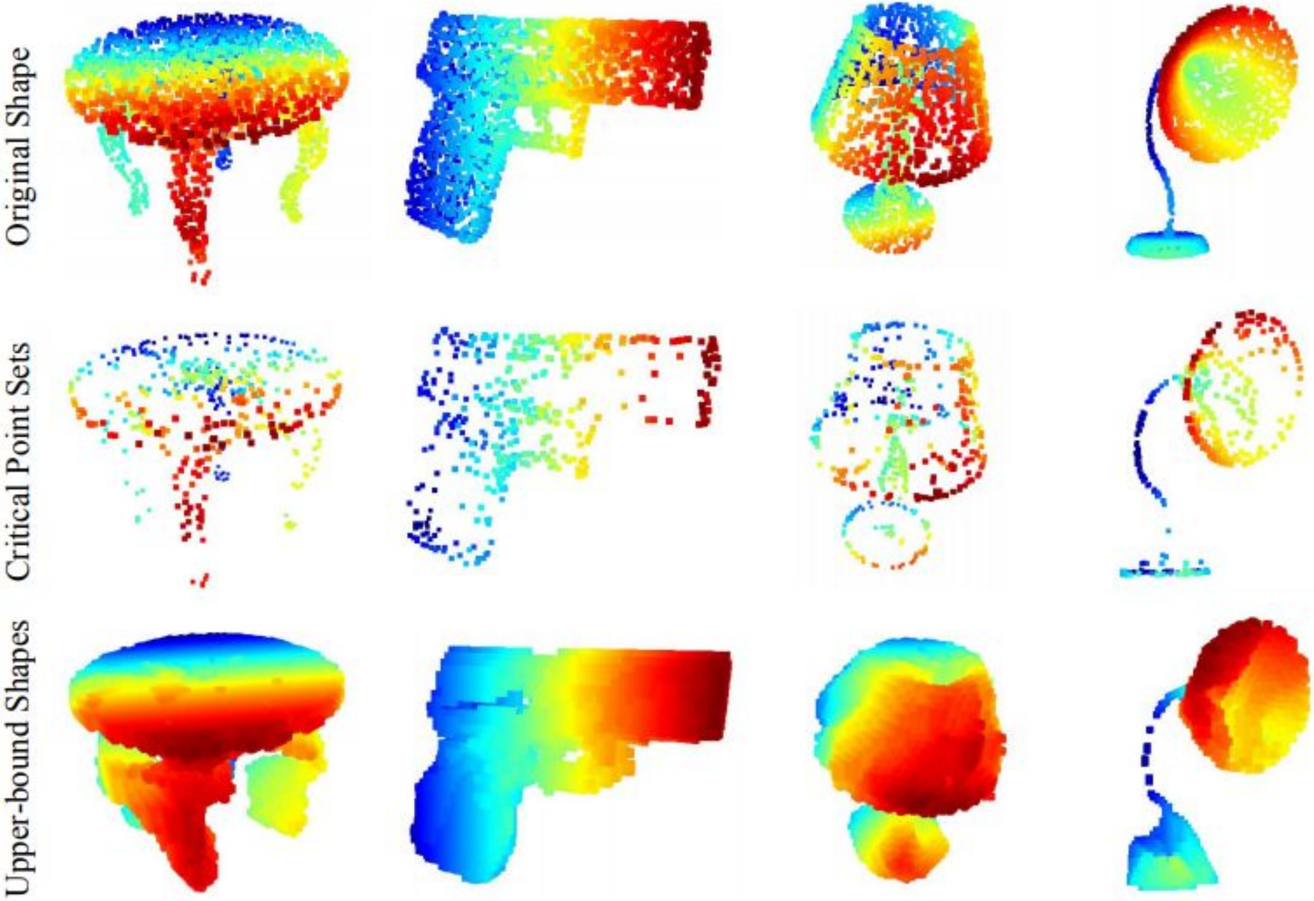
Also applied to features, to align features from different input points clouds
- Transformation matrix has much higher dimension in feature space, so regularizer term is used to **constrain the feature transformation to be close to orthogonal matrix** -> optimization becomes more stable

$$L_{reg} = \|I - AA^T\|_F^2,$$

# Theoretical Analysis

Application of function Analysis (해석학) to prove that PointNet can successfully approximate our target function *f*

- Uses epsilon-delta strategy
- Proves that as long as the neurons at the max pooling layer (K) is sufficiently high, can approximate *f*
- More details in the supplementary material.


Original Shape
Critical Point Sets
Upper-bound Shapes