

# Homework 3 Released

- Find it on Piazza
- Write a 3D object detector on the same dataset as HW2
- We still have a leaderboard, and rubric protocol is similar to HW2
- Due: March 8, 23:59PM
- Network training could take more than a day. Please start early.

# L15: Zero-shot 3D Understanding

Hao Su

Thank Dr. Songfang Han for helping to prepare slides

# Many Successes of Machine Learning

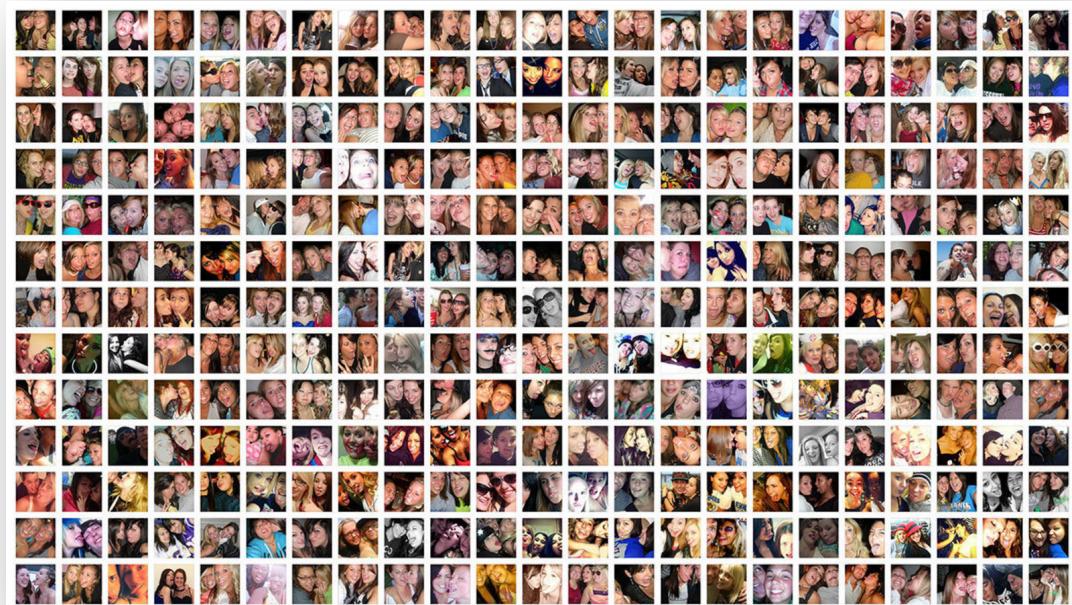


Image classification  
Speech understanding  
3D shape classification

...



# Made Possible By ....



Lots of data

For i.i.d samples between train/  
test sets, even generalize!

Lots of computing power



# Common Framework

**For each task:**

Dataset:  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$

Learner:  $y = f(x; \theta)$

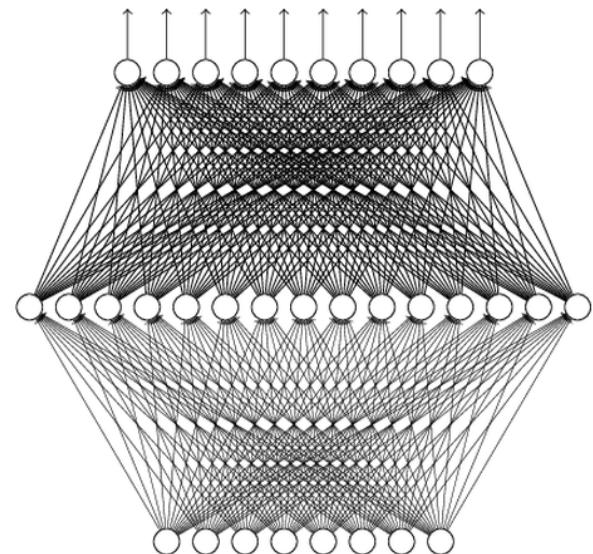
Referee (loss function):  $L(y, y')$

**Universality Theorem:**

Any continuous function  $f$

$$f : R^N \rightarrow R^M$$

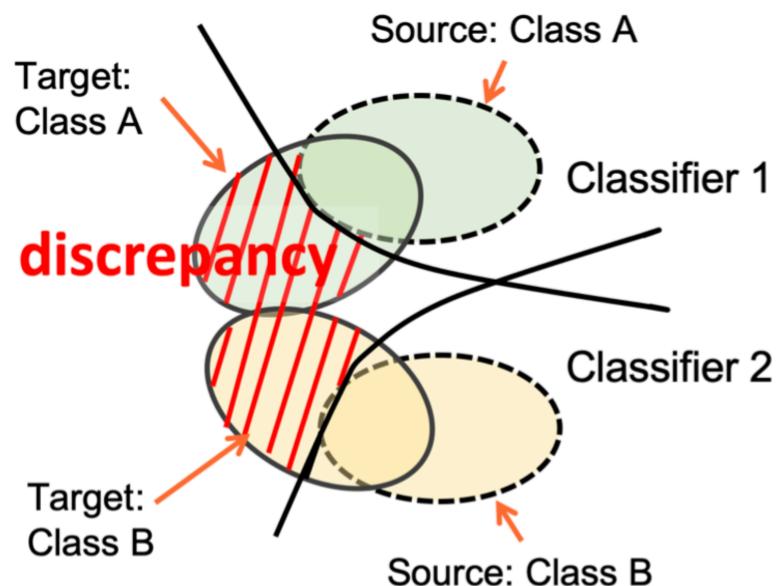
can be realized by a network



# Discrepancy between Training and Test

- Source Domain  $\sim P_S(X_S, Y_S)$ , lots of labeled data
- Target Domain  $\sim P_T(X_T, Y_T)$ , unlabeled or limited labels

Discrepancy between training and test set causes bad performance.



Can We Transfer 3D Understanding from  
Training Category to **Unseen** Categories?

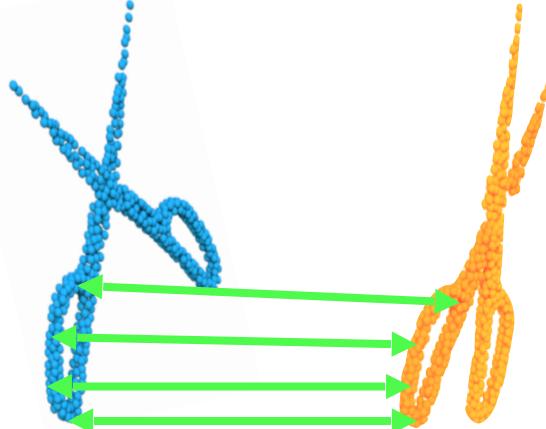
# Agenda

- Correspondence-based Part Segmentation
- Compositional Generalizability View of Zero-shot Part Segmentation (*read by yourself*)

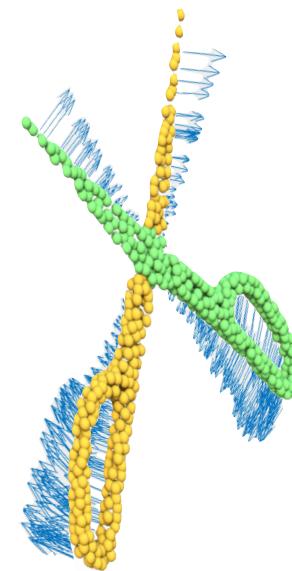
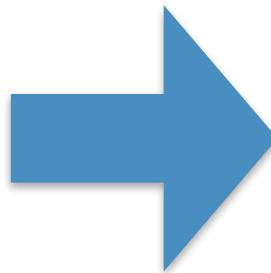
# **Correspondence-based Part Segmentation**

# Motivation

- **Key observation:** Correspondence assumes no object and categories information and focus on local regions.



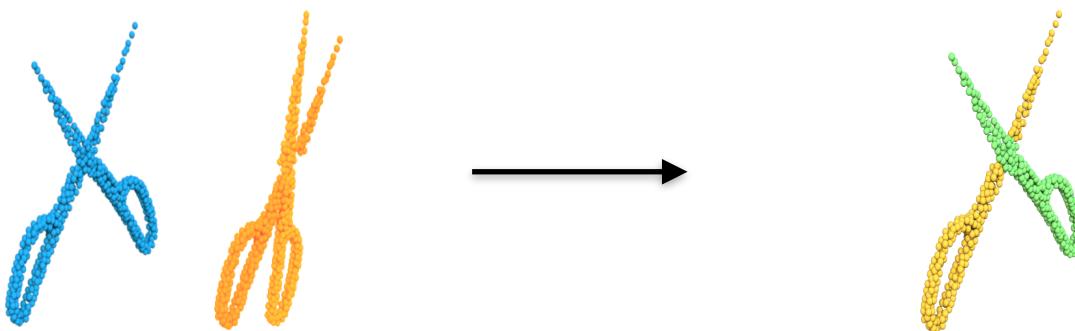
*Correspondence*



*Segmentation*

Correspondence is more **Generalizable** to novel instances and novel object categories.

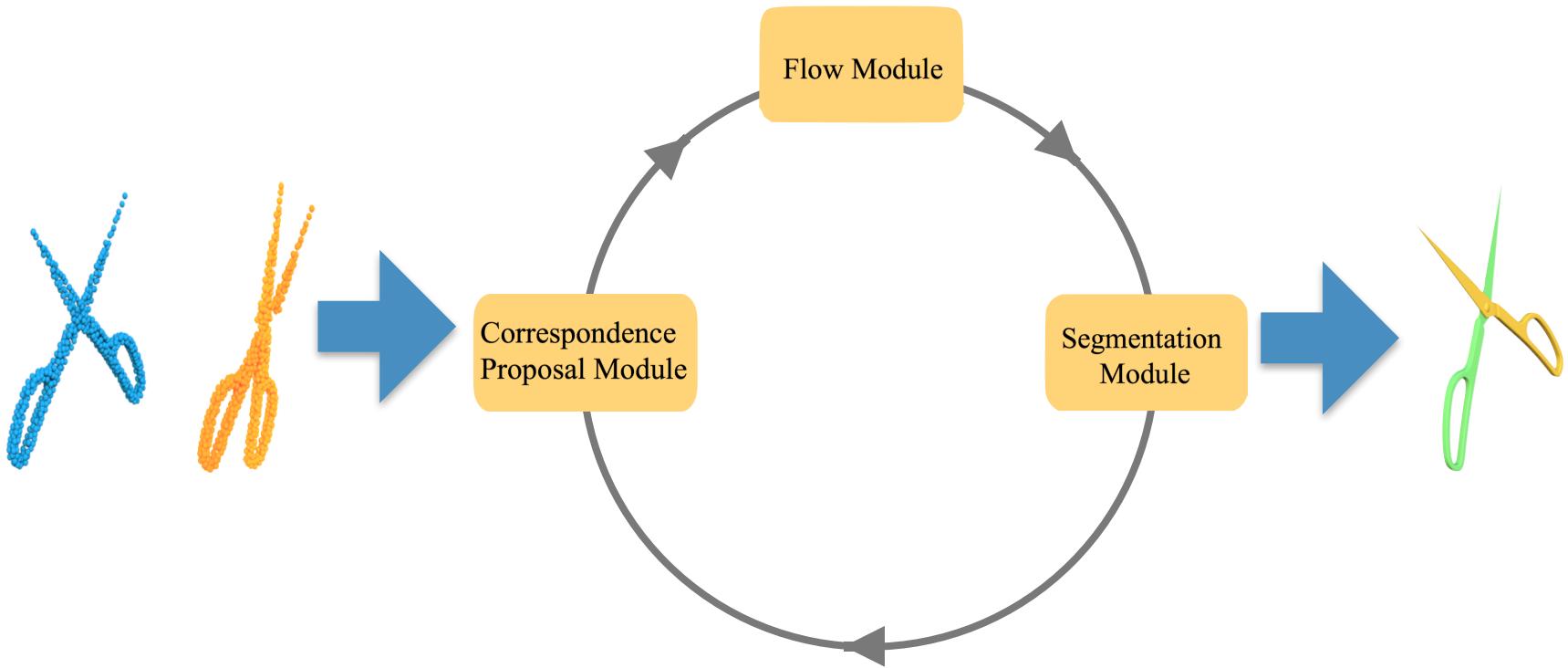
# Problem Definition: Two Frame Reasoning



Input: two point sets with different  
articulation states

Output: part segmentation

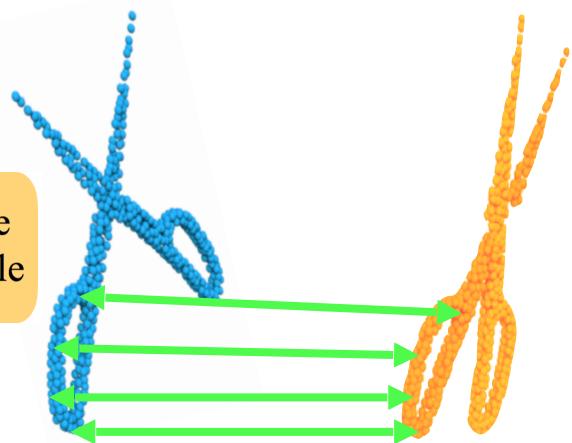
# Overall Pipeline



# Correspondence Proposal Module

Input

Point cloud 1



Correspondence  
Proposal Module

Point cloud 2

Output

Points in  
Point cloud 2

Points in Point  
cloud 1

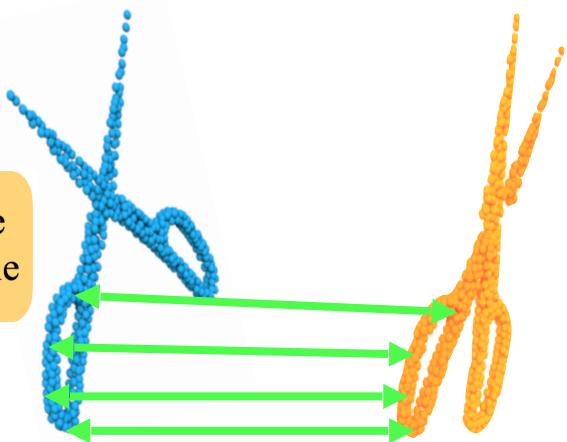
Matching  
probability

# Correspondence Proposal Module

Input

Point cloud 1

Point cloud 2



Output

Points in  
Point cloud 2

Points in Point  
cloud 1

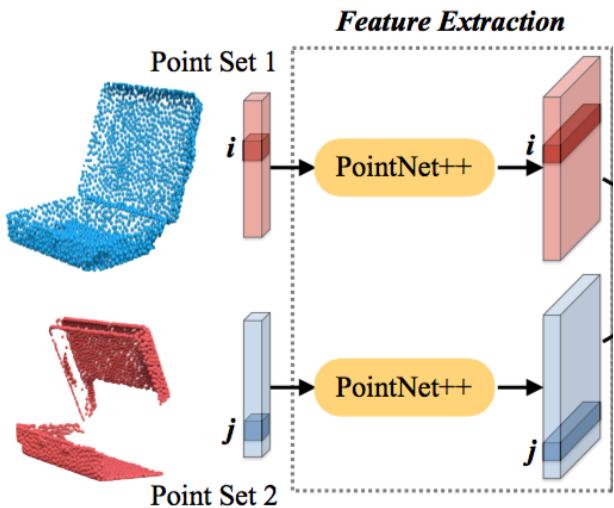
Matching  
probability

Idea:

- extract features of every point in p.c.1 & p.c.2
- compare each pair of points across p.c.1 & p.c.2

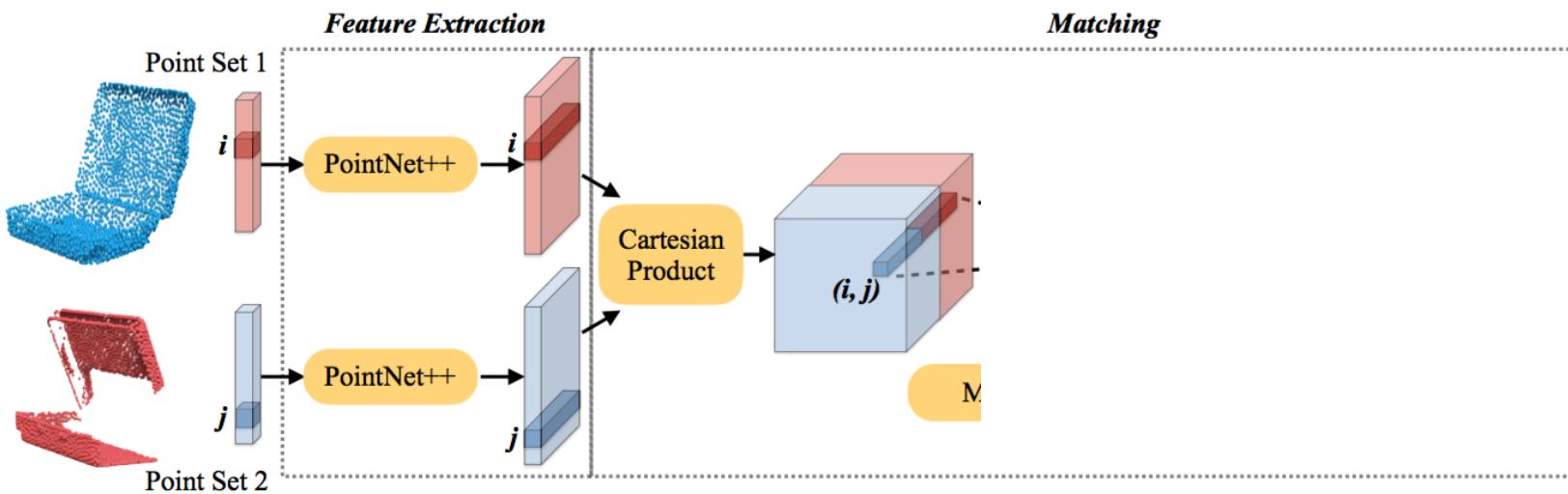
# Correspondence Proposal Module

- Extract point feature with PointNet++ for point set 1 and point set 2 separately



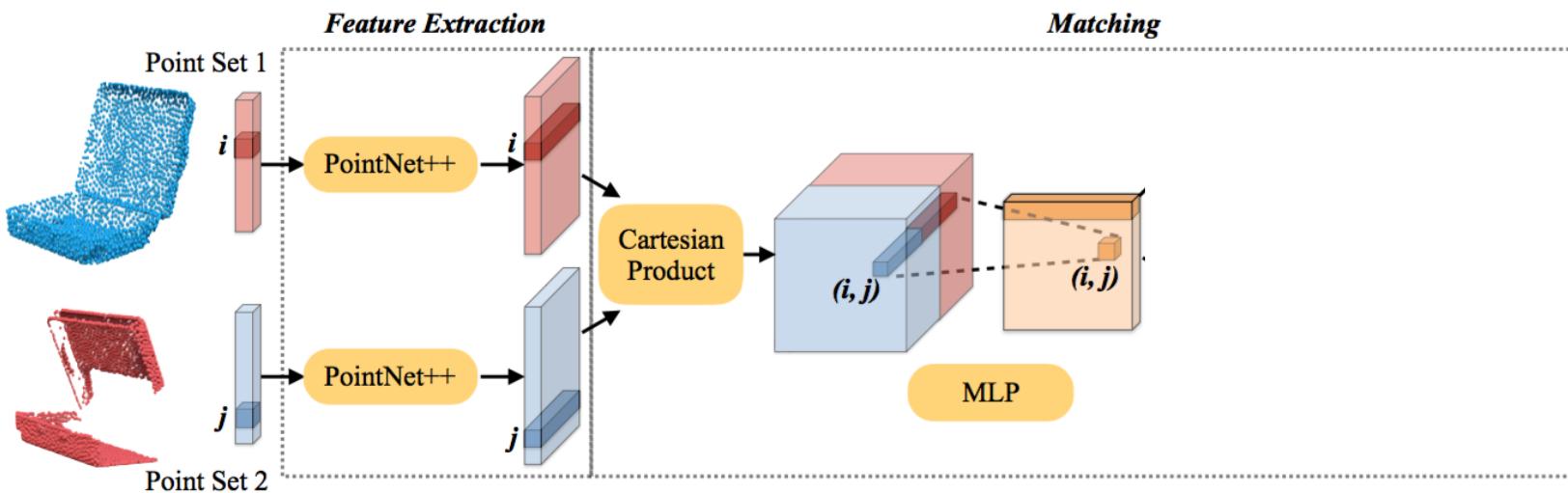
# Correspondence Proposal Module

- For each pair, concatenate the extracted point feature



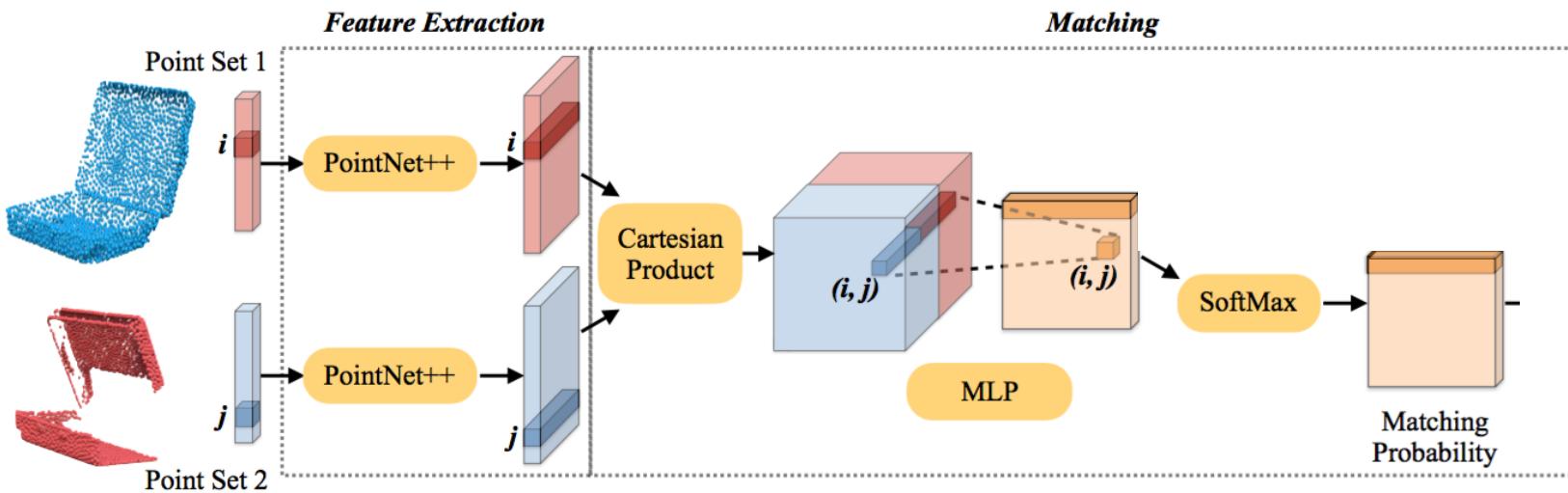
# Correspondence Proposal Module

- For each pair, predict how likely the two points match



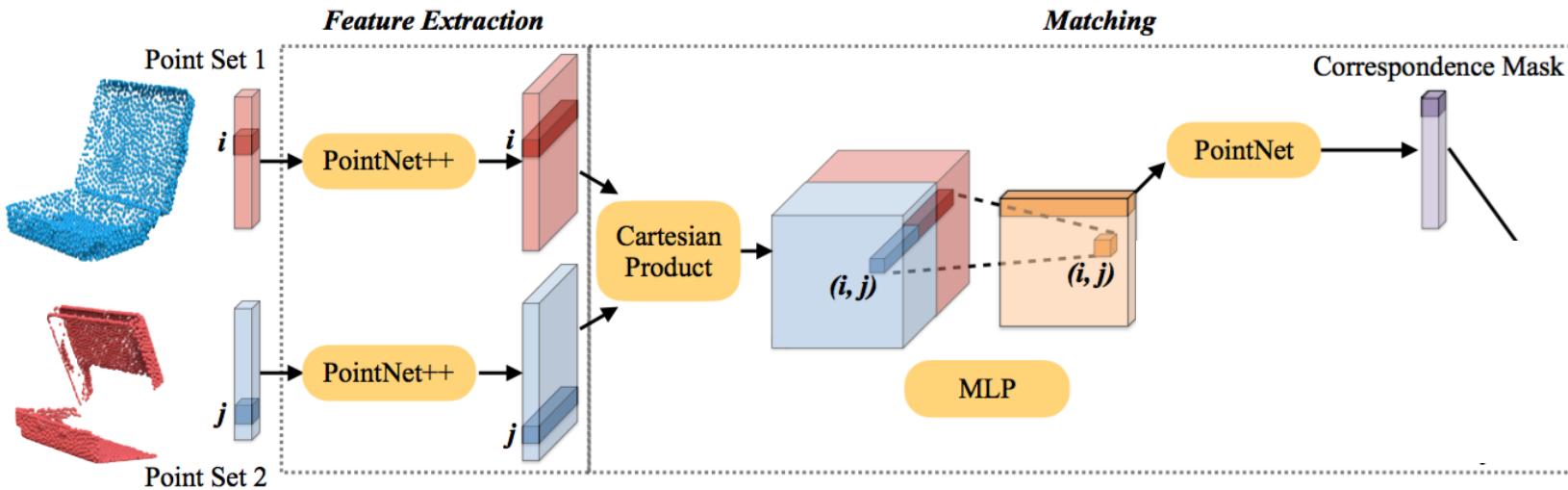
# Correspondence Proposal Module

- Convert the confidence value into a probability with SoftMax operation along each row



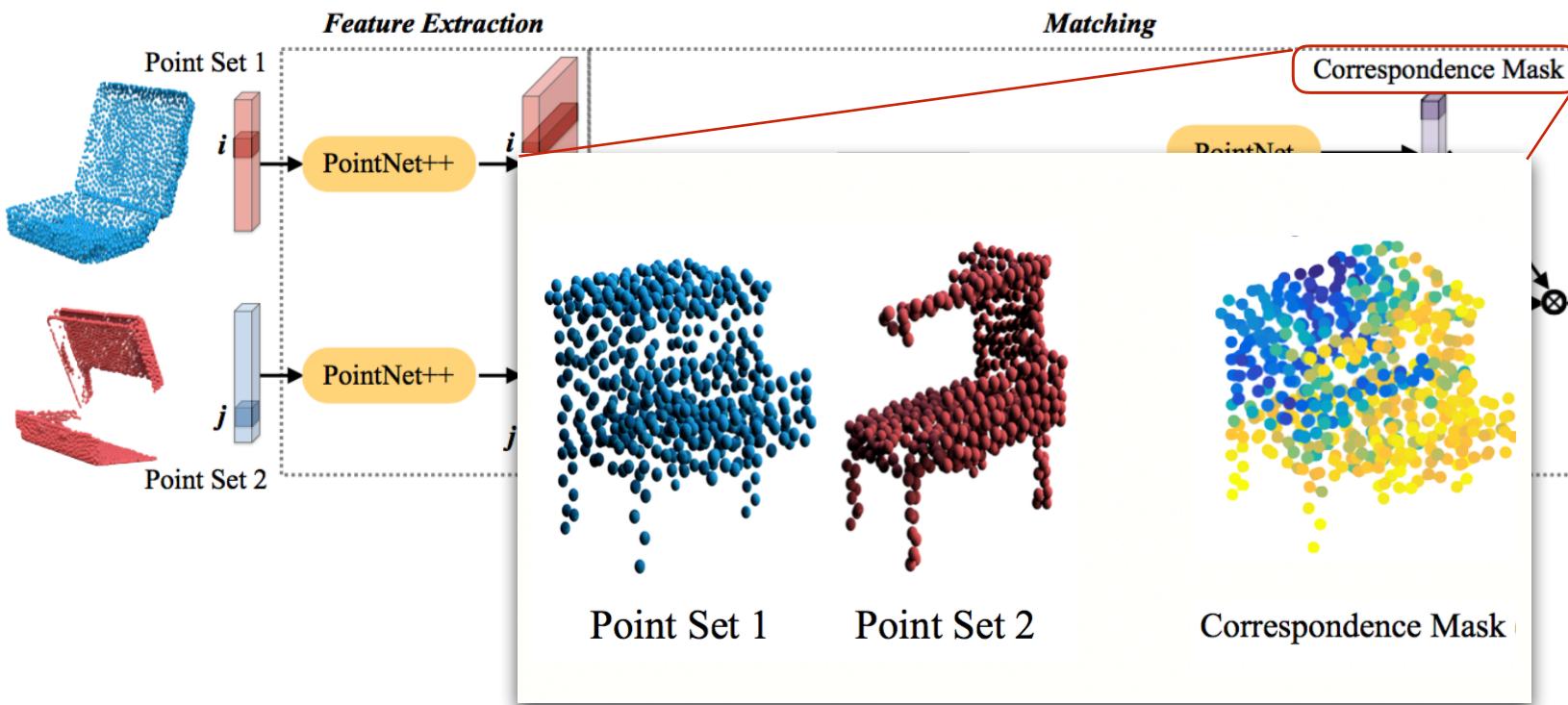
# Correspondence Proposal Module

- Point cloud 1 may have points not in point cloud 2
- So we predict whether the corresponding point exists as a binary label



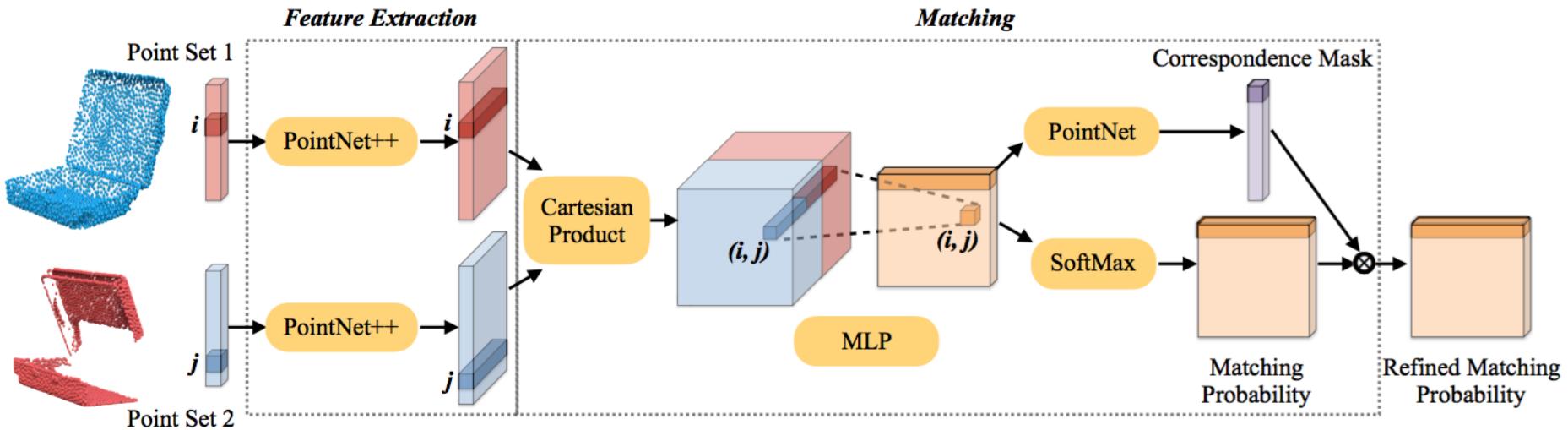
# Correspondence Proposal Module

- Visualizing the existence prediction label as a mask



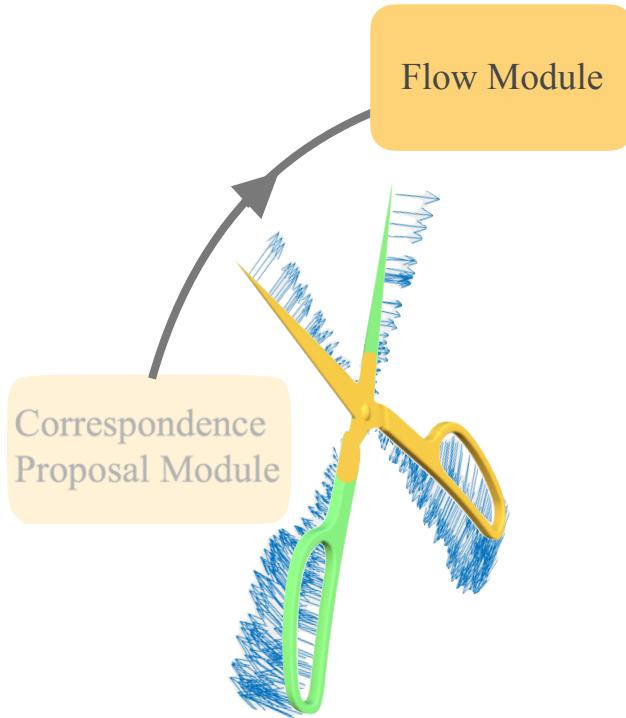
# Correspondence Proposal Module

- Downweight those rows of matching probability if the corresponding point might be missing



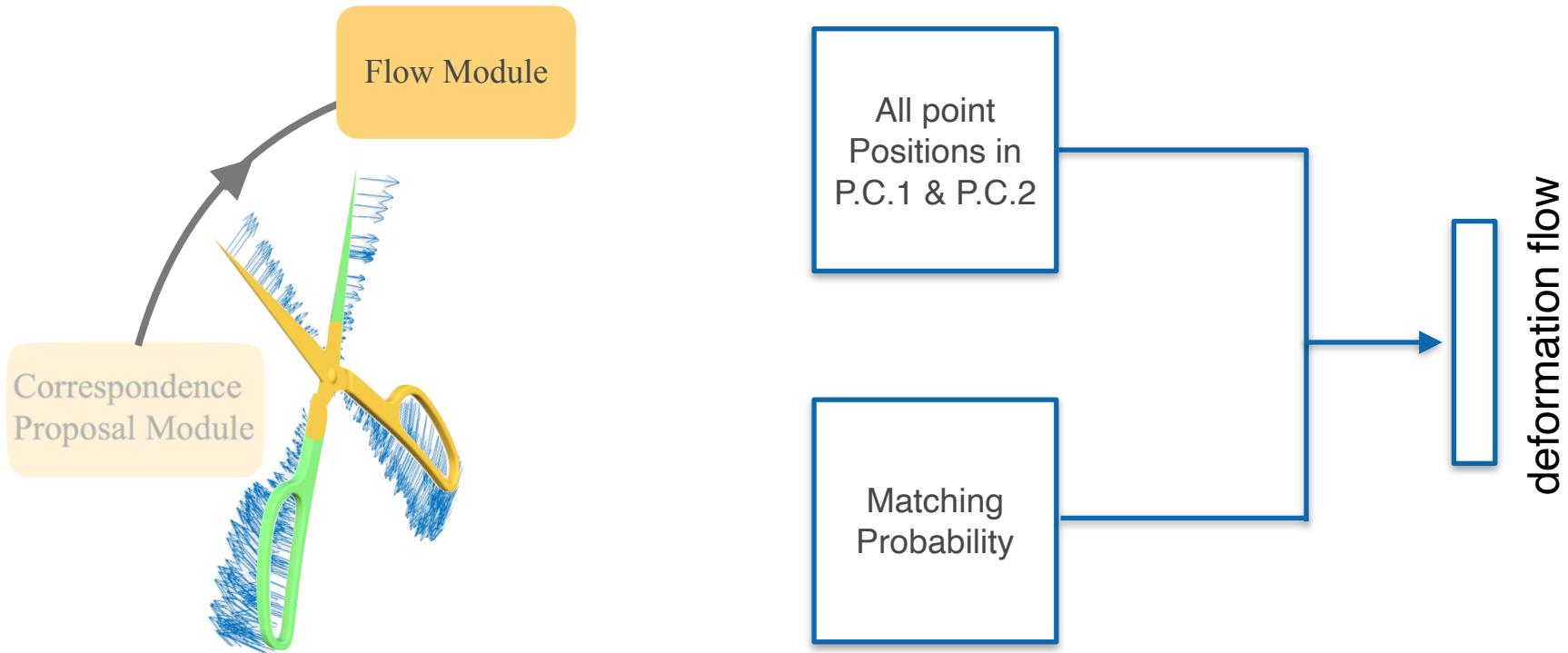
# Flow Module

- Deformation flow:
  - At each point in point cloud 1  $P$ ,
  - Associate a vector denoting the offset to the corresponding point in point cloud 2  $Q$



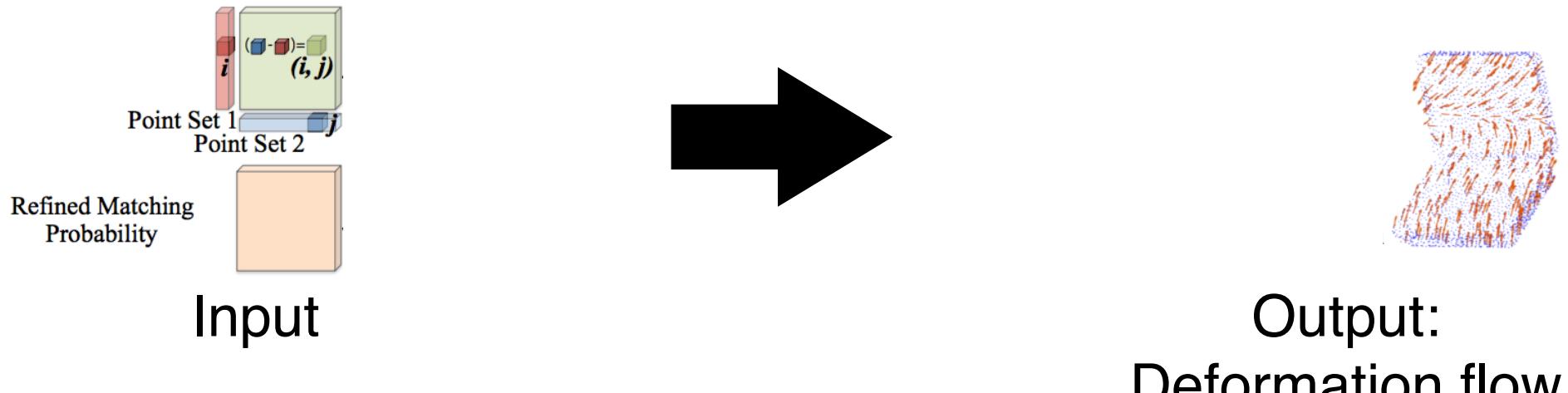
# Flow Module

- Deformation flow:
  - At each point in point cloud 1  $P$ ,
  - Associate a vector denoting the offset to the corresponding point in point cloud 2  $Q$



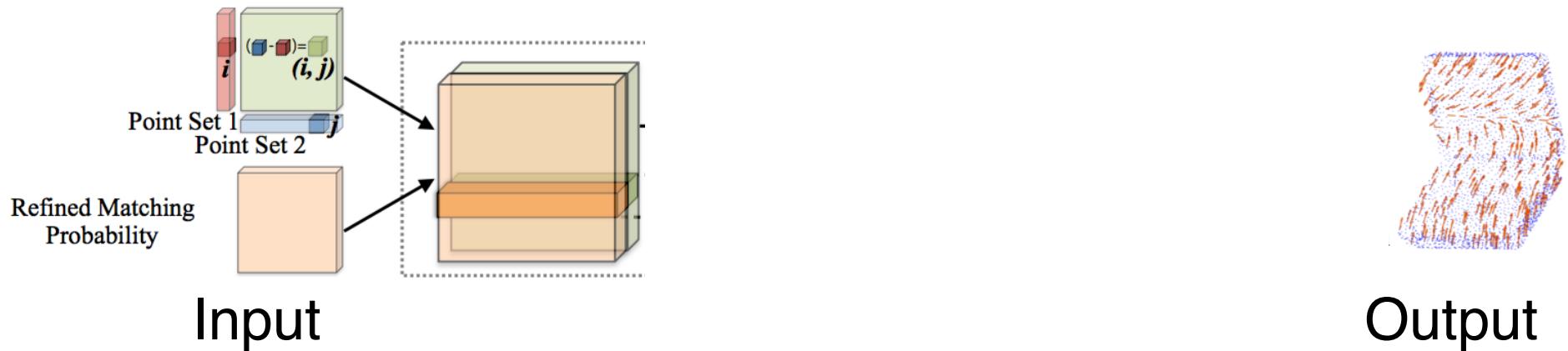
# Input and Output

- Input:
  - Matching probability matrix
  - Pairwise 3D displacement matrix  $M$ , with
$$m_{i,j} = q_j - p_i$$
- Output:
  - Displacement from point set1 to point set 2



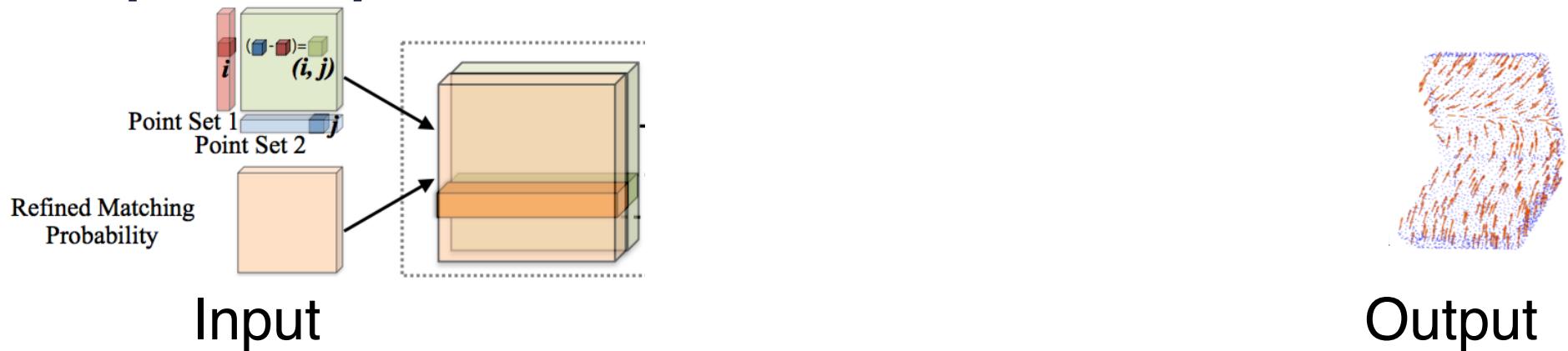
# Flow Module Architecture

- Concatenate matching probability matrix with pairwise displacement matrix
- Each row corresponds to a point in point set 1:
  - all possible displacements
  - and correspondences



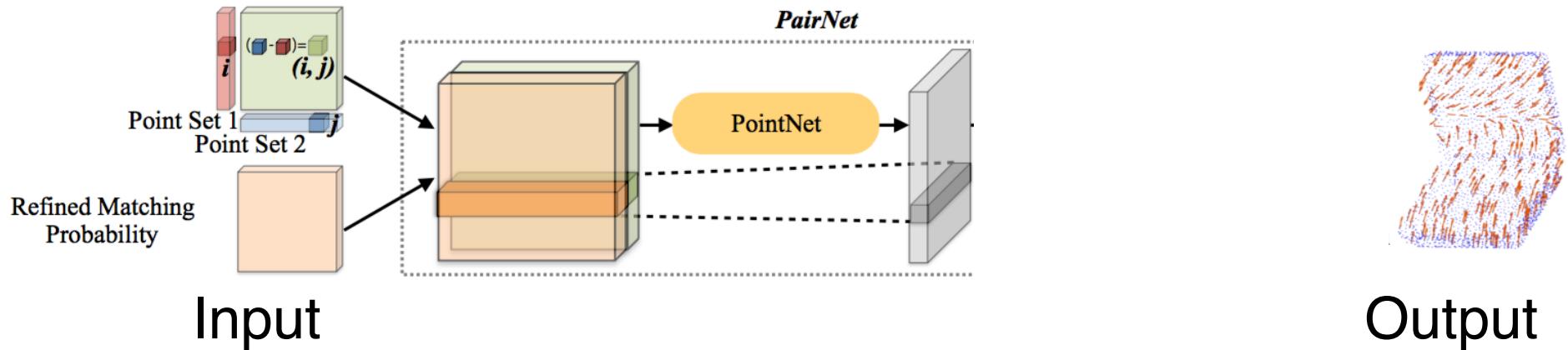
# Flow Module Architecture

- Concatenate matching probability matrix with pairwise displacement matrix
- Each row corresponds to a point in point set 1:
  - all possible displacements
  - and correspondences
- **Let us vote for the deformation vector for each point in point set 1!**



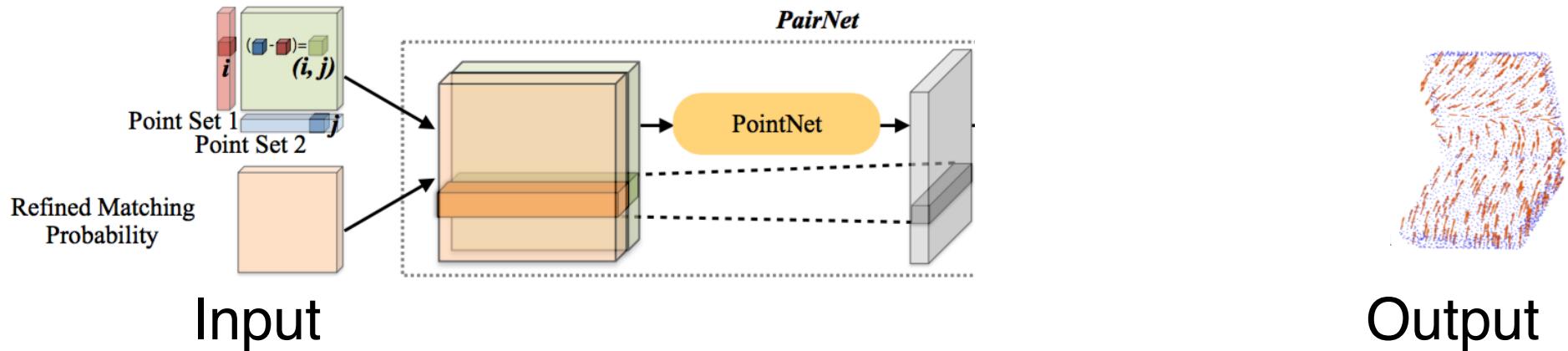
# Vote for the Deformation Vector

- Correspondence information aggregation:
  - Aggregate the information for each point in point set 1 by PointNet and store as a row vector



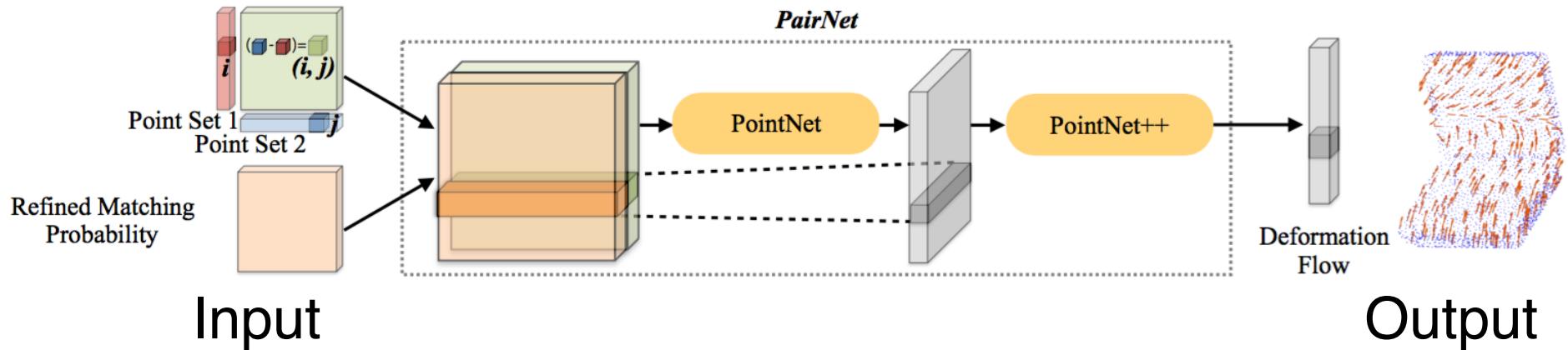
# Denoising Correspondence Information

- Note that neighboring points in point set 1 should have similar displacements
- Recall that PointNet++ will form local groups and capture local correlations

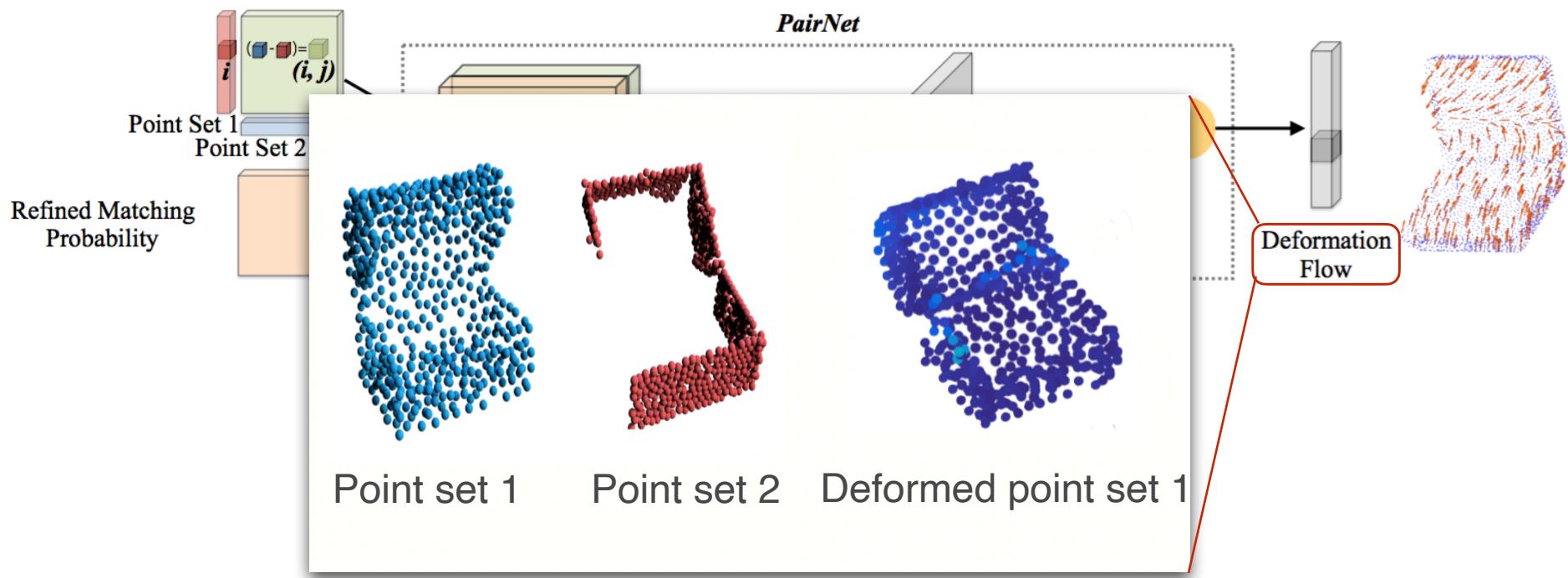


# Denoising Correspondence Information

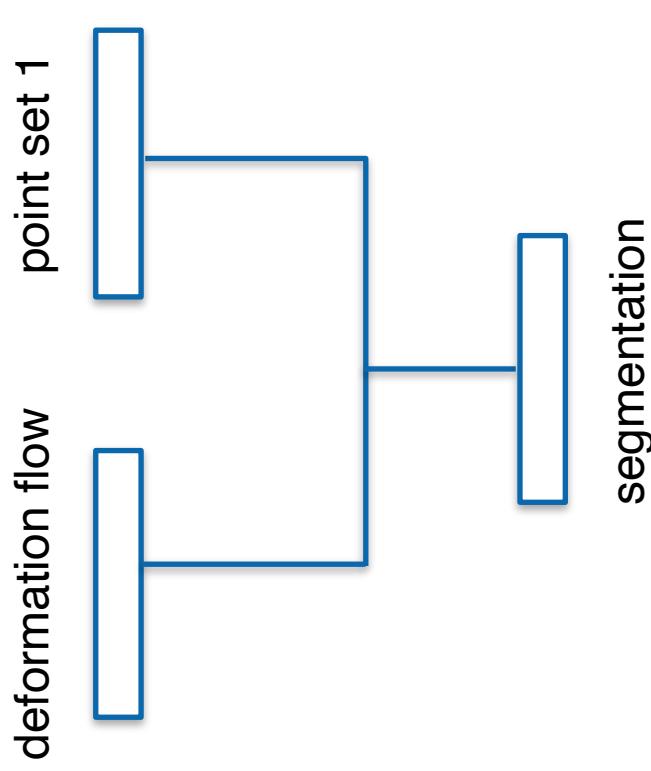
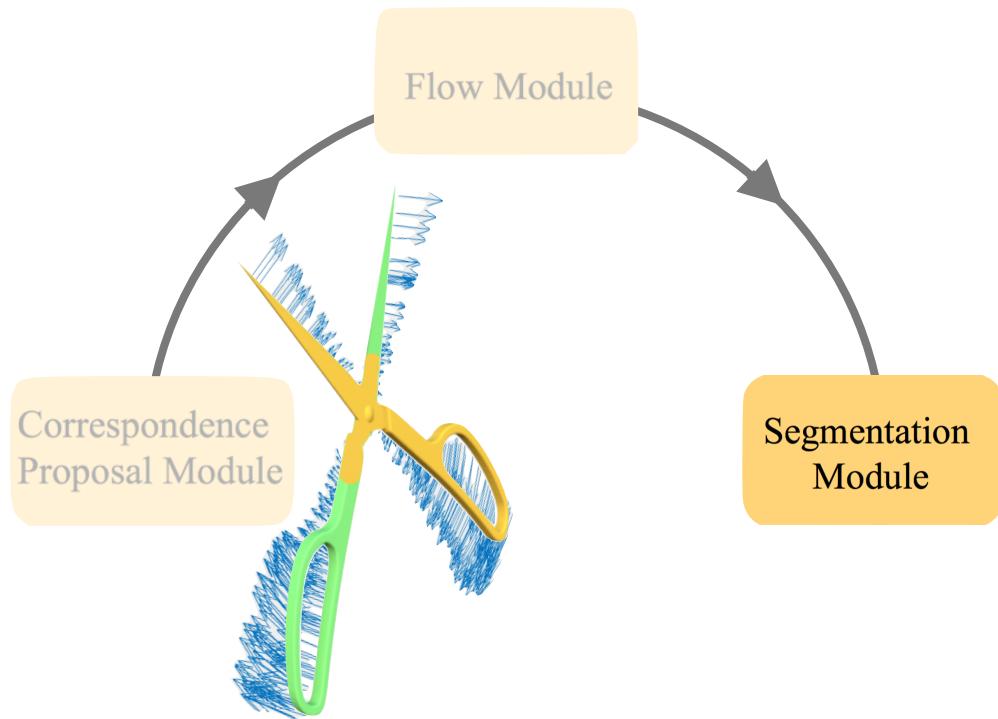
- In our correspondence information matrix
  - each row aggregates the correspondence information for each point in point set 1
- Treat each row as the features of a point, and use PointNet++ to predict a deformation vector for each point



# Visualization: Deforming Point Set 1 by Predicted Deformation Flow

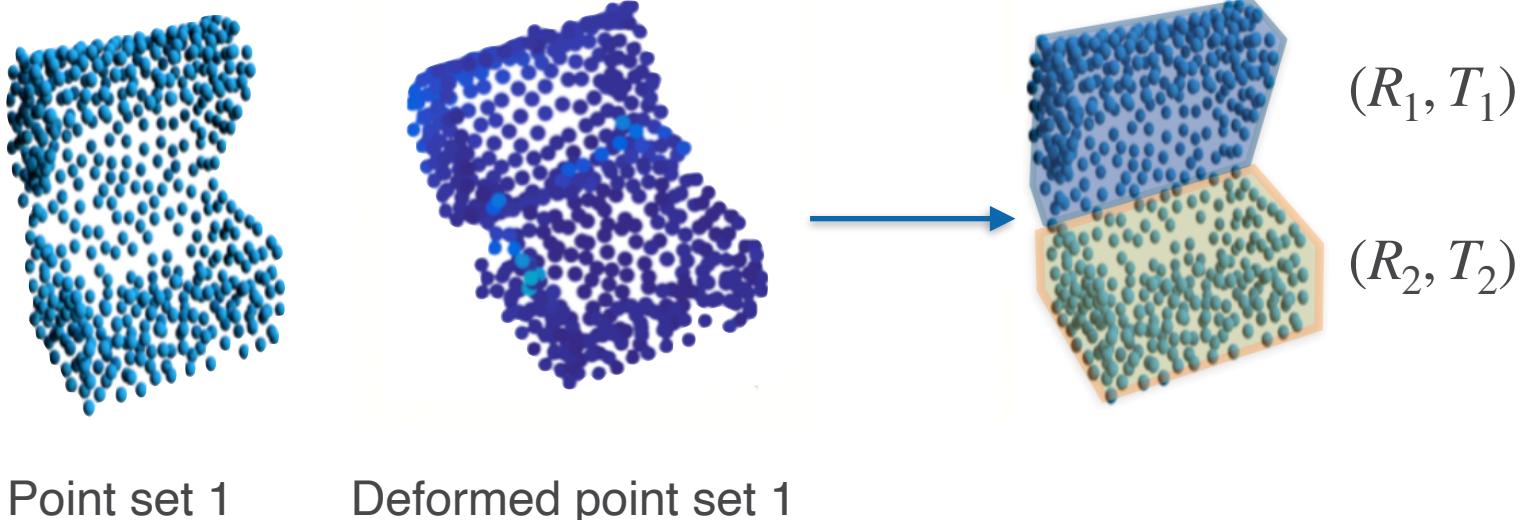


# Segmentation Module



# Segmentation Module

- How to segment parts from the deformation flow?
- Rigid part motion assumption: points from the same part share an identical transformation  $(R, T)$

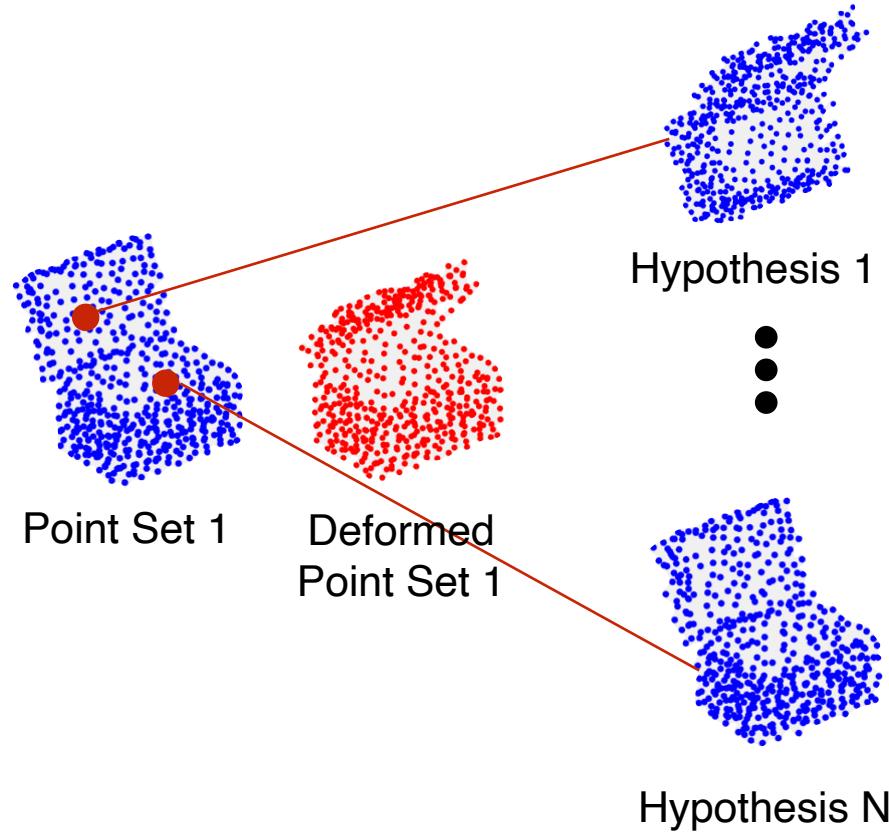


# Segmentation Module

- A neural net-based differentiable sequential RANSAC
- Three steps:
  - Rigid motion hypothesis set generation
  - Support prediction for each rigid motion hypothesis
  - Recurrent part segmentation neural network

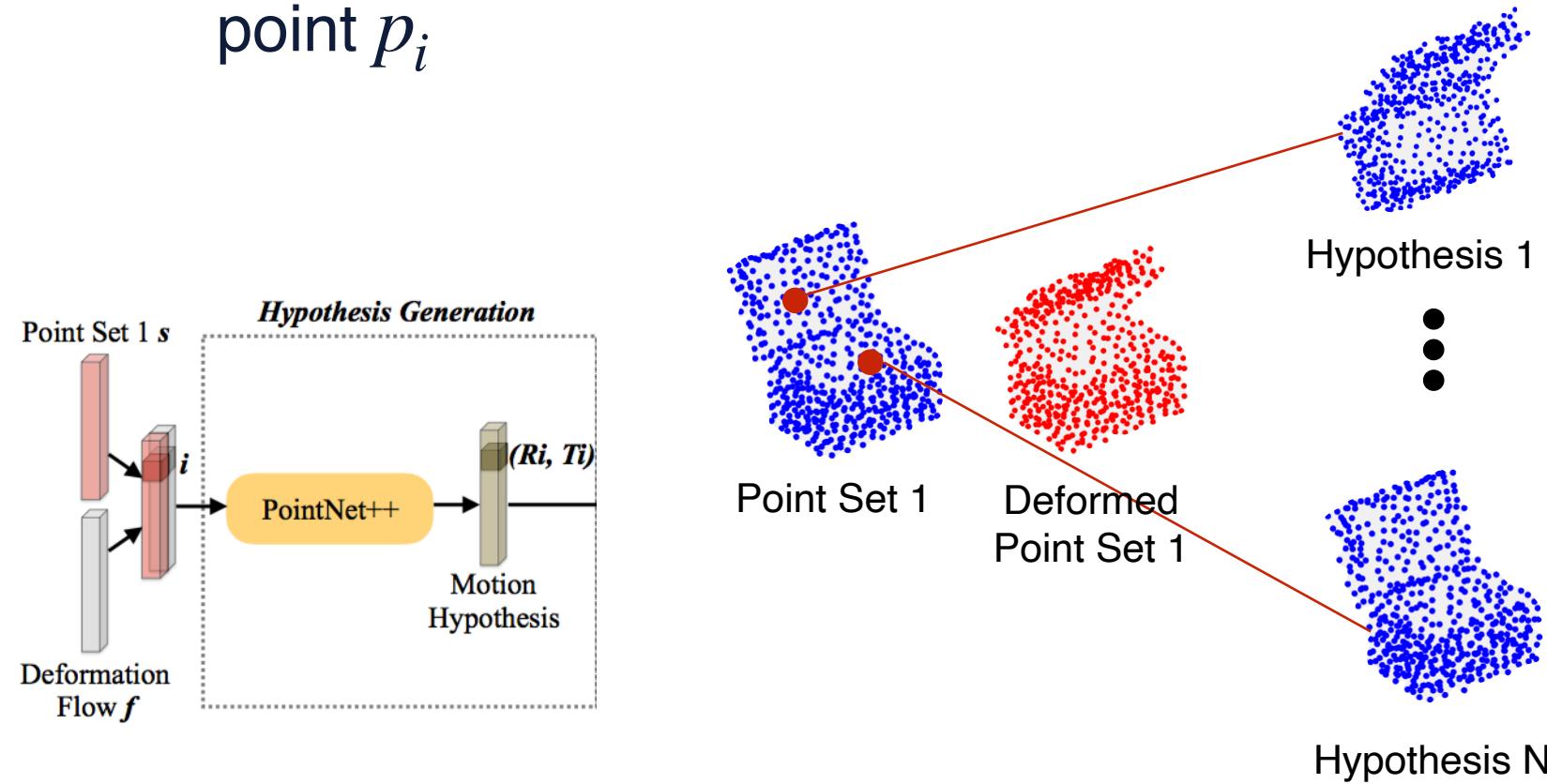
# Rigid Motion Hypotheses Generation

- Rigid motion hypothesis set generation
  - Compute the motion hypothesis  $(R_i, T_i)$  for each point  $p_i$



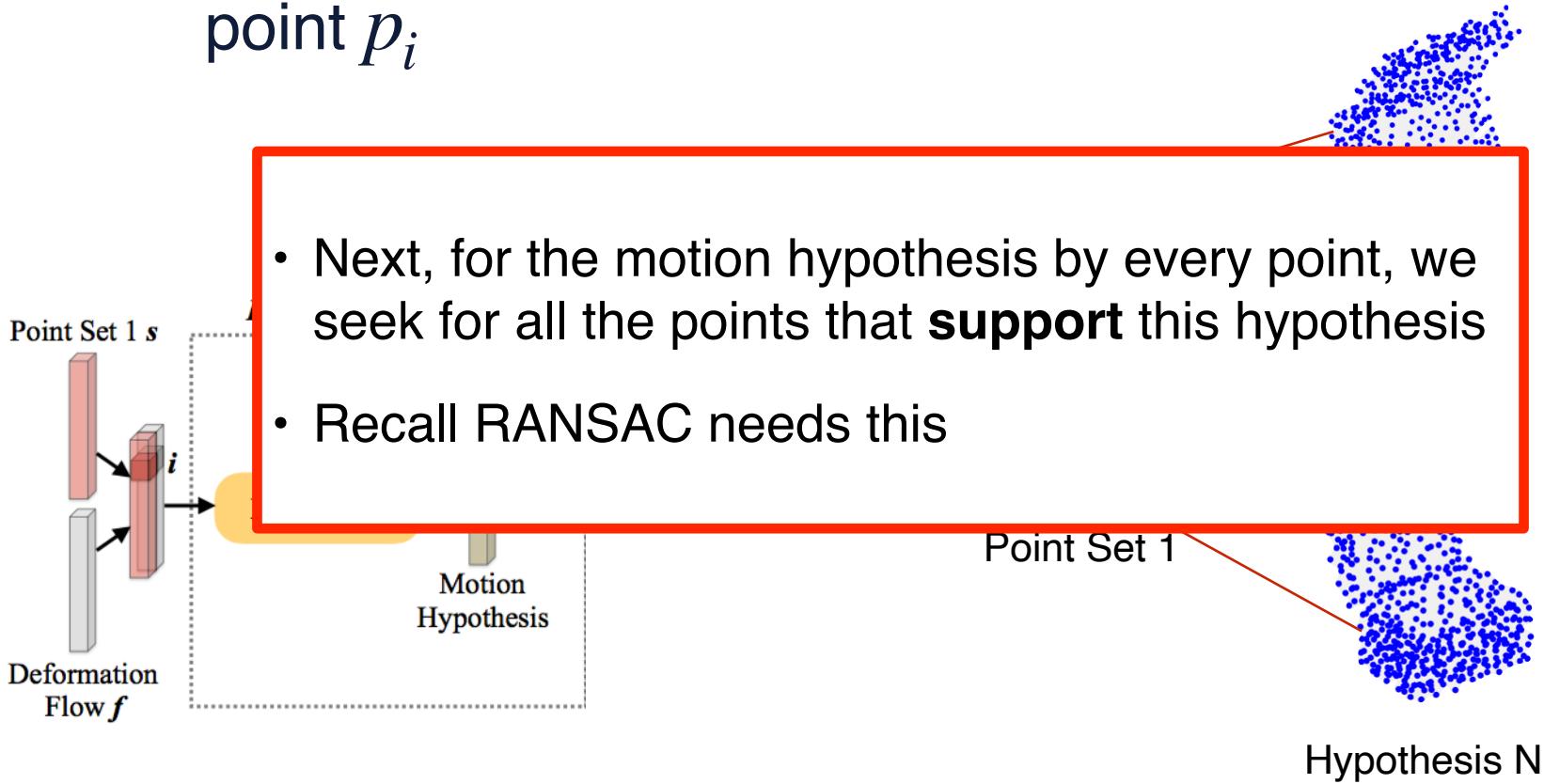
# Rigid Motion Hypotheses Generation

- Rigid motion hypothesis set generation
  - Compute the motion hypothesis  $(R_i, T_i)$  for each point  $p_i$



# Rigid Motion Hypotheses Generation

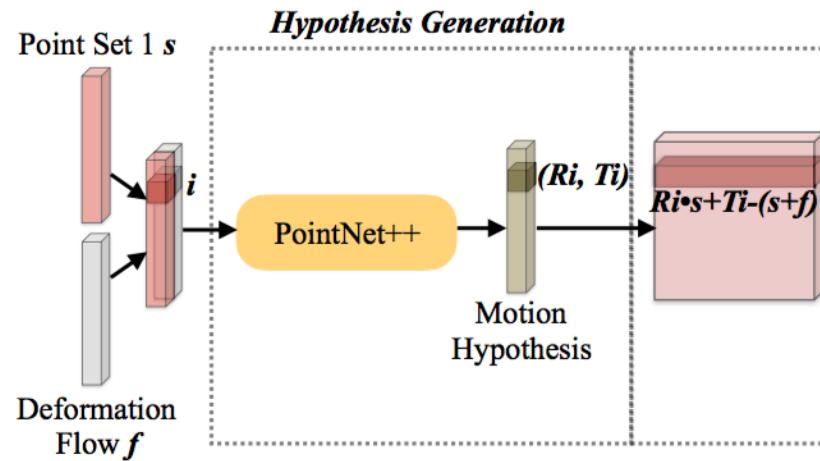
- Rigid motion hypothesis set generation
  - Compute the motion hypothesis  $(R_i, T_i)$  for each point  $p_i$



# Build Error Matrix

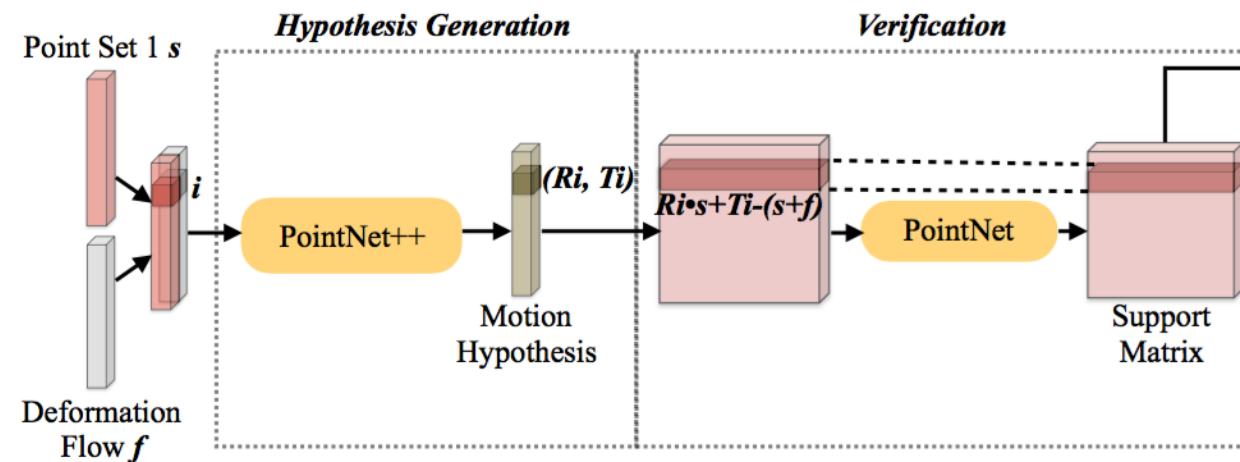
- For each motion hypothesis  $(R_i, T_i)$ , compute the difference between the transformed point set and the deformed point set with displacement  $f$

$$e_{i,j} = R_i p_j + T_i - (p_j + f_j)$$



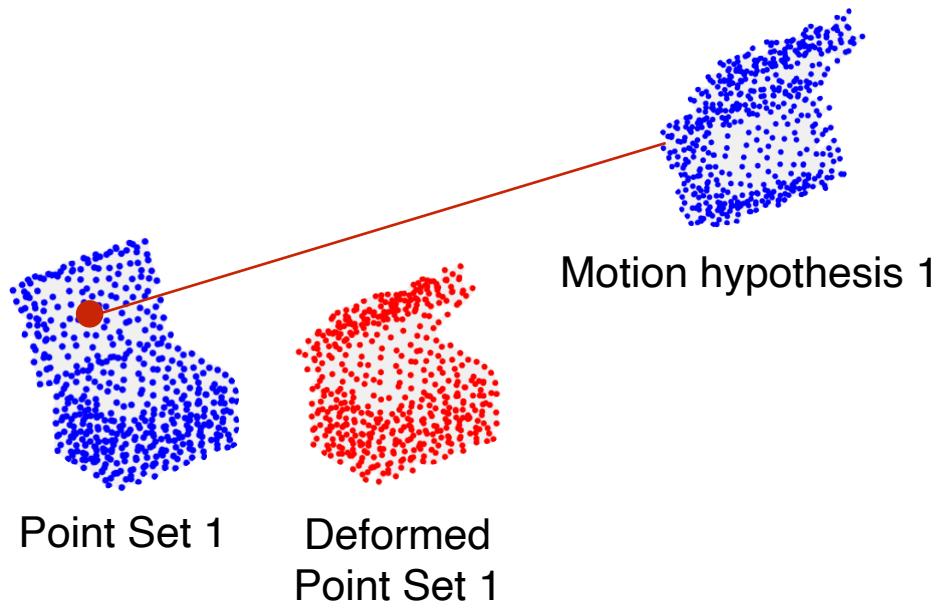
# Support Prediction

- Input: error matrix
- Output: support matrix
  - for each hypothesized rigid motion, the probability of every point under this rigid motion



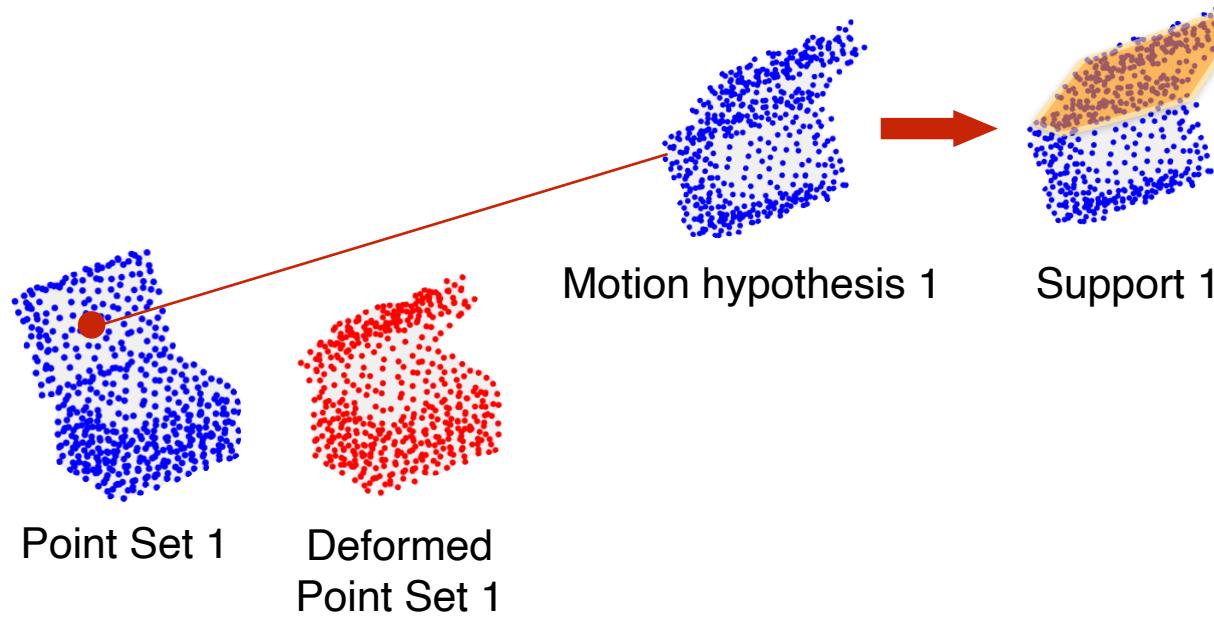
# Support Prediction

- Visualization



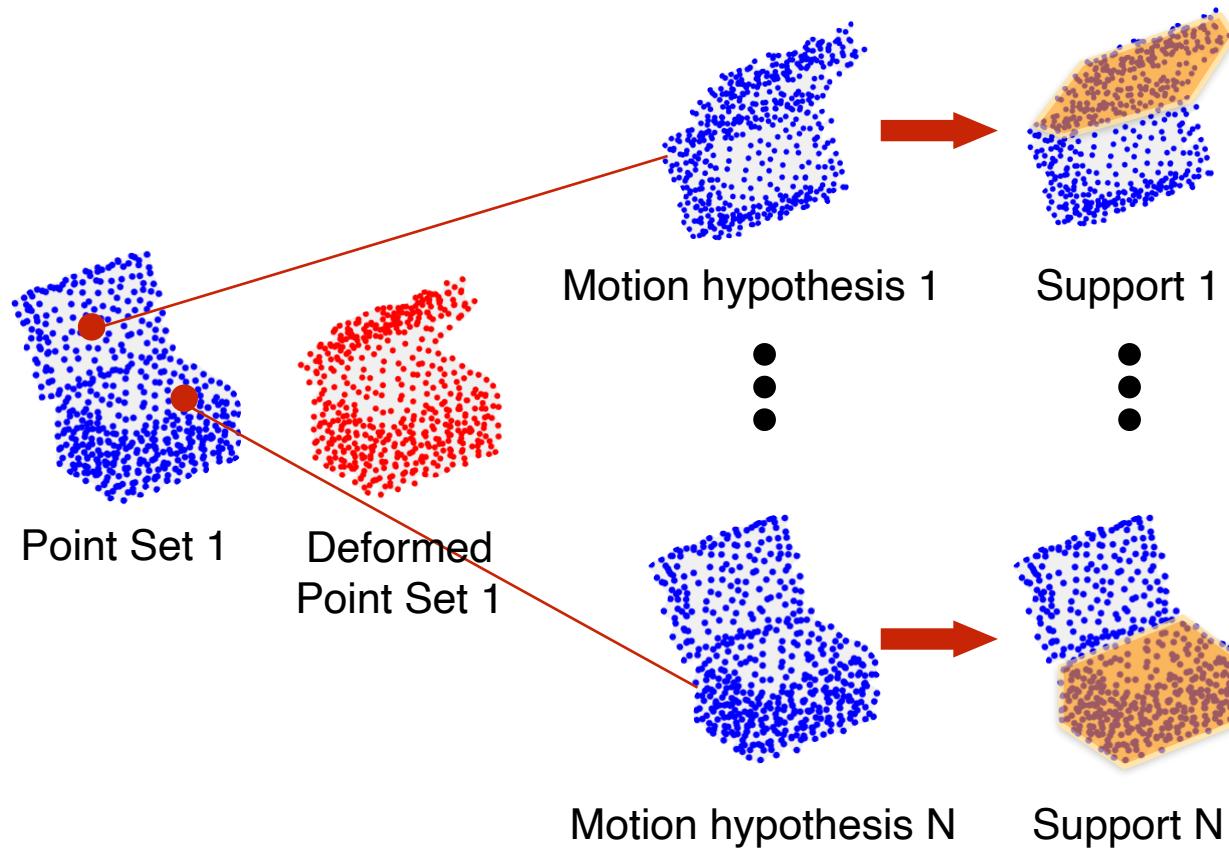
# Support Prediction

- Visualization



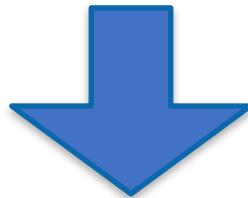
# Support Prediction

- Illustration



# Part Segmentation from Support Prediction

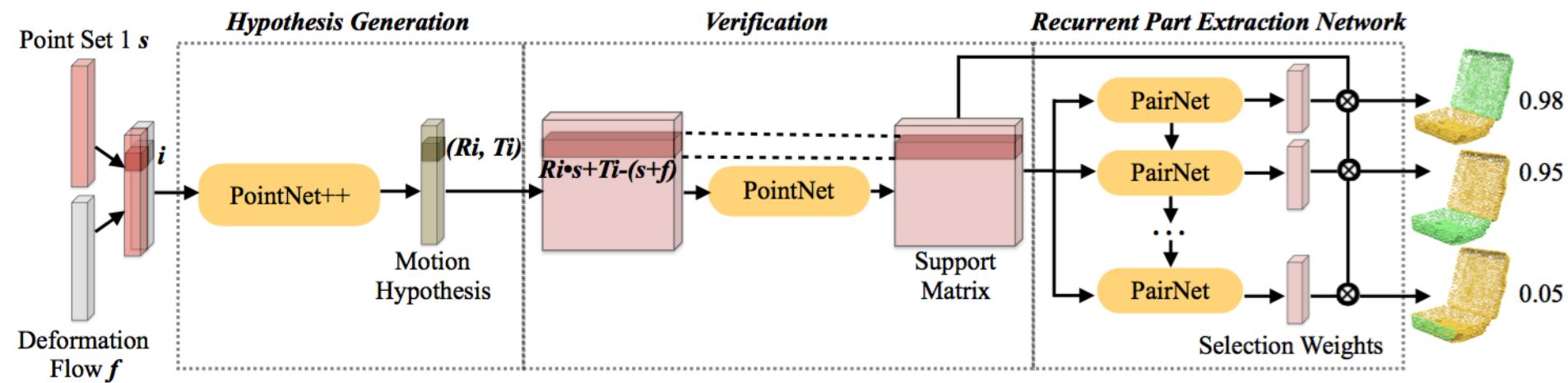
- Support prediction gives points that conform to the same motion transformation
- Cluster support points of similar motion hypothesis into a group
- Use a weighted average of the supports, generating one segment.



Recurrent part segmentation network

# Recurrent Part Segmentation Network

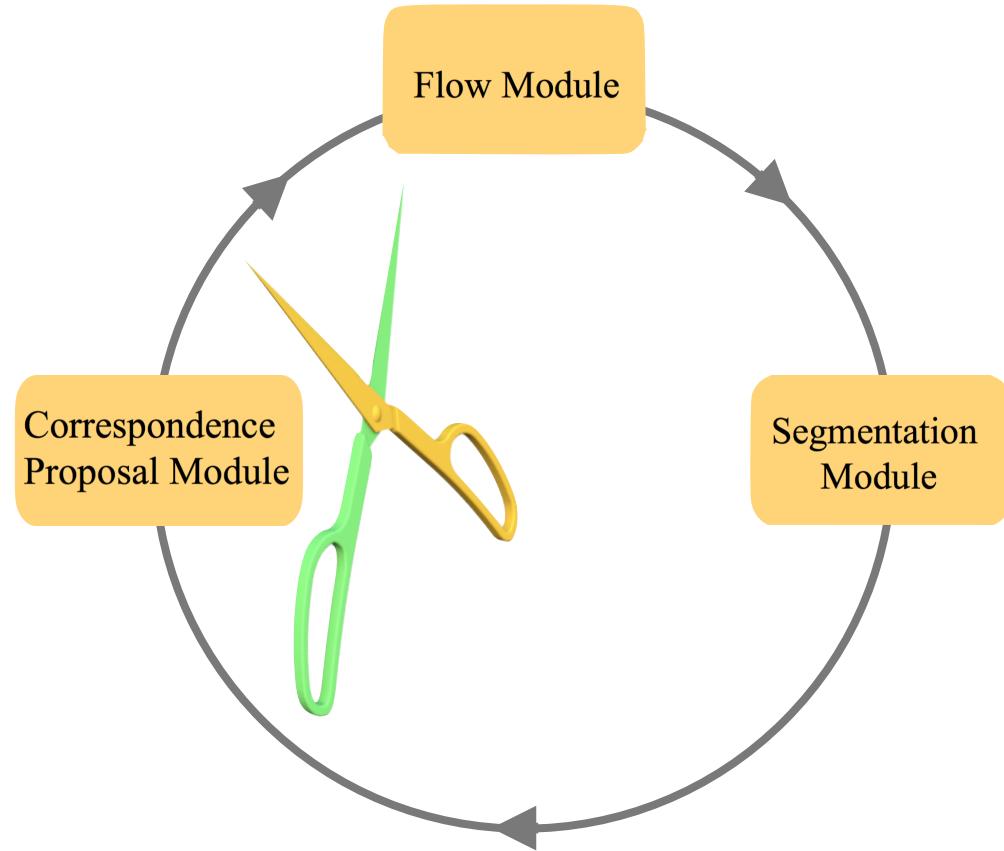
- Recurrent part segmentation neural network.
  - Split a set of points (segmentation) with the same transformation
  - Adjust the internal hidden state to encode already split regions



Yi et al., "Deep Part Induction from Articulated Object Pairs", SIGGRAPH Asia 2018

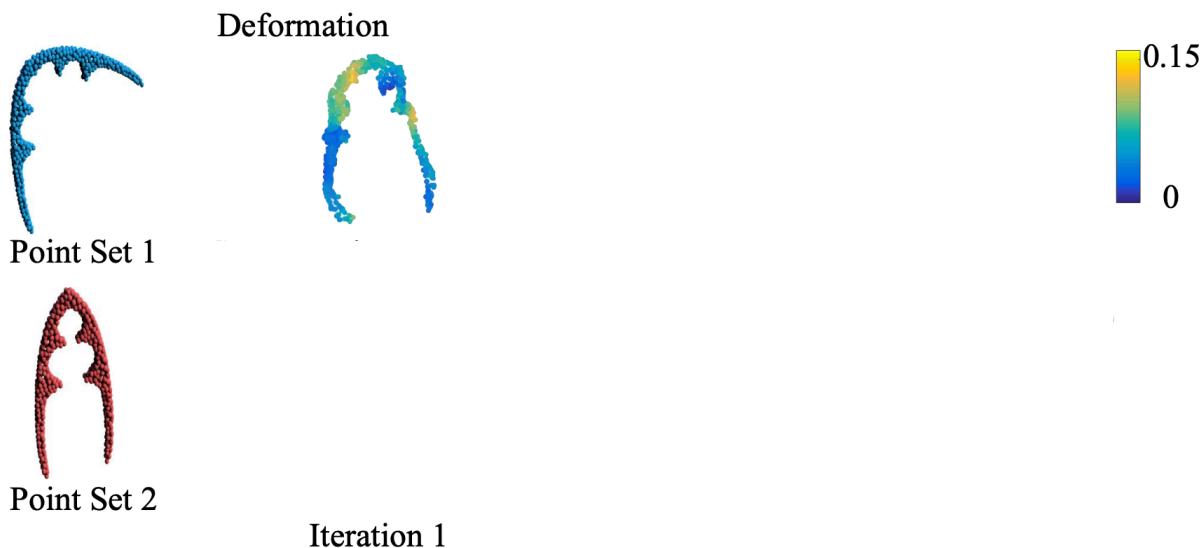
**Details omitted due to time constraints**

# Iterative Mutual Reinforcement Procedure

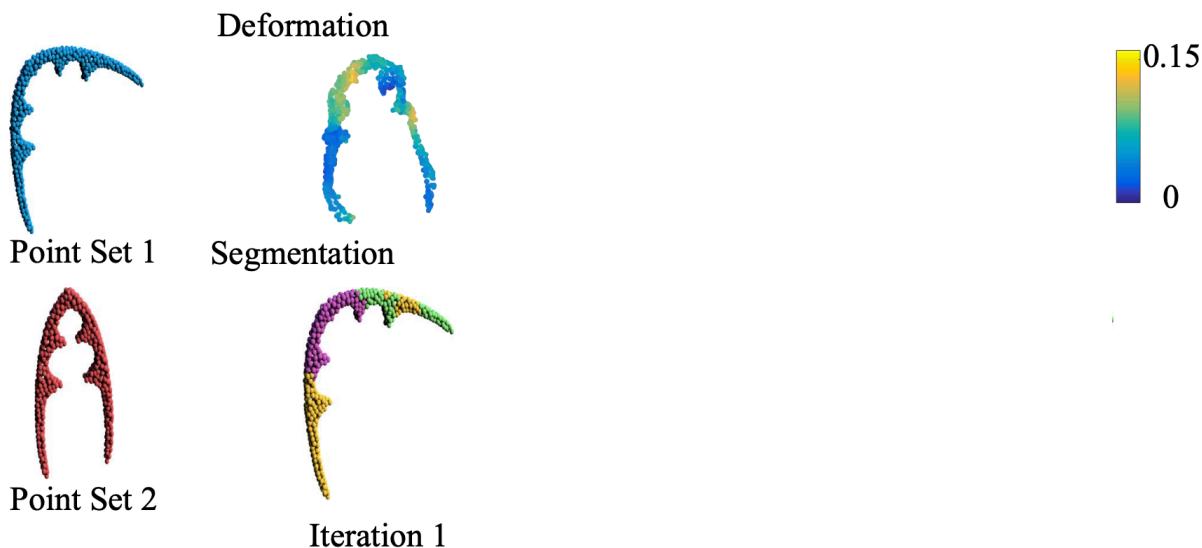


Segmentation mask helps to **restrict the possible regions** when estimating correspondences

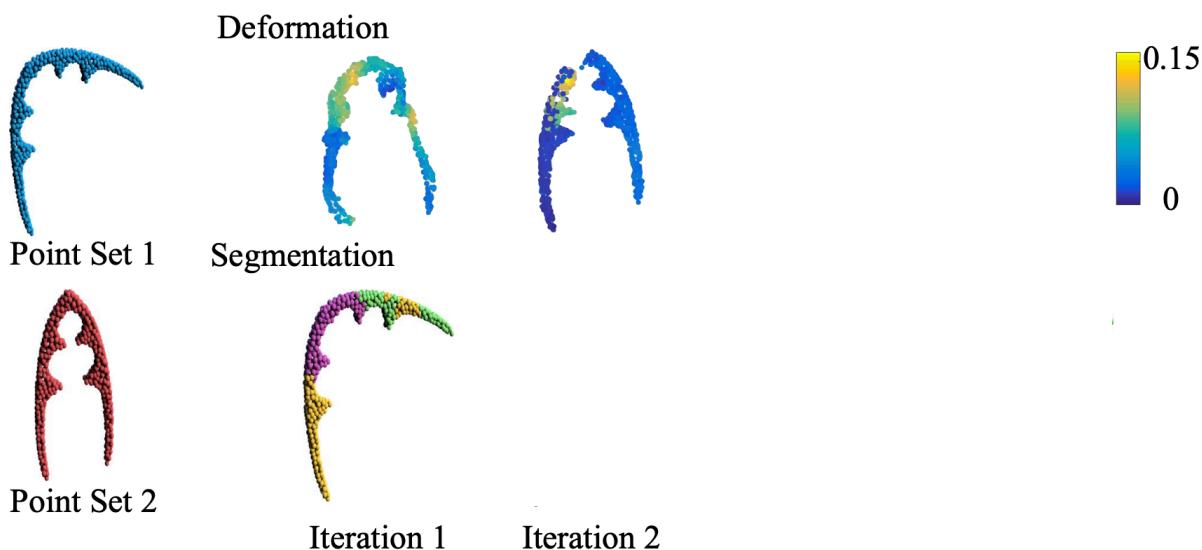
# Iterative Mutual Reinforcement Procedure



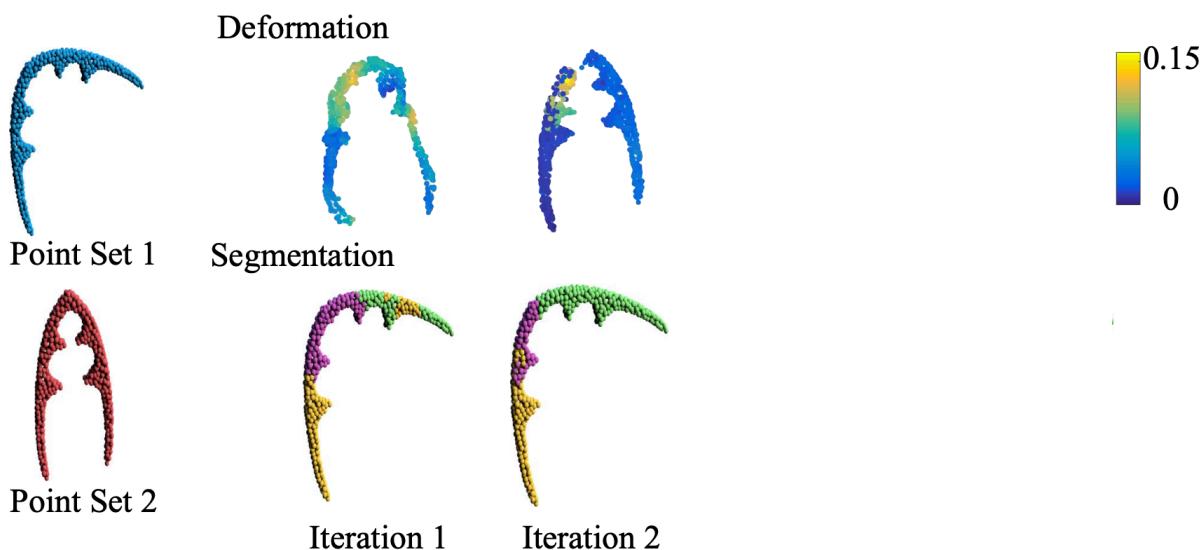
# Iterative Mutual Reinforcement Procedure



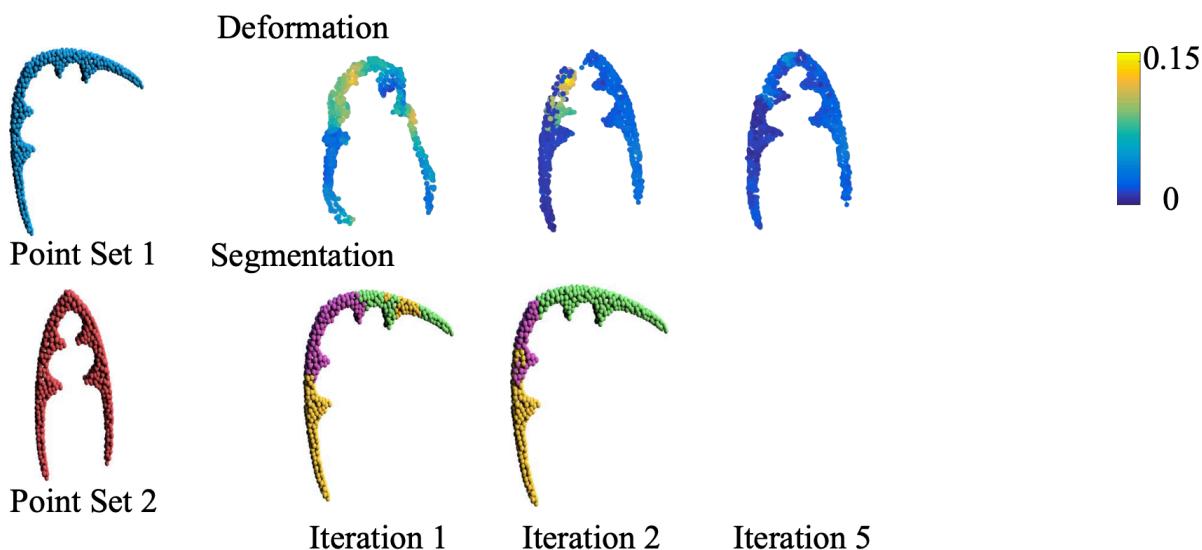
# Iterative Mutual Reinforcement Procedure



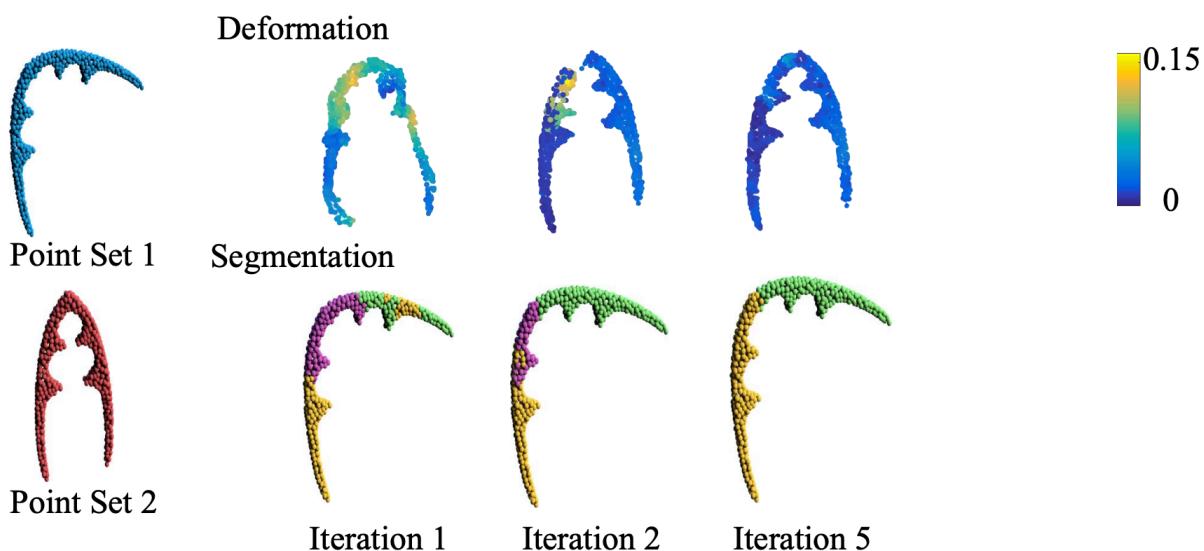
# Iterative Mutual Reinforcement Procedure



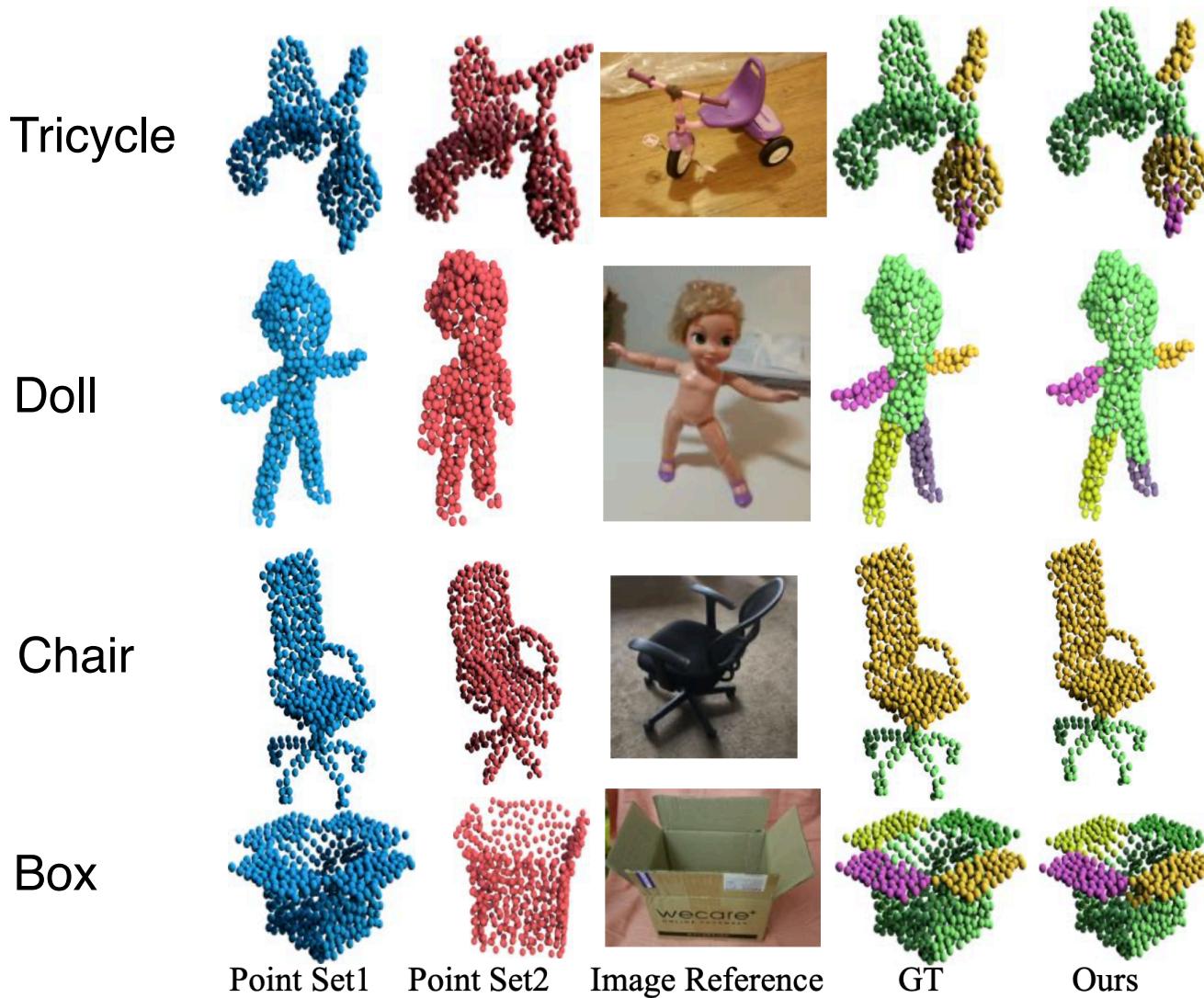
# Iterative Mutual Reinforcement Procedure



# Iterative Mutual Reinforcement Procedure



# Segmentation Results on UnSeen Objects



# Summary

- A **correspondence network** that predicts the matching probability, which can be applied to **object tracking**.
- A **flow network** that predicts the displacement between two point sets.
- A neural net-based **differentiable sequential RANSAC** which is a combination between algorithmic structure and learnable neural nets module.

# Compositional Generalizability

## View of Zero-shot Part Segmentation (*read by yourself*)

# Perspective: The World has Combinatorial Complexity



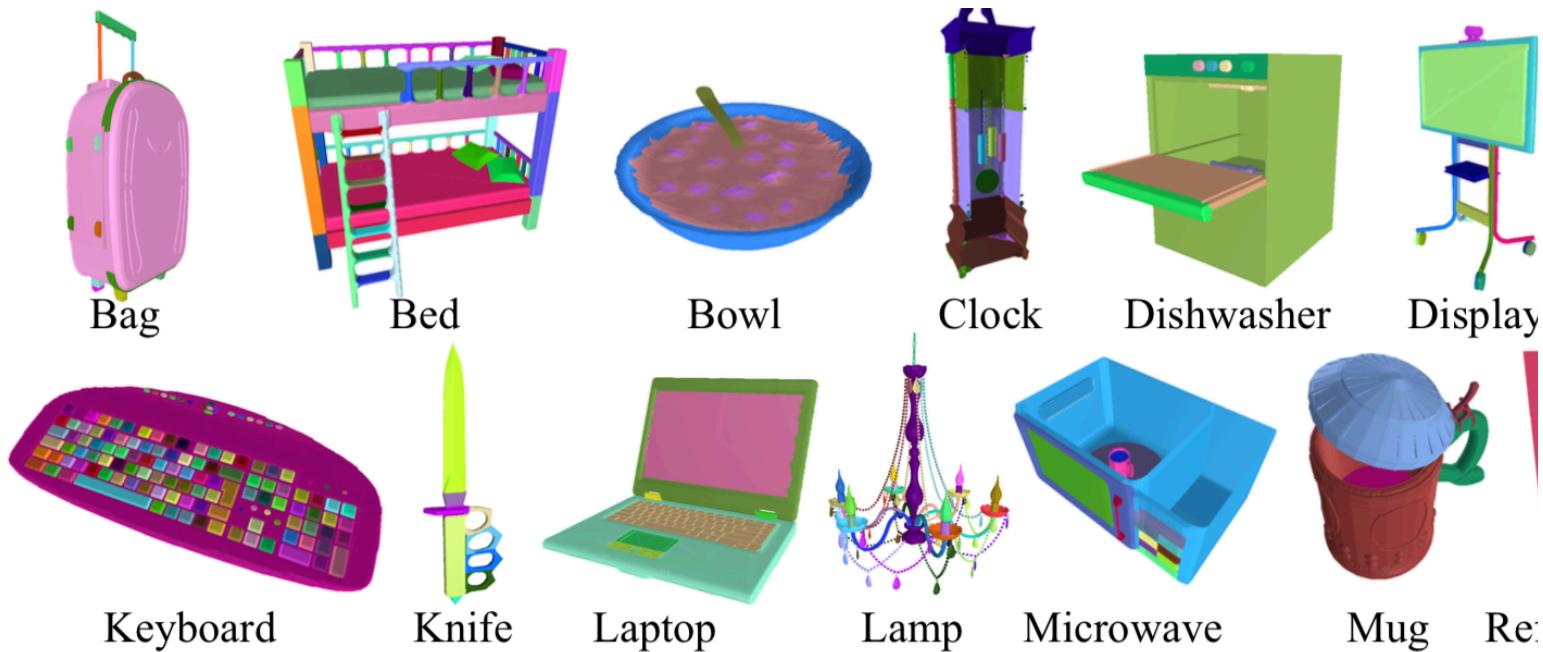
# How to train a smart learner?

- However, human can learn unknown things from just a few examples .
- One character of human intelligence is **systematic compositionality**
  - the capacity to understand and produce a potentially infinite number of **novel combinations of known components.**

(Chomsky, Montague, Fodor, Marcus, Pinker, etc.)

# PartNet Database

**573,585** part instances over **26,671** 3D models covering **24** object categories

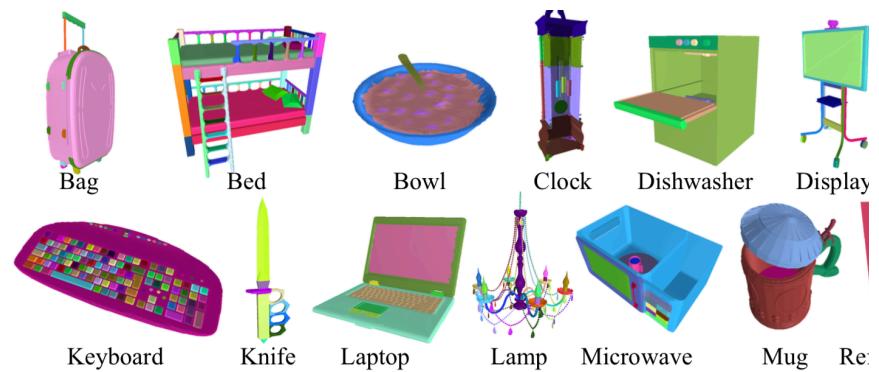


# Task: Zero-shot Part Segmentation

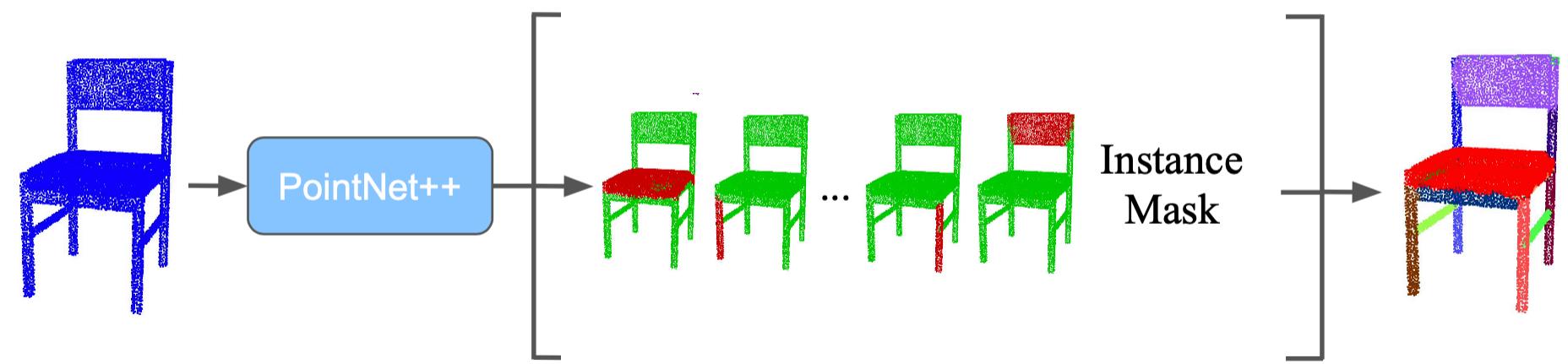
Train set  
(Chair only)



Test set  
(New categories)

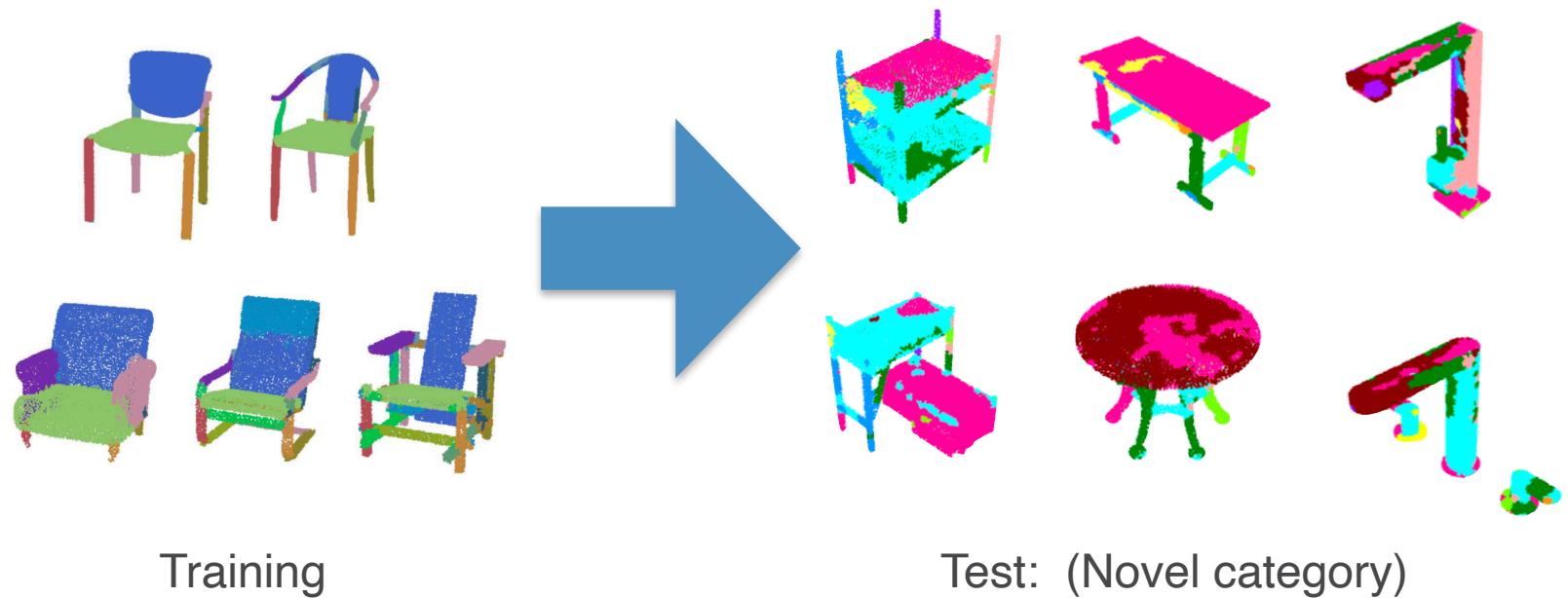


# Literature: Top-down Segmentation Pipeline



# Literature: Top-down Segmentation Pipeline

- Part segmentation



Fail to generalize to unseen categories due to  
the difference of global shapes

# Learning to Group

- Design disciplines:
  - Restrict the field of view to local regions.
  - Bottom-to-up fashion. (Initial part segmentation)
  - A module to merge two parts at each time.

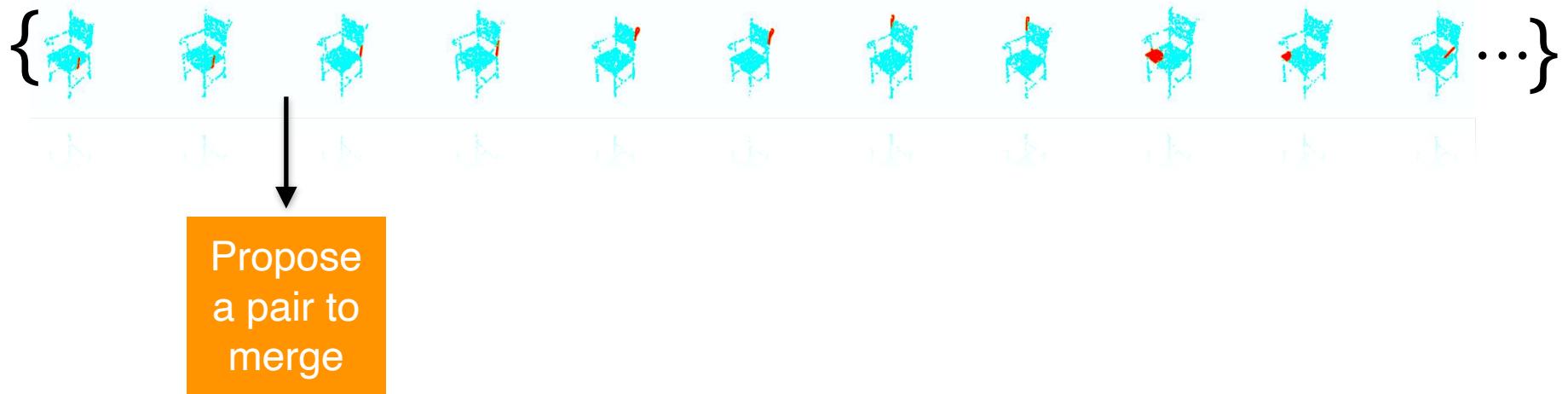
# Basic Pipeline

Cluster points into many small segments by geometry features



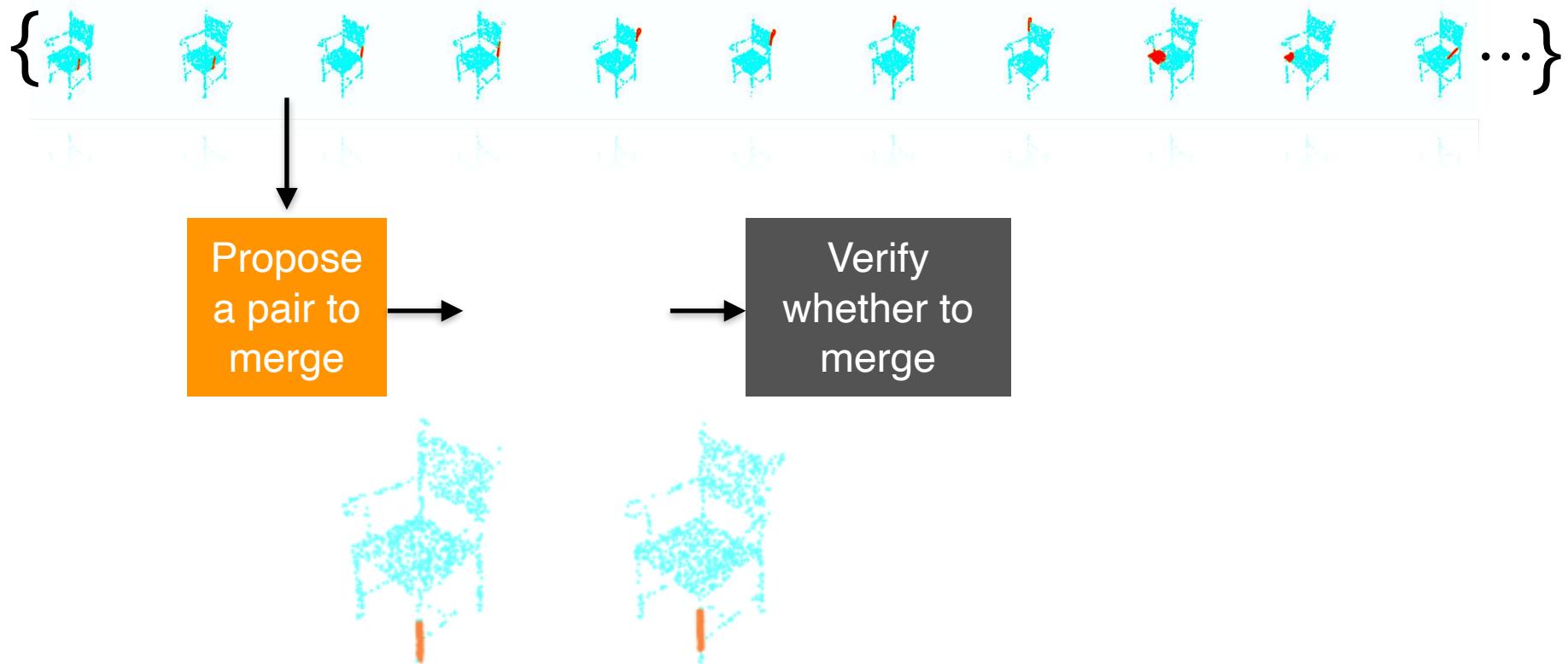
# Basic Pipeline

## Sub-Part Pool



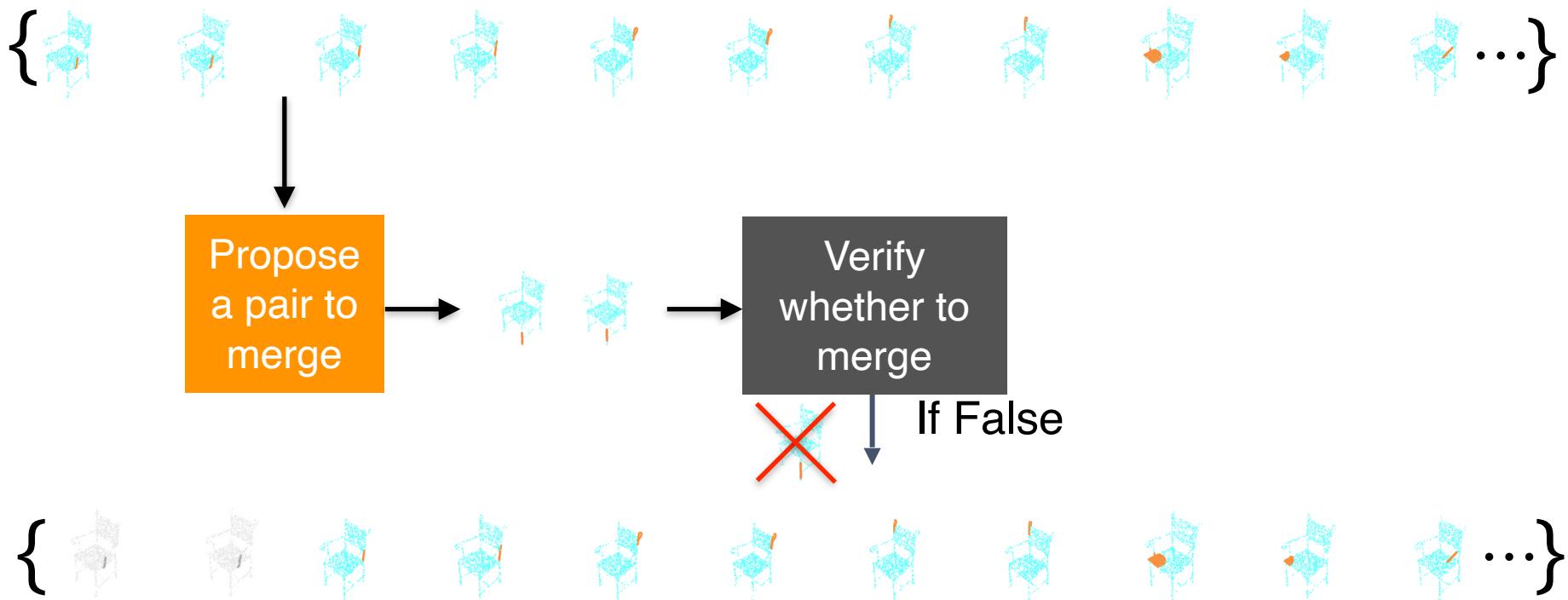
# Basic Pipeline

## Sub-Part Pool



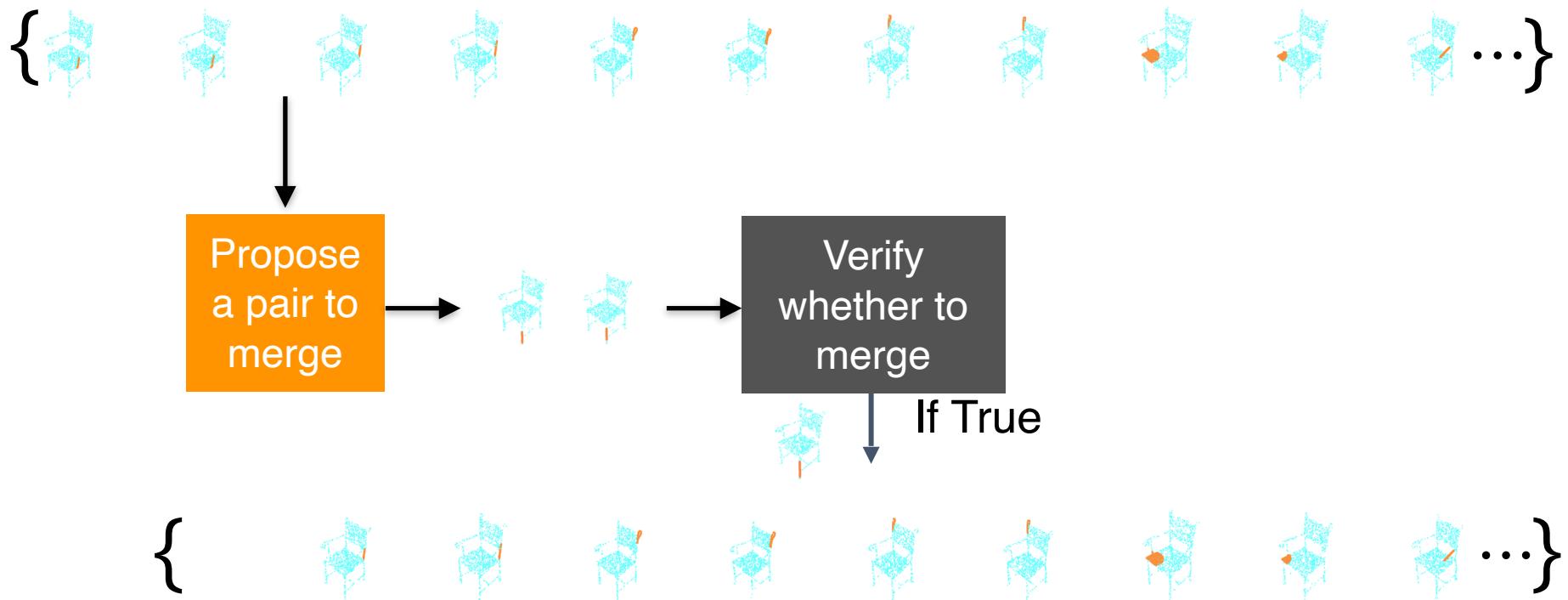
# Learning to Group

## Sub-Part Pool



# Learning to Group

## Sub-Part Pool



# Learning to Group

## Sub-Part Pool



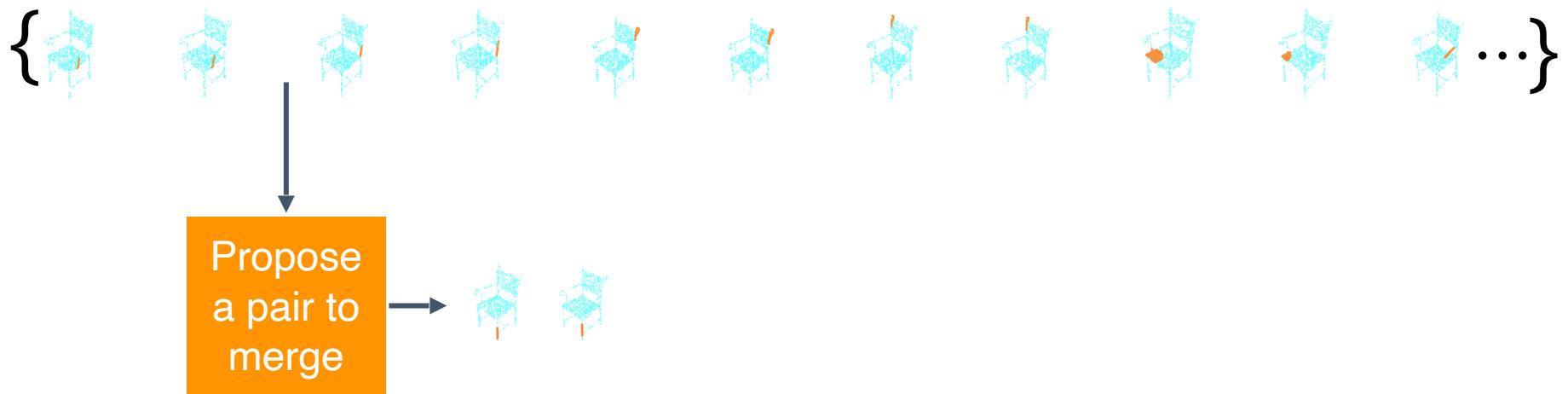
If all pairs verified to be false



Final

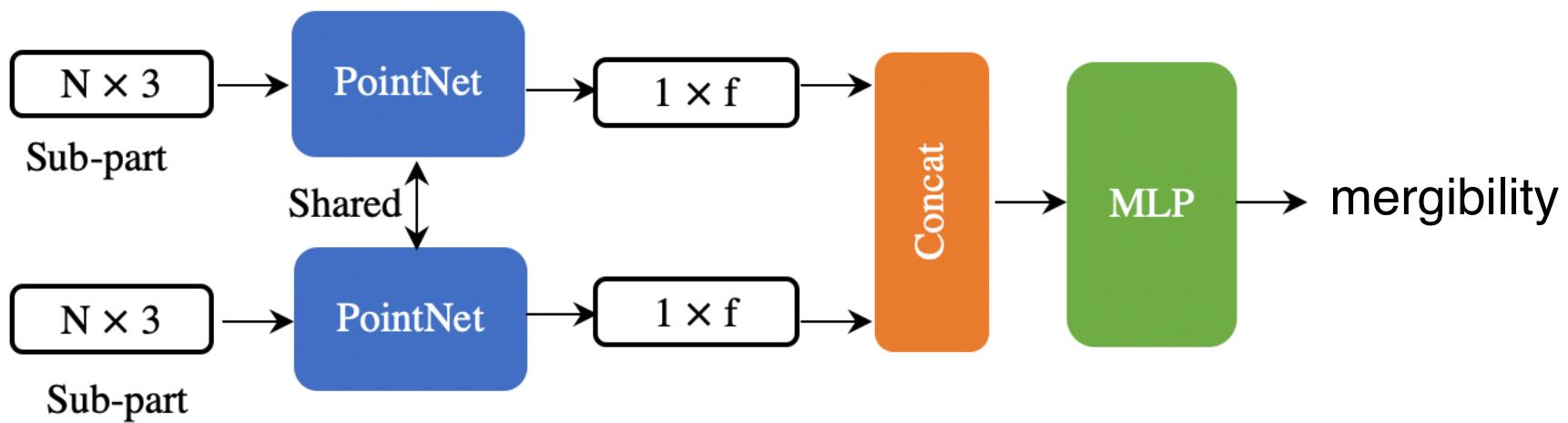
# Details of Pair Proposals

## Sub-Part Pool



# Proposal Network

- Learn a network to predict the “mergibility”
- Architecture:



# Training: RL Framework

- How to learn the proposal network?
- Proposal: about decisions of which parts to merge
- We are seeking to learn a “policy” network — Reinforcement learning!

# Proposal Network as a Policy

- In the language of RL, a policy is denoted as a distribution  $\pi$  over possible actions  $\mathcal{P}$ . You can use the policy to sample an action:

$$a \sim \pi(\cdot | \mathcal{P})$$

- For our problem,  $\mathcal{P}$  includes all the pairs, and  $a_i = (P_{i_1}, P_{i_2})$  is a specific pair

# Reward Design

- To learn the proposal policy  $\pi$ , we still need to know whether a proposal is good or bad.
- Let  $M(a)$  be the score of a pair (reward):
  - $M(a) = 1$  if
    - $(P_{i_1}, P_{i_2})$  has the same part label, and
    - merging  $(P_{i_1}, P_{i_2})$  would create a subpart overlaps with some groundtruth part  $\geq 80\%$
  - $M(a) = 0$  otherwise

# Policy Gradient

- We learn  $\pi$  as a network by optimizing the following problem:

$$\underset{\pi}{\text{maximize}} \mathbb{E}_{a \sim \pi(\cdot | \mathcal{P})} [M(a)]$$

*Read by yourself*

# Policy Gradient

- We learn  $\pi$  as a network by optimizing the following problem:

$$\underset{\pi}{\text{maximize}} \mathbb{E}_{a \sim \pi(\cdot | \mathcal{P})} [M(a)]$$

- Gradient descent to update policy network
  - Note that:

$$\begin{aligned}\nabla_{\pi} L &= \nabla \mathbb{E}_{a \sim \pi(\cdot)} M(a) = \nabla \int \pi(a) M(a) \\ &= \int \nabla \pi(a) M(a) = \int \pi(a) \frac{1}{\pi(a)} \nabla \pi(a) M(a) \\ &\approx \frac{1}{n} \sum_{a_i} \frac{1}{\pi(a_i)} \nabla \pi(a_i) M(a_i) = \frac{1}{n} \sum_{a_i} \nabla \log \pi(a_i) M(a_i)\end{aligned}$$

- Network update by chain rule:  $\nabla_{\theta} L = \nabla_{\theta} \pi \nabla_{\pi} L$

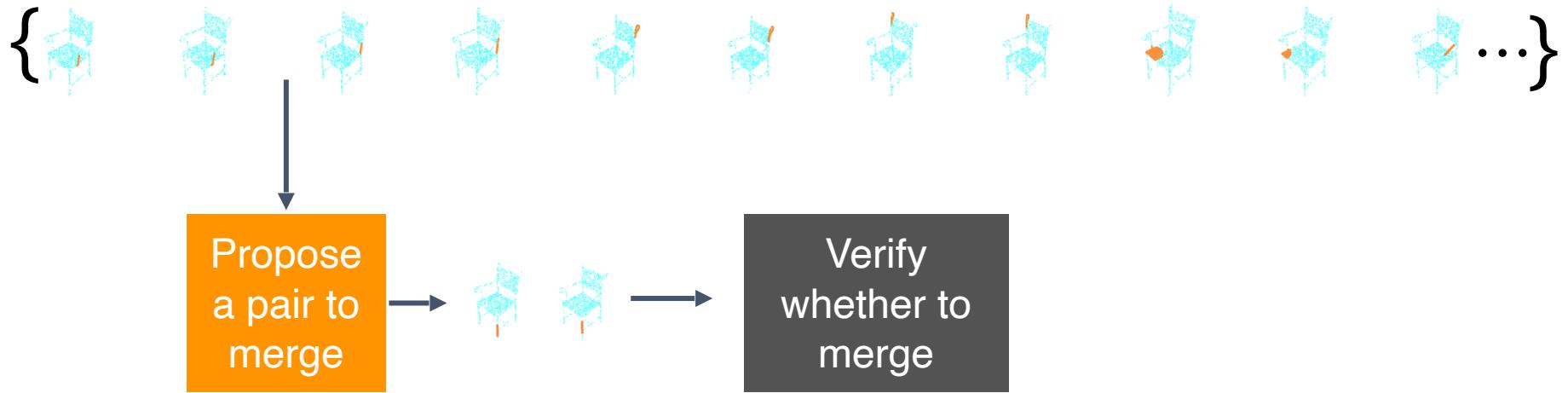
**Read by yourself**

# Summary of Pair Proposal Training

- Proposal policy  $\pi$  to sample pairs of sub-parts to merge
- Reward  $M$  to sub-parts that belongs to the same part and have large overlap with ground-truth part
- Train the network by maximizing the expectation of reward

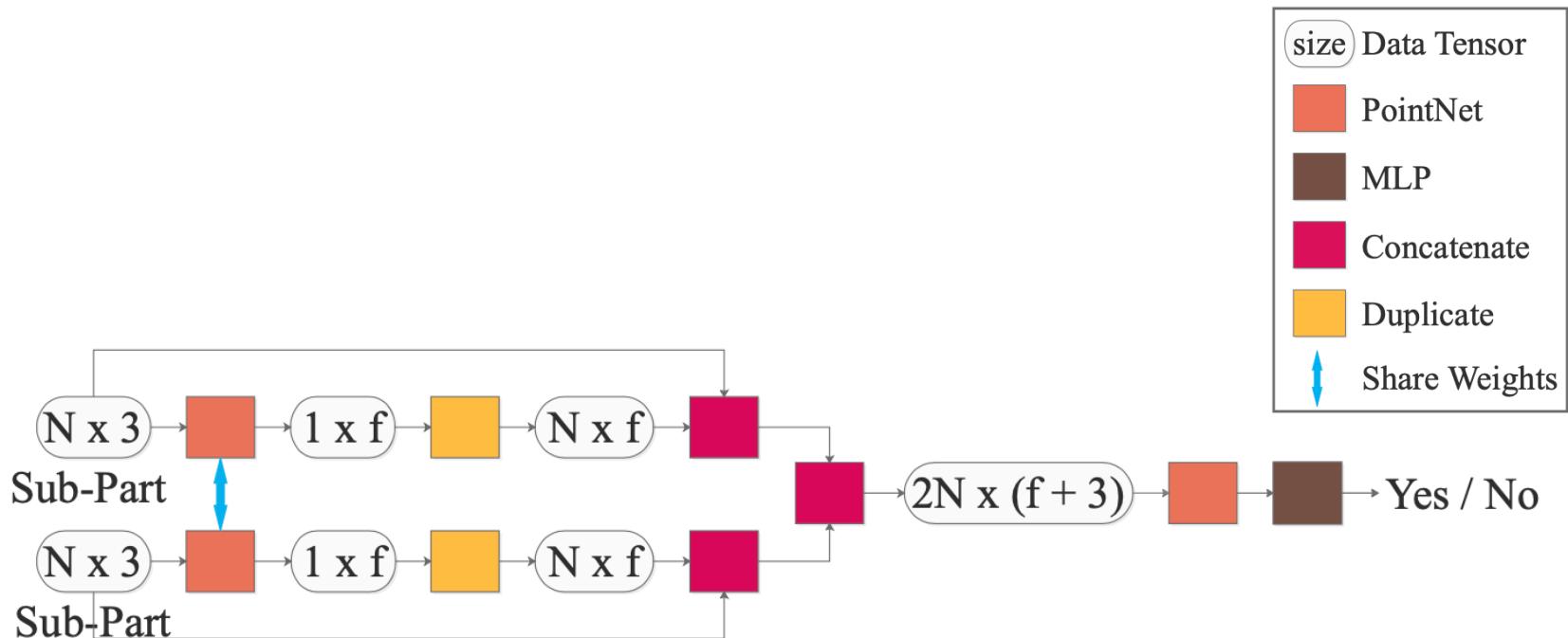
# Details of Verification

## Sub-Part Pool

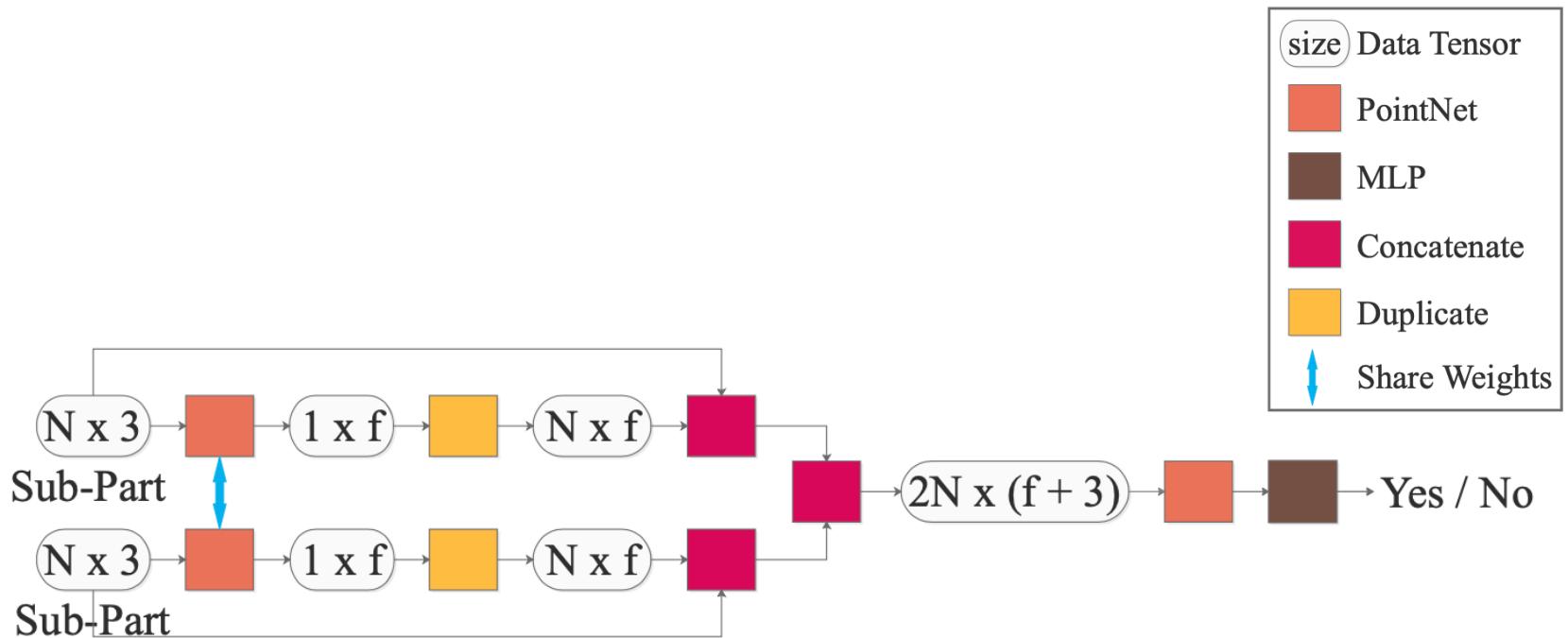


# Network

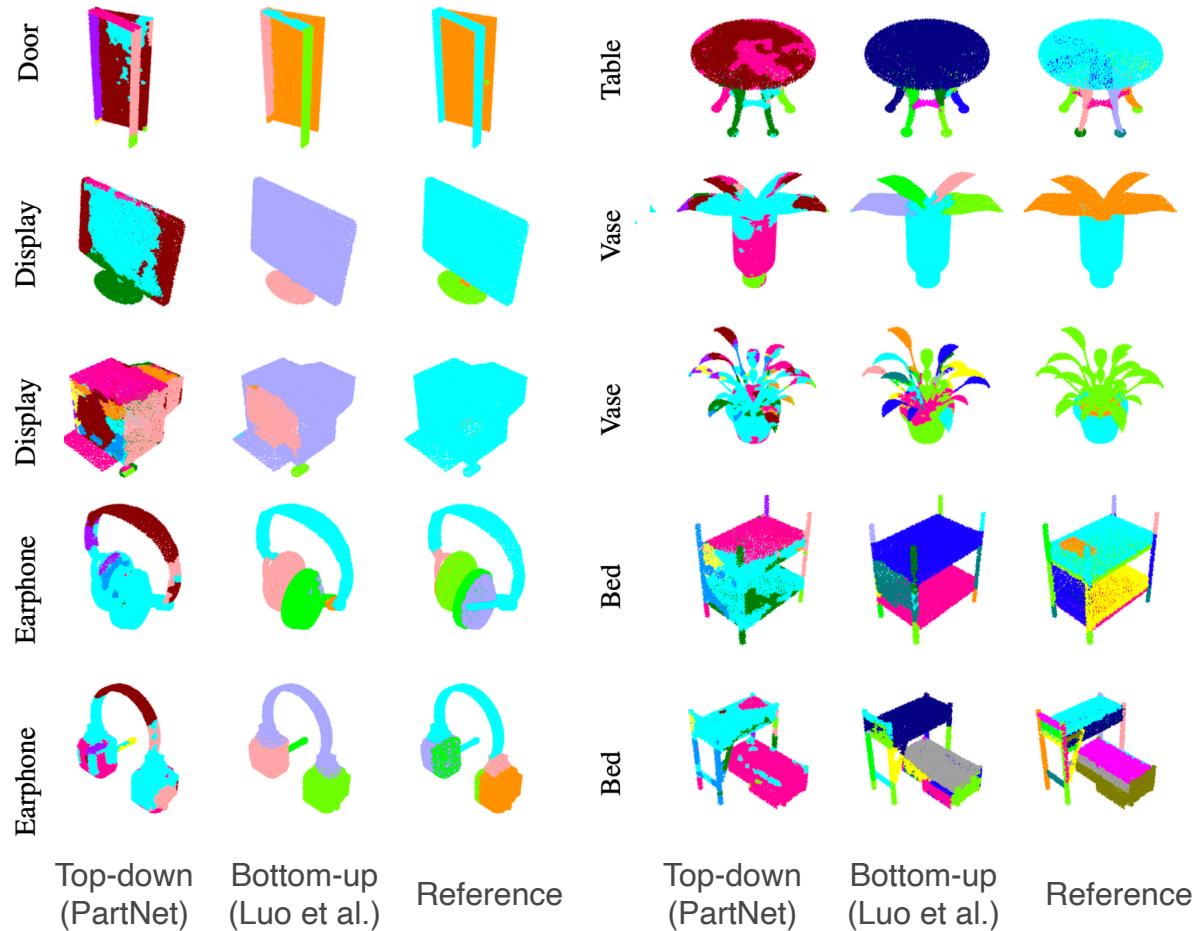
- Verify whether the selected pair sub-parts should be merged or not.



# Training: Binary Classification



# Results on Unseen Categories



Luo, Tiange, et al. "Learning to Group: A Bottom-Up Framework for 3D Part Discovery in Unseen Categories.", ICLR 2020.

# Summary

- A **correspondence network** that predicts the matching probability between two point cloud in **Transformer** fashion.
- A neural net-based **differentiable sequential RANSAC** which is a combination between algorithmic structure and learnable neural nets module.
- A part discovery pipeline in **bottom-up** fashion.