FlowNet 3D

1. Problem definition

$P = \{x_i \mid i=1,\cdots,n_1\}$, $Q = \{y_j \mid j=1,\cdots,n_2\}$, $x_i, y_j \in R^3$

$d_i = x_i' - x_i$

goal: $D = \{d_i \mid i=1,\cdots,n_1\}$

2. Summary:

FlowNet 3D: estimate scene flow from a pair of $\underbrace{\text{consecutive}}_{t \text{ and } t+1}$ point clouds
end-to-end

two new layers:

① flow embedding layer: correlate two point clouds

② set upconv layer: propagate features from one set to the other

3. FlowNet3D Architecture

① set conv layer: from PointNet++

hierarchical feature learning, translation-invariant

n points $p_i = \{x_i, f_i\}$, $x_i \in R^3$, $f_i \in R^C$, $i = 1,\cdots,n$

$\downarrow$ set conv layer

$\underbrace{\text{sub-sampled}}_{\text{farthest point sampling}}$ n' points $p_j' = \{\underbrace{x_j'}_{\text{region center}}, f_j'\}$, $x_j' \in R^3$, $f_j' \in R^{C'}$, $j = 1,\cdots,n'$

For each region centered at x_j'

$$f_j' = \max_{\{i \mid \|x_i - x_j\| \le r\}} \{h(\underbrace{f_i, x_i - x_j'}_{\text{concatenate}})\}$$

$h: R^{C+3} \longrightarrow R^{C'}$, a MLP

max: element-wise max pooling

② flow embedding layer

mix two point clouds

input: $\{p_i = (x_i, f_i)\}_{i=1}^{n_1}$, $\{q_j = (y_j, g_j)\}_{j=1}^{n_2}$

$x_i, y_j \in R^3$, $f_i, g_j \in R^C$

output: $\{o_i = (x_i, e_i)\}_{i=1}^{n_1}$

For each x_i:

$$e_i = \max_{\{j \mid \|y_j - x_i\| \leq r\}} \{h(f_i, g_j, y_j - x_i)\}$$

* For each x_i, we consider multiple softly corresponding points y_j and make a "weighted" decision
* $\|y_j - x_i\|$ alternative: $dist(f_i, g_j)$
  but the previous is better.
* $\{o_i\}$ further go through several set conv layers.

③ Set upconv layer: flow refinement

input: $\{p_i = \{x_i, f_i\} \mid i = 1, \cdots, \underline{n}\}$ 降采样后的点数

$\{x_j' \mid j = 1, \cdots, \underline{n'}\}$ P中的点数

output: $\{x_j', f_j'\}_{j=1}^{n'}$

For each region centered at x_j'

$$f_j' = \max_{\{i \mid \|x_i - x_j\| \leq r\}} \{\underbrace{h(f_i, x_i - x_j')}_{\text{concatenate}}\}$$

* alternative way to upsample: 3D interpolation

$$f_j' = \sum_{\{i \mid \|x_i - x_j'\| \leq r\}} \underbrace{w}_{\text{normalized inverse-distance weight function}}(x_i, x_j')f_i$$

but the previous is better.

* a final regression layer to output R^3 predicted scene flow

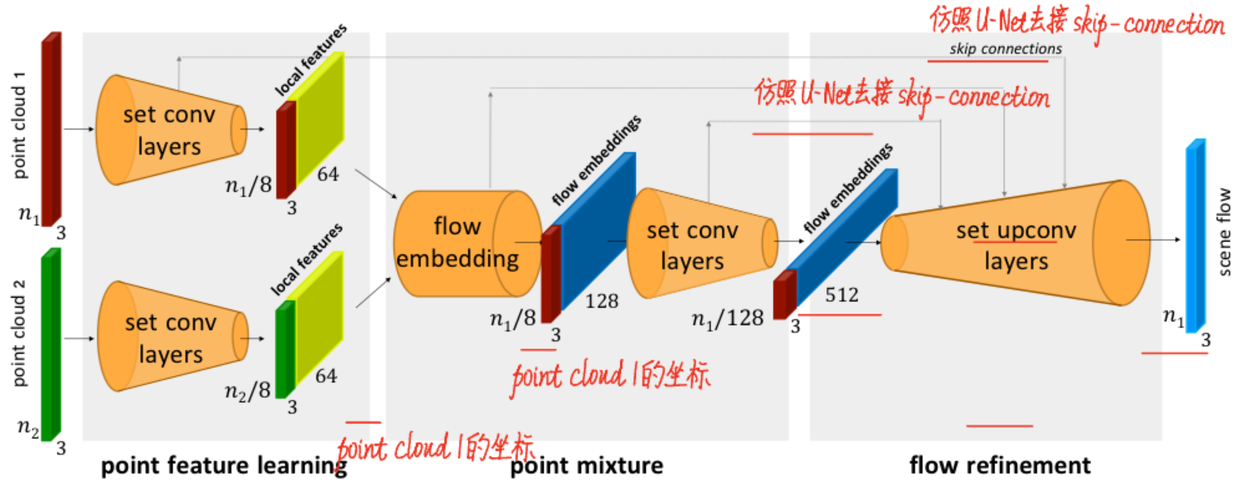

Figure 3: **FlowNet3D architecture.** Given two frames of point clouds, the network learns to predict the scene flow as translational motion vectors for each point of the first frame. See Fig. 2 for illustrations of the layers and Sec. 4.4 for more details on the network architecture.

encoder ⟶ merge ⟶ decoder

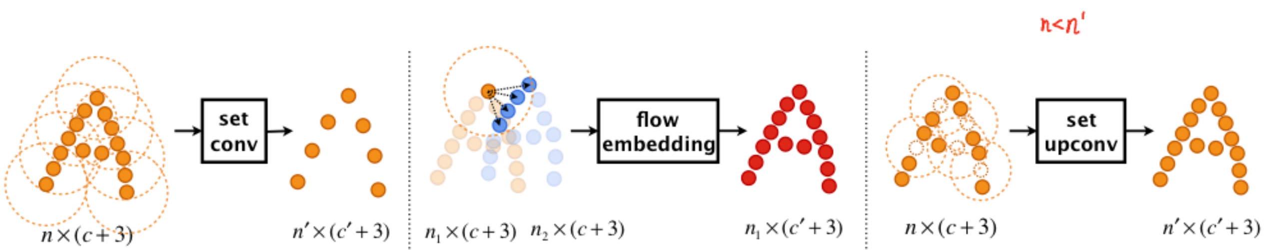

Figure 2: **Three trainable layers for point cloud processing.** *Left:* the *set conv* layer to learn deep point cloud features. *Middle:* the *flow embedding layer* to learn geometric relations between two point clouds to infer motions. *Right:* the *set upconv* layer to up-sample and propagate point features in a learnable way.

4. Other notes

① training loss:

$P = \{x_i\}_{i=1}^{n_1}$, $Q = \{y_j\}_{j=1}^{n_2}$

$D = \underbrace{F}_{\text{FlowNet3D}}(P, Q; \underbrace{\Theta}_{\text{parameters}}) = \{d_i\}_{i=1}^{n_1}$

groundtruth: $D^* = \{d_i^*\}_{i=1}^{n_1}$

backward flow: $\{d_i'\}_{i=1}^{n_1} = F(P', P; \Theta)$

where $P' = \{x_i + d_i\}_{i=1}^{n_1}$

$$L(P, Q, D^*, \Theta) = \frac{1}{n_1} \sum_{i=1}^{n_1} \{\|d_i - d_i^*\| + \lambda \underbrace{\|d_i' + d_i\|}_{\text{cycle-consistency term}}\}$$

② down-sample introduces noise ⟶ inference with random re-sampling

5. Application

<1> 3D scan registration

<2> motion segmentation