A. Super-keypoints discovering and learning

Given a pretrained feature extractor, we aim to cluster the prototypes of keypoints from all the base categories into super-keypoints. In this section, to indicate the category of a keypoint, we add the superscript y in its keypoint and prototype notation: j_r^y and $\mathbf{w}_r^y$ denote the r^{th} keypoint and its corresponding prototype of a category y. The set of all prototypes are then $\mathbf{W}_{base} = \bigcup_{y \in C_{base}} \{\mathbf{w}_r^y\}_{r=1}^{T_y}$.

We adopt the k-nearest neighbor based density peaks clustering [2, 7] since it does not requires the number of clusters to be known in advance. Note that two keypoints cannot be in the same cluster if they belong to the same category or they are two far from each other. Hence, we define for each keypoint j_r^y a set $V(j_r^y)$ that contains its valid neighbors:

$$V(j_r^y) = \{j_{\tilde{r}}^{\tilde{y}} | \tilde{y} \neq y, \tilde{r} \in \{1, \dots, T_{\tilde{y}}\}, \frac{\mathbf{w}_r^{y\top} \mathbf{w}_{\tilde{r}}^{\tilde{y}}}{\|\mathbf{w}_r^y\| \|\mathbf{w}_{\tilde{r}}^{\tilde{y}}\|} \geq \xi\}, \quad \text{for } j_r^y \in \mathbf{W}_{base},$$

where ξ is a threshold. In this work, we set ξ to 0.8.

We compute the density ρ_r^y for each keypoint j_r^y as follow:

$$\rho_r^y = \frac{1}{|KNN(j_r^y)|} \sum_{j_{\tilde{r}}^{\tilde{y}} \in KNN(j_r^y)} \frac{\mathbf{w}_r^{y\top} \mathbf{w}_{\tilde{r}}^{\tilde{y}}}{\|\mathbf{w}_r^y\| \|\mathbf{w}_{\tilde{r}}^{\tilde{y}}\|}, \quad \text{for } j_r^y \in \{1, \dots, T\},$$

where $KNN(j)$ is the set of $\min(k, |V(j_r^y)|)$ nearest valid neighbors of j_r^y.

Intuitively, the density of a keypoint measures the strength of the connections between the keypoint and its nearest valid neighbors. Having higher density equivalents to having higher chance to be a cluster center. We gradually take out the keypoint with the highest density to be a new cluster center and assign its valid neighbors to the cluster. Algorithm 1 shows how to infer cluster assignments for the keypoints.

Algorithm 1 Clustering keypoints in to super-keypoints

Input: clustering densities: $\{\rho_r^y\}_{r=1,y\in C_{base}}^{T_y}$, sets of valid neighbors of keypoints: $\{V(j_r^y)\}_{r=1,y\in C_{base}}^{T_y}$.
Initialization: set of unassigned keypoints $A = \{j_r^y\}_{r=1,y\in C_{base}}^{T_y}$, cluster count $L = 0$.
While $|A| > 0$:
  $j_r^y \leftarrow \arg\max_{j_r^y \in A} \rho_r^y$; ▷ Take out the keypoint with highest density
  remove j_r^y from A;
  $L \leftarrow L + 1$;
  $z_r^y \leftarrow L$;
  $B \leftarrow \{y\}$; ▷ B is the set of categories of keypoints in the underlying cluster
  For $j_{\tilde{r}}^{\tilde{y}} \in A$:
    If $j_{\tilde{r}}^{\tilde{y}} \in V(j_r^y)$ and $\tilde{y} \notin B$:
      remove $j_{\tilde{r}}^{\tilde{y}}$ from A;
      $z_{\tilde{r}}^{\tilde{y}} \leftarrow L$;
      $B \leftarrow B \cup \{\tilde{y}\}$;
Return: cluster assignments: $\{z_r^y\}_{r=1,y\in C_{base}}^{T_y}$, number of clusters (super-keypoints): L.

B. Detailed expectation maximization algorithm in novel category adaptation

Recall section 5.1 in the main paper, we aim to estimate the keypoint prototypes for a novel category y, given that we have estimated the parameters for the set of super-keypoints $\mathbf{M} = \{\mu_1, \mu_2, \cdots, \mu_L\}$.

The keypoint prototypes can be estimated by maximizing the following objective:

$$\mathcal{L}_{MAP} = \sum_{r=1}^{T_y} \log p(\mathbf{w}_r | \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) = \sum_{r=1}^{T_y} \log \sum_{z_r=1}^{L} p(\mathbf{w}_r, z_r | \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}). \tag{1}$$

where $\mathbf{w}_r$ is the prototype for the r^{th} keypoint, $\{\mathbf{o}_r^i\}_{i=1}^K$ is the keypoint features extracted from K support images, and $\mathbf{M} = \{\mu_1, \mu_2, \cdots, \mu_L\}$ are the super-keypoints' parameters, and z_r is the super-keypoint assignment of the keypoint. Here, we marginalize over all possibilities of the super-keypoint assignment z_r as we do not know which super-keypoint should the keypoint belong to in practice.

To deal with missing random variable z_r, we adopt the expectation-maximization algorithm, which alternates between estimating the super-keypoint assignments of novel keypoints $\{z_r\}_{r=1}^{T_y}$ (the expectation step) and optimizing for the prototypes $\{\mathbf{w}_r\}_{r=1}^{T_y}$ with the prior encoded in the assigned super-keypoints (the maximization step).

We introduce $q_r(z_r)$, a variational distribution that approximates the posterior $p(z_r|\mathbf{w}_r, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})$ over the super-keypoint assignment for the r^{th} keypoint. We then derive the lower bound for $\mathcal{L}_{MAP}$ as follows:

$$
\begin{aligned}
\mathcal{L}_{MAP} &= \sum_{r=1}^{T_y} \log p(\mathbf{w}_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) \\
&= \sum_{r=1}^{T_y} \sum_{z_r=1}^{L} q_r(z_r) \log p(\mathbf{w}_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) \\
&= \sum_{r=1}^{T_y} \sum_{z_r=1}^{L} q_r(z_r) \log \frac{p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})}{p(z_r|\mathbf{w}_r, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})} \\
&= \sum_{r=1}^{T_y} \sum_{z_r=1}^{L} q_r(z_r) \log \frac{q_r(z_r)}{p(z_r|\mathbf{w}_r, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})} \frac{p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})}{q_r(z_r)} \\
&= \underbrace{\sum_{r=1}^{T_y} \mathcal{D}_{\text{KL}}(q_r(z_r) \,||\, p(z_r|\mathbf{w}_r, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}))}_{\text{Kullback-Leibler (KL) divergence terms}} + \underbrace{\sum_{r=1}^{T_y} \mathbb{E}_{q_r(z_r)}[\log \frac{p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})}{q_r(z_r)}]}_{\text{lower bound}}.
\end{aligned}
$$

The lower bound can be written as:

$$
\text{Lower bound} = \sum_{r=1}^{T_y} \mathbb{E}_{q_r(z_r)}[\log p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})] + \sum_{r=1}^{T_y} \mathcal{H}(q_r(z_r)), \tag{2}
$$

where $\mathcal{H}(q_r(z_r)) = -\mathbb{E}_{q_r(z_r)}[q_r(z_r)]$ is the entropy of q_r.

B.1. Expectation step

In this step, we aims to find the most likely super-keypoint for each novel keypoint, i.e., we find $q_r(z_r)$ that best approximates the posterior $p(z_r|\mathbf{w}_r, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})$. Since $\mathcal{L}_{MAP}$ does not depend on $q_r(z_r)$, instead of minimizing the KL divergence between the variational distribution and the true posterior, we maximize the lower bound term for $q_r(z_r)$. Also note that keypoints of the same category must be assigned to different super-keypoints, so we constrain our optimization problem as follows:

$$
\begin{aligned}
&\underset{q_r(z_r),\, r\in\{1,\ldots,T_y\}}{\text{maximize}} \quad \sum_{r=1}^{T_y} \mathbb{E}_{q_r(z_r)}[\log p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})] \underbrace{-\mathbb{E}_{q_r(z_r)}[\log q_r(z_r)]}_{\mathcal{H}(q_r(z_r))} \\
&\text{subjects to} \quad q_r(z_r = z) \in \{0, 1\} \quad \text{and} \quad \sum_{r=1}^{T_y} q_r(z_r = z) \leq 1 \,\text{for}\, z \in \{1, \ldots, L\}.
\end{aligned}
$$

The constraints are introduced to reflect the nature of our task: they ensure that the novel keypoints are aligned with different super-keypoints. The constraints also make the entropy terms $\mathcal{H}(q_r(z_r))$ in the lower bound become constant with respect to $\{q_r(z_r)\}_{r=1}^{T_y}$. The objective is now equivalent to find the optimal bipartite matching between the novel keypoints and the super-keypoints such that the sum of log joint probability $\log p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})$ of the matching pairs are maximized. Below we show how to evaluate the log joint probability.

The log joint probability $\log p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})$ can be split into two terms:

$$\log p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) \tag{3}$$
$$= \log p(\mathbf{w}_r|z_r, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) + \log p(z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) \tag{4}$$

The first term is the posterior over the keypoint prototype given features extracted from the support samples and the super-keypoint assignment:

$$p(\mathbf{w}_r|z_r, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) = p(\mathbf{w}_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mu_{z_r}) \tag{5}$$
$$= \frac{p(\mathbf{w}_r, \{\mathbf{o}_r^i\}_{i=1}^K|\mu_{z_r})}{p(\{\mathbf{o}_r^i\}_{i=1}^K|\mu_{z_r})} \tag{6}$$
$$= \frac{p(\{\mathbf{o}_r^i\}_{i=1}^K|\mathbf{w}_r)p(\mathbf{w}_r|\mu_{z_r})}{\prod_{i=1}^K p(\mathbf{o}_r^i|\mu_{z_r})} \tag{7}$$
$$= \frac{\prod_{i=1}^K \mathcal{N}(\mathbf{o}_r^i|\mathbf{w}_r, \phi^2) \times \mathcal{N}(\mathbf{w}_r|\mu_{z_r}, \sigma^2)}{\prod_{i=1}^K \mathcal{N}(\mathbf{o}_r^i|\mu_{z_r}, \phi^2 + \sigma^2)}. \tag{8}$$

Here $p(\mathbf{o}_r^i|\mu_{z_r})$ is computed by marginalizing: $\int_{\mathbf{w}_r} p(\mathbf{o}_r^i|\mathbf{w}_r)p(\mathbf{w}_r|\mu_{z_r})d\mathbf{w}_r = \int_{\mathbf{w}_r} \mathcal{N}(\mathbf{o}_r^i|\mathbf{w}_r, \phi^2)\mathcal{N}(\mathbf{w}_r|\mu_{z_r}, \sigma^2)d\mathbf{w}_r = \mathcal{N}(\mathbf{o}_r^i|\mu_{z_r}, \phi^2 + \sigma^2)$

The second term is the posterior over the super-keypoint assignment given features extracted from support samples:

$$p(z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) = \frac{p(\{\mathbf{o}_r^i\}_{i=1}^K|\mu_{z_r})p(z_r)}{\sum_{z=1}^L p(\{\mathbf{o}_r^i\}_{i=1}^K|\mu_z)p(z)} \tag{9}$$
$$= \frac{\prod_{i=1}^K \mathcal{N}(\mathbf{o}_r^i|\mu_{z_r}, \phi^2 + \sigma^2) \times p(z_r)}{\sum_{z=1}^L \prod_{i=1}^K \mathcal{N}(\mathbf{o}_r^i|\mu_z, \phi^2 + \sigma^2) \times p(z)}. \tag{10}$$

The prior over the super-keypoints $p(z)$ is assumed to be uniform or estimated as statistic from base keypoints. We do not observe any difference in performance of the two choices.

As we can see that, the evaluation for the joint probability $\log p(\mathbf{w}_r, z_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})$ can be done independently for each novel keypoint without accessing to coordinate information of the keypoints. Hence, the simple matching cannot perform complex reasoning about their correlation as well as the poses of object. In the main paper, we refer to this matching method as the **simple matching**.

B.2. Maximization step

After finding the optimal super-keypoint assignments $z_r^\star$ for each novel keypoint, the optimal keypoint prototype $\mathbf{w}_r$ is then:

$$\mathbf{w}_r^{MAP} = \arg\max_{\mathbf{w}_r} \log p(\mathbf{w}_r, z_r^\star|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) \tag{11}$$
$$= \arg\max_{\mathbf{w}_r} \log p(\mathbf{w}_r|z_r^\star, \{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M}) + \underbrace{\log p(z_r^\star|\{\mathbf{o}_r^i\}_{i=1}^K, \mathbf{M})}_{\text{constant w.r.t } \mathbf{w}_r} \tag{12}$$
$$= \arg\max_{\mathbf{w}_r} \log p(\mathbf{w}_r|\{\mathbf{o}_r^i\}_{i=1}^K, \mu_{z_r^\star}) \tag{13}$$
$$= \arg\max_{\mathbf{w}_r} \log p(\{\mathbf{o}_r^i\}_{i=1}^K|\mathbf{w}_r) + \log p(\mathbf{w}_r|\mu_{z_r^\star}) \tag{14}$$
$$= \arg\max_{\mathbf{w}_r} \frac{1}{K}\sum_{i=1}^K \log \mathcal{N}(\mathbf{o}_r^i|\mathbf{w}_r, \phi^2) + \log \mathcal{N}(\mathbf{w}_r|\mu_{z_r^\star}, \sigma^2) \tag{15}$$
$$= \arg\min_{\mathbf{w}_r} \frac{1}{K}\frac{1}{\phi^2}\sum_{i=1}^K (\mathbf{o}_r^i - \mathbf{w}_r)^\top(\mathbf{o}_r^i - \mathbf{w}_r) + \frac{1}{\sigma^2}(\mathbf{w}_r - \mu_{z_r^\star})^\top(\mathbf{w}_r - \mu_{z_r^\star}). \tag{16}$$
$$\tag{17}$$

Setting the derivative of the objective with respect to $\mathbf{w}_r$ to zero, we obtain:

$$\mathbf{w}_r^{MAP} = \frac{\frac{1}{K}\frac{1}{\phi^2}\sum_{i=1}^{K}\mathbf{o}_r^i + \frac{1}{\sigma^2}\mu_{z_r^\star}}{\frac{1}{\phi^2}+\frac{1}{\sigma^2}} \tag{18}$$

$$= \alpha\frac{1}{K}\sum_{i=1}^{K}\mathbf{o}_r^i + (1-\alpha)\mu_{z_r^\star}, \tag{19}$$

where $\alpha = \frac{\frac{1}{\phi^2}}{\frac{1}{\phi^2}+\frac{1}{\sigma^2}}$. In this work, we set the hyperparameters ϕ and σ such that $\alpha = 0.5$, which balances the effect of visual cues from support samples and the prior knowledge from the super-keypoint.

C. Matching keypoints and super-keypoints with optimal transport

Given the score matrix $\mathbf{S}$, we resort to optimal transport to find the maximum score matching between the keypoints and the super-keypoints, i.e., finding a transport matrix $\mathbf{Q}$ such that it maximizes the inner product $\langle \mathbf{Q}, \mathbf{S}\rangle$. Typically, for an image of category y, the number of keypoints is smaller than the number of the available super-keypoints, i.e. $T_y < L$. Hence, there are $L - T_y$ super-keypoints that cannot be matched with any keypoint. Following [5], we introduce a dummy keypoint to match with the redundant super-keypoints. An augmented score maxtrix $\bar{\mathbf{S}}$ is then constructed by appending to $\mathbf{S}$ an additional row, representing the matching scores between the dummy keypoint and the super-keypoints:

$$\begin{aligned}
&\bar{\mathbf{S}}_{r,z} = \mathbf{S}_{r,z}, \quad \text{for } r, z \in \{1,\dots,T_y\}\times\{1,\dots,L\},\\
&\bar{\mathbf{S}}_{T_y+1,z} = \pi, \quad \text{for } z \in \{1,\dots,L\},
\end{aligned}$$

where π is a learnable parameter. The matching problem is now extended to finding the optimal transport matrix $\bar{\mathbf{Q}}^\star \in \mathbb{R}^{(T_y+1)\times L}$:

$$\begin{aligned}
\bar{\mathbf{Q}}^\star &= \arg\max_{\bar{\mathbf{Q}}} \langle \bar{\mathbf{Q}}, \bar{\mathbf{S}}\rangle,\\
\text{subject to } &\bar{\mathbf{Q}}\mathbf{1}_L = a, \bar{\mathbf{Q}}^\top \mathbf{1}_{T_y} = b,\\
\text{where } &a = \frac{1}{L}[\mathbf{1}_{T_y}, L-T_y]^\top \quad \text{and} \quad b = \frac{1}{L}\mathbf{1}_L.
\end{aligned}$$

Here $\mathbf{1}_d$ denotes all-one vector of dimension d. The constraints ensure that each keypoint is matched to one super-keypoint and vice versa, while the dummy keypoint is able to match with $L - T_y$ super-keypoints. $\bar{\mathbf{Q}}^\star$ can be solved efficiently with the Sinkhorn algorithm [1], which is differentiable and can be used to train the network end-to-end on the base categories. After obtaining the optimal $\bar{\mathbf{Q}}^\star$, we simply discard its last row to produce $\mathbf{Q}^\star$, which is used in the estimation of keypoint prototype $\mathbf{w}_r^{\text{ESCAPE}}$.

D. Training matching network

After discovering super-keypoints from the base keypoints, we are ready to train our matching network. Specifically, for a base category $y \in C_{base}$ with interested keypoints $\{j_r\}_{r=1}^{T_y}$, its keypoints have the corresponding super-keypoints $\{z_r\}_{r=1}^{T_y}$. Let $\bar{\mathbf{S}}^{i,y}$ denote the score matrix computed for an annotated sample $(\mathbf{I}^i, P^i)$ of y, and let $\bar{\mathbf{Q}}^{i,y}$ denote the optimal transport matrix for $\bar{\mathbf{S}}^{i,y}$. We train the matching network with the following loss:

$$\mathcal{L}_{SuperGlue} = -\frac{1}{n}\sum_{y\in C_{base}}\sum_{(\mathbf{I}^i,P^i)\in\mathcal{D}_{base}^y}\Big[\sum_{r=1}^{T_y}\log\bar{\mathbf{Q}}_{r,z_r}^{i,y} + \sum_{z\in{}^-Z_y}\log\bar{\mathbf{Q}}_{T_y+1,z}^{i,y}\Big],$$

where ${}^-Z_y = \{1,\dots,L\}\setminus\{z_r\}_{r=1}^{T_y}$ is the set of super-keypoints that are not assigned to any keypoints of y, and so they are matched with the dummy keypoint. Intuitively, we minimize the negative log-likelihood of the correct assignments.

E. Architecture of matching network

As in [5], our matching network relies heavily on the attention mechanism [6]. Below we provide a background on the attention and show how we leverage it in the design of our matching network.

E.1. Preliminaries: attention

An attention function operates on a set of queries and a set of key-value pairs. It outputs a set of features with the same number of elements as the queries. Suppose that the queries, keys and values are respectively organized into the matrices $\mathbf{X}_q \in \mathbb{R}^{q\times d}, \mathbf{X}_k \in \mathbb{R}^{k\times d}, \mathbf{X}_v \in \mathbb{R}^{k\times d}$. The attention function is then:

$$\text{Attention}(\mathbf{X}_q, \mathbf{X}_k, \mathbf{X}_v) = \text{softmax}(\frac{\mathbf{X}_q \mathbf{X}_k^\top}{\sqrt{d}})\mathbf{X}_v.$$

Intuitively, for each query, the output is a weighted sum of the values in which the weight for a value is obtained by measuring the strength of the connection between the query and the corresponding key. In this way, a query can 'attend' to all the key-value pairs.

Multi-head attention. We adopt the multi-head attention from [6]. A multi-head attention first linearly projects the queries, keys and values inputs into multiple versions and applies attention to each of them. It then concatenates the outputs of the attentions and linearly projects it one more time.

$$\begin{aligned}\text{Multi-Head-Attention}(\mathbf{X}_q, \mathbf{X}_k, \mathbf{X}_v) &= \text{Concat}(\text{head}_1, \dots, \text{head}_h)\mathbf{O},\\ \text{where head}_i &= \text{Attention}(\mathbf{X}_q^{(i)}, \mathbf{X}_k^{(i)}, \mathbf{X}_v^{(i)}).\end{aligned}$$

Here, a triplet $(\mathbf{X}_q^{(i)}, \mathbf{X}_k^{(i)}, \mathbf{X}_v^{(i)})$ is the i^{th} linearly projected version of $(\mathbf{X}_q, \mathbf{X}_k, \mathbf{X}_v)$; $\mathbf{O}$ is a weight matrix.

E.2. Self-attention and cross-attention layers

The architecture of our matching network involves two parts: self-attention and cross-attention. The self-attention is simply letting a sequence to attend to itself whereas the cross-attention is letting a sequence to attend to another one. We also leverage residual connections, layer normalization and linear projection in our designs. Fig. 1 shows the architecture of the self-attention and cross-attention layers.

E.3. Matching network

Our matching network takes two sequences of features as inputs: the keypoint features $\{\mathbf{u}_r\}_{r=1}^{T_y}$ and the super-keypoint features $\{\mathbf{v}_z\}_{z=1}^{L}$. Let us organize $\{\mathbf{u}_r\}_{r=1}^{T_y}$ and $\{\mathbf{v}_z\}_{z=1}^{L}$ into two matrices $\mathbf{U} \in \mathbb{R}^{T_y\times d}$ and $\mathbf{V} \in \mathbb{R}^{L\times d}$ respectively. The matching network transforms the two sequences as follows:

$$\begin{aligned}
\mathbf{U}^{(0)} &= \mathbf{U}, \quad \mathbf{V}^{(0)} = \mathbf{V} \\
\hat{\mathbf{U}}^{(i-1)} &= \text{Self-Attention}_{(2i)}(\mathbf{U}^{(i-1)}), && i = 1 \dots U \\
\hat{\mathbf{V}}^{(i-1)} &= \text{Self-Attention}_{(2i+1)}(\mathbf{V}^{(i-1)}), && i = 1 \dots U \\
\mathbf{U}^{(i)} &= \text{Cross-Attention}_{(2i)}(\mathbf{U}^{(i-1)}, \mathbf{V}^{(i-1)}), && i = 1 \dots U \\
\mathbf{V}^{(i)} &= \text{Cross-Attention}_{(2i+1)}(\mathbf{V}^{(i-1)}, \mathbf{U}^{(i-1)}), && i = 1 \dots U \\
\mathbf{U}^\star &= \mathbf{U}^U, \quad \mathbf{V}^\star = \mathbf{V}^U
\end{aligned}$$

The output of the matching network $\{\mathbf{u}_r^\star\}_{r=1}^{T_y}$ and $\{\mathbf{v}_z^\star\}_{z=1}^{L}$ are rows of $\mathbf{U}^\star$ and $\mathbf{V}^\star$ respectively.

To implement the network, we modified the code from https://github.com/lucidrains/vit-pytorch/blob/main/vit_pytorch/vit_1d.py. We also adopt the default layer configuration from the above code, summarized in Tab. 1.

F. More experiments

Data augmentation for training includes random scaling ($[-15\%, 15\%]$) and random rotation ($[-15°, 15°]$). We train our network with Adam optimizer [4] for 60 epochs. The learning rate is initialized as $5e^-4$ and scheduled to $5e^-5$, $5e^-6$ at epoch number 40, 50 respectively. With one GeForce GTX 1080 Ti, the pretraining for the feature extractor with batch size 32 takes roughly 3 hours whereas the training of the matching network with batch size 16 takes 10 hours.

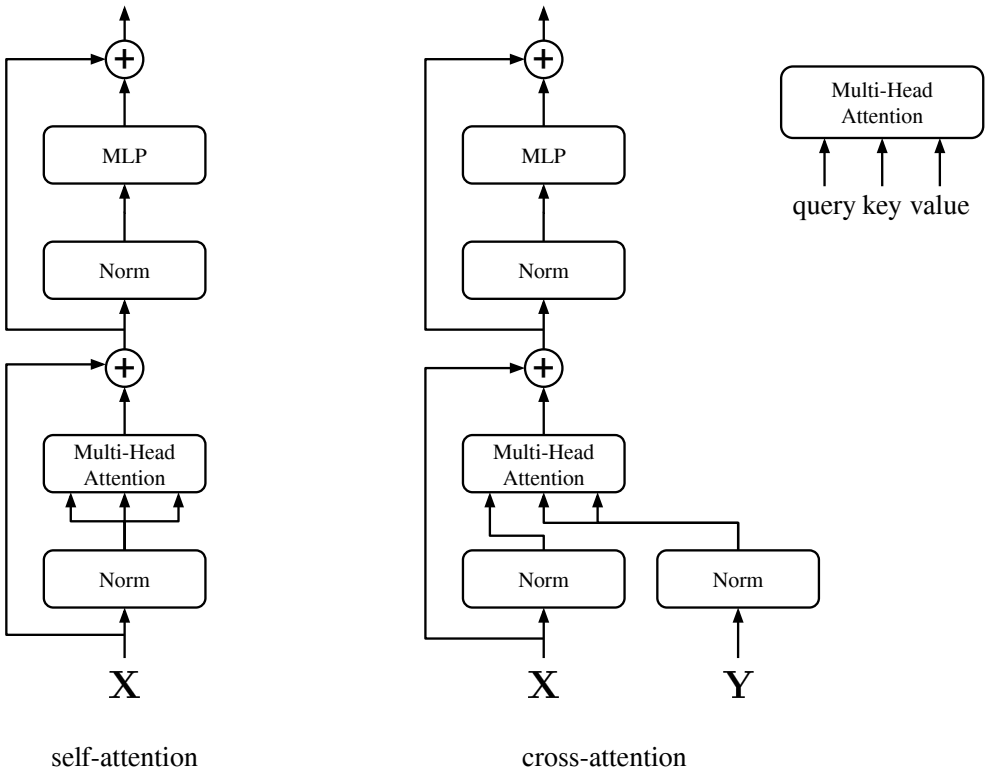

Figure 1. Architecture of the self-attention block and the cross attention block. MLP is a block of a linear projection, GELU activation [3] and another linear projection.

Table 1. Configuration of the matching network.

Number of layers U	Hidden dim	MLP dim	Heads
6	1024	2048	8

Table 2. Cross supercategory results.

Method	Human body	Human face	Vehicle	Furniture	Average
POMNet	73.82	79.63	34.92	47.27	58.91
CapeFormer	**83.59**	82.76	**44.02**	47.38	64.44
ESCAPE	80.60	**84.13**	41.39	**55.49**	**65.29**

F.1. Cross super-category

Similar to 'dummy keypoint' (*cf*. Sec. C.1 in supp), ESCAPE can generalize to cross super-category setting by utilizing a 'dummy super-keypoints'. This ensures that keypoints significantly different from the learned super-keypoints are matched to the dummy one during inference, thus preventing unrelated super-keypoints from influencing query predictions. ESCAPE exhibits superior performance as shown in the table below. This highlights the robust generalization capabilities of our approach.

References

[1] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013. 4

[2] Mingjing Du, Shifei Ding, and Hongjie Jia. Study on density peaks clustering based on k-nearest neighbors and principal component analysis. *Knowledge-Based Systems*, 99:135–145, 2016. 1

[3] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 6

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[5] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 4

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4, 5

[7] Wang Zeng, Sheng Jin, Wentao Liu, Chen Qian, Ping Luo, Wanli Ouyang, and Xiaogang Wang. Not all tokens are equal: Human-centric visual analysis via token clustering transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11101–11111, 2022. 1