Matching Is Not Enough: A Two-Stage Framework for Category-Agnostic Pose Estimation

Min Shi[1, 2, *] Zihao Huang[1, *] Xianzheng Ma[2] Xiaowei Hu[2] Zhiguo Cao[1, †]
[1] School of AIA, Huazhong University of Science and Technology
[2] Shanghai AI Laboratory
{min_shi, zihaohuang, zgcao}@hust.edu.cn, {maxianzheng, huxiaowei}@pjlab.org.cn

Abstract

Category-agnostic pose estimation (CAPE) aims to predict keypoints for arbitrary categories given support images with keypoint annotations. Existing approaches match the keypoints across the image for localization. However, such a one-stage matching paradigm shows inferior accuracy: the prediction heavily relies on the matching results, which can be noisy due to the open set nature in CAPE. For example, two mirror-symmetric keypoints (e.g., left and right eyes) in the query image can both trigger high similarity on certain support keypoints (eyes), which leads to duplicated or opposite predictions. To calibrate the inaccurate matching results, we introduce a two-stage framework, where matched keypoints from the first stage are viewed as similarity-aware position proposals. Then, the model learns to fetch relevant features to correct the initial proposals in the second stage. We instantiate the framework with a transformer model tailored for CAPE. The transformer encoder incorporates specific designs to improve the representation and similarity modeling in the first matching stage. In the second stage, similarity-aware proposals are packed as queries in the decoder for refinement via cross-attention. Our method surpasses the previous best approach by large margins on CAPE benchmark MP-100 on both accuracy and efficiency. Code available at github.com/flyinglynx/CapeFormer

1. Introduction

Humans can quickly grasp the essentials of a keypoint for arbitrary objects (*e.g.,* the nose of an animal) and then pinpoint the same keypoint on another object in the same category. However, most pose estimation models are still category-specific, which cannot be applied to a new category unless they are trained with sufficient category-specific data. Recently, category-agnostic pose estimation (CAPE) [40] is proposed to address such a generic pose estimation problem. CAPE aims to localize instance keypoints for arbitrary categories given one or few support images with keypoint annotations (support keypoints).

Unlike category-specific pose estimation models that learn to recognize specific keypoints for a fixed category, CAPE models learn to represent and compare keypoints for open-world categories. As shown in Fig. 1 (a), existing CAPE approaches [40] match the support keypoints across the query image in an embedding space. Then, the match-

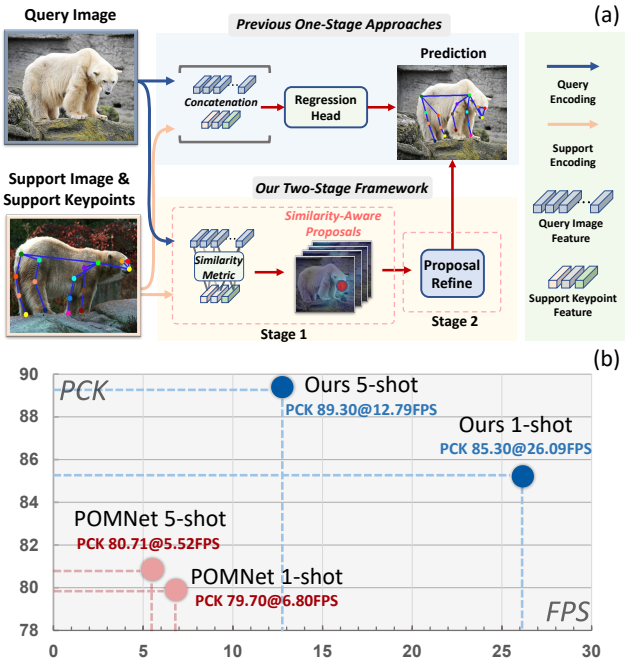

Figure 1. **Comparison with existing methods.** (a) Compared with previous one-stage matching paradigm that entirely relies on keypoint matching, we propose a novel two-stage framework, which learns to correct unreliable matching results in the second stage. (b) The proposed approach attains better accuracy and efficiency compared with the previous best approach.

*: Equal contribution. †: Corresponding author.

ing results are used as clues for keypoint localization. Although a sophisticated feature embedding pipeline is elaborated [40], we find that such a one-stage pipeline shows inferior accuracy and efficiency for CAPE tasks. Specifically, the prediction heavily relies on matching results. However, due to the open set nature in CAPE, the similarity results often contain some noises: 1) insufficient similarity when support and query instances differ significantly in poses, textures, or styles; 2) false positive similarity is triggered when points in the query image share similar appearance with support keypoints. For example, one-stage matching paradigm has difficulties distinguishing the left and right of keypoints due to similar visual characteristics. Thus, the left eye in support keypoints can be indiscriminately predicted to the right eye in the query image.

In this paper, we remedy these issues from two aspects. First, to refine the matching results, we introduce a two-stage framework for CAPE, where matched keypoints in the first stage are viewed as similarity-aware position proposals. The model learns to refine the proposals in the second stage. In addition, to construct a robust matching process, we also improve the representation quality and similarity metrics.

Specifically, we instantiate the two-stage framework with a transformer, termed CAPE transformer (CapeFormer). The transformer encoder encodes the support keypoints and query images in the first stage, with specific designs to improve the representation quality. We first design a query-support joint refine encoder to mutually transfer information among support keypoints and the query images, thus narrowing the gap between the support and query instances (poses or styles). Then, we add a support keypoint identifier to alleviate the ambiguity between two support keypoints when they are geometrically close or with similar appearance. After the feature encoding, similarity-aware position proposals are generated for the second stage. We directly match the support keypoint with query image features using a learnable inner-product [34]. The similarity peaks are selected as the similarity-aware position proposals for support keypoints. In the second stage, the position proposals are packed with support keypoint features as the queries in the transformer decoder. In each decoder layer, queries extract relevant features with cross-attention, which are used as clues to update the proposals.

We evaluate our method on a category-agnostic pose estimation benchmark MP-100 [40]. Our method outperforms the previous best approach POMNet [40] by large margins, with an improvement of 5.6% and 8.6% under 1-shot and 5-shot settings, respectively. When training and test categories almost have no properties in common, a more significant improvement (up to 10.5%) can be obtained compared with POMNet. Note that, our method is trained end-to-end and shows better efficiency compared with previous best approach, as in Fig. 1 (b).

Our contributions can be summarized as follows:

- We propose a two-stage framework for category-agnostic pose estimation tackling the noisy matching results.
- We instantiate the two-stage framework with a transformer model, termed CapeFormer, with specific designs to enhance the representation and similarity modeling in the matching pipeline.
- CapeFormer outperforms the previous best approach significantly in both accuracy and efficiency.

2. Related Work

2.1. Pose Estimation

Pose estimation aims to localize the semantic keypoints of instances. Most pose estimation approaches are targeted for one specific category, *e.g.*, human [1, 19, 45, 46], animals [3, 17, 20], or vehicles [29, 36]. According to how points are localized, pose estimation methods can be categorized into regression-based [18, 21, 28, 37, 38], heatmap-based [4, 44], offset-based [12] and query-based [33] ones. In this work, we address category-agnostic pose estimation, which has not been well-studied yet. Compared with category-specific pose estimation, category-agnostic pose estimation focuses on constructing a generic representation and similarity metrics: keypoints are predicted by comparing the support keypoints and the query images in the embedding space.

2.2. Category-Agnostic Vision Models

Category-agnostic vision models aim to build a generic model that can be applied to any categories on a particular task, *e.g.*, object detection [14], segmentation [43], object counting and localization [24, 34]. This setting is also known as the few-shot learning. Recently, POMNet [40] first proposes the category-agnostic pose estimation task and elaborates a dataset MP-100. Most category-agnostic models can be classified into meta-learning and metric-learning approaches. Meta-learning approaches [11] aim to initialize the model with a set of optimal parameters, where the model can quickly fit the new tasks by learning from a few support samples. Metric learning-based approaches match the support samples with query images in an embedding space. Then, the matching results are used as clues for subsequent tasks. For pose estimation, POMNet [40] encodes query images and support keypoints with a transformer. Then, a regression head directly infers the similarity from the concatenated support keypoint and query image features. We find that simply using matching results for point localization is challenging. Hence, we extend such a matching paradigm with a two-stage framework to correct the unreliable matching results for accurate predictions.

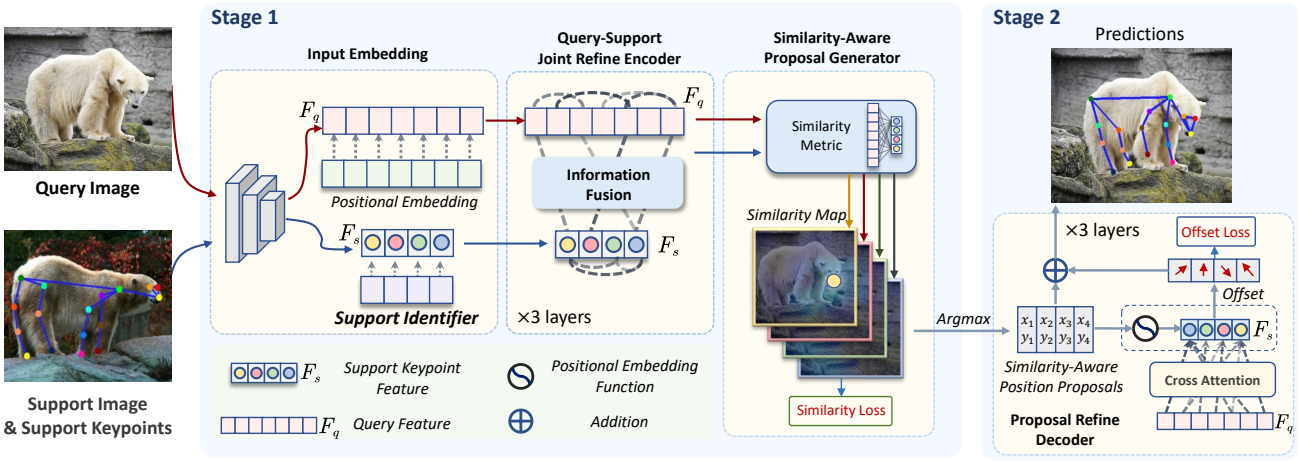

Figure 2. **Illustration of the proposed CapeFormer.** CapeFormer consists of a matching stage where similarity-aware position proposals are generated, and a refine stage that further calibrates the proposals. A support identifier and query-support joint refine encoder are proposed to improve the representation quality.

2.3. Semantic Correspondence

Semantic correspondence (SC) [8, 15] aims to match the points that have identical semantic meaning with a dense correspondence field. Hence, SC models can be directly applied on the CAPE: localize the corresponding points for the support keypoints with the estimated correspondence field. Typical SC models first match each pair of pixels and use the matching score to construct correspondence. Similar to our two-stage design that alleviates the noise on similarity matching, a series of SC models refine the matching scores with geometry constraints [26], high-dimensional convolutions [31], and attention mechanism [6, 7]. However, we find directly applying SC models for CAPE is inefficient and yields inferior performance. Besides, most techniques for the matching refine in SC are based on the dense matching volume, which is not available in CAPE, as only the sparse support keypoints are matched.

3. CAPE Transformer

Class-agnostic pose estimation (CAPE) aims to estimate instance keypoints given one or a few support images with keypoint annotations. Specifically, we denote the query image as I_q and the support image as I_s, which has K predefined support keypoints. The CAPE model will infer the positions of the corresponding K keypoints in the query images. For simplicity, we use the 1-shot setting for illustration (the number of support images equals one).

3.1. Two-Stage Framework for CAPE

Different from the previous one-stage paradigm, CapeFormer features a two-stage framework. As in Fig. 2, the first stage consists of the input embedding, a support-query joint refine encoder, and a similarity-aware proposal generator. A proposal refine decoder forms the second stage. The input embedding encodes the query image and support keypoints into the embedding space (Sec. 3.2). We denote the features for the query image and support keypoints as F_q and F_s, which are further refined with the query-support joint refine encoder (Sec. 3.3). The proposal generator measures the similarity between F_s and F_q and the similarity peaks are selected as the similarity-aware position proposals (Sec. 3.4). Then, in the second stage, the similarity-aware proposals are gradually refined into the final predictions (Sec. 3.5).

3.2. Input Embedding

The input embedding is illustrated in Fig. 3. We first project the support and query image into feature maps with a shared backbone. The flattened feature map of the query image is used as its representation, denoted by $F_q \in \mathbb{R}^{c \times n}$, where n equals the number of pixels in the feature map and c denotes the embedded dimension. To obtain the features of the support keypoint, we conduct a "soft" ROI pooling [30] on the support image feature map. K soft masks are generated by placing a Gaussian kernel centered at support keypoints, and then multiplied with the support feature map. The summation of masked support feature map is adopted as the support keypoint feature $F_s \in \mathbb{R}^{K \times c}$. For multiple support images, *e.g.*, 5-shot settings, the support keypoint features from different images are averaged.

Support keypoint identifier. When encoding support keypoints, each support keypoint's positional and context information is discarded during ROI pooling. Hence, when two support keypoints are geometrically close or with similar appearance (*e.g.*, left and right eyes), they can be indistinguishable in the embedding space. Such ambiguity can lead to inaccurate predictions that are duplicated or mirror-symmetric to the ground truth.

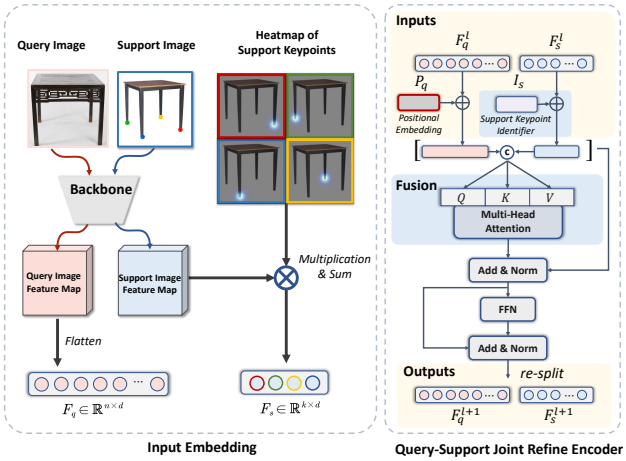

Figure 3. **Illustration of the input embedding and the support-query joint refine encoder.**

To alleviate such ambiguity, we propose a simple but effective support keypoint identifier that gives an "identity" to each support keypoint. The implementation is embarrassingly simple: K embedding vectors are generated via sinusoidal encoding, denoted by $I_s \in \mathbb{R}^{K\times c}$. This identifier is added to F_s ahead of attention calculation, which will be discussed later in this section. Such a simple modification can lead to over 3% improvement with negligible computation overhead.

3.3. Query-Support Refine Encoder

Directly matching the support keypoints with the query image right after the input embedding is challenging, as instances in the support and query images can be different, such as different colors, textures, and poses. Thus, the target and support keypoints may not possess sufficient similarity in the embedding space [10, 34]. An intuitive way to alleviate this is to fuse relevant query and support features. Hence, the query and support features can be pulled closer in the embedding space.

To this end, we implement information fusion and transfer among support and query with self-attention. As illustrated in Fig. 3, the inputs of the l^{th} encoder layer are query and support features, denoted as F_q^l and F_s^l. We augment the F_q^l with positional embedding as in DETR [5], and add the support keypoint identifier into the F_s^l. The augmented F_q^l and F_s^l are concatenated into a feature sequence with the length of $n+K$. The sequence is then fed into the standard multi-head self-attention layer [39]. The output sequence from the self-attention layer is processed by a feed forward network (FFN). Then, the sequence are re-split to the support keypoint and query image features, which are used as the inputs for the next encoder layer.

3.4. Similarity-Aware Proposal Generator

The similarity-aware proposal generator matches the support keypoint features with the query features to obtain the similarity maps, whose peaks are selected as the similarity-aware proposals. Previous state-of-the-art approach POMNet [40] measures similarity implicitly: a regression head directly infers similarity maps (heatmaps) from the concatenation of support keypoint and query features. However, regressing similarity without explicit similarity modeling has the risk of overfitting in class-agnostic setting [34]. Moreover, considerable computation overhead is introduced. For example, given 100 support keypoint features with 256 channels as in our case, the similarity maps are obtained by forwarding 512-channel feature maps with an FCN for 100 times.

To embrace both efficiency and generality, we use a learnable inner-product [34] to model similarity explicitly. Specifically, we first calculate a channel attention weight $a \in \mathbb{R}^{K\times c}$ conditioned on $F_s \in \mathbb{R}^{K\times c}$. Then, the similarity map M is obtained as:

$$M = (W_q F_q)(\boldsymbol{a} \circ W_s F_s)^T , \tag{1}$$

where W_q and W_s are two learnable projection matrices in $\mathbb{R}^{c\times c}$ and $\circ$ denotes the Hadamard product. Then, similarity maps $M \in \mathbb{R}^{n\times K}$ are reshaped into 2D with the shape of $H \times W$. The channel attention helps stressing the key patterns during inner product computation [34].

The maximums from the similarity maps are selected as the similarity-aware position proposals for support keypoints. To reduce the round-off errors on the down-scaled resolutions, instead of directly selecting the maximums, we first generate a 3×3 local window around the similarity peaks and normalize this local window with `softmax`. Then, we take the expectation of the coordinates in each local window as the proposal, which can be formulated as:

$$x_k = \sum_{i,j\in W_k} i \cdot \frac{\exp\left(M_{k,ij}\right)}{\sum_{i,j\in W} \exp\left(M_{ij}\right)} , \tag{2}$$

$$y_k = \sum_{i,j\in W_k} j \cdot \frac{\exp\left(M_{k,ij}\right)}{\sum_{i,j\in W} \exp\left(M_{ij}\right)} , \tag{3}$$

where W_k denotes the local window for the k^{th} support keypoint and i, j indexes the horizontal and vertical coordinates. Here we denote the similarity-aware position proposals for all the support keypoints as $P \in \mathbb{R}^{K\times 2}$.

3.5. Proposal Refine Decoder

As mentioned in Sec. 1, due to the unreliable matching process, directly using the matched keypoints from the first stage as predictions is prone to error. Intuitively, each support keypoint can be corrected by jointly considering other

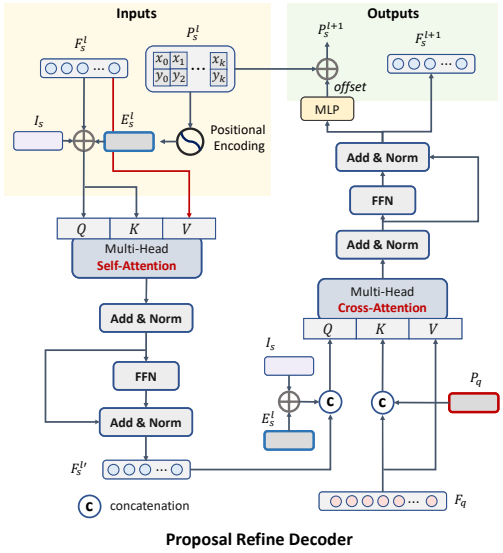

Figure 4. **Illustration of the proposal refine decoder.** The decoder consists of a self-attention and a cross-attention layer

support keypoint predictions and keypoint-related information in the query image. Hence, we devise the second stage to unleash the potential of correcting the raw similarity-aware position proposals.

Specifically, as shown in Fig. 4, we design a proposal refine decoder, where the support keypoint embedding together with the keypoint positions are fed into the decoder layer. Each layer consists of two steps: 1) self-attention among support keypoints makes each proposal aware of other keypoints' positions and contents; 2) cross-attention that helps extract relevant content from the query feature for each support keypoint. By integrating these two attention mechanisms, the decoder can further refine the position proposals and produce more accurate predictions.

Self-attention among support keypoints. Self-attention enables interactions among the support keypoints, which makes each proposal aware of other keypoints' positions and contents. To prepare the inputs for the self-attention layer, the support keypoint positions are first converted into sinusoidal positional encoding $E_s^l \in \mathbb{R}^{K\times c}$. Then, the support keypoint identifier I_s and position embedding E_s^l are added into F_s^l forming the query and key for the first self-attention layer. The support keypoint feature F_s^l is directly used as the value. The multi-head self-attention layer is followed by a transformer-style feed-forward network (FFN) with residual connection and layer normalization [2]. We denote the support feature processed by the FFN as $F_s^{l\prime}$.

Cross-attention between support and query features. Cross-attention helps extract relevant contents from the query feature F_q for each support keypoint. As illustrated in Fig. 4, we use the concatenation of F_s' and the summation of I_s and E_s^l as the query for the cross-attention layer. Correspondingly, query image features F_q and positional embedding P_q are concatenated as the key, and F_q are used as the value. Instead of addition, we concatenate the positional embedding and support identifier for key and value, following DAB-DETR [23]. An FFN processes the output from the cross-attention layer to form the support keypoint features for the next decoder layer, *i.e.*, F_s^{l+1}.

Position update. We use the refined support feature F_s^{l+1} to update the keypoint positions P^l to P^{l+1}. An MLP takes F_s^{l+1} as input and infers the offset $O^{l+1} \in \mathbb{R}^{K\times 2}$ pointing to the ground truth. Then, the keypoint positions are updated by:

$$P^{l+1} = \sigma\left(O^l + \sigma^{-1}\left(P^l\right)\right), \tag{4}$$

where σ and σ^{-1} denote the `sigmoid` and its inverse function. Note that the similarity-aware proposals are used as the keypoint positions for the first decoder layer, *i.e.*, P^1. The updated keypoint positions from the last decoder layer are used as the final prediction.

3.6. Supervision Signal

CapeFormer incorporates two supervision signals: the similarity loss and the offset loss. The similarity loss facilitates the learning of representation and the similarity metric in the first stage. The offset loss supervise the learning of proposal refinement.

Similarity loss. The similarity loss is imposed on the similarity maps M from the similarity-aware proposal generator. We simultaneously constrain the shape of similarity map and the accuracy of similarity-aware proposals with a heatmap term $\mathcal{L}_h$ and an expectation term $\mathcal{L}_e$.

For heatmap loss, we use `sigmoid` to normalize similarity map M. Then, $\mathcal{L}_h$ are computed with ℓ_2 loss as:

$$\mathcal{L}_h = \frac{1}{K}\sum_{i=1}^{K}\frac{1}{H\cdot W}\left\|\text{sigmoid}\left(M_i\right) - H_i\right\|_2, \tag{5}$$

where H_i denotes the ground truth heatmap for the i^{th} keypoint. For $\mathcal{L}_e$, we first normalize the similarity map with `softmax` and then take the expectation of the coordinates as:

$$E\left(P_k\right) = \sum_{j=1}^{H}\sum_{i=1}^{W}\frac{\exp\left(M_{k,ij}\right)}{\sum\exp\left(M_k\right)}\cdot(i,j), \tag{6}$$

where $E(P_k)$ denotes the coordinate expectation for the k^{th} keypoint. H and W denote the height and width for similarity map. Then we use ℓ_1 loss to calculate $\mathcal{L}_e$ as:

$$\mathcal{L}_e = \frac{1}{K}\sum_{i=1}^{K}\left|E\left(P_i\right) - \hat{P}_i\right|_1, \tag{7}$$

where $\hat{P}_i$ denotes the ground truth coordinate of the i^{th} point.

Table 1. **Results on the MP-100 dataset.** Performance (PCK) under 1-shot and 5-shot settings. The proposed CapeFormer achieves the best PCK on all the splits under 1-shot and 5-shot settings.

Method	1-shot						5-shot					
	Split 1	Split 2	Split 3	Split 4	Split 5	Average	Split 1	Split 2	Split 3	Split 4	Split 5	Average
ProtoNet [35]	46.05	40.84	49.13	43.34	44.54	44.78	60.31	53.51	61.92	58.44	58.61	58.56
MAML [11]	68.14	54.72	64.19	63.24	57.20	61.50	70.03	55.98	63.21	64.79	58.47	62.50
Fine-tune [27]	70.60	57.04	66.06	65.00	59.20	63.58	71.67	57.84	66.76	66.53	60.24	64.61
POMNet [40]	84.23	78.25	78.17	78.68	79.17	79.70	84.72	79.61	78.00	80.38	80.85	80.71
CapeFormer	**89.45**	**84.88**	**83.59**	**83.53**	**85.09**	**85.31**	**91.94**	**88.92**	**89.40**	**88.01**	**88.25**	**89.30**

Offset regression loss. Supposing a total of L decoder layers are used, we define the offset regression loss $\mathcal{L}_o$ as:

$$\mathcal{L}_o = \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{K}\left|P_i^{l+1} - \hat{P}_i\right|_1 . \tag{8}$$

The overall training loss is calculated as $\lambda_h \cdot \mathcal{L}_h + \mathcal{L}_e + \mathcal{L}_o$, where λ_h is a pre-defined weight for the heatmap loss.

4. Experiments

4.1. Implementation Details

Network architecture. We use the ResNet-50 [13] pre-trained from ImageNet [9] as the backbone following POMNet [40]. Different from POMNet, the backbone is shared by query and support images in our cases for efficiency. Both the query-support joint refine encoder and the proposal refine decoder have three layers. Refer to supplementary for more details on network architectures.

Training strategy. We use Adam [16] optimizer to train the model for 200 epochs with the batch size of 16. The learning rate is set as $1e^{-5}$ and decays by $10\times$ on the 160^{th} and 180^{th} epoch. The weight for heatmap loss is empirically set as 2.0 to make different loss terms lie in the same magnitude. Data augmentation and pre-processing are kept the same with POMNet [40] for fair comparisons.

Dataset and metric. We use MP-100 dataset [40] to train and evaluate our method. Samples are collected from the existing category-specific pose estimation datasets [22, 32, 41]. MP-100 dataset encompasses over 18k images from 100 different categories. Keypoint numbers of different categories vary from 8 to 68. To train and evaluate the model, the samples are organized into five splits. On each split, the training, validation, and test categories have no overlaps, *i.e.,* the categories for evaluation are not accessed during training. We use the Probability of Correct Keypoint (PCK) [42] as the quantitative metric. The threshold for PCK is set to 0.2 following POMNet [40].

4.2. Results on MP-100

We compare our method with the previous best CAPE method POMNet [40] and three baselines: ProtoNet [35], MAML [11], and Fine-tune [27]. PCKs of different approaches on the MP-100 dataset under 1-shot and 5-shot settings are reported in Table 1, where we can see our method significantly outperforms POMNet [40], and improves the average PCK by 5.6% under the 1-shot setting and 8.6% under the 5-shot setting.

1-shot CapeFormer can surpass other methods with five support images, demonstrating the robustness of the proposed two-stage framework when the support information is limited. When increasing the shots from one to five, CapeFormer obtains a significant improvement of 4.0% on PCK. For POMNet, the improvement from 1-shot to 5-shot is 1.0%. This shows that CapeFormer can better utilize the additional information when more support samples are available. Note that we use the same way as POMNet [40] to encode the support keypoint features for fair comparisons. We also include a comparison with the semantic correspondence approaches in the supplementary.

Table 2. **Cross super-category evaluation on MP-100 dataset.** Experiments are conducted under the 1-shot setting.

Method	Human Body	Human Face	Vehicle	Furniture
ProtoNet [35]	37.61	57.80	28.35	42.64
MAML [11]	51.93	25.72	17.68	20.09
Fine-tune [27]	52.11	25.53	17.46	20.76
POMNet [40]	73.82	79.63	34.92	47.27
CapeFormer	**83.44**	**80.96**	**45.40**	**52.49**

4.3. Cross Super-Category Pose Estimation

In the MP-100 dataset, training, validation, and test categories do not overlap. However, some categories may still share similar characteristics, such as the faces of different animals. To better evaluate the generalization ability on significantly different categories, we conduct a cross-super-category evaluation following POMNet [40]. Four super-categories: human face, human body, vehicle, and furniture, are collected from the MP-100 dataset as the test categories, while the other categories are used as the training samples.

As in Table 2, CapeFormer achieves the state-of-the-art performance. The advantages on the human body and vehicle are particularly remarkable, with an improvement

Figure 5. **Qualitative results of our method.** We visualize the keypoint predictions under 1-shot setting. The left column denotes the support image with corresponding keypoints. The middle column denotes the results without proposal refine decoder, and the right column are results from the full method.

of 9.62% and 10.48% on PCK, respectively. Compared with the standard setting, the improvement in cross super-category setting is more significant, which further demonstrates the generalization ability and robustness of the proposed method. The improvement of the human face is limited (1.36%). A plausible reason is that the human face has much more keypoints (68) than the training categories (the largest keypoint number in training is 38). Such discrepancy renders great difficulties for generalization: the keypoints are densely arranged, and the training data does not contain such a fine-grained matching case.

4.4. Ablation Study

In this section, we first ablate the two-stage framework and the proposed modules in Table 3. Then, we validate some detailed design choices, including the implementation of training loss and support keypoint identifier. We perform experiments on the test set of MP-100 split1 under the 1-shot setting following [40]. We set some variants of the encoder for ablations on network components. (i) **DETR** [5] encoder removes the query-support interactions, *i.e.*, the information fusion among support keypoints and query images is disabled; (ii) **QSR** denotes the proposed query-support joint refine encoder.

Two-stage framework. First we focus on the effectiveness of the second stage. Comparing No. 3 and No. 5 in Table 3, the decoder brings an improvement of 4.13% on PCK, demonstrating the necessity of the second stage.

We compare one-stage and two-stage paradigms qualitatively in Fig. 5. "One-Stage" denotes the direct matching results from No. 3 setting in Table 3, here and after. The direct matching results are prone to error when query and support instances are different, *e.g.*, the dog on the right part of the first row. We also visualize how the decoder corrects the predictions in Fig. 6. For the alpaca face, the inner left canthus (red point) is close to the annotation of the right inner canthus (yellow point) due to the occlusion. Hence, the similarity-aware position proposal for the left inner canthus is wrongly generated at the right inner canthus. The decoder gradually pushes it to the correct position.

Table 3. **Ablation study on different components in CapeFormer.** Support ID stands for the support keypoint identifier in Sec. 3.2. QSR denotes our query-support joint refine encoder.

	Support ID	Encoder	Decoder	Paradigm	PCK
No.1	−−	DETR	−−	one-stage	80.32
No.2	−−	QSR	−−	one-stage	82.86
No.3	✓	QSR	−−	one-stage	85.32
No.4	−−	QSR	✓	two-stage	85.81
No.5	✓	QSR	✓	two-stage	**89.45**

Representation. Here we analyze the effectiveness of query-support joint refine encoder and the support keypoint identifier. i) Comparing No.1 with No.2 in Table 3, the QSR encoder brings an absolute improvement of 2.54% on PCK over DETR encoder, which suggests the effectiveness of support-query mutual information fusion. 2) Comparing No.2 with No.3 in Table 3, introducing the support iden-

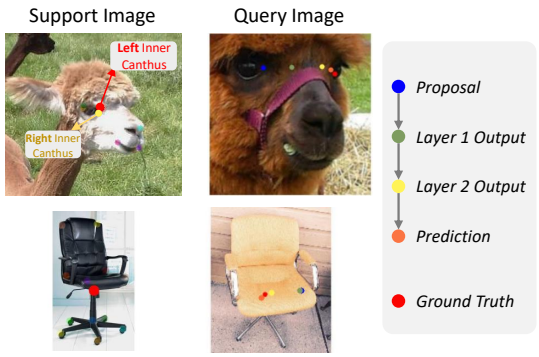

Figure 6. **The decoder can help correct unreliable proposals.** The left column shows the support keypoint (red point) and the right column shows the refining process in the second stage. Note that in the first row, the left inner canthus is invisible due to occlusion, which looks to be on the right.

Table 4. Analysis of different terms in the similarity loss.

$\mathcal{L}_h$	$\mathcal{L}_e$	PCK
✓	−−	88.74
−−	✓	89.41
✓	✓	**89.45**

Table 5. Comparison of different support keypoint identifiers.

Type	PCK
−−	85.81
Learnable	88.96
Fixed	**89.45**

tifier improves the PCK by 2.46%. Note that the No. 3 model excludes the decoder, which is one-stage. Its performance can still achieve state-of-the-art on split1 test set, which suggests that the support identifier and QSR decoder significantly improve representation quality.

To intuitively show how the encoder and decoder work, we visualize the encoder/decoder attention maps in Fig. 7. The attention shows which location the support keypoint will focus during the information fusion (first stage) and the proposal correction (second stage). As can be observed, in the encoder, the support keypoint attends to keypoints with similar appearance, while in the decoder, the attention becomes more focused on the corresponding keypoints for proposal correction.

Loss function. We analyze the similarity loss in Table 4. i) Only using the heatmap supervision (row 1) yields the worst performance, indicating that the expectation of loss is beneficial for precise localization. ii) As shown in row 3, the combination of heatmap loss and expectation attains the best performance, as both the shape of similarity map and the proposal position can be supervised.

Support keypoint identifier. We compare two different implementations of support keypoint identifier: learnable embedding and fixed sinusoidal embedding in Table 5. Using fixed sinusoidal embedding improves PCK by 0.49% compared with learnable embedding. The learnable embedding is similar to the learnable query embedding in DETR-based detection methods [5, 25], which can be viewed as a learnable content embedding or anchors [23]. However, our

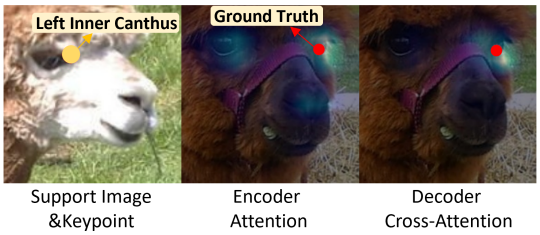

Figure 7. **Visualization of attention map.** We visualize the encoder and decoder attention for the support keypoint at the left inner canthus. The support keypoint coarsely attends to keypoints with similar appearance in the encoder while focusing more on the corresponding keypoints for position update in the decoder.

Table 6. **Inference and Training Efficiency.** We test the efficiency on RTX3090 under 1-shot setting.

Method	Params (M)	GFLOPs	FPS	Memory (GB)
POMNet [40]	48.21	38.01	6.80	13.8 × 2
One-Stage	26.86	22.65	36.90	7.5
CapeFormer	31.14	23.68	26.09	7.8

support keypoint identifier is fixed and serves a different purpose: distinguishing different support keypoints. Learnable support identifier deteriorates the performance as they may overfit the training categories more easily.

4.5. Performance and Efficiency Trade-off

We evaluate the efficiency under 1-shot setting for POMNet and CapeFormer in Table 6. The training memory is evaluated with a batchsize of 16. i) Comparing row 1 and 3, CapeFormer significantly reduces the parameters (−35.4%) and FLOPs (−37.7%) compared with POMNet due to the shared backbone and efficient similarity metric. ii) Comparing rows 2 and 3, the second stage only introduces limited parameters and FLOPs, which shows a satisfying trade-off between performance and efficiency. iii) CapeFormer is efficient for training, which consumes only 8GB memory on a single GPU (almost 1/4 of POMNet).

5. Conclusion

In this work, we formulate a two-stage framework for category-agnostic pose estimation (CAPE) and instantiate it with a novel transformer model. Unlike previous one-stage approaches that entirely rely on matching results, the matching results in the first stage are viewed as similarity-aware position proposals which are refined in the second stage. We also propose specific designs to improve the representation and similarity modeling. On the CAPE benchmark MP-100, our method outperforms the previous best method by large margins on both accuracy and efficiency.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (Grant No. U1913602 and 61876211).

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter V. Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 3686–3693, 2014. 2

[2] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv Computer Research Repository*, abs/1607.06450, 2016. 5

[3] Jinkun Cao, Hongyang Tang, Hao-Shu Fang, Xiaoyong Shen, Cewu Lu, and Yu-Wing Tai. Cross-domain adaptation for animal pose estimation. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 9497–9506, 2019. 2

[4] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):172–186, 2018. 2

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 213–229, 2020. 4, 7, 8

[6] Seokju Cho, Sunghwan Hong, Sangryul Jeon, Yunsung Lee, Kwanghoon Sohn, and Seungryong Kim. Cats: Cost aggregation transformers for visual correspondence. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 9011–9023, 2021. 3

[7] Seokju Cho, Sunghwan Hong, and Seungryong Kim. Cats++: Boosting cost aggregation with convolutions and transformers. *arXiv Computer Research Repository*, abs/2202.06817, 2022. 3

[8] Christopher B. Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Krishna Chandraker. Universal correspondence network. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 2406–2414, 2016. 3

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 248–255, 2009. 6

[10] Qi Fan, Wenjie Pei, Yu-Wing Tai, and Chi-Keung Tang. Self-support few-shot semantic segmentation. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2022. 4

[11] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1126–1135, 2017. 2, 6

[12] Zigang Geng, Ke Sun, Bin Xiao, Zhaoxiang Zhang, and Jingdong Wang. Bottom-up human pose estimation via disentangled keypoint regression. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 14671–14681, 2021. 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 770–778, 2016. 6

[14] Shuqiang Jiang, Sisi Liang, Chengpeng Chen, Yaohui Zhu, and Xiangyang Li. Class agnostic image common object detection. *IEEE Trans. Image Process.*, 28(6):2836–2846, 2019. 2

[15] Seungryong Kim, Dongbo Min, Bumsub Ham, Stephen Lin, and Kwanghoon Sohn. FCSS: fully convolutional self-similarity for dense semantic correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(3):581–595, 2019. 3

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations*, 2014. 6

[17] Rollyn Labuguen, Jumpei Matsumoto, Salvador Blanco Negrete, Hiroshi Nishimaru, Hisao Nishijo, Masahiko Takada, Yasuhiro Go, Ken ichi Inoue, and Tomohiro Shibata. Macaquepose: A novel "in the wild" macaque monkey pose dataset for markerless motion capture. *Frontiers in behavioral neuroscience (2021)*, 2021. 2

[18] Jiefeng Li, Siyuan Bian, Ailing Zeng, Can Wang, Bo Pang, Wentao Liu, and Cewu Lu. Human pose regression with residual log-likelihood estimation. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 11005–11014, 2021. 2

[19] Jiefeng Li, Can Wang, Hao Zhu, Yihuan Mao, Hao-Shu Fang, and Cewu Lu. Crowdpose: Efficient crowded scenes pose estimation and a new benchmark. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 10863–10872, 2018. 2

[20] Shuyuan Li, Jianguo Li, Hanlin Tang, Rui Qian, and Weiyao Lin. Atrw: A benchmark for amur tiger re-identification in the wild. In *Proceedings of ACM International Conference on Multimedia*, pages 2590–2598, 2020. 2

[21] Yanjie Li, Shoukui Zhang, Zhicheng Wang, Sen Yang, Wankou Yang, Shu-Tao Xia, and Erjin Zhou. Tokenpose: Learning keypoint tokens for human pose estimation. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 11293–11302, 2021. 2

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 740–755, 2014. 6

[23] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic anchor boxes are better queries for detr. In *Proceedings of International Conference on Learning Representations*, 2022. 5, 8

[24] Erika Lu, Weidi Xie, and Andrew Zisserman. Class-agnostic counting. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, pages 669–684, 2019. 2

[25] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 3631–3640, 2021. 8

[26] Juhong Min, Jongmin Lee, Jean Ponce, and Minsu Cho. Hyperpixel flow: Semantic correspondence with multi-layer

neural features. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 3394–3403, 2019. 3

[27] Akihiro Nakamura and Tatsuya Harada. Revisiting fine-tuning for few-shot learning. *arXiv Computer Research Repository*, abs/1910.00216, 2019. 6

[28] Xuecheng Nie, Jianfeng Zhang, Shuicheng Yan, and Jiashi Feng. Single-stage multi-person pose machines. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 6950–6959, 2019. 2

[29] N. Dinesh Reddy, Minh Vo, and Srinivasa G. Narasimhan. Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 1906–1915, 2018. 2

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017. 3

[31] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelovic, Akihiko Torii, Tomás Pajdla, and Josef Sivic. Neighbourhood consensus networks. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 1658–1669, 2018. 3

[32] Christos Sagonas, Epameinondas Antonakos, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 faces in-the-wild challenge: database and results. *Image Vis. Comput.*, 47:3–18, 2016. 6

[33] Dahu Shi, Xing Wei, Liangqi Li, Ye Ren, and Wenming Tan. End-to-end multi-person pose estimation with transformers. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 11059–11068, 2022. 2

[34] Min Shi, Hao Lu, Chen Feng, Chengxin Liu, and Zhiguo Cao. Represent, compare, and learn: A similarity-aware framework for class-agnostic counting. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 9529–9538, 2022. 2, 4

[35] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 4077–4087, 2017. 6

[36] Xibin Song, Peng Wang, Dingfu Zhou, Rui Zhu, Chenye Guan, Yuchao Dai, Hao Su, Hongdong Li, and Ruigang Yang. Apollocar3d: A large 3d car instance understanding benchmark for autonomous driving. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 5452–5462, 2019. 2

[37] Xiao Sun, Jiaxiang Shang, Shuang Liang, and Yichen Wei. Compositional human pose regression. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 2621–2630, 2017. 2

[38] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 1653–1660, 2014. 2

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017. 4

[40] Lumin Xu, Sheng Jin, Wang Zeng, Wentao Liu, Chen Qian, Wanli Ouyang, Ping Luo, and Xiaogang Wang. Pose for everything: Towards category-agnostic pose estimation. In *Proceedings of European Conference on Computer Vision (ECCV)*, volume abs/2207.10387, 2022. 1, 2, 4, 6, 7, 8

[41] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 5525–5533, 2016. 6

[42] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2878–2890, 2013. 6

[43] Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen. Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 5217–5226, 2019. 2

[44] Feng Zhang, Xiatian Zhu, Hanbin Dai, Mao Ye, and Ce Zhu. Distribution-aware coordinate representation for human pose estimation. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 7091–7100, 2020. 2

[45] Song-Hai Zhang, Ruilong Li, Xin Dong, Paul L. Rosin, Zixi Cai, Han Xi, Dingcheng Yang, Haozhi Huang, and Shi-Min Hu. Pose2seg: Detection free human instance segmentation. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR)*, pages 889–898, 2019. 2

[46] Jian Zhao, Jianshu Li, Yu Cheng, Li Zhou, Terence Sim, Shuicheng Yan, and Jiashi Feng. Understanding humans in crowded scenes: Deep nested adversarial learning and a new benchmark for multi-human parsing. In *Proceedings of ACM International Conference on Multimedia*, pages 792–800, 2018. 2