Supplementary Material:

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

This supplementary material includes the following five parts.

Part 1 presents more implementation details, including network architecture and the training recipe (in main paper, Sec. 4.1, line 559).
Part 2 presents the comparison with the semantic correspondence approaches.

Part 3 presents more qualitative results (Fig. 5 in main paper).

Part 4 shows more visualizations of the attention maps (Fig. 7 in the main paper).

Part 5 includes the visualizations of the similarity maps and the similarity-aware position proposals.

Part 1: Implementation Details on Network Architecture and Training Recipe

Network architecture. Here, we specify more details on network architecture. For all the encoder and the decoder layers, the embedding dimension is set to 256. The raw feature maps with 2048 channels from the backbone are first compressed to 256 channels with a 1×1 convolution. The number of heads for all the self- and the cross-attention layers are set to eight. The detailed architecture for the feed-forward network and the offset prediction head (in line 483 of the main paper) are illustrated in Fig. A1. Note that the overall design of the encoder and decoder has been illustrated in Fig. 3 and Fig. 4 of the main paper.

For the similarity-aware proposal generator, we use the same configuration as in BMNet [7]. The detailed architecture is illustrated in Fig. A2.

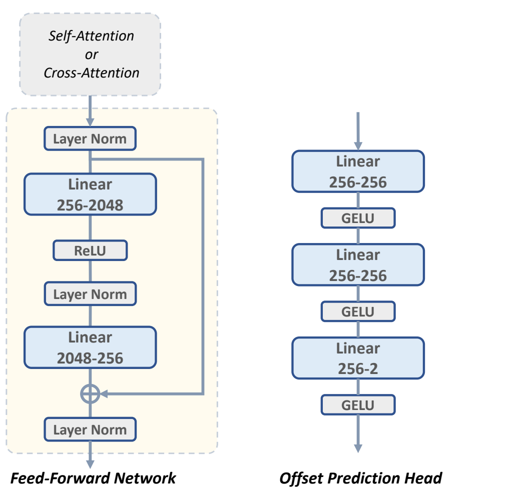

Figure A1. **Architecture of the feed-forward network and the offset prediction head.**

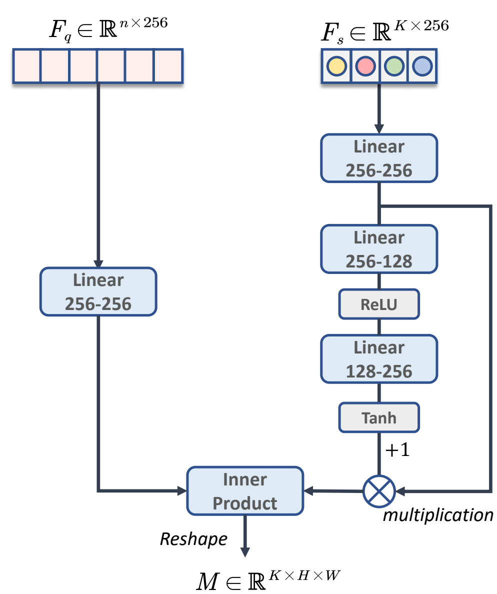

Figure A2. **Architecture of the similarity-aware proposal generator**.

Training recipe. We use `pytorch` [6] and MMPose [4] framework to develop the method. We set the dropout rate of attention layers and feed-forward networks to be 0.1 following DETR [1]. We use a linear warm-up strategy for $1,000$ iterations. The warm-up ratio is set to be 0.001.

To generate the keypoint heatmaps, we set the variance of the Gaussian kernel to 2.0. Note that, for the support keypoints, the heatmaps are used as the soft masks to obtain the support keypoint features (Sec. 3.2 in the main paper). Meanwhile, the heatmaps for keypoints in the query images are used as the ground truth for the heatmap loss (see Eq. (5) in the main paper).

Minor design choice. Here we ablate some minor design choices, including hyper-parameters and parameter sharing. For hyper-parameters, we analyze the impact of different λ_h, *i.e.*, the weight for heatmap loss in Table A1. We test the PCK under 1-shot setting on MP-100 split1. The results show that 2.0 is an optimal choice, and the impact of λ_h is insignificant. The other hyper-parameters are not further fine-tuned.

We also notice that POMNet [9] adopts separated backbones for the support and query images, while the proposed CapeFormer adopts a shared one. The offset prediction head can also be shared across the decoder layers. We compare different parameter-sharing settings in Table A2. Although using separated backbones, as in POMNet, can increase the performance slightly, we adopt a shared backbone considering the efficiency.

Table A1. The impact of different heatmap loss weights (λ_h).

λ_h	1	2	4	8	10
PCK	89.36	**89.45**	89.26	89.03	88.58

Table A2. The impact of parameter sharing for backbone and offset prediction head.

	Backbone	Offset Head	PCK
Parameter Sharing	✓	×	89.45
	×	×	**89.74**
	✓	✓	89.08

Part 2: Comparison with Semantic Correspondence Here we show the comparison between CAPE and the semantic correspondence models [3, 5], and then discuss their differences. As mentioned in the main paper, SC can naturally conduct CAPE. However, we find that directly applying SC models is not an optimal choice for CAPE. As shown in Table A3 (CATs++ [2] is currently one of the best practices on SC), both the pre-trained and the fine-tuned CATs++ (denoted by *) fall behind POMNet [9] and CapeFormer in terms of accuracy and efficiency, but outperform the metric learning baseline ProtoNet [8]. We find the fine-tuned CATs++ hard to converge during training, thus failing to achieve better results.

Table A3. **Quantitative comparison with the semantic correspondence models**. * denotes that the model is fine-tuned on MP-100 dataset.

Method	ProtoNet	POMNet	CapeFormer	CATs++	CATs++*
PCK	46.05	84.23	**89.45**	66.76	59.18
GFLOPs	–	38.01	**23.68**	36.23	36.23

Then we discuss the differences, which are three folds: *task, methodology, and data bias*. For **task**, SC aims to predict a dense correspondence field for every pixel in the source (support) image. The core of this process is to compute a correlation map with the size of $hw \times hw$. In contrast, CAPE only computes similarity maps for input support keypoints. Hence, directly using SC is inefficient.

In terms of **methodology**, SC researchers find coarse correlation maps unreliable and noisy due to similar appearance and repetitive patterns. Hence, similar to our two-stage design, post-processing techniques are designed for correlation results. Unfortunately, most of these techniques in SC models can not be adapted to CAPE as they heavily rely on the full-size $hw \times hw$ correlation map, such as the commonly-used geometry consistency constraints.

Another difference can be the **bias on the training data**, most training data for SC models is rigid objects, *e.g.*, aeroplanes, faces, or bicycles. For CAPE, one needs to consider non-rigid deformation and occlusions.

Part 3: Qualitative Results

More qualitative results are shown in Fig. A3 and Fig. A4. Note that here we also add the qualitative comparison with the previous best method POMNet [9].

Figure A3. More qualitative results #1. One-stage removes the decoder in the proposed CapeFormer.

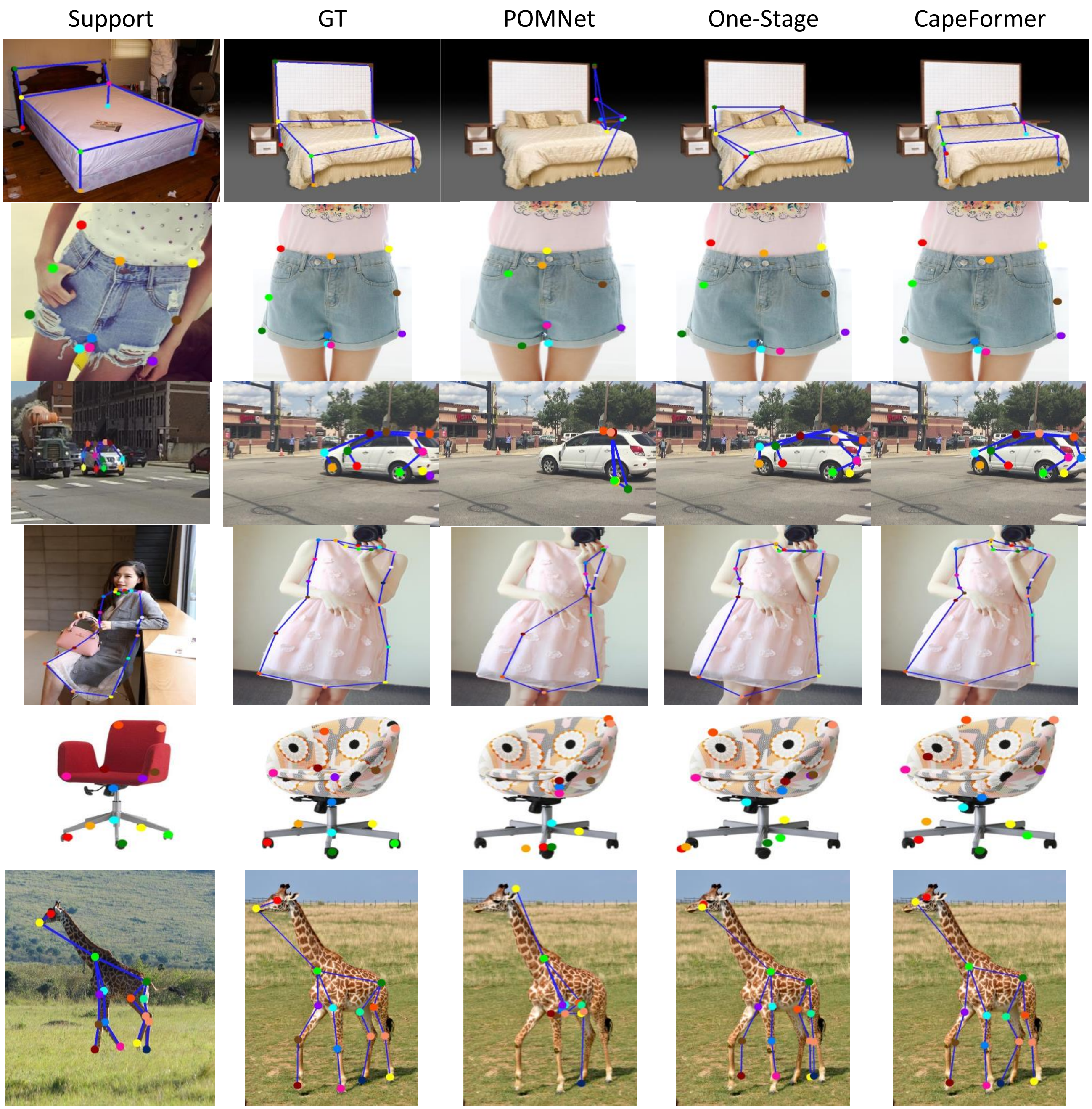

Figure A4. More qualitative results #2. One-stage removes the decoder in the proposed CapeFormer.

Part 4: Attention Visualization

More visualizations of the encoder attention maps. We visualize more attention maps for the query-support joint refine encoder in Fig. A5. We only visualize the support-to-query attention of the last encoder layer. The encoder attention indicates which location in the query image the support keypoints will attend to during information fusion.

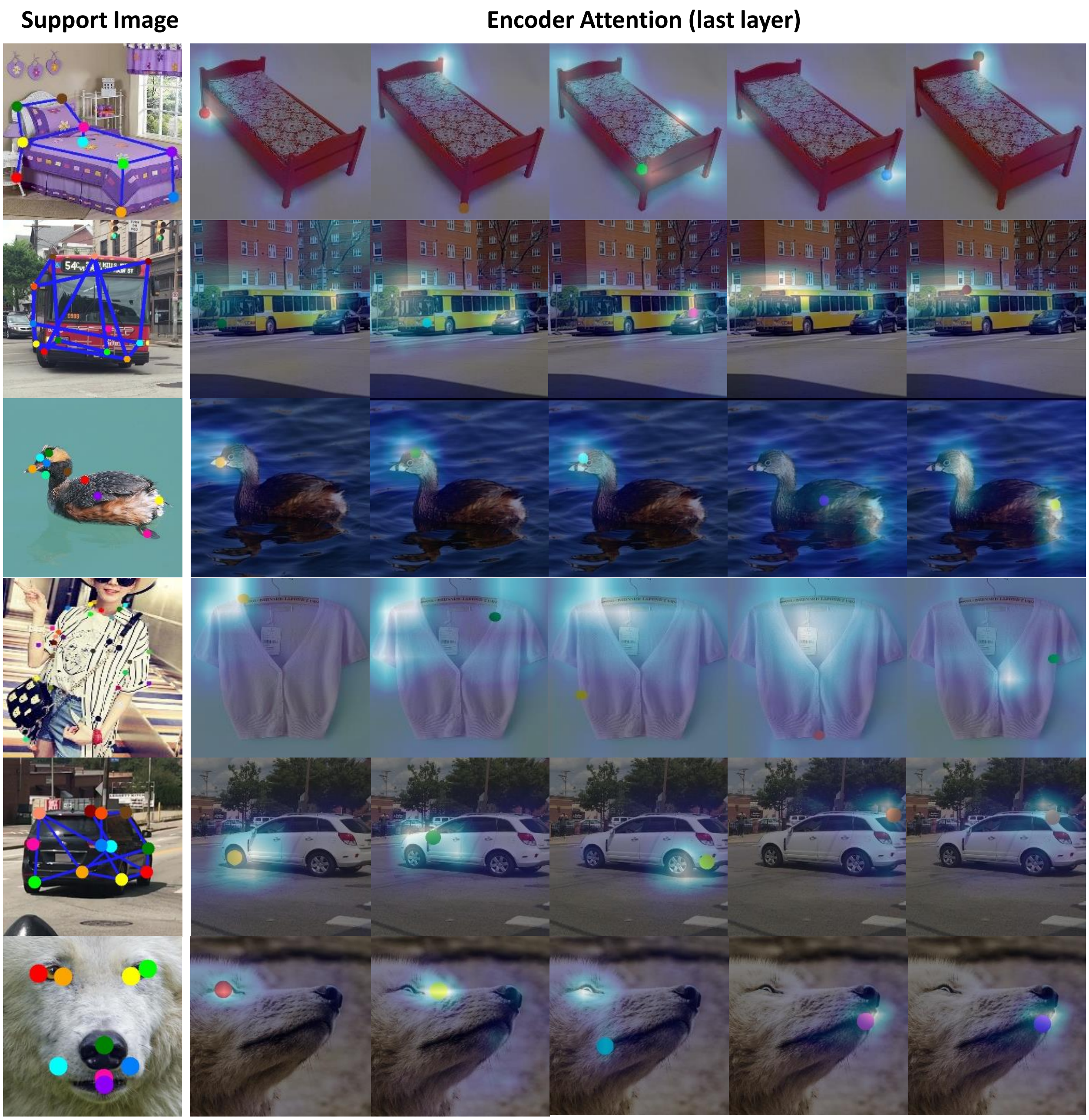

Figure A5. **Visualizations of the encoder attention map.** The left column shows the support image with support keypoints. The other columns represent the attention maps for different support keypoints. The support keypoint for each attention map is marked in the attention maps, whose color is identical to the corresponding keypoints in the support images.

More visualizations for the decoder attention maps. More visualizations of the cross-attention in the last proposal refine decoder layer are shown in Fig. A6. The attention shows which features will be extracted for each support keypoint during position update.

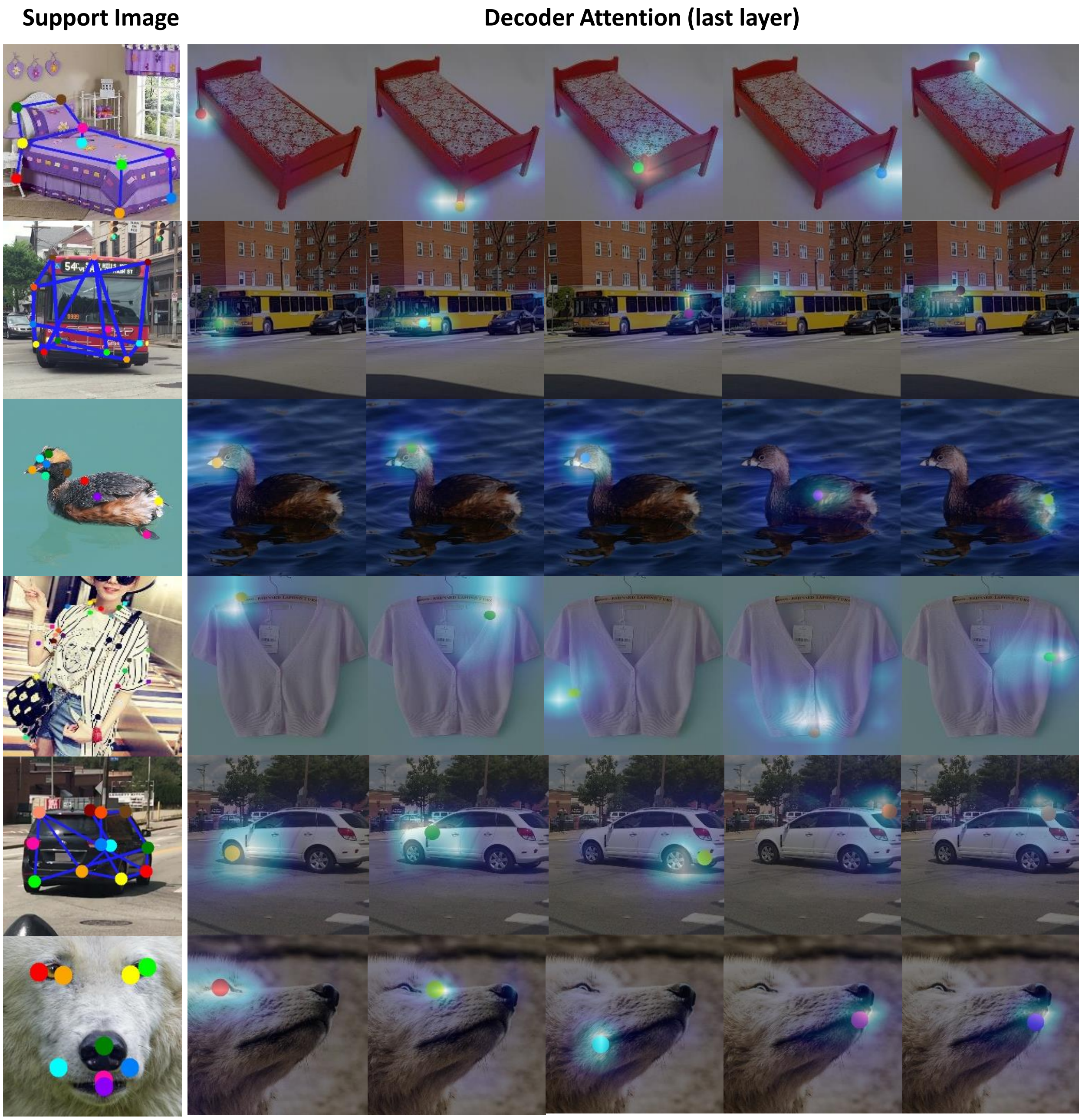

Figure A6. **Visualizations of the decoder attention map.** The left column shows the support image with support keypoints. The other columns represent the attention maps for different support keypoints. The support keypoint for each attention map is marked in the attention visualizations, whose color is identical to the corresponding keypoints in the support images.

Part 5: Visualizations of Similarity Map and Similarity-Aware Proposals

We visualize the similarity maps and similarity-aware proposals in Fig. A7.

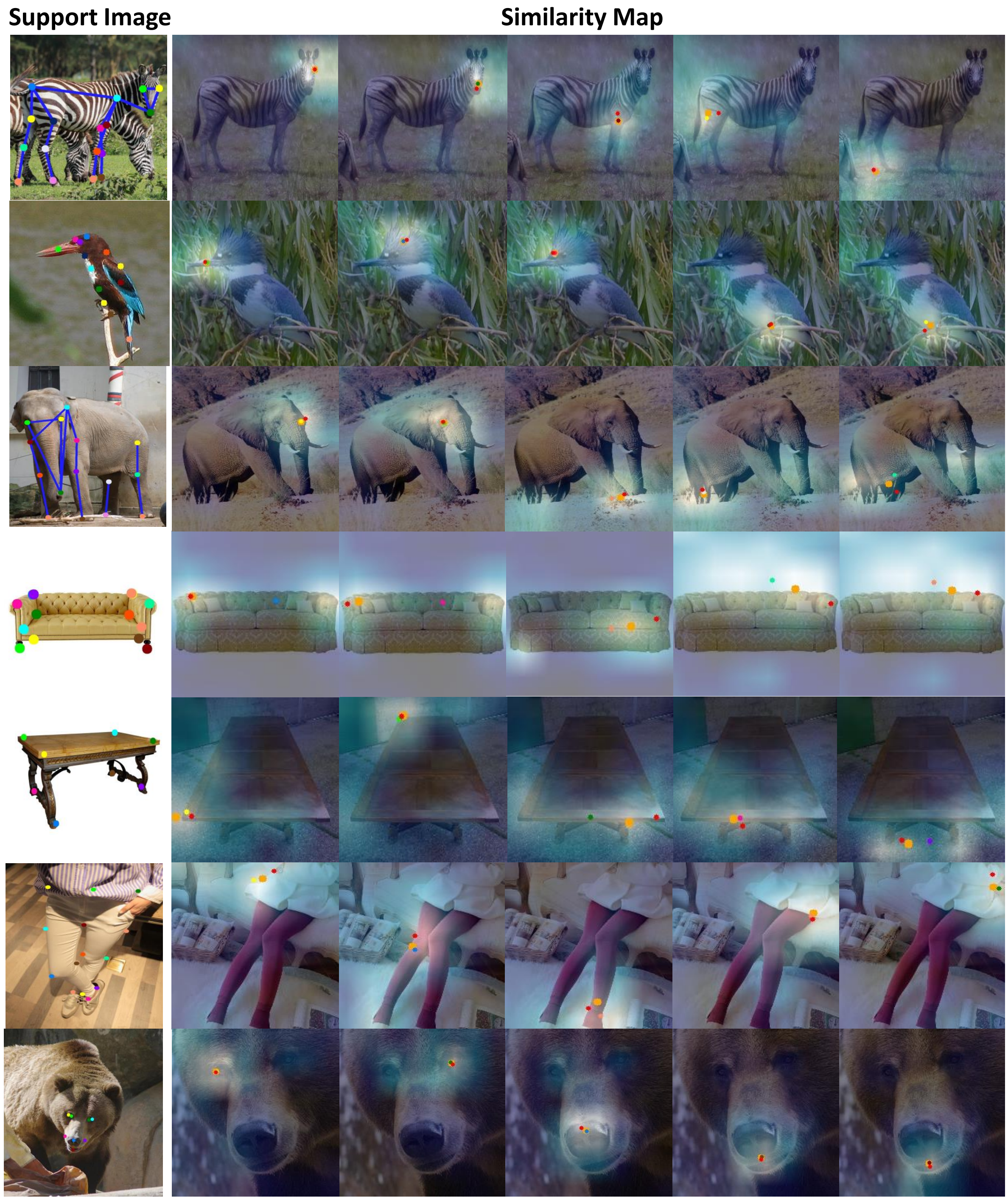

Figure A7. Visualizations of similarity maps and similarity-aware proposals. The left column shows the support images with support keypoints. The other columns visualize the similarity maps for different support keypoints. The keypoints are marked on the similarity maps, whose colors are identical to the corresponding keypoints in the support images. We mark the ground truth annotations with red points and the final predictions with orange points.

References

[1] 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. 2

[2] Seokju Cho, Sunghwan Hong, and Seungryong Kim. Cats++: Boosting cost aggregation with convolutions and transformers. *arXiv Computer Research Repository*, abs/2202.06817, 2022. 4

[3] 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. 4

[4] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. `https://github.com/open-mmlab/mmpose`, 2020. 2

[5] 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. 4

[6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019. 2

[7] 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

[8] 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. 4

[9] 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. 3, 4, 5