

Table 8: Similarity analysis for the base classes and novel classes on various splits

	Split1	Split2	Split3	Split4	Split5	Body	Face	Vehicle	Furniture
Cosine Similarity	0.65	0.64	0.65	0.65	0.63	0.62	0.61	0.54	0.59

A Appendix / Supplemental Material

In this Supplementary Material, we provide more contents to complement our main text.

A.1 More Implementation Details

Our method is implemented upon the codebase of CapeFormer [59], due to the relation with multi-class pose estimation. As in [59], we crop and resize the image patch to 256×256 for training and test images. Data augmentations to enrich the dataset in the training stage, *e.g.*, random scaling and random rotation. We employ Adam [25] optimizer to train our model for 60 epochs with batch size 1. The learning rate is set as $1e^{-5}$ with decay 0.1 at the 40-th and 50-th epoch. For experiment environment, we employ 4 NVIDIA 3090ti GPUs on Ubuntu 20.04 system with 64 GB Intel 9700K CPU. We adopt the random seed 10 in all experiments unless stated otherwise.

Note that we random shuffle the keypoint order of each class, because the keypoint order matters weak-shot pose estimation. To clearly show the influence, we report the mAPs \uparrow obtained using related and shuffled orders on S1 using 3 labeled images per base class in Tab. 7. We could find that the Sim.Base. [68] and Hedlin *et al.* [18] achieve higher performances on related keypoint order, but dramatically degrade on shuffled keypoint order. Our method are relatively robust due to matching-based supervision, where the keypoint order matters negligibly (probably better initial matching only). Therefore, our model is able to adaptively learn transferrable keypoints, without overfitting the related knowledge in annotations.

Table 7: The impact of using related and shuffled keypoint orders.

Method	Related	Shuffled
Sim.Base. [68]	19.9	9.2
Hedlin <i>et al.</i> [18]	59.5	18.6
MetaPoint [9]	27.5	27.1
Ours	60.6	60.2
Oracle*	74.6	74.4

As for the number of base/novel classes, and the number of labeled/unlabeled training samples. For the standard splits (*i.e.*, MP-100 Split1-Split5), we follow [69, 59] and employ 70/20 for base/novel classes. For the cross super-class splits, we follow [69, 59] and employ 99/1, 98/2, 95/5 and 97/3 for base/novel classes in Body, Face, Fur. and Veh. splits. For the Novel Bird split, we employ 92/8 for base/novel classes. The numbers of samples and labels for base/novel classes are determined by the learning paradigm. Specifically, all methods in weak-shot learning employ 3/30 labeled/unlabeled training samples for each base/novel class. For the few-shot methods (with 1-shot), we employ 3/1 labeled/unlabeled training samples for each base/novel class. For the unsupervised methods, we employ 30 unlabeled training samples for each novel class.

A.2 Split Similarity Analysis

We measure the similarity via the GT keypoints and a ResNet-50 backbone pre-trained on ImageNet. Specifically, we firstly extract an image feature vector for each image, by averaging the feature vectors of GT keypoints on the backbone feature map. Then, we extract a class feature vector, by averaging the image feature vectors belonging to the same class. After that, for each novel class, we compute its similarity to base classes by averaging its similarities to all base classes using class feature vectors. Finally, we obtain the similarity from novel classes to base classes by averaging the similarities of all novel classes.

As shown in Table 8, we summarize the similarities for all dataset splits. We could see that, the similarities of standard splits (*i.e.*, Split1-Split5) are robust and intermediate due to random splitting. The similarities of cross super-class splits are significantly lower than the standard splits.

Table 9: Results(mAP \uparrow) using different numbers of labeled images for each base class.

Method	#1	#2	#3	#5	#10
Hedlin <i>et al.</i> [18]	15.4	16.8	18.6	25.9	30.0
MetaPoint [9]	21.3	26.0	27.1	32.9	40.3
Ours	43.8	56.8	60.2	61.3	65.8
Oracle*	55.3	70.6	74.4	76.4	81.5

Table 10: Results(mAP \uparrow) using different numbers of unlabeled images for each novel class.

Method	#0	#15	#30	#45	#60
Hedlin <i>et al.</i> [18]	18.6	24.7	27.0	29.0	29.5
MetaPoint [9]	27.1	28.7	31.5	32.7	33.6
Ours	60.2	66.1	70.4	72.2	72.7
Oracle*	74.4	82.2	86.1	87.5	88.6

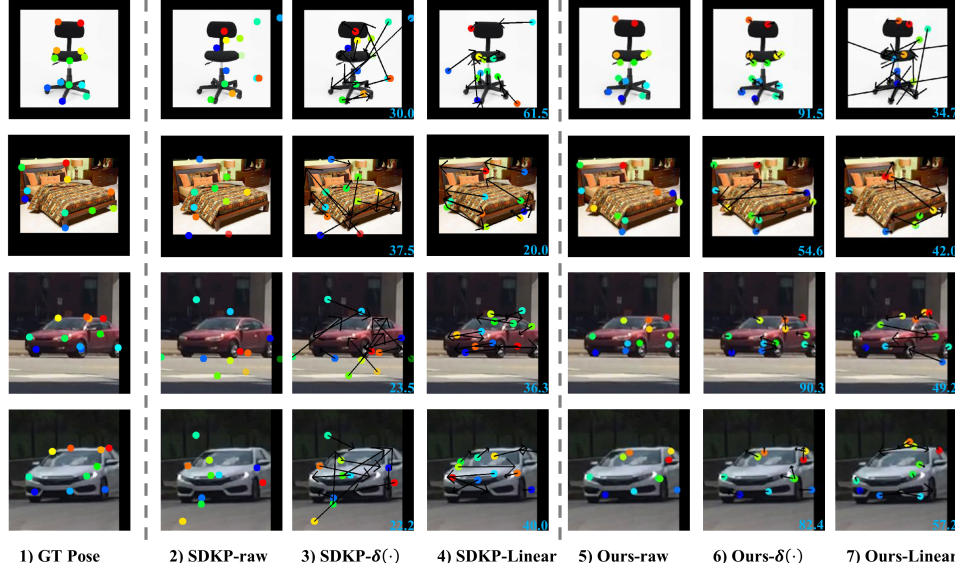


Figure 6: We visualize the results of SDKP [18] and ours on the challenging cross super-class splits, *i.e.*, Furniture split (the top two rows) and Vehicle split (the bottom two rows). The first column shows the GT keypoints on test images. For each method, we respectively show the estimated raw keypoints, estimated keypoints matched to GT, and estimated keypoints after linear regression. In image patches for evaluations, the numbers colored by blue at the right-bottom corners show the F1-score measured by GT. We could see that our keypoints better cover the objects and outperform the baseline SDKP [18] for extremely out-of-distribution classes.

A.3 Analysis of Training Image Number

Because our model is built upon pre-trained diffusion model, we naturally wonder the performances of our model in a small scale of training data. Firstly, in Tab. 9, we use different numbers of labeled images per base class without any image from novel classes. Surprisingly, given only 1 labeled image per base class, our model is able to achieve satisfying mAPs \uparrow for novel classes and reach about 80% of the upper-bound. Note that the baseline Hedlin *et al.* [18] also leverages pre-trained diffusion model, but cannot learn transferrable knowledge for multiple novel classes. Although MetaPoint [9] can learn transferrable knowledge, it requires abundant training images. By contrast, our method leverages diffusion model to learn keypoint prompts and correspondences from base classes, and transfer them to support the unsupervised learning of multiple novel classes.

Secondly, in Tab. 10, we use different numbers of unlabeled images per novel class with 3 images per base class. The performances of our model are dramatically improved if more unlabeled images are available. But our results seem gradually saturated, may due to the aggravated imbalance between the images number of base classes and novel classes, and we remain this issue for future works.

A.4 More Visualizations in Cross-Super Class Transfer

In this section, we provide more visualizations using cross-super class transfer splits as in [69, 59, 9], *i.e.*, the base classes and novel classes come from different super-class. As shown in Fig. 6, the estimated keypoints of baseline SDKP [18] and our method all degrade in such extremely challenging

Table 11: Results on dataset splits for cross super-class transfer.

Method	Metric	Body	Face	Fur.	Veh.
Hedlin <i>et al.</i>	mAP \uparrow	18.5	16.1	8.8	7.8
	L2 \downarrow	14.6	26.9	68.9	54.9
MetaPoint	mAP \uparrow	47.9	22.1	35.6	16.6
	L2 \downarrow	13.1	22.4	43.7	34.7
Ours	mAP \uparrow	73.8	37.0	50.9	33.5
	L2 \downarrow	9.2	16.4	26.2	23.7
Oracle*	mAP \uparrow	78.4	53.7	77.3	43.2
	L2 \downarrow	8.9	15.1	20.9	19.5

Table 12: Results on dataset splits using fewer base classes.

Method	Metric	S5	S6	S7	S8
Hedlin <i>et al.</i>	mAP \uparrow	24.1	18.7	15.5	12.6
	L2 \downarrow	47.1	58.8	60.0	66.5
MetaPoint	mAP \uparrow	36.9	36.1	33.8	31.8
	L2 \downarrow	39.6	38.7	41.2	43.5
Ours	mAP \uparrow	64.5	56.1	53.4	48.9
	L2 \downarrow	22.3	27.9	28.9	31.6
Oracle*	mAP \uparrow	76.0	73.3	73.2	73.7
	L2 \downarrow	20.3	23.1	22.9	22.3

Table 13: The efficiency comparison against previous works.

Method	All Param(M)	Trainable Param(M)	FPS(task/s)
Hedlin <i>et al.</i> [18]	1066.28	0.05	10.4
MetaPoint [9]	31.30	31.30	25.0
Ours	1073.30	7.07	10.3

cases. However, our method still better fits the object structures against the baseline, *i.e.*, our keypoints roughly cover the cars in the bottom two rows. Generally, such challenges inevitably exist in class-wise transfer as in related works [69, 59, 9], and we will keep in mind and improve in our future works.

Besides, we also visualize the keypoints in matching-based evaluation (*i.e.*, col-3 and col-6) and regression-based evaluation (*i.e.*, col-4 and col-7). We can see that some keypoints mapped by the regressor are worse, *e.g.*, for both methods in the second row. The reason may be that the trained regressor may not fit all estimations satisfyingly. Therefore, our matching-based evaluation inspired by [13] are complementary and irreplaceable.

A.5 Generalization Analysis

Using out-of-distribution novel classes. The distribution gap between base classes and novel classes matters the our class-wise transfer learning. As in related works, [69], we explore this by using one of four super-classes (*i.e.*, human body, human face, furniture and vehicle) as novel classes respectively while using other classes as base classes. As shown in Tab. 11, our method outperforms baselines by large margins (*e.g.*, +25.9% for Body against MetaPoint) in such challenging splits, indicating that our model is more generalizable against distribution shift.

Using different numbers of base classes. We of course prefer the method that could learn from fewer base classes and generalize to more novel classes. Therefore, we construct S6, S7, and S8 based on S5 by gradually moving 10 random classes from base classes to novel classes, and the ratio between base classes and novel classes in S8 is 40:50. As shown in Tab. 12, all mAPs degrade due to fewer base classes, but our model achieves the best results, indicating the robustness of our model.

A.6 Efficiency Analysis

In this section, we analyse the efficiency of our model against strong baselines, including all parameters, trainable parameters, and FPS. We use the official prints “task/s” provided by MMPose[12] as the results of FPS. As shown in Tab. 13, our method is not as efficient as MetaPoint [9], but our method achieves significantly better performances. Compared with Hedlin *et al.* [18], our method uses more parameters in estimating keyness and extracting correspondences, which are only employed in the training stage, and thus the FPS of our method is comparable against Hedlin *et al.* [18] in the test stage.