Test-time Adaptation vs. Training-time Generalization: A Case Study in Human Instance Segmentation using Keypoints Estimation

Kambiz Azarian Debasmit Das Hyojin Park Fatih Porikli
Qualcomm AI Research*
{kambiza, debadas, hyojinp, fporikli}@qti.qualcomm.com

Abstract

We consider the problem of improving the human instance segmentation mask quality for a given test image using keypoints estimation. We compare two alternative approaches. The first approach is a test-time adaptation (TTA) method, where we allow test-time modification of the segmentation network's weights using a single unlabeled test image. In this approach, we do not assume test-time access to the labeled source dataset. More specifically, our TTA method consists of using the keypoint estimates as pseudo labels and backpropagating them to adjust the backbone weights. The second approach is a training-time generalization (TTG) method, where we permit offline access to the labeled source dataset but not the test-time modification of weights. Furthermore, we do not assume the availability of any images from or knowledge about the target domain. Our TTG method consists of augmenting the backbone features with those generated by the keypoints head and feeding the aggregate vector to the mask head. Through a comprehensive set of ablations, we evaluate both approaches and identify several factors limiting the TTA gains. In particular, we show that in the absence of a significant domain shift, TTA may hurt and TTG show only a small gain in performance, whereas for a large domain shift, TTA gains are smaller and dependent on the heuristics used, while TTG gains are larger and robust to architectural choices.

1. Introduction

Human instance segmentation is an important task that requires finding the image regions of different individuals in a scene. This task has multiple applications ranging from video conferencing [15, 7], matting [34, 41, 17], generation [27, 23], autonomous driving [37], robotics [31], extended reality [38], pedestrian tracking [25, 24, 40], etc. For example, in extended reality applications, human instance segmentation can be used to detect human contours. Along with 3D information, the human contours can be used to render virtual objects at arbitrary locations, which can then enhance augmented reality experiences. For human-robot interaction, segmentation of individuals is required for understanding spatial relationships that allows the robot to plan movements and perform complex tasks. Since human instance segmentation has multiple use cases, there has been considerable effort in recent years to improve the performance of such models.

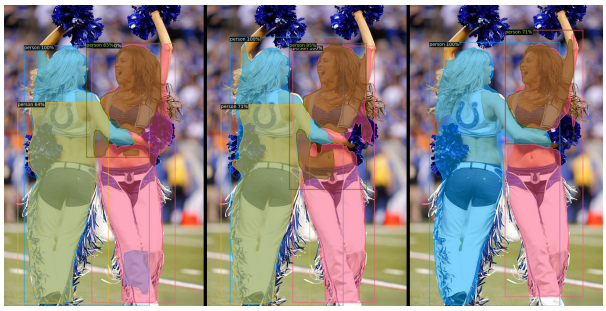

Figure 1. From left to right: baseline, test-time adapted and training-time generalized Mask-RCNN.

Most methods for human instance segmentation are built on top of general instance segmentation frameworks. Such methods can be broadly divided into single-stage and multi-stage methods. Single-stage methods [8, 32, 3] normally use parallel branches for detection and segmentation. The detection branch is used to localize each individual instance, while the segmentation branch learns to annotate each pixel based on the feature information densely. The output of the detection branch and the segmentation branch are employed to obtain masks for all the instances in the scene. Since detection and segmentation operate independently of each other, single-stage methods are considerably faster than multi-stage counterparts. However, the segmentation step does not utilize the localization information; thus, single-stage methods perform poorer than multi-stage ones.

Two-stage methods [10, 22, 12] generally follow a se-

*Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

quential approach of firstly detection and then segmentation. Multi-stage methods [4, 6] go a step further and repeat this two-step method multiple times. The detection stage normally crops out a region of interest that localizes a particular object. Subsequently, segmentation is applied for different regions of interest to produce masks for different instances in the scene. Since segmentation and detection depend on each other during training or inference, segmentation quality is much better compared to single-stage methods. Hence, in this paper, we opt for the popular two-stage framework Mask-RCNN [12], which also carries a pose estimation head to aid human instance segmentation.

Compared to general instance segmentation, human instance segmentation is relatively more challenging for a couple of reasons. Firstly, there is large intra-class variation within the human category due to various outfits, poses, and deformability of the human body. Secondly, the number of humans and their locations in a scene can be random, which increases the complexity of model inference. Finally, humans tend to interact with each other and other objects causing occlusions and obstructions, resulting in unnatural 2D shapes and poses. To tackle such challenges, pose estimation [5, 9, 16] has been used to enhance human instance segmentation.

There have been different approaches to using pose estimation for human segmentation. The most popular method [35] follows a sequential approach, where humans are firstly detected, and their corresponding poses are estimated. The sparse keypoints are then grouped, and the generated heatmaps are used to segment the humans. Although the proposed sequential approach is seminal, it has the weakness of propagating the pose estimation errors to the segmentation stage, which can negatively affect the performance, especially for occluded humans. Alternatively, there are methods [26, 12] that carry out a joint estimation of pose and instance segmentation, which do not suffer from propagation issues. We build on the latter framework, i.e., Mask-RCNN [12] in this paper. Specifically, we consider how human pose estimation can be used to enhance the performance of a human instance-segmentation model (trained on a source dataset) on test images from a target dataset. Our evaluations are done on the OCHuman [35] and COCOPersons [19, 35] datasets from which we draw conclusions about the feasibility and extent of such improvement using generalization, adaptation, or both.

Our main contributions are summarized as follows:

- We propose a *test-time* method for adapting a human segmentation network to a single unlabelled test image. It involves backpropagating keypoint pseudo-labels to adjust the backbone weights. We devise three keypoints head variants in addition to the Mask-RCNN's keypoints head. Two of the variants are transformer based, and all augment the pose estimates to include keypoint visibility/occlusion indicators.

- We propose a *training-time* method for enhancing the performance of a human segmentation network on new domains. It involves augmenting the backbone features with those generated by the keypoints head and using the aggregate feature vector for segmentation. We show our method achieves competitive performance despite its simplicity.

- We evaluate the performance of both methods and, through ablations, identify factors that limit test-time adaptation (TTA) gains. We show that for a small domain shift, TTA may hurt performance, and training-time generalization (TTG) delivers only a small gain. For a large domain shift, TTA gains are small and dependent on the heuristics used, while TTG gains are more prominent and relatively insensitive to architectural details.

2. Related Work

Non-Adaptive Instance Segmentation: Non-adaptive Instance segmentation methods generally have enhanced architectures either through single-stage approaches [8, 32, 3] or multi-stage approaches [10, 22, 12]. Single-stage methods have a distributed approach where they produce feature maps for the whole image and then extract the feature maps for each instance to produce the corresponding masks. For example, InstanceFCN [8] generates instance-specific scoring maps and outputs instance masks using an assembly module that contains operations like repooling and mask-voting. In similar spirit, CondInst [29] dynamically generated convolutions conditioned on each instance to produce instance-specific segmentation masks. On the other hand, YOLACT [3] is an efficient method that has parallel branches for generating fixed number of prototype masks and mask coefficients. Multi-stage methods generally have a two step procedure where firstly objects are detected using bounding boxes. Then, features are extracted from the region of interest to produce the desired masks. Mask R-CNN [12] is a popular two-stage instance segmentation framework that has an additional head for predicting segmentation masks for each region extracted from the bounding box. In this paper, we focus specifically on the Mask R-CNN architecture. QueryInst [10] is also a recent query-based multi-stage framework that uses sequences of dynamic convolution and multi-head self-attention blocks to produce more refined instance segmentation masks. More comprehensive review of non-adaptive instance segmentation methods can be found in [11].

Adaptive Instance Segmentation: There have been very few works on adaptive instance segmentation. Most of these works target segmentation of biomedical entities like nuclei,

cells etc. For example, in [20], the authors propose a multi-step procedure to tackle domain shift that includes inpainting of images, producing domain-invariant features and a task reweighting scheme to remove source bias. In [21], the authors extended the framework by adding a mechanism of feature similarity maximization. Li et al. [18] also proposed domain adaptation for nuclei segmentation by category-specific feature alignment and self-training using pseudo-labels. In addition to augmented pseudo-labelling, Hsu et al. [13] proposed to use a domain separation module as well as a self-supervised consistency loss. Recently, in [18], the authors propose domain adaptation for mitotic cells by aligning pixel-level feature distributions and also additional supervision through a semantic head. For adaptive human instance segmentation [28] is the only work that is known to us. In this work, the authors propose to jointly estimate pose and instance masks of clinicians by adapting from a source dataset. Adaptation is carried out by a feature normalization strategy and self-training procedure where pseudo-labels are refined using geometric consistency of augmentations.

Human Instance Segmentation: Human instance segmentation involves segmenting humans with the aid of additional information like pose. Earlier works that used pose to segment out human instances include Pose2Seg [35], PersonLab [26] and Pose2Instance [30]. Pose2Seg is a two-stage framework where keypoint heatmaps are initially generated, which are then transformed and aggregated to pass through a segmentation decoder and produce human instance masks. Similarly, PersonLab detected keypoints but then grouped into masks using a geometric embedding descriptor. In Pose2Instance, the authors used the distance transform of keypoints as priors for human masks. More recent works on human instance segmentation include LSNet [36] and PosePlusSeg [1]. In LSNet, the authors use pose attention module and keypoint sensitive combination to aggregate information from multiple sampling points. In the PosePlusSeg framework, the authors proposed a refinement network for improved quality of poses and instances obtained from separate heads. In this paper, we use a more simpler approach where we have multiple heads for pose estimation and instance segmentation but use pose estimation task to select relevant features for adaptation and generalization. Additional works on human instance segmentation include ideas such as self-supervised consistency of human structures across videos [14], iterative refinement using pose with shape prior and part attention [39], deformable convolutions with geometric transformation between keypoint offsets [2].

3. Method

We first describe the task of human segmentation using keypoints, and then we detail our TTA and TTG approaches.

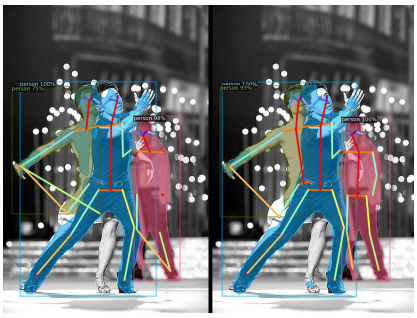

Figure 2. High quality keypoints estimates can improve human segmentation masks (left to right: baseline and TTA Mask-RCNN, where the man's mask spill-over has significantly been reduced).

Finally, we describe the heuristics and keypoints head variants we devised to enhance TTA and TTG gains.

3.1. Human Segmentation using Keypoints

Segmentation networks may perform poorly especially if the test image exhibits a large domain shift with respect to the training dataset, e.g., indoor vs. outdoor or daylight vs. nighttime. For *human* instance segmentation, such a domain shift may be due to severe occlusion of human subjects. Figure 2 depicts a case where the baseline network's mask for the man significantly spills over to that of the woman. Interestingly, despite the network's poor segmentation performance, it estimates the human keypoints fairly well. This observation motivated us to devise methods for enhancing human instance segmentation using keypoints estimates. We consider two such methods, i.e., TTA and TTG (c.f., Figure 3). Since we need pose estimates, we consider (the rather common) architecture (e.g., Mask-RCNN) where the network, in addition to its instance-segmentation head, has a separate keypoints head, with a common backbone.

3.2. Test-time Adaptation

Figure 3 (top) depicts our test-time-adaptation method. In this approach we assume availability of a fully trained human instance segmentation/keypoints estimation network, i.e., $\mathbf{m} = \{\mathbf{m}_b, \mathbf{m}_m, \mathbf{m}_k\}$, and a single unlabelled test image. We also allow for *test-time* adaptation of network weights using backpropagation, but not access to the source dataset. These assumptions make our TTA setup very realistic, but simultaneously very challenging.

We give the steps of our TTA method in Algorithm 1. It includes multiple rounds (e.g., 3) of weight adaptation, each consisting of converting the keypoints estimates to pseudo-labels, plugging them in the keypoints loss together with the keypoint estimates, and backpropagating the resulting self-supervised loss to adjust the backbone weights. While our method works for any pseudo-label conversion method,

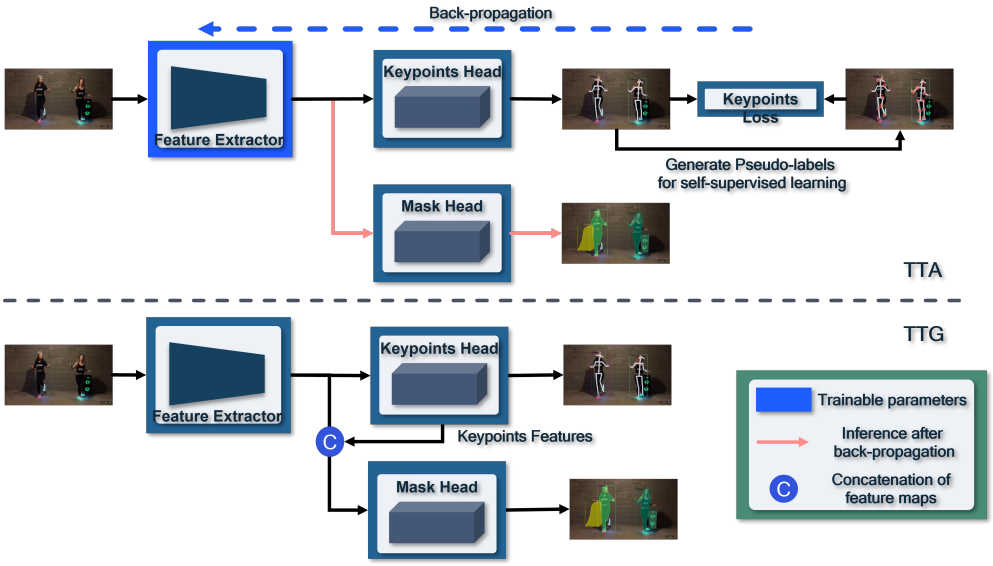

Figure 3. Test-time adaptation (top) and training-time generalization (bottom) frameworks for enhancing human segmentation using pose estimates.

$\mathbf{f}$, and keypoint loss, $\mathcal{L}_{key}$, we use simple candidates, i.e., for person bounding boxes scoring above 0.5, we take keypoints with a minimum probability of 0.05, and declare the location in their 56×56 heat-maps with the highest value as pseudo-labels. We use the multi-category cross-entropy as our keypoint loss. Finally, adapted segmentation masks are generated by running the mask head on the adapted feature map.

Algorithm 1: TTA (test-time adaptation)

Given: Model $\mathbf{m} = \{\mathbf{m}_b, \mathbf{m}_m, \mathbf{m}_k\}$, test-image x^{tgt}
Step 1 (test-time): Adapt backbone to test image
  Initialize n, $i \leftarrow 0$, $\mathbf{m}_b^0 \leftarrow \mathbf{m}_b$
  For $i < n$
    Estimate keypoints, i.e., $y_{key}^i = \mathbf{m}_k\big(\mathbf{m}_b^i(x^{tgt})\big)$
    Generate pseudo-labels, i.e., $\tilde{y}_{key}^i = \mathbf{f}\big(y_{key}^i\big)$
    Update backbone, i.e., compute $\mathbf{m}_b^{i+1}$ by back-propagating self-supervized $\mathcal{L}_{key}(y_{key}^i, \tilde{y}_{key}^i)$
  Save adapted model, i.e, $\mathbf{m}_b^{\text{TTA}} \leftarrow \mathbf{m}_b^n$
Step 2 (test-time): Get TTA masks for test image
  Infer TTA masks, i.e., $y_{mask}^{\text{TTA}} = \mathbf{m}_m\big(\mathbf{m}_b^{\text{TTA}}(x^{tgt})\big)$

3.3. Training-time Generalization

Figure 3 (bottom) shows our training-time generalization method. We assume availability of a human segmentation and pose estimation network, i.e., $\mathbf{m} = \{\mathbf{m}_b, \mathbf{m}_m, \mathbf{m}_k\}$ (which does not need to be trained), and allow offline access to the labelled source dataset. We do not, however, assume the availability of any test images or knowledge about the target domain. We do not allow test-time adaptation of network weights either. These assumptions make our TTG setup both realistic and challenging.

We give the steps of our TTG method in Algorithm 2. Training-time generalization consists of splitting the keypoints head into two subnets, i.e., $\mathbf{m}_k = \mathbf{m}_k^{\text{reg}} \circ \mathbf{m}_k^{\text{fe}}$, for feature extraction and regression, respectively (the split details are given in the sequel). The mask head also needs to be modified to accommodate the extra keypoints features. The TTG model $\mathbf{m}^{\text{TTG}} = \{\mathbf{m}_b, \mathbf{m}_m^{\text{TTG}}, \mathbf{m}_k\}$ is then trained on the labeled source dataset using the original segmentation and pose losses. At test-time, the segmentation masks are generated by running the TTG mask head on the aggregated feature map.

Algorithm 2: TTG (training-time generalization)

Given: Model $\mathbf{m} = \{\mathbf{m}_b, \mathbf{m}_m, \mathbf{m}_k\}$, source-dataset $\mathcal{X}^{src} = \{(x_i^{src}, t_i^{src})\}$
Step 1 (training-time): Train generalized model on source dataset
  Split keypoints head into feature-extractor & regressor subnets, i.e., $\mathbf{m}_k = \mathbf{m}_k^{\text{reg}} \circ \mathbf{m}_k^{\text{fe}}$
  Modify mask head, i.e., $\mathbf{m}_m^{\text{TTG}}$ to accommodate the extra keypoints features, i.e., $\mathbf{m}_k^{\text{fe}} \circ \mathbf{m}_b(x^{tgt})$
  Train TTG model, i.e., $\mathbf{m}^{\text{TTG}} = \{\mathbf{m}_b, \mathbf{m}_m^{\text{TTG}}, \mathbf{m}_k\}$ on $\mathcal{X}^{src} = \{(x_i^{src}, t_i^{src})\}$
Step 2 (test-time): Get TTG masks for test image x^{tgt}
  Infer TTG masks using aggregated features, i.e., $y_{mask}^{\text{TTG}} = \mathbf{m}_m^{\text{TTG}}\big(\mathbf{m}_b(x^{tgt}), \mathbf{m}_k^{\text{fe}} \circ \mathbf{m}_b(x^{tgt})\big)$

3.4. Heuristics and Keypoints Head Variants

The performance of the TTA and TTG methods greatly depends on the quality of the pseudo-labels and features

generated by the keypoints head, respectively. This motivates us to devise three keypoints head variants besides the Mask-RCNN's original. In the following, we discuss each variant and the heuristics behind it. Note that the input feature map from the backbone to all these heads is of dimension $N \times 256 \times 14 \times 14$, where N is the number of person bounding boxes.

- **Mask-RCNN:** $\mathbf{m}_k^{\text{fe}}$ consists of 8 2d-convolutional layers, each with 512 output channels, a 3×3 kernel size and a stride of 1. Hence, there are 512 extra keypoints features. $\mathbf{m}_k^{\text{reg}}$ consists of a 2d-transposed-convolutional layer with 17 (i.e., number of keypoints) output channels, a 4×4 kernel and a stride of 2, followed by a bilinear upsampler to increase the keypoints heatmap resolution to $N \times 17 \times 56 \times 56$.

- **Variant1:** While the Mask-RCNN keypoints head estimates the keypoint positions, it does not predict whether they are visible or occluded. This may hurt the TTA performance, e.g., in Figure 4, the position of the left person's elbow has correctly been estimated, however, not specifying it as occluded has caused parts of the bouquet to be included in the TTA mask. To address this all variants predict if a keypoint is visible or occluded, e.g., Variant 2 is identical to Mask-RCNN except that the 2D-transposed-convolutional layer has $51 (= 3 \times 17)$ outputs to allow two additional outputs per location for visible/occluded prediction.

- **Variant2:** As Figure 5 shows, the keypoint estimates from a convolutional head can be of very low quality, severely impacting the TTA/TTG gains. To enhance keypoint estimation (through global attention) Variant2 uses transformers, i.e., $\mathbf{m}_k^{\text{fe}}$ consists of a transformer decoder with 6 layers, 8 heads and 17 queries of width 256, operating on the backbone feature map. The input queries are trainable parameters, while the output queries are mapped to keypoints and decoded by a 3-layer MLP to $N \times 51 \times 14 \times 14$ keypoints heatmap (after reshaping). Hence, there are 51 extra keypoints features. $\mathbf{m}_k^{\text{reg}}$ consists of a bilinear upsampler to increase the heatmap resolution to 56×56.

- **Variant 3:** As will be shown (c.f., Table 1), while Variant 2 shows slightly better (i.e., 1%) $\mathbf{AP}_{key}$ numbers on the target datasets, it lags Mask-RCNN and Variant 1 on the source dataset by a large margin (4%). This is because in Variant 2, the transformer output queries are directly decoded to 14×14 heatmaps, without using convolutional layers. Variant 3 gets around this problem by using the last transformer layer's value projections, and attention weights to form a separate $15 \times 14 \times 14$ feature map for each keypoint. Hence there are $(255 = 17 \times 15)$ extra keypoints features. $\mathbf{m}_k^{\text{reg}}$ consists of two group 2D-convolutional and one group 2D-transposed-convolutional layers that further process keypoint feature maps, independently from one another, followed by a bilinear upsampler to get the 56×56 heatmaps.

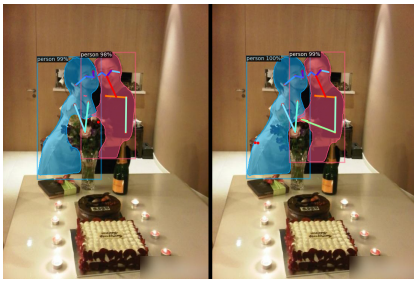

Figure 4. Backpropagating the left person's elbow pseudo-label, without specifying it as occluded, degrades TTA performance by inclusion of part of the bouquet as that person's mask (left to right: baseline and TTA for Mask-RCNN).

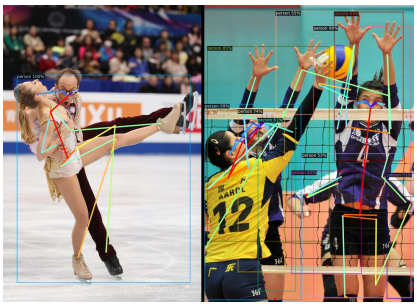

Figure 5. Keypoint estimates from a convolutional head can be of very low quality.

4. Experiments

This section consists of experimental details, TTA vs. TTG comparison results and ablation studies.

4.1. Experimental Details

We report evaluation results on the COCOPersons [35] and OCHuman [35] datasets. The COCOPersons dataset, which consists of 60K images, is a refined split obtained from MS-COCO [19]. In this split, non-person categories as well as person categories with small annotations are removed since they do not contain keypoint annotations. The OCHuman dataset is highly challenging and mainly consists of occluded humans, e.g., the average MaxIOU for each person in COCOPersons is 0.08 while that of OCHuman is 0.67. Both these datasets contain 17 keypoints.

We use Mask-RCNN [12] for our TTA and TTG experiments, though our methods apply to any architecture consistent with Figure 3. More specifically, we use the detectron2 [33] codebase with ResNet-50-FPN and ResNet-

Table 1. Keypoint head's $\mathbf{AP}_{key}$ (ResNet-50-FPN, 4 seeds). The large difference ($> 30\%$) between COCOPersons *val* and OCHuman *val* and *test* shows the large domain-shifts involved. Variant 2, despite a weaker performance on source generalizes better (1%) to the target domains, thanks to using transformers.

Model	COCO Persons *val*	OCHuman *val*	OCHuman *test*
Mask-RCNN	64.85 (0.10)	32.21 (0.32)	31.91 (0.23)
Variant 1	64.87 (0.17)	32.19 (0.17)	31.83 (0.32)
Variant 2	59.90 (0.11)	33.45 (0.54)	32.91 (0.23)
Variant 3	63.94 (0.12)	32.30 (0.31)	31.67 (0.28)

101-FPN as backbones and the mask and keypoints losses implemented therein. The three variants in this work differ from the standard Mask-RCNN only in their keypoints head as detailed in Section 3.

We train all models and variants offline on the COCOPersons *train* set. For baseline and TTG experiments, we keep model weights frozen throughout evaluations on COCOPersons *val* split and OCHuman *val* and *test* splits. For TTA experiments, we reset the weights to their pre-adaptation state before evaluating each test image. We then adapt the model to the image $n = 3$ times with a TTA learning rate of $1\mathrm{e}{-3}$, after which we segment the image to get the TTA person masks (c.f. Algorithm 1). This process is repeated for each test image in the dataset. We report $\mathbf{AP}_{key}$ and $\mathbf{AP}_{mask}$ mean and standard deviation across multiple runs (4 and 3 seeds for ResNet50-FPN and ResNet-101-FPN, respectively).

4.2. TTA vs. TTG Results

Table 1 gives $\mathbf{AP}_{key}$ numbers for Mask-RCNN and its variants. The large drop, i.e., more than 30%, in Mask-RCNN's $\mathbf{AP}_{key}$ when moving from COCOPersons *val* to OCHuman *val* and *test* attests to the significant domain shift between these datasets. Variant 1 uses the same keypoints head as Mask-RCNN, hence its numbers are similar. For the source dataset, i.e., COCOPersons *val*, Variant 2 is lagging Mask-RCNN and Variant 3's $\mathbf{AP}_{key}$ by 4% to $\%5$, however it shows around 1% advantage over the target datasets, i.e., OCHuman *val* and *test*. This is likely due to Variant 2 not using convolutional layers and directly decoding keypoint tokens into spatial heatmaps. Figure 6 depicts the keypoints estimates from the three variants for a test image. As both Table 1 and Figure 6 suggest, use of transformers does not improve the quality of keypoint pseudo-labels, which is a major factor limiting TTA gains.

Table 2 gives $\mathbf{AP}_{mask}$ numbers for various models over the source dataset, i.e., COCOPersons *val*. Note that while TTG improves $\mathbf{AP}_{mask}$ by around 1% percent, TTA degrades it by about the same amount. TTG improves $\mathbf{AP}_{mask}$ by using extra, i.e., keypoint, ground-truth labels for training the *segmentation* head. TTA hurts $\mathbf{AP}_{mask}$ because there is not enough domain shift between COCOPersons *train* and *val* to justify adjusting the model's already optimized weights based on a single unlabelled test image.

Figure 6. Keypoints pseudo-label quality is a major limiting factor for TTA. From left to right: Variant 1, 2 and 3.

Table 2. $\mathbf{AP}_{mask}$ on COCOPersons *val* (ResNet-50-FPN, 4 seeds). Without a large domain-shift TTA hurts performance and TTG shows only a small gain (1%).

Model	Baseline	TTA	TTG
Mask-RCNN	60.11 (0.12)	58.59 (0.25)	61.25 (0.12)
Variant 1	60.22 (0.25)	58.62 (0.15)	61.45 (0.10)
Variant 2	59.55 (0.04)	58.55 (0.04)	60.80 (0.15)
Variant 3	60.12 (0.15)	59.02 (0.24)	60.97 (0.03)

Tables 3 and 4 give $\mathbf{AP}_{mask}$ numbers for the target datasets, i.e., OCHuman *val* and *test*. Here the domain shift is large enough (i.e., more than 40% drop in $\mathbf{AP}_{mask}$) to enable TTA to improve various variants' performances using a single unlabelled image. The largest gain is around 1% and corresponds to Variant 2, due to this variant's slightly higher $\mathbf{AP}_{key}$. It is noteworthy that TTA has little, if any benefit, for Mask-RCNN as it does not distinguish between the visible and occluded keypoints as discussed earlier. More importantly though, the TTG gains are significantly higher than those offered by TTA, e.g., 3.77% for TTG compared to TTA's 1.05% for Variant 2 on OCHuman *val*. Furthermore, the TTG gains for all variants are similar, i.e., TTG $\mathbf{AP}_{mask}$ numbers are within a standard deviation from one another. This is because TTG does not use the keypoint *estimates* directly; it instead uses the keypoint *features* that are richer in information, e.g., implicitly infer keypoints' visibility/occlusion. It also reconfirms that TTG gain comes from using the extra keypoint ground-truth labels for training the segmentation head and not from the details of the keypoint head architecture.

Tables 5, 6 and 7 give our results for ResNet-101-FPN backbone, and show very similar trends, i.e., in the absence of a significant domain shift, TTA may hurt and TTG show only a small gain (c.f., Table 5), whereas with a large

Table 3. $\mathbf{AP}_{mask}$ on OCHuman *val* (ResNet-50-FPN, 4 seeds). With a large domain-shift TTA improves all variants, however TTA gains are smaller than TTG, and more sensitive to keypoint head's architecture.

Model	Baseline	TTA	TTG
Mask-RCNN	17.74 (0.25)	17.66 (0.15)	21.48 (0.10)
Variant 1	17.93 (0.19)	18.00 (0.19)	21.56 (0.17)
Variant 2	18.15 (0.27)	18.79 (0.28)	21.51 (0.51)
Variant 3	18.41 (0.29)	18.44 (0.29)	21.20 (0.14)

Table 4. $\mathbf{AP}_{mask}$ on OCHuman *test* (ResNet-50-FPN, 4 seeds). Same trends as Table 3, i.e., TTA gains are smaller and more heuristic dependent than TTG.

Model	Baseline	TTA	TTG
Mask-RCNN	17.48 (0.15)	17.51 (0.15)	21.30 (0.18)
Variant 1	17.42 (0.33)	17.69 (0.22)	21.09 (0.07)
Variant 2	17.92 (0.21)	18.50 (0.11)	20.93 (0.08)
Variant 3	17.99 (0.06)	18.04 (0.12)	20.64 (0.21)

Table 5. $\mathbf{AP}_{mask}$ on COCOPersons *val* (ResNet-101-FPN, 3 seeds). Same trends as ResNet-50-FPN, i.e., without a large domain-shift, TTA hurts and TTG gives a small gain (1%).

Model	Baseline	TTA	TTG
Mask-RCNN	60.87 (0.10)	59.22 (0.21)	61.73 (0.09)
Variant 1	60.93 (0.10)	59.14 (0.10)	61.82 (0.02)
Variant 2	60.71 (0.15)	59.49 (0.06)	61.69 (0.12)
Variant 3	61.15 (0.10)	59.70 (0.28)	61.81 (0.19)

Table 6. $\mathbf{AP}_{mask}$ on OCHuman *val* (ResNet-101-FPN, 3 seeds). Same trends as with ResNet-50-FPN, i.e., TTG gains are larger and less dependent on pose head's architectural nuances.

Model	Baseline	TTA	TTG
Mask-RCNN	19.21 (0.36)	19.58 (0.23)	22.67 (0.35)
Variant 1	19.10 (0.40)	19.45 (0.40)	22.60 (0.23)
Variant 2	19.32 (0.38)	20.30 (0.17)	22.51 (0.32)
Variant3	19.77 (0.18)	20.46 (0.25)	22.50 (0.15)

enough domain shift, TTA gains are smaller and more dependent on the heuristics used, while TTG gains are larger and less sensitive to model variations (c.f., Tables 6, 7).

Table 8 compares the TTG $\mathbf{AP}_{mask}$ for Variant 2 against some of the existing methods in the literature as described in Section 2 (numbers are cited from the referenced papers).

Table 7. $\mathbf{AP}_{mask}$ on OCHuman *test* (ResNet-101-FPN, 3 seeds). Same trends as with ResNet-50-FPN, i.e., TTG gains are larger and more consistent across various variants.

Model	Baseline	TTA	TTG
Mask-RCNN	18.76 (0.02)	18.97 (0.28)	21.95 (0.54)
Variant 1	18.71 (0.27)	18.93 (0.23)	22.42 (0.22)
Variant 2	18.92 (0.19)	19.92 (0.11)	22.09 (0.58)
Variant 3	19.68 (0.05)	20.26 (0.28)	22.27 (0.38)

Table 8. Comparison of TTG $\mathbf{AP}_{mask}$ for Variant 2 against methods from the literature.

Model	Backbone	COCO Persons *val*	OCHuman *val*	OCHuman *test*
Mask-RCNN [12]	ResNet-50-FPN	53.2	16.3	16.9
Pose2Seg [35]	ResNet-50-FPN	55.5	22.2	23.8
CondInst [29]	ResNet-50-FPN	54.8	20.3	20.1
YOLACT [3]	ResNet-101-FPN	50.2	13.2	13.5
Variant 2 (ours)	ResNet-50-FPN	60.8	21.5	20.9
Variant 2 (ours)	ResNet-101-FPN	61.7	22.5	22.1

Table 9. TTA $\mathbf{AP}_{mask}$ on OCHuman *val* with different learning rates (ResNet-50-FPN, 4 seeds).

Model	Lr			
	0.5e-3	1e-3	2e-3	4e-3
Variant 2	18.49 (0.29)	18.75 (0.29)	18.92 (0.21)	18.57 (0.38)

While an apple to apple comparison is difficult due to the nuances in each paper setup, we observe that our simple TTG method provides a competitive performance especially noting that we report *average* values across multiple seeds. This reconfirms our observation that TTG gains are less dependent on the heuristics used and more robust to architectural variations.

4.3. Ablation Studies

In this section we report several ablation studies that shed light on some of the factors limiting the performance of test-time adaptation. Tables 9, 10 and 11 give the TTA $\mathbf{AP}_{mask}$ for Variant 2 on OCHuman *val* when sweeping the *adaptation* learning-rate, min-person-score (i.e., threshold for rejecting all keypoints for a detected person) and min-keypoint-prob (i.e., threshold for rejecting an individual keypoint), respectively. As the tables show, our choices for these parameters (i.e., $1\mathrm{e}{-3}$, 0.5 and 0.05, respectively) lie within the relevant sweet-spots, however, since we do not assume any knowledge about the target domain, we cannot use the *optimal* values (as Table 9 indicates a learning-rate of $1\mathrm{e}{-3}$ is not optimal) and have to rely on judgment calls which is an important limiting factor. For example as Figure 7 shows, increasing the min-person-score from 0.5 to 0.8 removes some of the false-positives from the volleyball scene, but also causes some true-positives to be missed in the ice-skating scene.

We note that dynamic selection of hyper-parameters such

Table 10. TTA $\mathbf{AP}_{mask}$ on OCHuman *val* with different min-person-score (ResNet-50-FPN, 4 seeds).

Model	Min. Person Score			
	0.5	**0.6**	**0.7**	**0.8**
Variant 2	18.78 (0.25)	18.70 (0.27)	18.74 (0.26)	18.70 (0.31)

Table 11. TTA $\mathbf{AP}_{mask}$ on OCHuman *val* with different min-keypoint-prob (ResNet-50-FPN, 4 seeds).

Model	Min. Keypoint Prob.			
	0.05	**0.1**	**0.2**	**0.4**
Variant 2	18.75 (0.30)	18.54 (0.25)	18.29 (0.25)	18.08 (0.30)

Figure 7. Increasing the min-person-score from 0.5 to 0.8 removes some false-positives from the volleyball scene (first and second from left, respectively), but also causes some true-positives to be missed in the ice-skating scene (third and fourth).

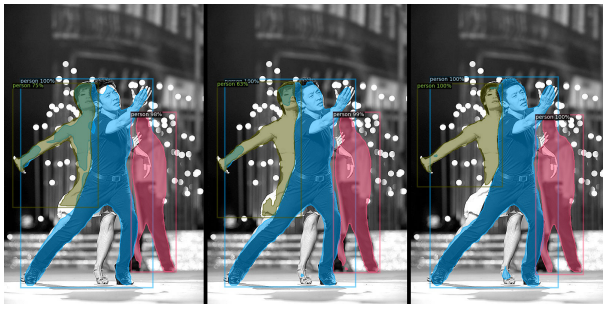

Figure 8. Dynamic selection of hyper-parameters, based on the test image, may improve TTA gain significantly, but is impractical in the absence of prior knowledge about the target domain. From left to right: baseline and test-time-adapted Mask-RCNN for TTA learning rates of $0.5\mathrm{e}{-3}$, and $4\mathrm{e}{-3}$, respectively. For this test image a learning rate of $0.5\mathrm{e}{-3}$ greatly cleans up the spill-over of the man's mask. A larger learning rate of $4\mathrm{e}{-3}$ degrades the performance by shrinking the woman's mask.

as TTA learning-rate, depending on the test image at hand, may significantly improve the TTA performance, e.g., for the particular image shown in Figure 8, a learning rate of $0.5\mathrm{e}{-3}$ greatly cleans up the spill-over of the man's mask. Whereas a larger learning rate of $4\mathrm{e}{-3}$ degrades the performance by shrinking the woman's mask. We also note that such dynamic hyper-parameter selection is impractical in the absence of prior knowledge about the target domain (i.e., another TTA limiting factor).

Table 12 reports numbers for the scenario where the backbone is only partially adaptable (i.e., stage 5 of ResNet-50-FPN). As the table shows, for smaller learning rates the partially adapted backbone yields smaller TTA gains compared to the fully-adaptable one. However for the large learning-rate of $4\mathrm{e}{-3}$ it retains more of the gain which is consistent with being partially adaptable, hence more regularized.

Table 12. TTA $\mathbf{AP}_{mask}$ on OCHuman *val* with backbone fully or partially adapted (ResNet-50-FPN, 4 seeds).

Model	Lr			
	0.5e-3	**1e-3**	**2e-3**	**4e-3**
Variant 2 w. backbone fully adapted	18.49 (0.29)	18.75 (0.29)	18.92 (0.21)	18.57 (0.38)
Variant 2 w. backbone's last-stage adapted	18.34 (0.37)	18.57 (0.39)	18.85 (0.28)	18.83 (0.35)

Table 13. $\mathbf{AP}_{mask}$ on OCHuman *val* with both TTG and TTA (ResNet-50-FPN, 4 seeds). Applying TTA on top of TTG has minimal gain, if any, as TTG already uses keypoint features.

Model	Baseline	TTG	TTG+TTA
Mask-RCNN	17.74 (0.25)	21.48 (0.10)	21.37 (0.22)
Variant 1	17.93 (0.19)	21.56 (0.17)	21.77 (0.27)
Variant 2	18.15 (0.27)	21.51 (0.51)	21.70 (0.41)
Variant 3	18.41 (0.29)	21.20 (0.14)	20.96 (0.43)

Finally, Table 13 gives $\mathbf{AP}_{mask}$ numbers for the case where a TTG model is test-time adapted. We note that TTA gains, if any, are even smaller as TTG already takes advantage of the features generated by the keypoints head.

5. Conclusion

We compared two approaches for enhancing human segmentation masks using keypoints estimation. First was test-time adaptation (TTA), where we allowed test-time adjustments of network weights based on the unlabeled test image without access to the labeled source dataset. It worked by back-propagating keypoints estimates, as pseudo-labels, to adjust the backbone weights. The second approach was training-time generalization (TTG), where we allowed offline access to the labeled source dataset but no test-time alteration of weights. We further did not assume the availability of any test images. Our TTG method worked by augmenting the backbone features with that of the keypoints head and using it to infer masks. We also devised three additional pose-head variants to improve keypoint pseudo-label quality, using keypoint visibility/occlusion prediction, and use of global-attention (i.e., transformers).

We evaluated both approaches and, through ablations, identified factors limiting the TTA gains, i.e., we showed that without a large domain shift, TTA hurts performance, and TTG shows only a small gain. In contrast, for a large domain shift, TTA gains are smaller and more heuristics dependent, while TTG's are larger and more robust.

References

[1] Niaz Ahmad, Jawad Khan, Jeremy Yuhyun Kim, and Youngmoon Lee. Joint human pose estimation and instance segmentation with poseplusseg. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(1):69–76, Jun. 2022.

[2] Yang Bai and Weiqiang Wang. Acpnet: anchor-center based person network for human pose estimation and instance segmentation. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1072–1077. IEEE, 2019.

[3] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9157–9166, 2019.

[4] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4974–4983, 2019.

[5] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7103–7112, 2018.

[6] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1290–1299, 2022.

[7] Lutao Chu, Yi Liu, Zewu Wu, Shiyu Tang, Guowei Chen, Yuying Hao, Juncai Peng, Zhiliang Yu, Zeyu Chen, Baohua Lai, et al. Pp-humanseg: Connectivity-aware portrait segmentation with a large-scale teleconferencing video dataset. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 202–209, 2022.

[8] Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *European conference on computer vision*, pages 534–549. Springer, 2016.

[9] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 2334–2343, 2017.

[10] Yuxin Fang, Shusheng Yang, Xinggang Wang, Yu Li, Chen Fang, Ying Shan, Bin Feng, and Wenyu Liu. Instances as queries. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6910–6919, 2021.

[11] Wenchao Gu, Shuang Bai, and Lingxing Kong. A review on 2d instance segmentation based on deep neural networks. *Image and Vision Computing*, page 104401, 2022.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[13] Joy Hsu, Wah Chiu, and Serena Yeung. Darcnn: Domain adaptive region-based convolutional neural network for unsupervised instance segmentation in biomedical images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1003–1012, 2021.

[14] Yalong Jiang, Wenrui Ding, Hongguang Li, Hua Yang, and Xu Wang. A self-supervised framework for human instance segmentation. In *European Conference on Computer Vision*, pages 479–495. Springer, 2020.

[15] Ziyu Jiang, Zhenhua He, Xueqin Huang, Zibin Yang, and Pearl Tan. Ce-peopleseg: Real-time people segmentation with 10% cpu usage for video conference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 914–922, 2021.

[16] Muhammed Kocabas, Salih Karagoz, and Emre Akbas. Multiposenet: Fast multi-person pose estimation using pose residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 417–433, 2018.

[17] Yaoyi Li and Hongtao Lu. Natural image matting via guided contextual attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11450–11457, 2020.

[18] Yonghui Li, Yao Xue, Liangfu Li, Xingjun Zhang, and Xueming Qian. Domain adaptive box-supervised instance segmentation network for mitosis detection. *IEEE Transactions on Medical Imaging*, 2022.

[19] 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 *European conference on computer vision*, pages 740–755. Springer, 2014.

[20] Dongnan Liu, Donghao Zhang, Yang Song, Fan Zhang, Lauren O'Donnell, Heng Huang, Mei Chen, and Weidong Cai. Unsupervised instance segmentation in microscopy images via panoptic domain adaptation and task re-weighting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4243–4252, 2020.

[21] Dongnan Liu, Donghao Zhang, Yang Song, Fan Zhang, Lauren O'Donnell, Heng Huang, Mei Chen, and Weidong Cai. Pdam: A panoptic-level feature alignment framework for unsupervised domain adaptive instance segmentation in microscopy images. *IEEE Transactions on Medical Imaging*, 40(1):154–165, 2020.

[22] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018.

[23] Fengling Mao, Bingpeng Ma, Hong Chang, Shiguang Shan, and Xilin Chen. Learning efficient text-to-image synthesis via interstage cross-sample similarity distillation. *Science China Information Sciences*, 64(2):1–12, 2021.

[24] Jiayuan Mao, Tete Xiao, Yuning Jiang, and Zhimin Cao. What can help pedestrian detection? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3127–3136, 2017.

[25] Wanli Ouyang, Hui Zhou, Hongsheng Li, Quanquan Li, Junjie Yan, and Xiaogang Wang. Jointly learning deep features, deformable parts, occlusion and classification for pedestrian detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1874–1887, 2017.

[26] George Papandreou, Tyler Zhu, Liang-Chieh Chen, Spyros Gidaris, Jonathan Tompson, and Kevin Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *Proceedings of the European conference on computer vision (ECCV)*, pages 269–286, 2018.

[27] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Learn, imagine and create: Text-to-image generation from prior knowledge. *Advances in neural information processing systems*, 32, 2019.

[28] Vinkle Srivastav, Afshin Gangi, and Nicolas Padoy. Unsupervised domain adaptation for clinician pose estimation and instance segmentation in the operating room. *Medical Image Analysis*, 80:102525, 2022.

[29] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *Computer Vision – ECCV 2020*, pages 282–298, Cham, 2020.

[30] Subarna Tripathi, Maxwell Collins, Matthew Brown, and Serge Belongie. Pose2instance: Harnessing keypoints for person instance segmentation. *arXiv preprint arXiv:1704.01152*, 2017.

[31] Andre Ückermann, Robert Haschke, and Helge Ritter. Realtime 3d segmentation for human-robot interaction. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2136–2143. IEEE, 2013.

[32] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. Solo: Segmenting objects by locations. In *European Conference on Computer Vision*, pages 649–665. Springer, 2020.

[33] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[34] Jingwen Ye, Yongcheng Jing, Xinchao Wang, Kairi Ou, Dacheng Tao, and Mingli Song. Edge-sensitive human cutout with hierarchical granularity and loopy matting guidance. *IEEE Transactions on Image Processing*, 29:1177–1191, 2019.

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

[36] Xiangzhou Zhang, Bingpeng Ma, Hong Chang, Shiguang Shan, and Xilin Chen. Location sensitive network for human instance segmentation. *IEEE Transactions on Image Processing*, 30:7649–7662, 2021.

[37] Ziyu Zhang, Sanja Fidler, and Raquel Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 669–677, 2016.

[38] Xiaomei Zhao, Fulin Tang, and Yihong Wu. Real-time human segmentation by bowtienet and a slam-based human ar system. *Virtual Reality & Intelligent Hardware*, 1(5):511–524, 2019.

[39] Desen Zhou and Qian He. Poseg: Pose-aware refinement network for human instance segmentation. *IEEE Access*, 8:15007–15016, 2020.

[40] Qinqin Zhou, Bineng Zhong, Yulun Zhang, Jun Li, and Yun Fu. Deep alignment network based multi-person tracking with occlusion and motion reasoning. *IEEE Transactions on Multimedia*, 21(5):1183–1194, 2018.

[41] Bingke Zhu, Yingying Chen, Jinqiao Wang, Si Liu, Bo Zhang, and Ming Tang. Fast deep matting for portrait animation on mobile phone. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 297–305, 2017.