Supplementary: Uncertainty-Aware Adaptation for Self-Supervised 3D Human Pose Estimation

The supplementary document is organized as follows:

- Section 1: Notations
- Section 2: Training algorithms
- Section 3: Network architecture
- Section 4: Qualitative analysis

1. Notations

Most of the notations used in this paper are summarized in Table 1. In the first part, we list the general architecture related notations. Next, we group other notations into a) output of B_L, b) output of B_R, c) datasets, and finally the adaptation training related notations for both d) pose-level and e) joint-level adaptation.

Table 1. Notation Table

	Symbol	Description
Miscellaneous	J	Total no. of joints (17) indexed by j
	E	Encoder as the common backbone CNN
	B_L	Localization branch (outputs heatmaps)
	B_R	Regression branch (outputs 3D pose)
	T_{FK}	Forward-kinematics operation
	T_c	Weak-perspective projection operation
o/p of B_L	$\tilde{h}^{(j)}$	Heatmap PDF for j^{th} joint $\in \mathbb{R}^{H' \times W'}$
	$\tilde{q}^{(j)}$	2D pose coordinates for the j^{th} joint $\in \mathbb{R}^2$
	$\tilde{w}^{(j)}$	Joint confidences for the j^{th} joint $\in [0,1]$
O/p of B_R	$\hat{p}^l$	Local pose vectors (parent-relative) $\in \mathbb{R}^{J \times 3}$
	$\hat{c}$	Camera parameters (3 angles, 1 scale, 2 translations)
	$\hat{p}^c$	Canonical 3D pose coordinates $\in \mathbb{R}^{J \times 3}$
	$\hat{p}$	Camera-relative 3D pose coordinates $\in \mathbb{R}^{J \times 3}$
	$\hat{q}$	projected 2D pose coordinates $\in \mathbb{R}^{J \times 2}$
Datasets	$\mathcal{D}_s$, $\mathcal{D}_t$	Labeled source and unlabeled target datasets (full-body)
	$\mathcal{D}_s^O$, $\mathcal{D}_t^O$	Source and target datasets with occlusion/truncation
	$\mathcal{D}_b$	A dataset of background images (other than human)
Pose-level	$\mathcal{U}(I)$	Pose-level uncertainty for a given image
	$\mathcal{L}_{Sup}^{(s)}$	Supervised loss on $\mathcal{D}_s$ samples (minimized)
	$\mathcal{U}^{(s)}$	Pose-uncertainty of $\mathcal{D}_s$ samples (minimized)
	$\mathcal{U}^{(b)}$	Pose-uncertainty of $\mathcal{D}_b$ samples (maximized)
	$\mathcal{U}^{(t)}$	Pose-uncertainty of $\mathcal{D}_t$ samples (minimized)
	$\mathcal{L}_{pSup}^{(t)}$	Loss on pseudo-label target subset $\mathcal{D}_t^{pl}$ (minimized)
	α_p^{th}	Threshold to select pseudo-labeled target subset $\mathcal{D}_t^{pl}$
Joint-level	$\mathcal{H}(I,j)$	Joint-level uncertainty (JU) for a given image, joint-id pair
	$\mathcal{L}_{Sup}^{OA}$	Occlusion-aware supervised loss on $\mathcal{D}_s^O$ (minimized)
	$\mathcal{H}_{\mathcal{J}_{outV}^s}^{(s)}$	JU of true out-view joints of $\mathcal{D}_s^O$ (maximized)
	$\mathcal{H}_{\forall j}^{(b)}$	JU of all joints for backgrounds $\mathcal{D}_b$ (maximized)
	$\mathcal{H}_{\mathcal{J}_{inV}^t}^{(t)}$	JU of pseudo-selected in-view joints of $\mathcal{D}_t^O$ (minimized)
	$\mathcal{H}_{\mathcal{J}_{outV}^t}^{(t)}$	JU of pseudo-selected out-view joints of $\mathcal{D}_t^O$ (maximized)
	$\mathcal{L}_{pSup}^{OA}$	Loss on pseudo-labeled target set $(I,j) \in \mathcal{J}_{inV}^t$ (minimized)
	α_q^{th}	Threshold to select pseudo-labeled target in-view set $\mathcal{J}_{inV}^t$
	α_h^{th}	Threshold to select pseudo-labeled target out-view set $\mathcal{J}_{outV}^t$

2. Training algorithms

In this section, we clearly discuss the training algorithms which could not be included in the main paper. Algo. 1 and Algo. 3 show the training algorithm for pose-level and joint-level adaptation respectively. We simultaneously train on samples from all the three datasets, *i.e.* on $\mathcal{D}_s$, $\mathcal{D}_t$, and $\mathcal{D}_b$ for pose-level adaptation and on $\mathcal{D}_s^O$, $\mathcal{D}_t^O$, and $\mathcal{D}_b$ for joint-level adaptation. The pseudo-label selection procedure is clearly explained in both the algorithms (refer Table 1 for a description of the notations). Though we use the above for pose-level adaptation for a fair prior-art benchmarking, one is always free to relax this assumption. **a)** Under pose-level DA, synthetic training on $\mathcal{D}_s^O$ (truncated+full) would make it applicable for both full and truncated target. In Fig. 5, notice the medium level uncertainty elicited by `MRPN(PU)` for truncated target (a desirable behaviour). **b)** On the other hand, joint-level adaptation already suits to both the scenarios (`MRPN(JU)` in Fig. 5

Algo. 2 shows a detailed training procedure to prepare the fusion network for the pose-level adaptation scenario. We prepare a separate fusion network for the joint-level adaptation. Table 3 reports relative contributions of B_R and B_L outputs against the fused. In case of joint level adaptation the loss-term in L3 of Algo. 2 is replaced by $\mathbb{1}_{(I,j)\in\mathcal{J}_{inV}^s}\mathcal{L}_p^{(j)}(\hat{p}, p_{gt})$ (the second loss-term in L6 of Algo. 3). Similarly, the loss-term in L4 of Algo. 2 is replaced by $\sum_{j\in\mathcal{J}_{inV}^t}\tilde{w}^{(j)}\mathcal{L}^{(j)}(\hat{p}, p_{gt}^{pl})$ (the second loss-term in L11 of Algo. 3).

We trained the framework on an NVIDIA P-100 GPU (16GB) with a batch size of 8. We employ separate Adam

Algorithm 1 Training algorithm for pose-level adaptation.

Input: Labeled source dataset $\mathcal{D}_s$, unlabeled target dataset $\mathcal{D}_t$, and the background dataset $\mathcal{D}_b$. Let Θ denote the learnable parameters of the *MRP-Net* architecture (excluding the fusion network).

while *iter* $<$ *MaxIter* **do**

A. Pseudo-label update *(after each $K_{interval}$).*

if *iter* $\pmod{K_{interval}} = 0$ **then**

  Compute $\mathcal{D}_t^{pl}$ where $\hat{q}_t$ and $\tilde{q}'_t$ are obtained using current state of network parameters Θ, as follows: $\mathcal{D}_t^{pl} = \{I_t : (|\hat{q}_t - \mathcal{F}_q(\tilde{q}'_t)| + |\tilde{q}_t - \mathcal{F}_q(\hat{q}'_t)|) < \alpha_p^{th}\}$

end if

B. Adaptation training *(for pose-level adaptation).*

Update Θ by minimizing $\mathcal{L}_h(\hat{h}, h_{gt})$, $\mathcal{L}_p(\hat{p}, p_{gt})$, and $\mathcal{U}^{(s)}$ (*i.e.* the first two terms under $\mathcal{L}_{Sup}^{(s)}$) on a mini-batch of $\mathcal{D}_s$ using separate Adam optimizers.

Update Θ by maximizing $\mathcal{U}^{(b)}$ on a mini-batch of $\mathcal{D}_b$ using Adam optimizer.

Update Θ by minimizing $\mathcal{U}^{(t)}$ on a mini-batch of $\mathcal{D}_t$ using Adam optimizer.

Update Θ by maximizing $\sum_j \tilde{w}^{(j)}\mathcal{L}^{(j)}(\tilde{h}, h_{gt}^{pl})$ and $\sum_j \tilde{w}^{(j)}\mathcal{L}^{(j)}(\hat{p}, p_{gt}^{pl})$ (*i.e.* the two terms under $\mathcal{L}_{pSup}^{(t)}$) using separate Adam optimizers.

end while

optimizers [2] for each loss term. Please refer Fig 1 for sensitivity analysis. Note that, we use fixed threshold values across all adaptation settings in Sec 4.1.

Importance of OOD images. We would like to reiterate that the background images represent an objective segregation of hard-OOD samples. The poses outside of the training distribution are critical to identify and we segregate them via the pseudo-label subset selection criteria (Eq. 4). Eq. 5 selectively imposes a strong loss on the more confident target samples. It is to be noted that, such segregation is highly subjective, and treating these soft-OOD samples as hard-OOD deteriorates the generalization performance.

Algorithm 2 Training algorithm for the fusion network.

Input: Labeled source dataset $\mathcal{D}_s$ and the pseudo-labeled target subset $\mathcal{D}_t^{pl}$. The network takes 3 inputs: a) 3D pose predictions via B_R (*i.e.* $\hat{p}$), b) 2D pose prediction via B_L (*i.e.* $\tilde{q}$), and c) the joint-confidences $\tilde{w}$ via B_L. Let θ^f denote the learnable parameters of the fusion network.

while *iter* $<$ *MaxIter* **do**

  Update θ^f to minimize $\mathcal{L}_p(\hat{p}^f, p_{gt})$ on a mini-batch of $\mathcal{D}_s$ using Adam optimizer.

  Update θ^f to minimize $\sum_{j=1}^{J} \tilde{w}^{(j)}\mathcal{L}^{(j)}(\hat{p}^f, p_{gt}^{pl})$ on a mini-batch of $\mathcal{D}_t^{pl}$ using Adam optimizer.

end while

Algorithm 3 Training algorithm for joint-level adaptation.

Input: Labeled source dataset $\mathcal{D}_s^O$, unlabeled target dataset $\mathcal{D}_t^O$, and the background dataset $\mathcal{D}_b$. Let Θ denote the learnable parameters of the *MRP-Net* architecture (excluding the fusion network).

while *iter* $<$ *MaxIter* **do**

A. Pseudo-label update *(after each $K_{interval}$).*

if *iter* $\pmod{K_{interval}} = 0$ **then**

  Compute $\mathcal{J}_{inV}^t$ and $\mathcal{J}_{outV}^t$, where $\hat{q}_t$ and $\tilde{q}'_t$ are obtained using the current state of the network parameters Θ, as follows:

  $\mathcal{J}_{inV}^t = \{(I_t, j) : \mathcal{H}(I_t, j)(|\tilde{q}_t^{(j)} - \mathcal{F}_q^{(j)}(\hat{q}'_t)|) < \alpha_q^{th}\}$

  $\mathcal{J}_{outV}^t = \{(I_t, j) : \mathcal{H}(I_t, j)(|\tilde{q}_t^{(j)} - \mathcal{F}_q^{(j)}(\hat{q}'_t)|) > \alpha_h^{th}\}$

end if

B. Adaptation training *(for joint-level adaptation).*

Update Θ by minimizing $\mathbb{1}_{(I,j)\in\mathcal{J}_{inV}^s}\mathcal{L}_h^{(j)}(\tilde{h}, h_{gt})$ and $\mathbb{1}_{(I,j)\in\mathcal{J}_{inV}^s}\mathcal{L}_p^{(j)}(\hat{p}, p_{gt})$ (*i.e.* the first 2 terms under $\mathcal{L}_{Sup}^{OA}$) on a mini-batch of $\mathcal{D}_s^O$ using separate Adam optimizers.

Update Θ to maximize $\mathcal{H}_{\mathcal{J}_{outV}^s}^{(s)} = \mathbb{1}_{(I,j)\in\mathcal{J}_{outV}^s}\mathcal{H}(I, j)$ on a mini-batch of $\mathcal{D}_s^O$ using Adam optimizer.

Update Θ to maximize $\mathcal{H}_{\forall j}^{(b)}$ on a mini-batch of $\mathcal{D}_b$ using Adam optimizer.

Update Θ to minimize $\mathcal{H}_{\mathcal{J}_{inV}^t}^{(t)} = \mathbb{1}_{(I,j)\in\mathcal{J}_{inV}^t}\mathcal{H}(I, j)$ on a mini-batch of $\mathcal{D}_t^O$ using Adam optimizer.

Update Θ to maximize $\mathcal{H}_{\mathcal{J}_{outV}^t}^{(t)} = \mathbb{1}_{(I,j)\in\mathcal{J}_{outV}^t}\mathcal{H}(I, j)$ on a mini-batch of $\mathcal{D}_t^O$ using Adam optimizer.

Update Θ to maximize $\sum_{j\in\mathcal{J}_{inV}^t} \tilde{w}^{(j)}\mathcal{L}^{(j)}(\tilde{h}, h_{gt}^{pl})$ and $\sum_{j\in\mathcal{J}_{inV}^t} \tilde{w}^{(j)}\mathcal{L}^{(j)}(\hat{p}, p_{gt}^{pl})$ (*i.e.* the two terms under $\mathcal{L}_{pSup}^{OA}$) using separate Adam optimizers. Here, $\tilde{w}^{(j)}$ is normalized such that $\sum_{j\in\mathcal{J}_{inV}^t} \tilde{w}^{(j)} = 1$.

end while

3. Network architecture

The architecture consists of an ImageNet initialized ResNet-50 (till *Res-4F*) which bifurcates into two branches, B_L and B_R as shown in Fig. 3. B_L is a convolutional decoder consisting of an alternate series of transposed convolution and general convolution which progressively increases the spatial resolution from 7×7 to 56×56. The final output of B_L is 17 heatmap PDFs, $\tilde{h}$ obtained via spatial softmax. These are then used to extract the correspond-

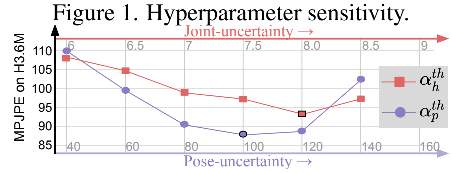
Table 2. Assets and the corresponding Licenses

Asset used	License
Human3.6M [1]	Limited license for academic use
MPI-INF-3DHP [3]	Limited license for academic use
3DPW [7]	Limited license for academic use
HumanEva [5]	Limited license for academic use
SURREAL [6]	Limited license for academic use

Table 3. Relative contribution of fusion network inputs on 3DPW, MPJPE (↓).

Methods	$\hat{p}$	$\tilde{q}$	$\tilde{q}+\tilde{w}$	Fused
Ours(H→3PDW)	100	122	115	**91**
Ours(JU:H→3DPW)	116	142	135	**98**

Figure 1. Hyperparameter sensitivity.



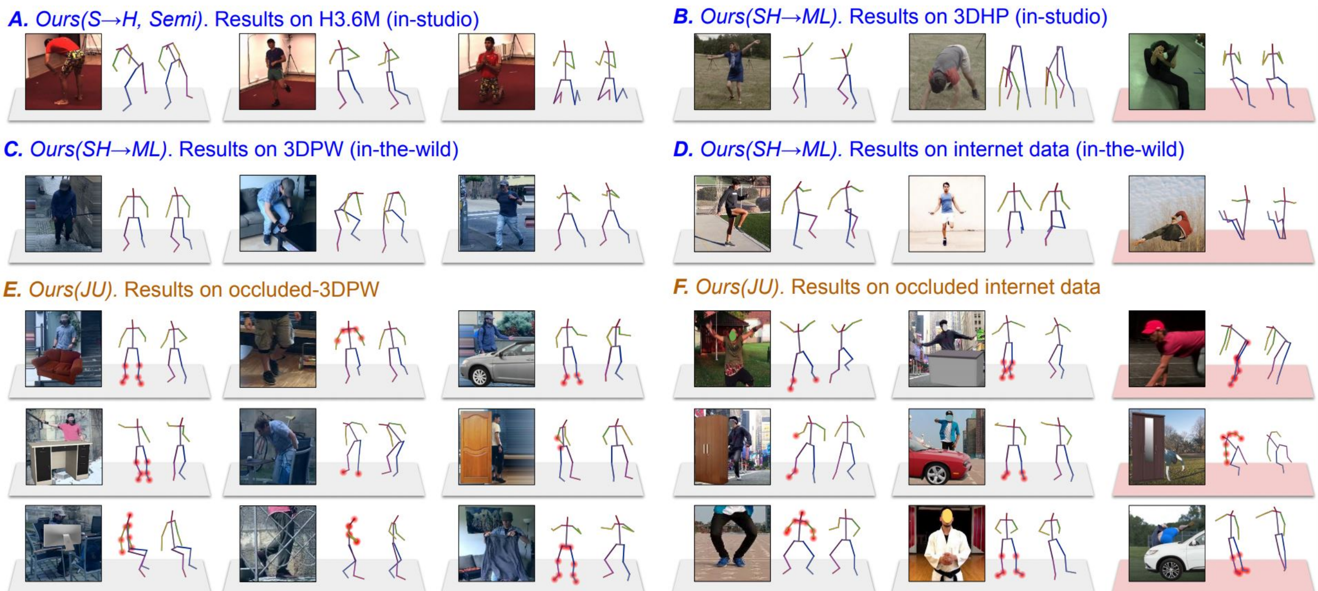


Figure 2. Qualitative analysis. 3D poses shown correspond to the original camera view and another azimuthal view at +30° or -30° depending on best viewing angle. For results in panel **E** and **F** the joints with uncertainty greater than a prefix threshold are highlighted with red-blobs. The model fails on rare poses, complex inter-limb occlusion and heavy background clutter as highlighted by red bases.

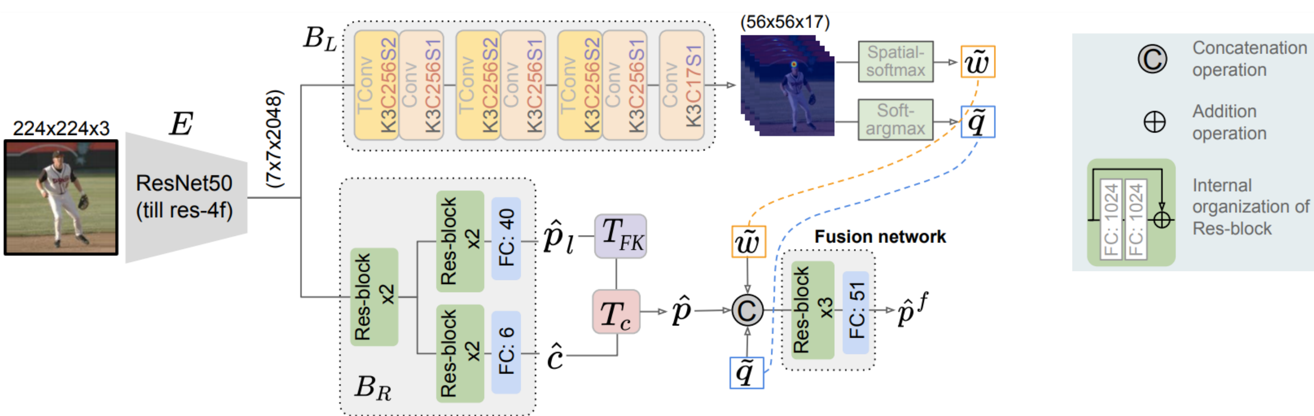


Figure 3. Detailed architecture of the proposed *MRP-Net*. On the right we show the legend. Here, *K3C256S2* denotes specifications of the convolutional layer, *i.e*. 3×3 filter size, 256 filters applied with a stride 2. Here, *TConv* denotes transposed convolution operation. *FC* denotes fully-connected layer. x2 and x3 depict number of residual blocks that are stacked to form the corresponding branch.

ing 2D joint coordinates, $\tilde{q}$ and joint confidence, $\tilde{w}$. B_R consists of a common branch with fully-connected residual blocks [8] which further divides into camera, $\hat{c}$ and pose prediction $\hat{p}_l$ sub-branches, each consisting of 2 residual blocks. The outputs, $\tilde{w}$, $\tilde{q}$, and $\hat{p}$ are concatenated and passed to the fusion network which is composed of a series of 3 residual blocks to regress the final 3D pose, $\hat{p}^f$. Fig. 3 shows the detailed architecture. Further, ablation performance with fusion network is shown in Table 4 (MPJPE of #5-7, Table 4). We see that a better adaptation further enhances the gain from fusion network.

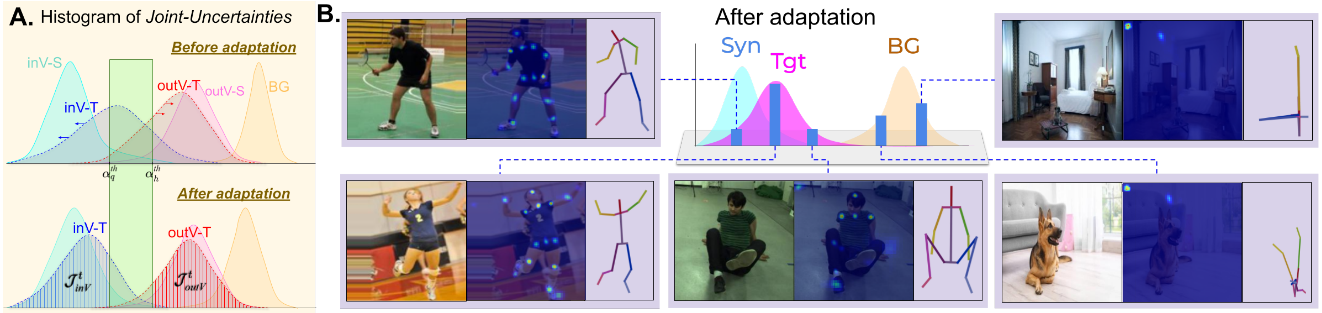


Figure 4. **A.** Shows histogram of the predicted *joint-uncertainties* for the true *in-view* and *out-view* joints separately for source (*i.e.* `inV-S` and `outV-S`) and target (*i.e.* `inV-T` and `outV-T`). `BG` denotes the histogram of all *out-view* joints for backgrounds. The shaded regions in the bottom panel depicts $\mathcal{J}^t_{inV}$ and $\mathcal{J}^t_{outV}$ which are segregated using the preset thresholds α^{th}_q and α^{th}_h respectively (edges of the green-box). Our adaptation algorithm succeeds to separate *inV-T* and *outV-T* over the course of adaptation training. **B.** Shows a similar analysis for *pose-uncertainties*. We show 5 different examples sampled from different regions of the histogram-bins. ***Results on right-panel:*** Notice that to maximize *pose-uncertainty* for backgrounds (OOD samples), *MRPN* estimates the 2D landmarks and 3D pose points separated towards opposite diagonal corners. Here, the 2D landmarks are collapsed to the top-left corner whereas the root joint (pelvis) of the model-based 3D predictions are seemed to have collapsed towards the bottom-right corner. ***Result on bottom-panel:*** For uncertain target instances, we see two peaks in the joint heatmap PDFs; one at the top-left corner (OOD-related) and the other near the actual joint location. During adaptation, the OOD-related peak suppress while the joint-related peak rises to simultaneously reduce the uncertainty while converging towards the true pose outcome. ***Results on the left panel:*** *Joint-level* uncertainty is indicated by the entropy of heatmap PDF.

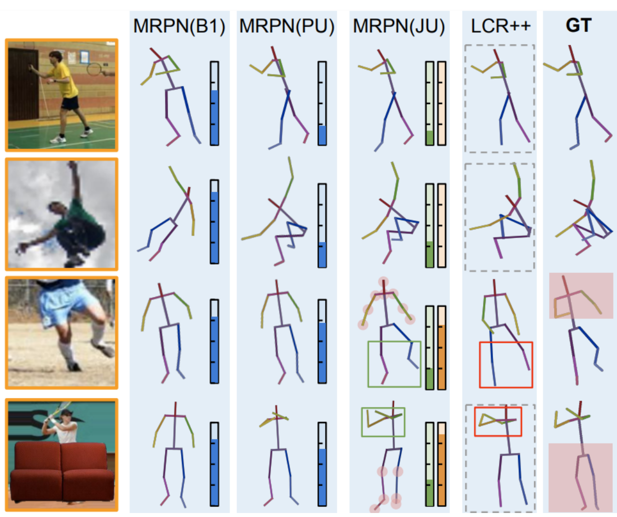


Figure 5. Every pose prediction of *MRPN* is associated with a measure of uncertainty barometer. The barometer height indicates high uncertainty. The blue, green and orange barometers indicate the average prediction uncertainty for the `full-pose`, `true-in-view` joints and `true-out-view` joints respectively. The dotted gray rectangles highlight the failure cases of LCR++ in predicting the correct 3D inter-limb depth though the 2D landmarks align with the GT. In the last 2 rows, the filled red-box under GT column segregates the true *out-view* joints. The *in-view* joint predictions of *MRPN(JU)* (unfilled green rectangles) performs better against the same for LCR++ (unfilled red rectangles) when compared against the same under GT.

4. Qualitative analysis

We perform a thorough qualitative study to interpret the behaviour of our network for a wide variety of in-distribution and out-of-distribution samples (see Fig. 2).

Table 4. Evaluation of #5-7 from Table 4 with fusion network.

No.	Method	$\mathcal{L}^{(s)}_{Sup} - \mathcal{U}^{(b)}$	$\mathcal{U}^t$	$\mathcal{L}^{(t)}_{pSup}$	w/o fuse	w/ fuse
5.	*B2(S→H)+DANN*	only $\mathcal{L}^{(s)}_{Sup}$	Standard DA		116.8	114.5 (2.3 ↓)
6.	*B2(S→H)*	✓	-	-	122.4	122.1 (0.3 ↓)
7.	*B2(S→H)*	✓	✓	-	113.4	110.7 (2.7 ↓)

The analysis in Fig. 4A shows that the proposed joint-level adaptation algorithm succeeds to separate *inV-T* and *outV-T* over the course of adaptation training, thereby aligning these with *inV-S* and *outV-S* respectively. In Fig. 5, *MRPN(B1)* indicates the occlusion-aware network before the adaptation training. *MRPN(PU)* and *MRPN(JU)* indicate the final networks after the *pose-level* and *joint-level* adaptations. Further we show the ground-truth (2D) and predictions on LCR++ [4]. *MRPN(PU)* is not tuned to work on occluded/truncated images and thus yields a higher uncertainty for the last two rows. Whereas, the uncertainty predictions of *MRPN(JU)* for the green and orange barometer yield the expected behaviour.

References

[1] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013. 3

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2

[3] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3D human pose estimation in the wild using improved cnn supervision. In *3DV*, 2017. 3

[4] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. LCR-Net++: Multi-person 2D and 3D pose detection in natural images. *IEEE transactions on pattern analysis and machine intelligence*, 42(5):1146–1161, 2019. 4

[5] Leonid Sigal, Alexandru O Balan, and Michael J Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International journal of computer vision*, 87(1):4–27, 2010. 3

[6] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *CVPR*, 2017. 3

[7] Timo Von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3D human pose in the wild using imus and a moving camera. In *ECCV*, 2018. 3

[8] Bastian Wandt and Bodo Rosenhahn. Repnet: Weakly supervised training of an adversarial reprojection network for 3D human pose estimation. In *CVPR*, 2019. 3