Supplementary Material for "Pretraining boosts out-of-domain robustness for pose estimation"

Alexander Mathis^{1,2*†} Thomas Biasi^{2*} Steffen Schneider^{1,3} Mert Yüksekgönül^{2,3} Byron Rogers⁴ Matthias Bethge³ Mackenzie W. Mathis^{1,2‡} ¹EPFL ²Harvard University ³University of Tübingen ⁴Performance Genetics

*Equal contribution.

[†]Correspondence: alexander.mathis@epfl.ch

[‡]Correspondence: mackenzie.mathis@epfl.ch

1. Additional information on the Horse-10 dataset

The table lists the following statistics: labeled frames, scale (nose-to-eye distance in pixels), and whether a horse was within domain (w.d) or out-of-domain (o.o.d.) for each shuffle.

Horse Identifier	samples	nose-eye dist	shuffle 1	shuffle 2	shuffle 3
BrownHorseinShadow	308	22.3	0.0.d	0.0.d	w.d.
BrownHorseintoshadow	289	17.4	0.0.d	0.0.d	0.0.d
Brownhorselight	306	15.57	0.0.d	w.d.	0.0.d
Brownhorseoutofshadow	341	16.22	0.0.d	w.d.	w.d.
ChestnutHorseLight	318	35.55	w.d.	w.d.	0.0.d
Chestnuthorseongrass	376	12.9	0.0.d	w.d.	w.d.
GreyHorseLightandShadow	356	14.41	w.d.	w.d.	0.0.d
GreyHorseNoShadowBadLight	286	16.46	w.d.	0.0.d	w.d.
TwoHorsesinvideobothmoving	181	13.84	0.0.d	0.0.d	w.d.
Twohorsesinvideoonemoving	252	16.51	w.d.	w.d.	w.d.
Sample1	174	24.78	0.0.d	0.0.d	0.0.d
Sample2	330	16.5	0.0.d	0.0.d	0.0.d
Sample3	342	16.08	0.0.d	0.0.d	0.0.d
Sample4	305	18.51	0.0.d	0.0.d	w.d.
Sample5	295	16.89	w.d.	0.0.d	0.0.d
Sample6	376	12.3	0.0.d	0.0.d	0.0.d
Sample7	262	18.52	w.d.	0.0.d	0.0.d
Sample8	388	12.5	w.d.	w.d.	0.0.d
Sample9	359	12.43	0.0.d	0.0.d	0.0.d
Sample10	235	25.18	0.0.d	0.0.d	0.0.d
Sample11	256	19.16	0.0.d	w.d.	0.0.d
Sample12	288	17.86	w.d.	0.0.d	w.d.
Sample13	244	25.78	w.d.	w.d.	w.d.
Sample14	168	25.55	0.0.d	0.0.d	0.0.d
Sample15	154	26.53	0.0.d	0.0.d	0.0.d
Sample16	212	15.43	0.0.d	0.0.d	0.0.d
Sample17	240	10.04	w.d.	0.0.d	0.0.d
Sample18	159	29.55	0.0.d	w.d.	0.0.d
Sample19	134	13.44	0.0.d	0.0.d	w.d.
Sample20	180	28.57	0.0.d	0.0.d	0.0.d
mean	270.47	18.89			
STD	73.04	6.05			

2. Learning schedule cross validation

Because of the extensive resources required to cross validate all models, we only underwent the search on MobileNetV2s 0.35 and 1.0, ResNet 50, and EfficientNets B0, B3, and B5 for the pretraining and from scratch variants. For all other models, the parameters from the most similar networks were used for training (i.e. EfficientNet-B1 used the parameters for EfficientNet-B0). The grid search started with the highest possible initial learning rate that was numerically stable for each model; lower initial learning rates were then tested to fine tune the schedule. Zero and nonzero decay target levels were tested for each initial learning rate. In addition to the initial learning rates and decay targets, we experimented with shortening the cosine decay and incorporating restarts. All cross validation experiments were performed on the three splits with 50% of the data for training.

For training, a cosine learning rate schedule, as in [4] with ADAM optimizer [2] and batchsize 8 was used. For the learning schedules we use the following abbreviations: Initial Learning Rates (ILR) and decay target (DT).

The tables below list the various initial learning rates explored during cross validation for each model with pretraining.

For the ImageNet pretrained case, the learning rate schedule without restarts was optimal on out of domain data, and the resulting optimal parameters are as follows:

Model	ILR								
MOBILENETV2-0.35	1E-2	5e-3	1E-3	5e-4					
MOBILENETV2-1.0	1E-2	5E-3	1E-3	5e-4					
ResNet-50	1E-3	5e-4	1E-4	5e-5					
EFFICIENTNET-B0	2.5e-3	1E-3	7.5e-4	5e-4					
EFFICIENTNET-B3	1E-3	5e-4	1E-4	5e-5					
EFFICIENTNET-B5	5E-4	1E-4							

MODELS	ILR d	& DT
MOBILENETV2s 0.35, 0.5	1E-2	0
MOBILENETV2s 0.75, 1.0	1E-2	1E-4
ResNets 50, 101	1E-4	1E-5
EFFICIENTNETS B0, B1	5e-4	1E-5
EFFICIENTNETS B2, B3, B4	5e-4	0
EFFICIENTNETS B5,B6	5e-4	1E-5

The initial learning rates explored for the from scratch models during cross validation are as follows:

MODEL	ILR								
MOBILENETV2 0.35	1E-2	5E-3	1E-3	5E-4					
MOBILENETV2 1.0	1E-1	1E-2	1E-3	1E-4					
ResNet 50	1E-3	5e-4	1E-4	5e-5					
EFFICIENTNET-B0	1E-3	5e-4	1E-4	5e-5					
EFFICIENTNET-B3	1E-3	5e-4	1E-4	5e-5					

For models trained from scratch, we found that using restarts lead to the best performance on out of domain data. The optimal learning rates found during the search are as follows:

MODELS	ILR o	& DT
MOBILENETV2s 0.35, 0.5	5e-2	5E-3
MOBILENETV2s 0.75, 1.0	1E-2	0
ResNet 50	5e-4	5e-5
EFFICIENTNETS B0, B3	1E-3	0

3. Baseline Performance on Horse-30

For comparison to Horse-10, we provide the train and test normalized errors for models trained on Horse-30. Here, Horse-30 was split into 3 shuffles each containing a train/test split of 50% of the horse images. Compared to Horse-10, we train these models for twice as long (60,000 iterations) but with the same cross-validated cosine schedules from Horse-10. Errors below are averaged over the three shuffles.

	HORSE-1	0 Errors	HORSE-3	0 Errors
MODELS	TRAIN	Test	TRAIN	TEST
MOBILENETV2 0.35	0.1342	0.1390	0.1545	0.1595
ResNet 50	0.0742	0.0815	0.0772	0.0825
EfficientNet-B4	0.0598	0.0686	0.0672	0.0750

4. Performance (PCK per bodypart) for all networks on Horse-10

The tables below show the PCK for several bodyparts for all backbones that we considered. They complete the abridged tables in the main text (Table 2 and 3) Thereby the bodyparts are abbreviated as follows: (FF=front foot; HF = Hind foot; HH = Hind Hock).

	Nose	Eye	Shoulder	Wither	Elbow	NearFF	OffFF	Hip	NearHH	NearHF	OffHF
MobileNetV2 0.35	90.7	94.1	97.6	96.9	96.7	92.3	93.7	96.4	94.1	94.2	92.5
MobileNetV2 0.5	94.1	96.1	99.2	98.3	98.0	93.8	95.4	96.7	97.2	97.2	97.0
MobileNetV2 0.75	96.0	97.5	99.2	98.0	99.0	96.6	96.8	98.8	97.6	98.0	97.4
MobileNetV2 1.0	97.7	98.8	99.7	99.1	99.0	97.6	97.3	99.4	98.4	98.5	98.9
ResNet 50	99.9	100.0	99.8	99.9	99.8	99.8	99.6	99.9	99.9	99.6	99.8
ResNet 101	99.9	100.0	99.9	99.8	99.9	99.8	99.7	99.8	99.9	99.7	99.9
EfficientNet-B0	99.7	99.9	100.0	99.9	100.0	99.6	99.5	100.0	99.9	99.7	99.7
EfficientNet-B1	99.8	99.9	100.0	99.8	99.9	99.5	99.8	100.0	99.8	99.8	99.8
EfficientNet-B2	99.9	99.9	100.0	99.9	100.0	99.8	99.7	99.9	99.8	99.7	99.7
EfficientNet-B3	99.9	99.9	99.9	99.9	99.9	99.7	99.6	99.7	99.8	99.6	99.9
EfficientNet-B4	100.0	100.0	99.9	99.8	99.9	99.6	99.7	99.9	99.7	99.8	99.8
EfficientNet-B5	99.9	99.9	100.0	99.9	100.0	99.7	99.8	99.6	99.8	99.8	99.9
EfficientNet-B6	99.9	99.9	99.9	99.8	100.0	99.8	99.9	99.8	99.8	99.7	99.8

Table 1: PCK@0.3 (%) for several bodyparts and all evaluated architectures on within domain horses.

Table 2: PCK@0.3 (%) for several bodyparts and all architectures on out-of-domain horses.

	Nose	Eye	Shoulder	Wither	Elbow	NearFF	OffFF	Hip	NearHH	NearHF	OffHF
MobileNetV2 0.35	45.6	53.1	65.5	68.0	69.1	56.4	57.6	65.9	65.9	60.5	62.5
MobileNetV2 0.5	52.7	61.0	76.7	69.7	78.3	62.9	65.4	73.6	70.8	68.1	69.7
MobileNetV2 0.75	54.2	65.6	78.3	73.2	80.5	67.3	68.9	80.0	74.1	70.5	70.2
MobileNetV2 1.0	59.0	67.2	83.8	79.7	84.0	70.1	72.1	82.0	79.9	76.0	76.7
ResNet 50	68.2	73.6	85.4	85.8	88.1	72.6	70.2	89.2	85.7	77.0	74.1
ResNet 101	67.7	72.4	87.6	86.0	89.0	79.9	78.0	92.6	87.2	83.4	80.0
EfficientNet-B0	60.3	62.5	84.9	84.6	87.2	77.0	75.4	86.7	86.7	79.6	79.4
EfficientNet-B1	67.4	71.5	85.9	85.7	89.6	80.0	81.1	86.7	88.4	81.8	81.6
EfficientNet-B2	68.7	74.8	84.5	85.2	89.2	79.7	80.9	88.1	88.0	82.3	81.7
EfficientNet-B3	71.7	76.6	88.6	88.7	92.0	80.4	81.8	90.6	90.8	85.0	83.6
EfficientNet-B4	71.1	75.8	88.1	87.4	91.8	83.3	82.9	90.8	90.3	86.7	85.5
EfficientNet-B5	74.8	79.5	89.6	89.5	93.5	82.2	84.1	91.8	90.9	86.6	85.2
EfficientNet-B6	74.7	79.7	90.3	89.8	92.8	83.6	84.4	92.1	92.1	87.8	85.3

5. CKA analysis of training & trained vs. from scratch networks

Figure 1 shows a linear centered kernel alignment (CKA) [3] comparison of representations for task-training vs. ImageNet trained (no task training) for ResNet-50



Figure 1: CKA comparison of representations for task-training vs. ImageNet trained (no task training) for ResNet-50. Left: Linear CKA on within domain horses (used for training) when trained from scratch vs. plain ImageNet trained (no horse pose estimation task training). Right: Same, but for Transfer Learning vs. from ImageNet. Matrices are the averages over the three splits. In short, task training changes representations.



Figure 2: **CKA comparison of representations when trained from scratch vs. from ImageNet initialization. A:** Top: Linear CKA between layers of individual networks of different depths on within domain horses (used for training the models). Bottom: Same, but for out-of-domain horses (not used in training). Matrices are the averages over the three splits. **B:** Quantification of similarity ratio plotted against depth of the networks.

6. Results of within domain performance on Animal Pose

In Main Figure 4 we show the performance when we train on only 1 species and testing on another (or all without cow/sheep vs. sheep.cow). Here, as a baseline we report the performance within domain, i.e. for each out-of-domain test species (cow and sheep) we trained on 90% of the cow data and tested on 10% cow data (see tables 3 and 4).

Table 3: Test performance on cow w	when trained on 90% of cow data
------------------------------------	---------------------------------

	Normalized Error
MobileNetV2 0.35	0.136
MobileNetV2 1.0	0.093
ResNet 50	0.062
EfficientNet-B0	0.060
EfficientNet-B3	0.054

Table 4: Test performance on sheep when trained on 90% of sheep data

	Normalized Error
MobileNetV2 0.35	0.385
MobileNetV2 1.0	0.248
ResNet 50	0.186
EfficientNet-B0	0.124
EfficientNet-B3	0.159

7. Full results on Horse-C

We show the full set of results for the Horse-C benchmark. We compute the corruptions proposed by Hendrycks et al. [1] using the image corruptions library proposed by Michaelis et al. [7].

The original Horse-30 dataset is processed once for each of the corruptions and severities. In total, Horse-C is comprised of 75 evaluation settings with 8,114 images each, yielding a total of 608,550 images. For a visual impression of the impact of different corruptions and severities, see Figures 3–6.



Figure 3: Noise corruptions for all five different severities (1 to 5, left to right). Top to bottom: Gaussian Noise, Shot Noise, Impulse Noise.

For evaluation, we consider MobileNetV2-0.35, MobileNetV2-1.0, ResNet-50 and the B0 and B3 variants of EfficientNet. All models are either trained on Horse-10 from scratch or pre-trained on ImageNet and fine-tuned to Horse-10, using the three validation splits used throughout the paper. In contrast to our other experiments, we now fine-tune the BatchNorm layers for these models. For both the w.d. and o.o.d. settings, this yields comparable performance, but enables us to use



Figure 4: Blur corruptions for all five different severities (1 to 5, left to right). Top to bottom: Defocus Blur, Motion Blur, Zoom Blur



Figure 5: Weather corruptions for all five different severities (1 to 5, left to right). Top to bottom: Snow, Frost, Fog, Brightness



Figure 6: Digital corruptions for all five different severities (1 to 5, left to right). Top to bottom: Contrast, Elastic Transform, Pixelate, Jpeg Compression

Net Type Pretrained Condition Corruption	mobilen False adapt	et_v2_0.35 base	True adapt	base	mobilen False adapt	et_v2_1.0 base	True adapt	base	resnet5 False adapt	0 base	True adapt	base	efficient False adapt	net-b0 base	True adapt	base	efficien False adapt	net-b3 base	True adapt	base
huiabtu ana	0.24	1.76	0.20	0.87	0.27	1.67	0.21	0.04	0.22	1.20	0.17	0.24	0.40	1.40	0.10	0.72	0.22	1.29	0.10	0.70
originatess	0.54	1.70	0.29	4.79	0.27	1.07	0.21	4.02	0.55	0.57	0.17	0.24	0.40	1.40	0.19	4.01	0.55	1.56	0.19	2.42
contrast	0.47	0.02	0.50	4.78	0.50	0.05	0.22	4.95	0.58	0.57	0.18	2.41	0.41	0.95	0.20	4.01	0.57	7.94	0.20	3.45
derocus_biur	0.81	3.22	0.09	2.20	0.01	3.51	0.00	3.35	0.85	3.44	0.60	1.54	0.67	2.05	0.51	2.54	0.71	2.04	0.54	2.01
elastic_transform	0.38	0.96	0.56	0.85	0.32	0.96	0.32	0.92	0.35	0.50	0.26	0.29	0.39	0.91	0.29	0.76	0.38	0.90	0.29	0.77
fog	1.57	6.62	0.41	1.55	1.17	7.20	0.30	2.23	1.09	7.51	0.26	0.56	1.63	5.31	0.27	1.15	1.28	6.80	0.25	1.11
frost	2.27	6.74	1.10	3.78	1.97	6.81	1.00	3.84	1.68	6.44	0.60	1.80	1.39	6.44	0.71	2.91	1.43	7.04	0.67	2.43
gaussian_noise	2.65	5.90	1.68	6.98	2.11	6.19	1.91	7.51	0.97	3.53	0.82	5.25	1.65	5.13	1.22	5.77	1.71	5.82	1.25	5.89
glass_blur	0.60	2.03	0.63	1.57	0.50	2.01	0.67	2.34	0.54	1.69	0.50	0.95	0.53	1.35	0.53	1.56	0.56	1.45	0.59	1.39
impulse_noise	2.36	5.75	1.73	6.88	1.86	6.07	1.91	7.46	0.83	3.47	0.81	5.56	1.45	4.83	0.89	5.46	1.46	5.80	0.86	5.71
jpeg_compression	0.64	1.32	0.52	1.12	0.50	1.49	0.48	1.30	0.39	0.62	0.34	0.39	0.51	1.10	0.43	1.06	0.47	1.13	0.45	1.00
motion_blur	0.83	2.81	0.68	1.84	0.73	2.99	0.68	2.69	0.80	2.29	0.56	1.08	0.68	1.88	0.56	1.72	0.66	2.07	0.56	1.66
none	0.30	0.88	0.26	0.72	0.25	0.87	0.20	0.73	0.27	0.40	0.17	0.19	0.33	0.84	0.18	0.66	0.30	0.83	0.18	0.66
pixelate	0.34	0.99	0.33	0.84	0.28	0.96	0.28	0.99	0.31	0.47	0.23	0.28	0.35	0.89	0.27	0.78	0.33	0.86	0.27	0.76
shot noise	2.27	5.31	1.29	6.52	1.65	5.57	1.29	6.95	0.72	2.83	0.63	4.40	1.28	4.55	0.82	4.94	1.32	5.63	0.80	4.90
snow	0.89	4 14	0.82	2 55	0.75	4 38	0.76	3 55	0.71	4.89	0.46	1.69	0.70	3 51	0.53	1.75	0.63	4 32	0.51	1 79
zoom_blur	0.98	2.34	0.82	1.74	0.88	2.58	0.89	2.39	0.93	2.16	0.69	1.11	0.93	1.75	0.70	1.60	1.02	1.95	0.71	1.56

Table 5: Summary results for evaluation of all models on the Horse-C dataset. Results are averaged across all five severities and three validation splits of the data. Adaptive batch normalization (adapt) is crucial for attaining good performance compared to fixing the statistics during evaluation (base).

the batch adaptation technique proposed by Schneider et al. [8] during evaluation on Horse-C, allowing a better estimate of model robustness.

On the clean data, using batch norm adaptation yields slightly improved performance for MobileNetV2s on clean withindomain data and deteriorates performance for EfficientNet models. Performance on clean ood. data is improved of all model variants when training from scratch, and improved for MobileNets and ResNets when using pre-trained weights.

We evaluate the normalized errors for the non-adapted model (Base) and after estimating corrected batch normalization statistics (Adapt). The corrected statistics are estimated for each horse identity and corruption as proposed in [8]. We average the normalized metrics across shuffles (and horses as usual). We present the full results for a pre-trained ResNet50 model for all four corruption classes in Tables 7 and 8 and contrast this to the within-domain/out-of-domain evaluation setting in Table 9.

For the ResNet50 model considered in detail, we find that batch normalization helps most for noise and weather corruptions, where we typically found improvements of 60 - 90% and of 30 - 70%, respectively. In contrast, blur corruptions and digital corruptions (apart from contrast, defocus blur) saw more modest improvements. It is notable that some of the corruptions—such as elastic transform or pixelation—likely also impact the ground truth posture.

Batch norm adaptation slightly improves the prediction performance when evaluating on different horse identities, but fails to close the gap between the w.d. and ood. setting. In contrast, batch adaptation considerably improves prediction performance on all considered common corruptions.

In summary, we provide an extensive suite of benchmarks for pose estimation and our experiments suggest that domain shift induced by different individuals is difficult in nature (as it is difficult to fix). This further highlights the importance of benchmarks such as Horse-10. Full results for other model variants are depicted in Table 5 and Table ??. We report average scores on Horse-C all models in the main text.

	Net Type	mobilenet_v2_0.35				mobilenet_v2_1.0			resnet_50			efficientnet-b0				efficientnet-b3					
	Pretrained	False	baca	True	baca	False	basa	True	basa	False	baca	True	baca	False	basa	True	basa	False	baca	True	basa
Corruption	Severity	adapt	base	adapt	base	auapi	base	auapi	base	adapt	base	adapt	Dase	auapi	Dase	auapi	base	auapt	base	auapi	base
1.51	1	0.20	1.02	0.26	0.75	0.25	0.00	0.20	0.70	0.20	0.44	0.16	0.20	0.24	0.02	0.10	0.67	0.20	0.96	0.17	0.60
brightness	2	0.30	1.02	0.26	0.75	0.25	1.20	0.20	0.78	0.29	0.44	0.16	0.20	0.34	1.01	0.18	0.67	0.30	0.86	0.17	0.68
	3	0.33	1.77	0.27	0.87	0.26	1.67	0.20	0.91	0.32	0.99	0.17	0.23	0.39	1.32	0.19	0.71	0.33	1.20	0.18	0.69
	4	0.35	2.20	0.30	0.93	0.27	2.14	0.22	1.02	0.34	1.71	0.18	0.26	0.42	1.67	0.20	0.74	0.34	1.61	0.20	0.70
contrast	5	0.40	2.55	0.35	1.00	0.30	2.38	0.24	1.17	0.39	2.74	0.19	0.32	0.46	2.07	0.22	0.78	0.38	2.25	0.21	0.72
contrast	2	0.31	6.91	0.20	1.38	0.25	7.93	0.20	1.02	0.27	7.86	0.17	0.27	0.33	5.22	0.18	1.03	0.30	7.36	0.18	0.75
	3	0.35	8.63	0.27	3.50	0.27	9.51	0.20	4.67	0.30	9.03	0.17	1.05	0.35	7.88	0.19	2.20	0.32	9.21	0.18	1.70
	4	0.48	9.54	0.29	8.13	0.36	10.22	0.22	8.17	0.38	10.05	0.18	3.90	0.40	9.48	0.20	6.85	0.36	9.60	0.20	5.66
defocus blur	5	0.88	9.77	0.38	9.92	0.68	10.25	0.29	9.03	0.69	0.64	0.22	0.43	0.65	9.68	0.26	9.17	0.56	9.66	0.25	8.10
deroedstordi	2	0.41	1.48	0.39	1.14	0.35	1.39	0.31	1.50	0.39	1.11	0.29	0.40	0.42	1.11	0.28	1.02	0.39	1.05	0.28	0.91
	3	0.65	3.21	0.59	1.92	0.50	3.08	0.52	3.23	0.63	3.23	0.47	0.94	0.58	1.78	0.44	2.31	0.58	2.06	0.44	1.56
	4	1.03	4.59	0.87	2.96	0.76	5.30	0.86	4.97	1.09	5.36	0.82	2.26	0.84	2.69	0.66	3.69	0.92	3.65	0.69	2.75
elastic_transform	1	0.32	0.92	0.29	0.75	0.27	0.90	0.23	0.00	0.30	0.43	0.20	0.22	0.35	0.87	0.22	0.69	0.33	0.86	0.21	0.69
	2	0.34	0.94	0.31	0.77	0.29	0.92	0.26	0.83	0.32	0.46	0.22	0.24	0.36	0.88	0.24	0.72	0.35	0.88	0.24	0.72
	3	0.38	0.96	0.35	0.82	0.31	0.95	0.31	0.91	0.35	0.50	0.25	0.28	0.39	0.91	0.29	0.75	0.37	0.90	0.28	0.76
	4	0.40	0.99	0.39	0.86	0.34	0.99	0.36	0.99	0.37	0.53	0.29	0.32	0.41	0.93	0.33	0.79	0.40	0.92	0.33	0.80
fog	1	0.87	5.11	0.34	0.96	0.65	5.27	0.24	1.11	0.60	5.78	0.20	0.28	0.89	3.46	0.21	0.81	0.71	4.76	0.21	0.76
-	2	1.26	6.42	0.36	1.21	0.94	6.93	0.26	1.57	0.82	7.33	0.22	0.33	1.30	4.89	0.23	0.96	1.03	6.53	0.22	0.86
	3	1.74	7.10	0.41	1.59	1.29	7.82	0.29	2.38	1.18	8.03	0.26	0.48	1.80	5.91	0.26	1.20	1.40	7.45	0.24	1.06
	4	2.17	6.97 7.49	0.43	2.33	1.54	8.29	0.32	2.47	1.24	7.97 8.46	0.28	1.13	2.27	5.82 6.45	0.28	1.21	1.47	7.93	0.26	1.15
frost	1	1.02	4.11	0.46	1.32	0.73	3.81	0.37	1.30	0.65	3.24	0.26	0.37	0.58	2.94	0.34	0.88	0.56	3.52	0.33	0.82
	2	1.98	6.45	0.86	2.99	1.63	6.32	0.75	2.97	1.30	6.39	0.45	1.15	1.08	6.01	0.57	1.86	1.14	7.02	0.55	1.57
	3	2.59	7.56	1.23	4.41	2.30	7.71	1.14	4.50	1.92	7.39	0.65	2.09	1.58	7.56	0.77	3.34	1.63	8.05	0.73	2.73
	5	3.01	7.92	1.63	5.50	2.41	8.32	1.54	5.61	2.00	7.71	0.90	3.07	2.03	8.12	1.01	4.77	2.05	8.47	0.95	3.99
gaussian_noise	1	0.93	3.08	0.51	3.73	0.59	2.99	0.50	4.81	0.34	0.61	0.27	0.68	0.58	1.55	0.41	1.36	0.53	1.73	0.40	1.45
	2	1.61	5.26	0.80	6.21	1.00	5.42	0.82	6.78	0.42	1.12	0.37	2.67	0.85	3.03	0.57	3.09	0.80	3.91	0.56	3.50
	4	3.62	7.15	2.29	8.48	2.99	7.55	2.65	8.75	1.16	5.62	1.02	8.02	2.23	7.35	1.55	8.70	2.39	7.96	1.60	8.54
	5	4.47	7.43	3.40	8.53	4.10	8.06	4.00	8.92	2.28	7.17	1.88	8.77	3.12	8.46	2.62	9.13	3.40	9.14	2.77	9.29
glass_blur	1	0.33	0.98	0.30	0.82	0.27	0.94	0.25	0.85	0.31	0.50	0.22	0.27	0.35	0.89	0.21	0.71	0.33	0.85	0.21	0.71
	2	0.38	1.11	0.37	0.95	0.32	1.11	0.33	1.07	0.35	0.68	0.28	0.35	0.39	0.98	0.28	0.82	0.36	0.93	0.30	0.81
	4	0.65	2.38	0.72	1.79	0.57	2.06	0.83	2.93	0.59	1.84	0.60	1.06	0.58	1.42	0.66	1.61	0.63	1.48	0.74	1.51
	5	1.09	4.16	1.18	2.91	0.88	4.51	1.34	5.04	0.95	4.26	0.93	2.41	0.83	2.33	1.01	3.50	0.98	2.89	1.17	2.80
impulse_noise	1	0.92	2.86	0.56	3.69	0.58	2.80	0.56	4.72	0.38	0.76	0.36	1.43	0.63	1.66	0.33	1.12	0.56	2.08	0.31	1.05
	3	2.10	6.26	1.21	7.45	1.44	6.53	1.42	7.95	0.47	2.69	0.47	5.74	1.18	4.45	0.47	5.56	1.12	4.33 5.87	0.43	5.83
	4	3.18	7.06	2.36	8.42	2.61	7.55	2.67	8.75	0.99	5.38	0.97	7.98	1.90	6.82	1.12	8.60	1.93	7.60	1.06	8.76
	5	4.07	7.38	3.64	8.58	3.68	8.07	3.95	8.93	1.76	6.99	1.68	8.78	2.67	8.07	1.90	9.10	2.84	8.90	1.94	9.38
jpeg_compression	2	0.45	1.05	0.35	0.88	0.37	1.18	0.33	0.96	0.31	0.45	0.24	0.28	0.40	0.92	0.29	0.77	0.37	0.93	0.29	0.77
	3	0.66	1.16	0.51	1.03	0.49	1.25	0.43	1.14	0.37	0.50	0.30	0.33	0.48	1.01	0.39	0.91	0.44	1.07	0.39	0.91
	4	0.75	1.45	0.61	1.25	0.58	1.66	0.56	1.49	0.42	0.70	0.38	0.44	0.57	1.23	0.49	1.17	0.53	1.27	0.53	1.11
motion blur	5	0.80	1.81	0.71	1.51	0.62	2.26	0.70	1.86	0.51	0.95	0.52	0.57	0.63	1.37	0.65	1.62	0.58	1.39	0.68	1.38
motion_oru	2	0.59	1.54	0.34	1.02	0.34	1.49	0.41	1.31	0.37	0.00	0.26	0.35	0.49	1.20	0.28	0.80	0.38	1.16	0.27	0.96
	3	0.73	2.69	0.61	1.47	0.63	2.59	0.60	2.27	0.70	1.90	0.51	0.77	0.64	1.67	0.53	1.41	0.61	1.69	0.53	1.36
	4	1.10	3.94	0.88	2.46	0.96	4.39	0.91	3.90	1.05	3.47	0.74	1.46	0.85	2.46	0.73	2.29	0.84	2.77	0.73	2.17
pixelate	1	0.30	4.75	0.27	0.74	0.25	0.89	0.21	4.97	0.28	4.32 0.41	0.95	0.21	0.33	0.85	0.88	0.69	0.31	0.84	0.88	0.69
Further	2	0.31	0.93	0.28	0.75	0.25	0.89	0.23	0.80	0.28	0.42	0.18	0.22	0.33	0.86	0.23	0.70	0.31	0.84	0.21	0.69
	3	0.33	0.98	0.31	0.82	0.26	0.93	0.26	0.91	0.29	0.46	0.22	0.27	0.35	0.88	0.24	0.73	0.32	0.86	0.25	0.73
	4	0.37	1.04	0.36	0.91	0.30	1.02	0.34	1.13	0.32	0.50	0.26	0.34	0.37	0.91	0.31	0.83	0.35	0.88	0.31	0.80
shot_noise	1	0.69	1.98	0.44	2.43	0.43	1.86	0.40	3.21	0.31	0.50	0.23	0.41	0.46	1.14	0.32	0.97	0.43	1.24	0.30	0.98
	2	1.19	4.19	0.65	5.61	0.68	4.14	0.62	5.99	0.37	0.88	0.31	1.52	0.66	2.10	0.44	1.90	0.60	2.97	0.41	2.15
	3	1.97	5.93	1.02	7.70	1.19	6.31	1.01	7.93	0.50	1.89	0.46	4.28	1.00	4.06	0.63	4.49	0.93	5.95	0.59	4.58
	5	4.12	7.41	2.50	8.44	3.49	7.98	2.56	8.84	1.51	4.08	1.29	8.34	2.46	8.30	1.60	0.55 9.01	2.74	9.55	1.63	7.82 8.98
snow	1	0.49	1.51	0.47	1.09	0.41	1.67	0.41	1.30	0.41	1.21	0.30	0.44	0.45	1.20	0.36	0.87	0.42	1.30	0.33	0.81
	2	0.74	3.60	0.71	2.38	0.63	4.19	0.70	3.51	0.61	5.65	0.43	1.38	0.60	2.60	0.51	1.42	0.55	3.73	0.48	1.34
	3	0.85	3.89	0.77	2.16	0.72	4.32	0.70	3.12	0.66	4.64	0.45	1.27	0.69	2.78	0.51	1.38	0.61	3.71	0.50	1.44
	5	1.15	6.35	1.00	3.92	0.97	6.18	1.03	5.23	0.94	6.85	0.54	2.82	0.83	6.39	0.63	2.93	0.75	7.23	0.62	2.93
zoom_blur	1	0.67	1.69	0.58	1.16	0.59	1.65	0.58	1.50	0.63	1.20	0.45	0.56	0.64	1.28	0.45	1.04	0.67	1.29	0.46	1.05
	2	0.85	2.05	0.78	1.51	0.76	2.13	0.85	2.05	0.80	1.70	0.64	0.87	0.81	1.51	0.63	1.37	0.88	1.58	0.65	1.35
	5 4	0.95	2.59	0.78	1.04	0.84	2.60	0.83	2.52	1.08	2.13	0.66	1.04	0.90	1.72	0.67	1.50	0.99	2.27	0.67	1.45
	5	1.34	2.95	1.01	2.41	1.21	3.48	1.15	3.23	1.24	3.12	0.87	1.68	1.21	2.27	0.93	2.24	1.36	2.70	0.92	2.14
none (Horse-10)	0	0.30	0.88	0.26	0.72	0.25	0.87	0.20	0.73	0.27	0.40	0.17	0.19	0.33	0.84	0.18	0.66	0.30	0.83	0.18	0.66

Table 6: Full result table on Horse-C. All results are averaged across the three validation splits. "none" denotes the uncorrupted Horse-10 dataset.

Noise		Base	Adapt	Δ_{abs}	$\Delta_{\rm rel}$	Blur		Base	Adapt	$\Delta_{\rm abs}$	$\Delta_{\rm rel}$
Corruption	Severity					Corruption	Severity				
Gaussian Noise	1	0.427	0.138	0.289	67.7%	Defocus Blur	1	0.137	0.100	0.037	27.0%
	2	2.187	0.201	1.986	90.8%		2	0.169	0.127	0.042	24.9%
	3	5.556	0.314	5.242	94.3%		3	0.369	0.233	0.136	36.9%
	4	7.843	0.649	7.194	91.7%		4	1.569	0.446	1.123	71.6%
	5	8.894	1.410	7.484	84.1%		5	3.480	0.763	2.717	78.1%
Impulse Noise	1	1.079	0.201	0.878	81.4%	Motion Blur	1	0.213	0.160	0.053	24.9%
	2	3.432	0.276	3.156	92.0%		2	0.290	0.224	0.066	22.8%
	3	5.393	0.360	5.033	93.3%		3	0.340	0.335	0.005	1.5%
	4	7.839	0.663	7.176	91.5%		4	0.864	0.501	0.363	42.0%
	5	8.923	1.339	7.584	85.0%		5	1.596	0.645	0.951	59.6%
Shot Noise	1	0.191	0.114	0.077	40.3%	Zoom Blur	1	0.331	0.288	0.043	13.0%
	2	0.986	0.152	0.834	84.6%		2	0.536	0.436	0.100	18.7%
	3	3.618	0.244	3.374	93.3%		3	0.654	0.467	0.187	28.6%
	4	7.225	0.516	6.709	92.9%		4	0.974	0.620	0.354	36.3%
	5	8.365	0.894	7.471	89.3%		5	1.217	0.640	0.577	47.4%

Table 7: Improvements using batch adaptation on the Horse-C Noise and Blur corruption subsets for a pre-trained ResNet50 model.

Weather		Base	Adapt	$\Delta_{\rm abs}$	$\Delta_{\rm rel}$	Digital		Base	Adapt	$\Delta_{\rm abs}$	$\Delta_{\rm rel}$
Corruption	Severity					Corruption	Severity				
Brightness	1	0.120	0.084	0.036	30.0%	Contrast	1	0.151	0.085	0.066	43.7%
	2	0.127	0.083	0.044	34.6%		2	0.211	0.084	0.127	60.2%
	3	0.141	0.084	0.057	40.4%		3	0.840	0.083	0.757	90.1%
	4	0.165	0.089	0.076	46.1%		4	3.700	0.085	3.615	97.7%
	5	0.205	0.097	0.108	52.7%		5	6.406	0.103	6.303	98.4%
Fog	1	0.156	0.097	0.059	37.8%	Elastic Transform	1	0.121	0.092	0.029	24.0%
	2	0.191	0.107	0.084	44.0%		2	0.127	0.101	0.026	20.5%
	3	0.289	0.126	0.163	56.4%		3	0.139	0.116	0.023	16.5%
	4	0.330	0.137	0.193	58.5%		4	0.154	0.133	0.021	13.6%
	5	0.764	0.176	0.588	77.0%		5	0.175	0.157	0.018	10.3%
Frost	1	0.193	0.125	0.068	35.2%	Jpeg Compression	1	0.136	0.108	0.028	20.6%
	2	0.672	0.249	0.423	62.9%		2	0.152	0.128	0.024	15.8%
	3	1.447	0.393	1.054	72.8%		3	0.170	0.138	0.032	18.8%
	4	1.680	0.449	1.231	73.3%		4	0.216	0.189	0.027	12.5%
	5	2.375	0.573	1.802	75.9%		5	0.305	0.276	0.029	9.5%
Snow	1	0.229	0.155	0.074	32.3%	Pixelate	1	0.117	0.087	0.030	25.6%
	2	0.737	0.252	0.485	65.8%		2	0.117	0.089	0.028	23.9%
	3	0.720	0.270	0.450	62.5%		3	0.125	0.100	0.025	20.0%
	4	1.873	0.386	1.487	79.4%		4	0.142	0.112	0.030	21.1%
	5	2.146	0.348	1.798	83.8%		5	0.156	0.132	0.024	15.4%

Table 8: Improvements using batch adaptation on the Horse-C Weather and Digital corruptions subsets for a pre-trained ResNet50 model.

	Base	Adapt	$\Delta_{\rm abs}$	$\Delta_{\rm rel}$
Identity (wd)	0.115	0.086	0.029	25.2%
Identity (ood)	0.271	0.247	0.024	8.9%

Table 9: Small improvements by using batch adaptation on the identity shift task for a pre-trained ResNet50 model. Note that the o.o.d. performance is still substantially worse (higher normalized error) than the within-domain performance.

8. Inference Speed Benchmarking

We introduced new DeepLabCut variants that can achieve high accuracy but with higher speed than the original ResNet backbone [5]. Here we provide a simple benchmark to document how fast the EfficientNet and MobileNetv2 backbones are (Figure 7). We evaluated the inference speed for one video with 11,178 frames at resolutions 512×512 , 256×256 and 128×128 . We used batch sizes: [1, 2, 4, 16, 32, 128, 256, 512], and ran all models for all 3 (training set shuffles) trained with 50% of the data in a pseudo random order on a NVIDIA Titan RTX. We also updated the inference code from its numpy implementation [6] to TensorFlow, which brings a 2 - 10% gain in speed.



Figure 7: **Speed Benchmarking for MobileNetV2s, ResNets and EfficientNets:** Inference speed for videos of different dimensions for all the architectures. **A-C:** FPS vs. batchsize, with video frame sizes as stated in the title. Three splits are shown for each network. MobileNetV2 gives a more than 2X speed improvement (over ResNet-50) for offline processing and about 40% for batchsize=1 on a Titan RTX GPU.

References

- [1] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv* preprint arXiv:1903.12261, 2019.
- [2] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.
- [3] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*, 2019.
- [4] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE Conference* on Computer Vision and Pattern Recognition, pages 2661–2671, 2019.
- [5] Alexander Mathis, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21(9):1281– 1289, 2018.
- [6] Alexander Mathis and Richard A. Warren. On the inference speed and video-compression robustness of deeplabcut. *BioRxiv*, 2018.
- [7] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. arXiv preprint arXiv:1907.07484, 2019.
- [8] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Removing covariate shift improves robustness against common corruptions. *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*, 2020.