Supplementary Material
for
Learning Pose Specific Representations by Predicting Different Views

Georg Poier David Schinagl Horst Bischof
Institute for Computer Graphics and Vision
Graz University of Technology
Austria

This supplementary material provides additional results supporting the claims of the main paper. First, we provide details about the mentioned adversarial training and compare the respective results quantitatively (§1). Additionally, we compare to a more recent pre-training baseline (§2) and provide the results when omitting the validation set for training on our novel dataset (§3). Finally, we show more examples from our qualitative investigation. Specifically, we show examples for predicted views (§4), investigate which inputs activate the neurons in the latent representation most (§5), and visualize the nearest neighbors in the latent space for our novel dataset (§6).

1. Adversarial training

In the main paper we stated that we also experimented with an additional adversarial loss term. Here, we give some details about the underlying intentions, the implementation and the results of these experiments.

In our work, the objective for the decoder g is based on the reconstruction loss. In this way, the decoder is penalized for any deviation from the second view's exact pixel values. However, more important for our task is the global structure of the image as this is affected crucially by the pose. That is, the decoder spends representational power on estimating exact pixel values, which are of little interest to us.

The adversarial training procedure, on the other hand, as proposed by Goodfellow *et al*. [1] corresponds to a minimax two-player game, where each player is implemented by a neural network. A generator network aims to generate samples from the data distribution and a discriminator network aims to discriminate generated samples from real samples. In this game, the loss for the generator is essentially provided by the discriminator, thus overcoming the need for explicit supervision, *e.g*., from corresponding target images.

Using this idea, we can train the decoder g of our method to match the distribution of real images, but lessen the focus on raw pixel differences. We do so by adding an additional adversarial term to the loss in Eq. (6) of the main paper,

$$\ell_{\text{semi}} = \ell_{\text{u}} + \lambda_{\text{l}}\,\ell_{\text{l}} + \lambda_{\text{a}}\,\ell_{\text{a}}, \tag{1}$$

where λ_{a} is a weighting factor and ℓ_{a} is based on how "real" the discriminator network h thinks a generated sample $\hat{\mathbf{y}}$ is. That is, since this yielded the best results, we define ℓ_{a} inspired by Least Squares GAN [2] as

$$\ell_{\text{a}} = \frac{1}{2}\left(h_j(\hat{\mathbf{y}}^{(j)}) - l_r\right)^2, \tag{2}$$

where l_r is the label value for real samples. The objective for the discriminator, on the other hand, is to push the real samples towards l_r and generated samples towards a distinct label value l_g, *i.e*.,

$$\ell_{\text{h}} = \frac{1}{2}\left(h_j(\mathbf{x}^{(j)}) - l_r\right)^2 + \frac{1}{2}\left(h_j(\hat{\mathbf{y}}^{(j)}) - l_g\right)^2. \tag{3}$$

In our case we set $l_r = 1$ and $l_g = 0$. For the adversarial part, we adopted the training procedure and discriminator architecture of DCGAN [6].

The decoder g needs to output an image closely resembling the image of the second view since the reconstruction loss is still part of its objective. Nevertheless, g is enforced to focus more on the overall structure of the image through the loss term provided by the discriminator.

Additionally, the discriminator can be improved, and thus provide better feedback, by conditioning it on additional input, as has been described, *e.g*., in [3]. We can condition it on the input from the first view and/or, in case of semi-supervised training, the pose. In the semi-supervised case, the provided pose information is the estimated pose for generated samples and the annotated pose for real samples.

n	100			1,000			10,000			43,640		
Metric (see main paper)	ME	FS80	JS80	ME	FS80	JS80	ME	FS80	JS80	ME	FS80	JS80
Semi-superv.	**29.12**	0.31	**0.63**	**22.96**	**0.44**	**0.71**	21.49	0.47	0.73	20.70	0.48	**0.74**
Semi-superv. & Adversarial	29.52	**0.32**	**0.63**	23.32	0.41	0.70	**20.67**	**0.48**	**0.74**	**20.23**	**0.49**	**0.74**

Table 1: **Comparison with additional adversarial loss on NYU-CS.** Comparison of semi-supervised training with (*Semi-superv. & Adversarial*) and without an additional adversarial loss (*Semi-superv.*) for different metrics on the NYU-CS dataset. Note, that these results differ slightly from the results in the main paper since this comparison was based on a previous version of the code base (*e.g.*, earlier PyTorch version, *etc*.). Best results in boldface.

n	189		
Metric (see main paper)	ME	FS80	JS80
Semi-superv.	**27.27**	0.30	**0.66**
Semi-superv. & Adversarial	27.79	**0.32**	0.65

Table 2: **Comparison with additional adversarial loss on MV-hands.** Comparison of semi-supervised training with (*Semi-superv. & Adversarial*) and without (*Semi-superv.*) an additional adversarial loss for different metrics on the MV-hands dataset. Best results in boldface.

In Tab. 1 and 2 we compare the results when using the additional adversarial term to the results of semi-supervised training without the adversarial term. We see that adversarial training can improve the results slightly for larger numbers of labeled samples n, but not in cases where only a small number of samples is labeled. Moreover, note that we obtained the presented results for the adversarial training by tuning hyper-parameters separately for different n and taking the best results. We found that, for different n, different conditioning types and settings for λ_{a} worked best. While for a small number of labeled samples, $n = 100$, conditioning the discriminator solely on the input and a small weight for the adversarial term ($\lambda_{\text{a}} = 0.01$) yielded best results, for larger n, conditioning on the pose and a larger weight $\lambda_{\text{a}} = 0.1$ had a positive impact on the results.

The results point out that training our method with an additional adversarial loss term bears potential to improve results. However, it appears that a sufficient amount of labeled samples is necessary so that the discriminator can exploit the pose conditioning and provide improved feedback for training the decoder. Additionally, it requires significant tuning to achieve improved performance. The semi-supervised approach, as described in the main paper, on the other hand, does not require such extensive hyper parameter tuning for different amounts of labeled data but yields consistently improved performance due to the pose specific latent representation.

Number of samples	100	1,000	10,000	43,640
Context Encoders [5]	53.4	53.4	53.3	53.8
Autoencoder	48.0	47.2	47.3	47.1
PreView (Ours)	**33.4**	**29.6**	**29.0**	**29.0**

Table 3: **Comparison of different pre-training methods on the NYU-CS dataset.** Mean joint error for learning a linear layer on top of the frozen latent representation with different numbers of labeled samples n. Best results in boldface.

2. Additional pre-training baseline

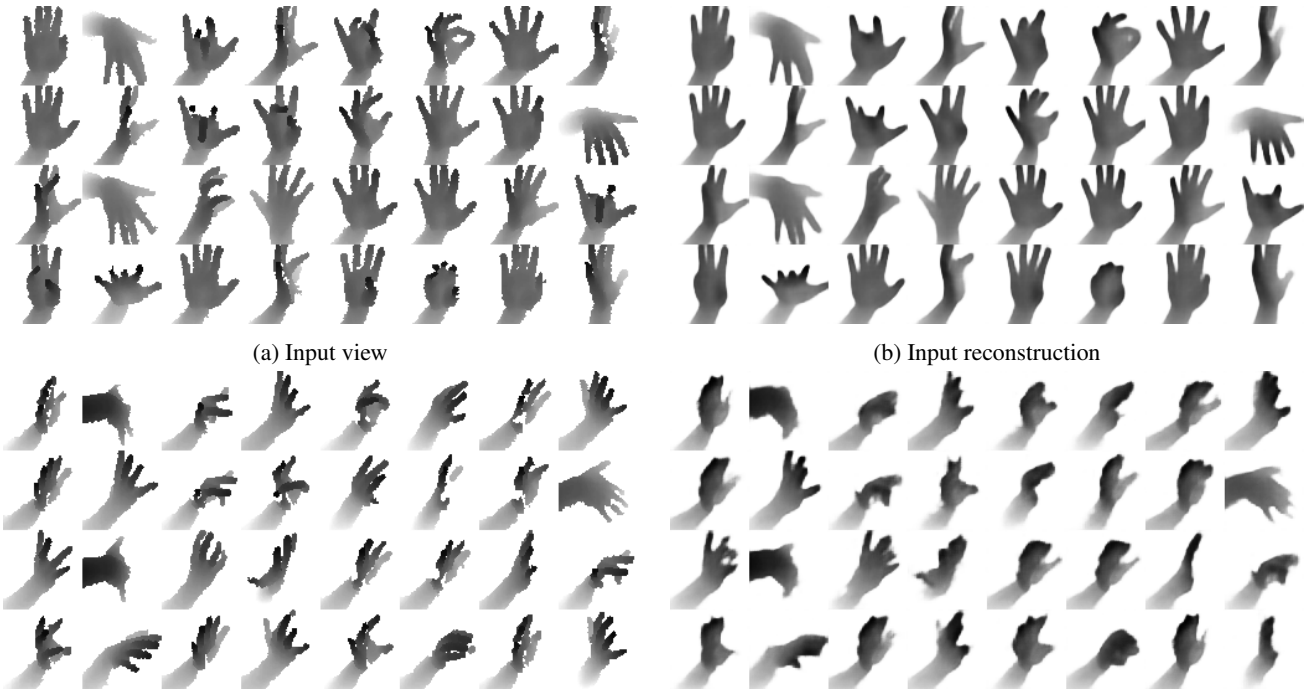
One of the reviewers suggested that a more recent pre-training baseline would make the paper stronger. In particular, a comparison to Context Encoders [5] was suggested. Context Encoders are trained to do inpainting. That is, large random contiguous parts of the input image are removed for training and the model should learn to inpaint the missing regions based on the context. The idea is that the model needs to learn to recognize the objects in the context in order to accomplish this task.

In Tab. 3 we compare the latent representations – pre-trained by different methods – based on their predictability for the pose (see main paper for details). The results show that the representations pre-trained using Context Encoders [5] are even less predictive for the pose than the representations learned by autoencoders.

One issue of the Context Encoder baseline is the "domain gap" between training and testing as has been discussed in [7]. That is, at training time parts of the input are missing, while the model is applied to full images at test time. Moreover, we believe that the main idea of Context Encoders does not really apply to pose estimation, where one part of the object does not necessarily contain pose information for other parts.

3. Results for Model Selection on MV-hands

In the main paper we compare the results on the MV-hands dataset when training on all data available for train-



(a) Input view (b) Input reconstruction

(c) Different view (d) Prediction for different view

Figure 1: **Input reconstruction vs. different view prediction.** Examples for generated views from the NYU validation set. Input view (a), reconstructions generated by the autoencoder (b), images from a different view (c), and the corresponding predictions from our method (d). Images from (a)-(d) with same grid index are corresponding. Visually, the autoencoder's input reconstructions resemble the input more closely than the predictions of our method match the different view. However, the latent representation learned by our method is much more predictive for the pose (*c.f.*, results in main paper).

n	189		
Metric (see main paper)	ME	FS80	JS80
DeepPrior++ [4]	36.56	0.17	0.54
Supervised	30.13	0.25	0.62
Semi-superv. Autoenc.	28.51	**0.30**	**0.64**
Semi-superv. PreView (Ours)	**28.38**	**0.30**	**0.64**

Table 4: **Comparison to the state-of-the-art and ablation experiments.** Results for different metrics on the MV-hands dataset. Best results in boldface.

ing, *i.e.*, using the 100 samples from the validation set for training, and omitting any model selection like early stopping or other hyperparameter optimizations. For comparison, in Tab. 4 we provide the results when employing the 100 validation samples for model selection during training, *i.e.*, early stopping.

4. Predicted views

In Fig. 1 we compare output images of input reconstruction (autoencoder) and view prediction (PreView), respectively. We can observe that the reconstructions of the input are cleaner (*e.g.* for the fingers) than the predictions for different views. Obviously, reconstructing the input is an easier task than predicting a different view. More importantly, input reconstruction can be performed without knowledge about the pose, as the results in the main paper suggest. Predicting different views, on the other hand, is a harder task but reveals pose information. In Fig. 2 we show view prediction examples on the MV-hands dataset. Altogether, these results underline that our latent representation is predictive for the different view as well as the pose.

5. Neuron activations

To investigate what each neuron in the latent space has learned, we search for the samples from the validation set, which activate a single neuron most. Fig. 3 shows these samples for each neuron. We find that many of the neurons are activated most for very specific poses. That is, the samples, which activate a neuron most, show similar poses.

(a) Different view (b) Prediction for different view

Figure 2: **View prediction examples on MV-hands data.** Target view (a), *i.e.*, ground truth images of the different view and the corresponding predictions from our method (b). Images with same grid index are corresponding.

6. Nearest neighbors

Similar to what we show in the main paper for the NYU dataset, in Fig. 4, we show nearest neighbors for the MV-hands dataset. That is, given a query image from the validation set, we find the closest samples from the training set according to the Euclidean distance in the learned latent representation space. Again, the nearest neighbors in the latent space exhibit very similar poses.

References

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In *Proc. NIPS*, 2014.

[2] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *Proc. ICCV*, 2017.

[3] M. Mirza and S. Osindero. Conditional generative adversarial nets. *ArXiv e-prints*, abs/1411.1784, 2014.

[4] M. Oberweger and V. Lepetit. DeepPrior++: Improving fast and accurate 3d hand pose estimation. In *Proc. ICCV Workshops*, 2017.

[5] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Proc. CVPR*, 2016.

[6] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proc. ICLR*, 2016.

[7] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proc. CVPR*, 2017.

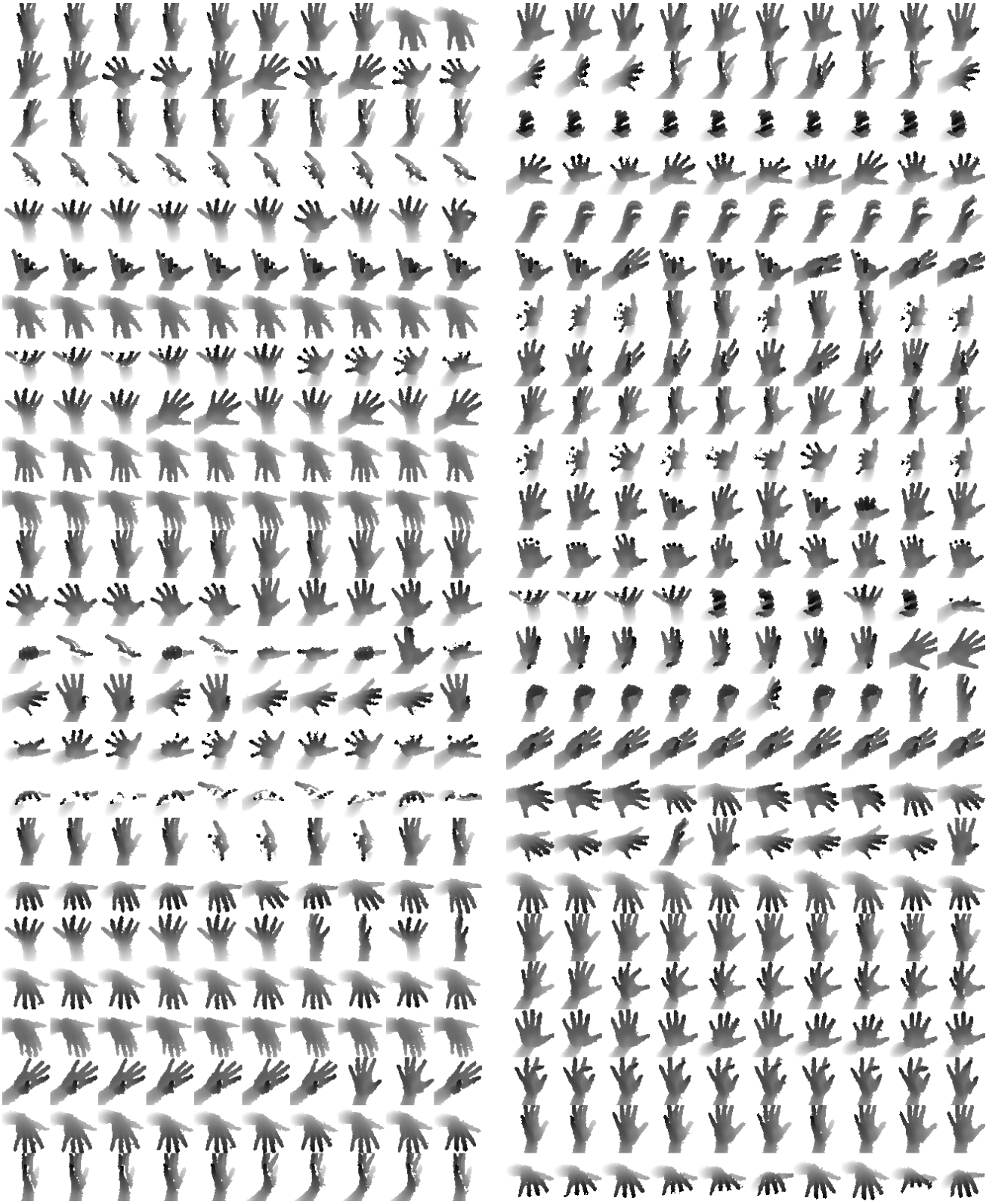


Figure 3: **Most activating samples.** Each row on the left and right side shows the ten samples from the validation set, which activate the same neuron in the learned latent representation most. Note, that we randomly perturbed detections to verify the robustness of our method. Hence, sometimes parts of the hand are cut off in the crops.

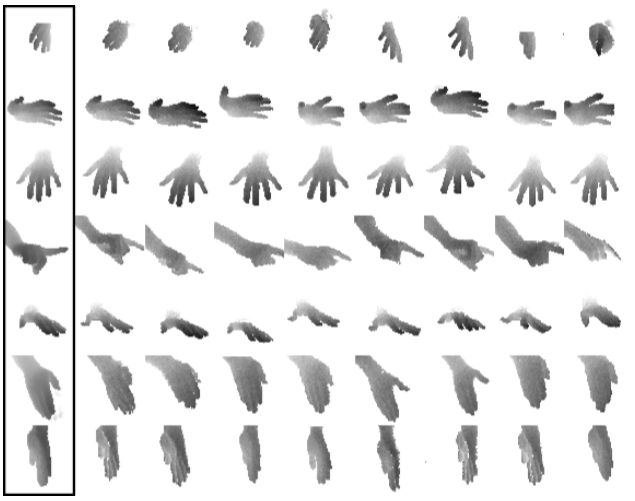


Figure 4: **Nearest neighbors in latent space.** Nearest neighbors from training set for query samples from validation set. Query images are shown in the marked, leftmost column, the remaining eight columns are the respective nearest neighbors.