Domain Adaptation for Head Pose Estimation Using Relative Pose Consistency

Felix Kuhnke and Jörn Ostermann, *Fellow, IEEE*

Abstract—Head pose estimation plays a vital role in biometric systems related to facial and human behavior analysis. Typically, neural networks are trained on head pose datasets. Unfortunately, manual or sensor-based annotation of head pose is impractical. A solution is synthetic training data generated from 3D face models, which can provide an infinite number of perfect labels. However, computer generated images only provide an approximation of real-world images, leading to a performance gap between training and application domain. Therefore, there is a need for strategies that allow simultaneous learning on labeled synthetic data and unlabeled real-world data to overcome the domain gap. In this work we propose relative pose consistency, a semi-supervised learning strategy for head pose estimation based on consistency regularization. Consistency regularization enforces consistent network predictions under random image augmentations, including pose-preserving and pose-altering augmentations. We propose a strategy to exploit the relative pose introduced by pose-altering augmentations between augmented image pairs, to allow the network to benefit from relative pose labels during training on unlabeled data. We evaluate our approach in a domain-adaptation scenario and in a commonly used cross-dataset scenario. Furthermore, we reproduce related works to enforce consistent evaluation protocols and show that for both scenarios we outperform SOTA.

Index Terms—Head Pose Estimation, Domain Adaptation, Consistency Regularization, Deep Learning

1 Introduction

Head pose estimation (HPE) describes the problem of predicting the orientation (head pose) of the human head. Head pose is a key factor in many biometric systems related to facial analysis and human behavior analysis. Accurate estimation of head pose can bring many benefits.

In driver assistance systems [1] HPE can be used for automatic assessment of the focus of attention. Head pose is used for visually assessing human-object and human-human interaction [2]. Further, it is the starting point for many gaze estimation methods [3]. In facial recognition systems head pose is important for pre-processing [4] or during training for data augmentation and to obtain pose-invariant recognition models [5]. Similar use cases apply for facial expression recognition [6] as pose-invariance is a desired property for facial analysis systems.

Motivated by the many possible applications, there has been a lot of progress in this field of research, most recently through deep learning methods. Collecting the required training data is still a challenging task for several reasons. Manual annotation is a problem, because humans cannot accurately annotate a 3D head pose from a 2D image. This has led to the creation of head pose datasets using devices like depth sensors and 3D head scans [1], [7], or special tracking equipment attached to the head [8], [9], [10]. However, with these recording setups, it is cumbersome and costly to reach a high diversity in subjects, environments and poses. Therefore it is still an open challenge to acquire suitable training data.

A solution is to use synthetic (rendered, computer generated) face images to provide inexpensive and virtually unlimited quantities of perfectly labeled data. Several methods train on synthetic [11], [12], [13], [14], [15], [16] or synthetically extended (warped) images (e.g., [17], [18] on 300W-LP dataset [19]). Unfortunately, learning-based approaches trained only on synthetic data (source domain) tend to perform poorly on real-world data (target domain) compared to methods trained on real-world data. This can be explained by the difference between domains (domain gap). In [16] we addressed this issue for HPE by introducing a method for domain adaptation (DA). DA methods typically use unlabeled data from a target domain during training, to overcome the domain gap. However, the performance of methods trained on synthetic head pose data is still inferior to methods trained on real-world head pose data.

In this work, an extended version of [20], our goal is to improve the performance of HPE on real-world data using only labels from a synthetic dataset in combination with an unlabeled real-world dataset. In [16] an adversarial training approach based on domain adversarial neural networks [21] is used to force the extraction of domain-invariant features. In contrast, we propose to tackle the problem using consistency regularization and relative pose labels.

Consistency regularization is a semi-supervised learning (SSL) technique. Semi-supervised learning utilizes labeled and unlabeled data simultaneously during training. This property was exploited successfully for domain adaptation using consistency regularization in [22]. Consistency regularization forces network outputs for the same input under different perturbations to be consistent. For visual tasks, these perturbations are typically implemented as various image augmentations, e.g., spatial transforms. However, head poses are not invariant to spatial image transforms, like flipping and rotation. If the ground truth pose is known, the pose label can be adjusted, however, the ground truth

- *The authors are with the Institut für Informationsverarbeitung, Leibniz University Hannover, 30167 Hannover, Germany.*
E-mail: kuhnke@tnt.uni-hannover.de.

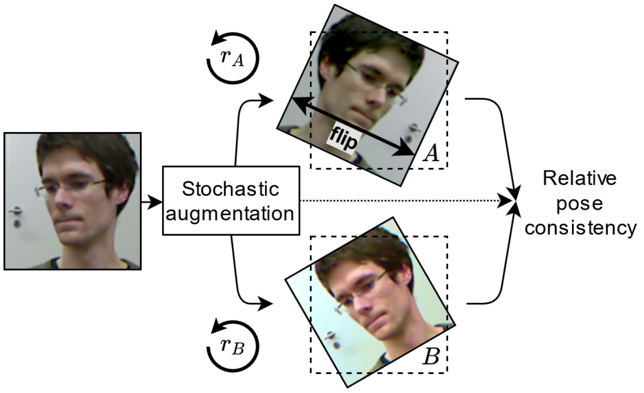

Fig. 1. From an image with unknown head pose (picture from [7]) two different augmented versions A and B are created. Augmentations can be color distortions, blurring, etc., but also pose-altering transforms like rotations (r_A, r_B) and flipping. In addition to a supervised loss, our method allows to train a network on unlabeled data with relative pose consistency between A and B. Relative pose consistency provides an unsupervised loss to enforce consistent predictions under different augmentations, and also to enforce predictions that comply with a relative pose.

pose is unknown for our target-domain data. In [20] and this extended work, we therefore propose to take advantage of relative pose.

The relative pose, which we store in a relative pose label, is the pose difference between two realizations of the same input (see Fig. 1 for an example). Recalling that training with consistency regularization requires different realizations of the same input, we implement relative pose label in a consistency training framework (see Figure. 2 for an overview of our method). This extends the consistency supervision from static augmentations to relative pose labels. As a consequence, the network is trained not only to make consistent predictions, but also predictions that comply with the relative poses. Our consistency-enforcing method does not require absolute pose information and can therefore be used with unlabeled data samples in semi-supervised or domain-adaption scenarios. We show the effectiveness of our approach on the popular BIWI Kinect Head Pose estimation benchmark [7].
Our main contributions are as follows:

- We show, for the first time that consistency regularization can be used for pose regression problems.
- We propose relative pose consistency, a novel extension to consistency regularization.
- We present an improved training framework compared to [20]
- We uncover effects of preprocessing on performance during reproduction of related work.
- We improve state-of-the-art results for two challenging HPE scenarios (domain-adaptation and cross-dataset).

2 Related Work

2.1 Head Pose Estimation

In recent years, traditional approaches based on facial landmarks and 3D face models are mostly superseded by deep learning methods. These methods estimate the pose directly from an image. In addition to images, different modalities such as depth images [7] or temporal information [14] can be used. In this review we will focus only on deep-learning methods for HPE from a single RGB image.

The first convolutional neural networks (CNN) to directly regress head pose from an image are presented by Anh et al. [23] and Patacchiola and Cangelosia [24]. Recent work proposes variations of loss functions and network architectures. Ruiz et al. [17] combine a regression loss with binned pose classification, by assigning continuous pose to discrete pose categories (bins). Shao et al. [25] use a similar combined loss but also evaluate the effect of adjusting the margin around the face image that is fed into a CNN. Similarly, Lathuilière et al. [26] evaluate various factors of deep regression like hyperparameter selection or image preprocessing in the context of head pose estimation. Wang et al. [15] present a coarse-to-fine approach, where head pose is coarsely classified in bins and later refined by regression. An attention based network structure for HPE is proposed by Yang et al. [18]. Their goal is to extract a set of representative features by learning a fine-grained structure mapping before a feature aggregation step. Zhou et al. [27] extend the work of [17] to full-range HPE by proposing a wrapped loss that allows training with the full range of yaw angles (-180°, 180°]. They further show that a small model, EfficientNet-B0 [28], can reach SOTA HPE performance. Hempel et al. [29] propose a continuous 6D rotation representation for full-range HPE and a geodesic distance-based loss. In contrast Gu et al. [14] present an approach for temporal prediction of facial features. They propose to use a recurrent neural network (RNN) on top of a VGG16 network [30] for joint estimation and tracking of head pose in videos. The above methods can be seen as orthogonal to our approach, because we are not trying to improve supervised performance with new losses or network architectures for HPE. For simplicity and comparability we focus on Mean Squared Error (MSE) loss and ResNet [31] network architecture. Nevertheless, our method can be applied to other loss functions or network architectures as well.

Another approach to HPE is multi-task learning [32], [33], [34], [35], [36]. In this setting multiple tasks like HPE, landmark detection, age estimation, etc., are solved simultaneously. A benefit of multiple tasks is that multiple data sources can be used for training, which considerably increases the amount of training data. In contrast, our method does not use additional labeled tasks or datasets.

An interesting unsupervised approach is presented by Mustikovela et al. [37]. In their work, a viewpoint estimation network is trained purely via self-supervision with an analysis-by-synthesis framework using a network similar to HoloGAN [38]. Similar to our work, they enforce flip consistency by applying a flip consistency loss. In contrast, their loss forces images synthesized from a flipped latent code to be consistent. Our loss forces the predicted labels from flipped images to be consistent.

Lastly, it is a common approach to use synthetic face datasets from 3D models for HPE [11], [12], [13], [14], [15], [16], [20]. This has the advantage of learning from a high amount of diverse images with perfect labels. To date, [14] and [13] are publicly available datasets. These related works train on synthetic or mixed datasets and evaluate on real-world datasets, however, most of them do not explore any domain adaptation or semi-supervised techniques to overcome the domain gap. This was only explored in our previous works [16], [20].

In [16], we improve HPE for an unlabeled target dataset by enforcing a network to extract domain-invariant features using a domain discriminator and an adversarial training loss. To account for an only partially-shared label space, we apply a weighted resampling of the source domain to filter out samples with poses that are outside the target domain. To find poses outside the target domain, a distance between source and target domain samples is estimated by a network trained on the source domain. The method has two drawbacks. First, in addition to the pose estimation network, a domain-discriminator network needs to be trained simultaneously during training. This leads to additional computations, hyperparameters and complexity. Second, the resampling of training data assumes the availability of a pose distance metric to find suitable training samples. It is not guaranteed that a network trained only on the source domain provides an accurate distance metric to select appropriate training samples.

In [20], we tackle the same problem but choose a completely different approach to avoid the aforementioned drawbacks. One goal of [20] is to find a method that does handle partially-shared label spaces like [16] but does not need an explicit filtering of training data. As a result, the approach presented in [20] does not require to resample the source data. Furthermore, the approach does not need an additional discriminator network with adversarial training. These factors and improved performance compared to [16] make the proposed solution an exciting new solution for HPE in domain-adaptation scenarios. In this iteration of [20] we further refine the solution and provide a more detailed evaluation of the method. We introduce a modification to the training framework that increases the performance relative to [20]. In addition, we apply our method to a cross-dataset task to verify that our method can be applied in different scenarios. In an attempt to enable meaningful performance comparisons to related work, we also investigate the reproducibility of head pose estimation experiments in the context of face detection (see Section 4.5).

2.2 Consistency Regularization

Consistency-enforcing methods provide state-of-the-art performance for semi-supervised learning. During training, consistent network predictions for unlabeled data under input and network perturbations are enforced. Although one can find many terms and variants like self-ensembling, consistency regularization, self-training, temporal ensembling, or pseudo-labels, the core principle of enforcing consistent outputs is similar. Consistency-enforcing methods have also been successfully applied to domain-adaption scenarios, where the unlabeled data is from another domain. While first used as semi-supervised methods, these principles are now popular for unsupervised pre-training of neural networks and paved the way for modern contrastive (self-supervised) methods like SimCLR [39], MoCo [40] and BYOL [41].

Laine and Aila [42] proposed two self-ensembling methods, Π-Model and temporal ensembling. Both methods enforce consistent network predictions for the same input under different stochastic input augmentations and network perturbations. In this case, dropout was used to provide network perturbations. The Π-Model randomly augments the same input twice during an iteration and forces consistent predictions. In contrast to the Π-Model, temporal ensembling forces network predictions over multiple previous training epochs to be consistent to the current prediction. Self-training and training with pseudo-labels, e.g., [43], [44], can be seen as a variant of temporal-ensembling. The Mean Teacher method by Tervainen et al. [45] adapted this idea but instead of reusing previous predictions, they added a teacher network that is an average of previous network weights. The teacher network's predictions and the current model's (named student) predictions are forced to be consistent. French et al. [22] applies the Mean Teacher method to domain adaptation and proposes modifications to improve DA performance. As we are the first to introduce consistency regularization to head pose estimation, we opt for simplicity and base our work on the Π-Model.

3 Method

Semi-supervised learning is typically used to learn from a large dataset, which is only partially labeled. We take up this idea for domain adaptation to learn from labeled synthetic images (source domain) and unlabeled real-world images (target domain). On the one hand, synthetic data provides perfect labels for a wide variety of poses. On the other, it only provides an approximation of real-world image features. Real data provides real-world features but lacks annotation quantity and quality. Combining them in a training scheme, where both datasets can be used simultaneously, is a promising way to improve performance on real-world images.

This concept is not limited to scenarios with synthetic and real-world data. A domain gap can decrease performance if the source data distribution is different from the target data distribution. For image datasets, different recording settings like lighting, camera, or subjects can already provide enough differences to cause reduced performance across domains. Our method can be applied in cross-dataset scenarios as well.

We will first introduce the required notations and baseline supervised learning. Then, we will describe the consistency regularization framework. Subsequently, we will describe the concept of relative pose labels and how these are embedded into the training framework. Finally, we discuss how we avoid degenerate solutions with consistency regularization and the effects of batch normalization while training with two dataset.

In a semi-supervised or domain-adaptation scenario, data is available from the labeled source domain $\mathcal{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$, where n_s is the number of data samples $x_i^s \in X_s$ and associated labels $y_i^s \in Y_s$. For head pose estimation, x is an image of a head and y is a vector of the three corresponding Euler angles of the head. We are interested in utilizing the unlabeled target data $\mathcal{D}_t = \{(x_i^t)\}_{i=1}^{n_t}$, which includes only samples but no labels.

A network f can be trained using the source data (X_s and Y_s) and a supervised loss. For head pose estimation the

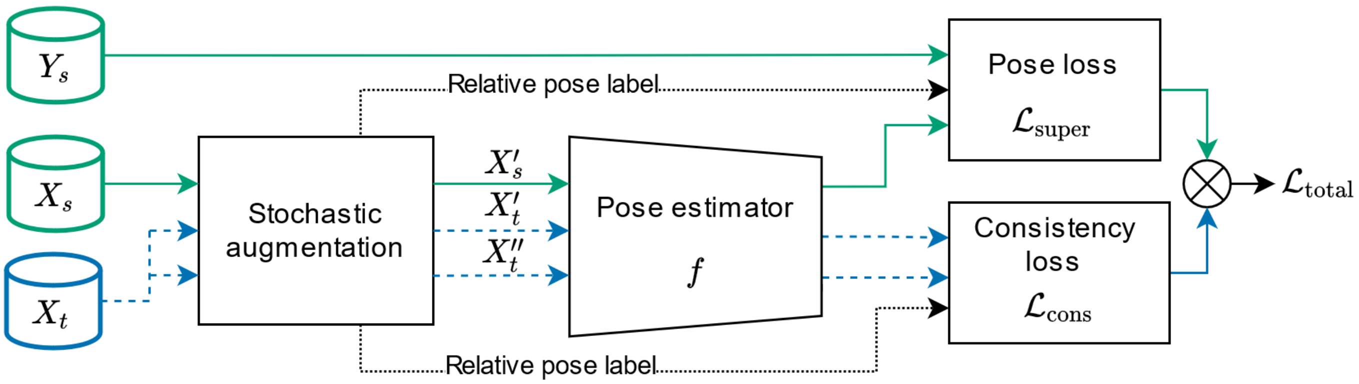

Fig. 2. Proposed framework for relative pose consistency regularized head pose estimation. Labeled data (in green) from the source domain and unlabeled data (in blue) from the target domain can both be used in a semi-supervised fashion. Input images X_s and X_t are perturbed by stochastic augmentations. The stochastic augmentation module can also change the pose of the input images by rotation and flipping. This information is stored in the relative pose label. Source data follows the supervised path (green) to train the pose estimator f. Target data is copied before the stochastic augmentation module which creates two different augmented versions of the target input. Note that even though the ground truth pose is unknown for X_t the relative pose between the augmented versions $X_t{}'$ and $X_t{}''$ can differ and is stored in a relative pose label. The relative pose label and predictions are fed into the consistency loss. The consistency loss provides supervision from consistency and relative pose labels. f is trained jointly on both losses.

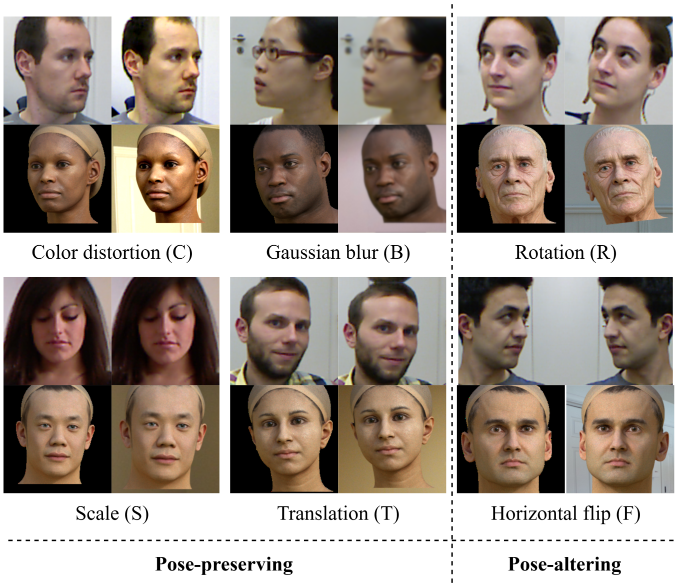

Fig. 3. Illustrations of the studied image augmentations. Each augmentation transforms the input image with random transformation parameters. The left images in each square show the inputs and the right images randomly transformed outputs. For synthetic images, a random background is added. Top row images from [7] and bottom row images from [14].

supervised loss is a measure of how similar two poses are and typically the Mean Squared Error is used as pose loss

$$\ell_{\text{mse}}(\hat{y}, y) = \|\hat{y} - y\|^2, \tag{1}$$

with the predicted Euler angles $\hat{y} = f(x)$, and the ground truth angles y.

3.1 Consistency Regularization Framework

Stochastic input perturbations are a central aspect of consistency-based models. In practice standard image augmentations like blurring, translation and scaling (usually implemented as random cropping), horizontal flipping, rotation, and color distortions provide appropriate image perturbations (see Fig. 3).

Given a sample x we create two randomly perturbed (augmented) inputs x' and x'' which are fed into the network f to produce predictions $f(x')$ and $f(x'')$.

A consistency loss $\mathcal{L}_{\text{cons}}$ enforces that both predictions are similar. This consistency loss is typically the Mean Squared Error or KL divergence [46]. We formulate our total loss

$$\mathcal{L}_{\text{total}} = \underbrace{\sum_{(x,y)\in\mathcal{D}_s} \ell_{\text{mse}}(f(x'), y)}_{\mathcal{L}_{\text{super}}} + \lambda \underbrace{\sum_{x\in\mathcal{D}_t} \ell_{\text{mse}}(f(x'), f(x''))}_{\mathcal{L}_{\text{cons}}}, \tag{2}$$

with λ controlling the relative effect of the consistency term in the overall loss.

The same stochastic perturbations are applied to both source and target images. Note that in SSL the consistency loss is typically applied to samples from both $\mathcal{D}_s$ and $\mathcal{D}_t$ [42], [45], [46]. Following [22], who use consistency regularization for domain adaptation, we apply the consistency loss only to samples from the target domain $\mathcal{D}_t$. Our framework is shown in Figure 2.

Unfortunately, *flipping and rotation* will *change the ground truth label* of a source-domain sample *and* produce target-domain inputs that *break the consistency assumption* that x' and x'' share the same label. We therefore need to distinguish between pose-preserving and pose-altering augmentations and need to redefine our loss functions for pose-altering augmentations. As shown in Fig. 3, **pose-preserving augmentations** are random color distortion, blurring, translation, and scaling and **pose-altering augmentations** are flipping and rotation. The required changes for pose-altering augmentations will be described in the next section.

3.2 Relative Pose Consistency

Pose-altering augmentations change the head pose. Knowing the spatial transformation and the true pose, an augmented image can be relabeled. However, this is not possible if the true pose is unknown. We create a new consistency loss based on the relative pose between augmented samples to benefit from pose-altering augmentations on our real-world unlabeled target data.

We will first give a short recap on pose representation and then provide the interdependence of image rotation and flipping to the orientation change of the head pose and required adaptions to the loss functions. Both augmentations require[1] that the pose is stored in Euler angles (Tait–Bryan angles) that describe intrinsic rotations around Z-Y′-X″. These are known as: roll, yaw, and pitch. This means that the rotation is performed by three successive rotations around the Z, Y′ and X″ axis. Recall that for intrinsic rotations the first rotation around Z will create a new coordinate system from which Y′ will be used for the second rotation and so on. For this representation a rotation around Z can be carried out independently from Y′ and X″ rotations. That means that any image rotation will result in an additive rotation term to the roll label.

Augmenting an image with (unknown) **rotation** r with two random rotations r_A and r_B would result in images A and B with rotations $r + r_A$ and $r + r_B$, respectively. One can easily see that the difference in rotation between the two augmented images is $r_B - r_A$, which is the relative pose difference between the images. To account for this difference we can change the consistency loss for the roll angle to:

$$\ell_{\text{mse}}(f(A)_{\text{roll}} + (r_B - r_A), f(B)_{\text{roll}}), \tag{3}$$

where $f(A)_{\text{roll}}$ and $f(B)_{\text{roll}}$ describe the predicted rotations. To use rotation augmentations for the source domain, one can simply replace r_B with 0 and $f(B)$ with the true rotation label r.

Flipping is performed by negating the yaw and roll angles for a flipped image. In the consistency loss, if we encounter an image with unknown pose, we can negate the predictions of the yaw and roll angles. A full example, showing all the angles, with A being flipped and random rotations would result in

$$\ell_{\text{mse}}\left(\begin{bmatrix} f(A)_{\text{pitch}} \\ f(A)_{\text{yaw}} \\ f(A)_{\text{roll}} \end{bmatrix} \odot \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ r_B - r_A \end{bmatrix}, f(B)\right), \tag{4}$$

where $\odot$ is the element-wise product. This complete example is also illustrated in Figure 1. Cases where B, or A and B are flipped are handled with negating B′s, or A′s and B′s yaw and roll angles, respectively. Note that the information provided by a relative pose only influences yaw and roll angles, as the pitch angle is untouched by rotation and flipping.

3.3 Avoiding Degenerate Solutions

Several works report difficulties when training with consistency regularization. In contrast to previous works, we apply consistency regularization to a regression problem and therefore use a different loss combination. For this reason, instead of class logits, we regularize the predicted pose angles. In the following we will address these difficulties and how we dealt with them.

The first difficulty is the selection of λ. [42] found that the network can get easily stuck in a degenerate solution if the unsupervised loss component ($\mathcal{L}_{\text{cons}}$) is too high in the beginning of the training. As a solution, they ramp-up λ from 0 to 1 during training. The same procedure was also adopted in [45]. In contrast, French et al. [22] replaced the ramp-up with a confidence threshold. They utilized the predicted class activations as probabilities and the loss of all samples with activations below the threshold is set to 0.

In our case we found that high λ values usually yield degenerate solutions, regardless of ramp-up or not. Our explanation is that in order to minimize the consistency loss the network can learn to output only a constant. However, we found a good indication on how to set λ comes from the supervised loss. As a simple rule, the regularization feedback should not be stronger than the supervised loss. Preliminary experiments showed that consistency training is quite robust and λ values in the range [0.1, 0.4] converge to similar performing networks. For $\lambda > 0.5$ the consistency loss became larger than the supervised loss and the overall performance decreased for both, source and target data.

Another issue with consistency regularization arises if the labels of the source and target domain do not come from the same underlying distribution [47]. This is ignored in many works, because it is assumed that the unlabeled data contains the same class distribution as the labeled data. As described in [16], this assumption usually does not hold for regression tasks like HPE. For classification, [22] introduced a class balance loss term that forces the network's mean class predictions to be uniform. This helped to avoid a degeneration to the most dominant class.

Following this approach, we introduce a weighted relative consistency regularization for HPE. To enforce a more evenly distributed feedback of the consistency loss, we reweight the consistency loss based on the pose predictions. Poses that are found often in a batch should be weighted down, whereas rarely appearing poses should be weighted up. In most natural image collections of faces, the poses are usually distributed around the pose that is facing the camera. The same holds true for most head pose datasets. Therefore, assuming a normal distribution of poses in a batch, we formulate our weighting:

$$w_p = 1 - e^{\left(\frac{p-\mu_P}{\sigma_P}\right)^2}, \tag{5}$$

where w_p is the weight given to a pose angle p (pitch, yaw, roll) and μ_P and σ_P are the mean and standard deviation of all $p \in P$ in a batch. In particular, we apply this weight to all angles independently. To keep λ constant between experiments, we rescale w_p with

$$\frac{Batchsize}{\sum w_p}, \tag{6}$$

so that the overall weight in a batch sums to one. We compare the results for weighted and unweighted predictions in our experiments in Section 4.

[1] For simplicity we describe our method for Z-Y′-X″ rotations, but other representations such as Z-X′-Y″ will also work.

Both λ and re-weighting are associated with the same underlying problem: an effective setting of the regularization strength. Although we have made two proposals, we think that uncertainty or curriculum approaches like [48] are paths worth looking into for future improvements.

3.4 Batch Normalization During Cross-Domain Training

Batch normalization (BN) [49], among other normalization methods, is a popular technique for training deep neural networks. BN normalizes activations in a network across the mini batch. Each feature is normalized by mean and standard deviation of the feature in the mini batch during training. For inference/evaluation typically a running estimate of the features mean and standard deviation is used. When working with multiple datasets (possibly from different domains) it is not realistic to assume that features in these datasets share the same statistics. Adapting feature statistics using batch normalization to adapt to different domains has been investigated, e.g., [50], [51]. Regardless, here we will discuss the effects of BN in our framework.

Using BN on the target domain implicitly affects training and evaluation performance. Similar to previous works like [16], [20], [22], we process mini batches of source and target data sequentially. The other possible method is to combine the data in every batch, e.g., half and half. We found that sequential processing provides stable training. With sequential processing, BN uses domain-dependent statistics during training but mixed (running) statistics during evaluation. However, this implicit mixing of domain feature statistics can have beneficial or detrimental effects on the evaluation performance for each domain. Assuming that domain statistics are very different, the mean statistics of two domains, which are typically used during evaluation, might not provide good results. Aside from mixed statistics, it is possible to use only target-domain statistics [51] during target evaluation. In our experiments, we found that mixed or target-only statistics decrease performance on the target dataset. The effect gets worse the more the distribution of the label space, and thus the distribution of features, differs between source and target.

We therefore propose to use BN in a way that does not use target-domain statistics during training or evaluation at all. If data from the target domain is fed trough the network, the running statistics of the source domain are used. The same applies for evaluation. This way, computation of BN statistics is completely independent of target-domain data. The advantage of this method is that we avoid any deteriorating effects from BN. The only drawback of this method is decreased training stability. In our experiments this is alleviated by using gradient clipping during training. We evaluate the performance of using ”no target BN” in our experiments in Section 4.4.

4 Experiments

In the following, we will analyze the performance of our method. First, we will give a description of the used datasets and the implementation details. We then compare our results with related work. In the following section, our ablation study, we show the effects of different components of our method. Finally, we conduct a series of experiments to reproduce related work. In doing so, we gain insight into some factors that influence the reproducibility of results and also obtain results with which we can compare our work more fairly.

TABLE 1
Augmentation parameters. Translation parameters are given relative to the image size and are applied independently for x and y translation. Values in ranges are sampled uniformly.

	Augmentation	Parameter	Probability

C	Color distortion	brightness=0.4, contrast=0.4
		saturation=0.4, hue=0.1
			0.8

B	Gaussian blur	$\sigma \in$ [0.2, 2]	0.5
S	Scale	[0.9, 1.1]	1.0
T	Translation	[-0.1, 0.1]	1.0
R	Rotation	[-20°, 20°]	1.0
F	Flip		0.5

4.1 Data

To validate our method we use revised datasets SynHead++ and Biwi+ proposed by [16]. These datasets are extensions of the popular face pose datasets Biwi Kinect Head Pose Database (Biwi) [7] and NVIDIA Synthetic Head Dataset (SynHead) [14]. For both datasets, [16] provides labels in Z-Y'-X”-angle representation and face bounding boxes. SynHead was artificially extended to include more poses, so that SynHead++ is a superset of Biwi+ in regard to pose labels. Here, we give a brief overview of the datasets.

Biwi+ is used as real-world, target-domain dataset. It contains 24 sequences of 20 different subjects recorded with a kinect sensor. **SynHead++** is used as synthetic, source-domain dataset. It contains images of 10 different rendered 3D head models. The total number of images is 15678 for Biwi+ and 653910 for SynHead++. All images are cropped to the given bounding boxes and scaled to 224 x 224 pixels. Exemplary images and illustrative augmentations are shown in Figure 3.

To make our work more comparable with previous works, we also perform experiments on the popular **300W-LP** [19] dataset. 300W-LP uses the images from 300W [52], a dataset that combined multiple datasets containing faces in unconstrained, ”in-the-wild” conditions. These images have been re-annotated with facial landmarks. Landmarks and image features are used by [19] to fit a 3D morphable model (3DMM) to the face images. With the image and fitted 3DMM they synthesize (render) face images with new head poses. 300W-LP includes 5488 real images and 55737 synthesized images. In addition all images are flipped to create a dataset with 122450 images. We processed the data the same way as Biwi+ in [16].

4.2 Implementation Details

For all our experiments, the pose estimator f is ResNet18 as provided by PyTorch [24] with last linear layer being replaced by a new linear layer with 512 inputs and 3 outputs for Euler angle estimation. This is consistent to previous work [16], [20].

We use different combinations of augmentations throughout the experiments with parameters provided in Table 1. If an augmentation would produce a roll angle $> \pm 89^\circ$, the rotation is performed in the opposite direction. We used the code[2] and parameters from [40] for color distortions and Gaussian blur. Similar to [22], we process mini batches of source and target data sequentially. Depending on the experiment, batch normalization is turned on or off during training on target domain batches. If it is turned off, we apply the running statistics that are computed during training on source domain batches to the target domain batches and enable gradient clipping of the complete network with maximum L2 norm of 5. For all experiments trained on SynHead++, we use stochastic gradient descent with momentum 0.9, Nesterov, a batch size of 84, and a learning rate set to 10^{-3}. For all experiments trained on 300W-LP, we use Adam [53], a batch size of 84, and a learning rate set to 10^{-4}.

For our "supervised only" baselines f is initialized with the default PyTorch pretrained ResNet18. The learning rate is ramped-up to warm start the optimization. During baseline training λ is set to 0. The baselines are trained for 35000 iterations which is equivalent to ≈5 epochs for SynHead++ or ≈24 epochs for 300W-LP.

For all consistency regularization experiments we fine tune a baseline model. To make the comparisons fair for all runs, we select one (supervised only) baseline model that is trained with all augmentations (full). From the 10 runs, the baseline model that performs most similar to the average performance is selected. For the consistency regularization experiments, all models are fine-tuned for 16000 iterations. λ is ramped-up to 0.2, to avoid deterioration from too strong regularization. For all experiments the performance at the end of training is reported, i.e. no early stopping is used. The performance for Biwi+ is reported without any augmentations.

We timed the execution of our approach and found that for one image it takes less than 3 ms on a consumer GPU (NVIDIA GTX 1080 TI). For the methods we reproduced (see Section 4.5) we found similar or slightly higher times but all of them would be suitable for real-time processing at 30 fps.

4.3 Head Pose Estimation Results

In this section, we compare our results to groups of similar experimental settings that we clustered from the related work. In Table 2 we measure the mean absolute error for every angle and the overall MAE (mean absolute error) of them. We report the mean and standard deviation of these values over 10 runs using different random seeds. Please note that this is not the standard deviation of the pose errors, but the standard deviation of the mean errors over all runs.

In the upper part of Table 2 we collected the results of related work. The lower part shows the results of reproduced related work (Section 2.1) and our work, all using Biwi+ as a common test set. Most work trains with real-world images and typically focuses on improving head pose estimation by improved network structures or supervised loss functions. In contrast, our main goal is to learn from synthetic images and unlabeled real-world images using a semi-supervised method. In addition to the differences in models and learning strategies, it is not always straightforward to compare head pose estimation results, even if the results are reported for the same dataset. For example, the evaluation protocol and processing of the Biwi dataset is not universally the same for all works. A deeper analysis of a factor that makes the comparability of results problematic is given later in Section 4.5. However, even though not always directly comparable, we think it can be valuable to discuss our results in a broader context of related work.

Cross-Validation. This section of Table 2 shows the performance for methods trained and evaluated on different splits of the same dataset. Compared to our results and other experimental settings, higher performance is likely explained by having no domain gap between training and test set. Both sets are from the relatively homogeneous Biwi dataset, which is recorded under lab conditions. The performance in this section is also dependent on the training/test split. Non surprisingly, a random split offers the best results, as splitting by persons introduces a small domain gap.

Self-Supervised shows that learning head pose can even be accomplished self-supervised. No pose labels are used during training. To gain pose predictions, a linear regressor is trained on 100 random test set samples to map network outputs to pose labels. While results are not on par with our or recent work, it shows the potential of self-supervised pose estimation. Training with our proposed relative pose labels can be seen as a self-supervised approach.

Cross-Dataset. Sometimes called inter-domain or cross-domain evaluation, these sections show results where the training set and test set are taken from different datasets. Most commonly, 300W-LP [19], a dataset created from real-world images is used for training. As 300W-LP uses real-images, the domain gap to Biwi should be small, compared to using fully synthetic images from SynHead++. However, the distribution of poses in 300W-LP does not match the one found in Biwi, which makes it challenging to gain high performance when training on 300W-LP only.

The section in the upper part of Table 2 shows results from related work. Similar to us, [15] uses rendered synthetic images but also a part of the Biwi dataset to train a HPE model, which makes a fair comparison difficult. Only [27] (WHENet and WHENet-V) publish better results than ours. WHENet-V, uses additional data from the (real-world) Panoptic Studio dataset [54] to better match the pose distribution of Biwi.

In an effort to create comparable results, we reproduced related work on Biwi+ in the section provided in the lower part of Table 2. More details can be found in Section 4.5. Using the proposed batch processing scheme, our method weighted relative consistency regularization (RCRw) trained on 300W-LP outperforms all reproduced work evaluated on Biwi+.

Domain Adaptation shows results for synthetic-to-real (SynHead++ to Biwi+) domain adaptation. Our proposed method RCRw, outperforms the partial domain adaptation method proposed in [16]. Although we get worse pitch performance, we achieve better yaw, roll, and average performance. For roll error, the improvement to [16] is nearly one degree. We also report our results from [20]. In comparison to [20], improved handling of batch normalization boosted

[2] https://github.com/facebookresearch/moco

TABLE 2
Head pose estimation results for experiments tested on variants of the Biwi dataset [7]. Variants: * random split (86%/14%^{1}, 80%/20%^{2}), † sequence split (16/8^{1},21/3^{2}), × no split, processed by the respective authors, + no split, processed by [16]. Experimental results are grouped in blocks describing the use of data during training and testing. We report mean and standard deviation of the average absolute angular errors in degree and mean absolute error (MAE) over all angles for 10 training runs. RCRw = Weighted Relative Pose Consistency Regularization.

Experiment	Method	Network	Training Set	Test Set	MAE	Pitch	Yaw	Roll
Cross-Validation	Gu [14]	VGG16	Biwi†^{1}	Biwi†^{1}	3.66	4.03	3.91	3.03
	Lathuilière [12]	VGG16	Biwi†^{2}	Biwi†^{2}	3.62	4.68	3.12	3.07
	Yang (FSA) [18]	Custom	Biwi†^{1}	Biwi†^{1}	3.60	4.29	2.89	3.60
	Ruiz (Hopenet) [17]	ResNet50	Biwi†^{1}	Biwi†^{1}	3.23	3.39	3.29	3.00
	Anh [23]	Custom	Biwi*^{1}	Biwi*^{1}	2.93	3.4	2.8	2.6
Self-Supervised	Mustikovela (SSV) [37]	Custom	300W-LP	Biwi†^{1}	6.8	9.4	6.9	4.2
	Mustikovela (SSV) [37]		300W-LP&Biwi†^{1}	Biwi†^{1}	5.8	8.5	4.9	4.2
Cross-Dataset	Shao [25]	ResNet50	300W-LP	Biwi×	5.99	7.25	4.59	6.15
	Ruiz (Hopenet) [17]	ResNet50	300W-LP	Biwi×	4.90	6.61	4.81	3.27
	Wang [15]	Custom	[15]&Biwi*^{2}	Biwi*^{2}	4.84	5.48	4.76	4.29
	Yang (FSA) [18]	Custom	300W-LP	Biwi×	4.00	4.96	4.27	2.76
	Zhou (WHENet) [27]	Eff.Net-B0	300W-LP	Biwi×	3.81	4.39	3.99	3.06
	Zhou (WHENet-V) [27]	Eff.Net-B0	300W-LP& [54]	Biwi×	3.48	4.10	3.60	2.73
Cross-Dataset with Consist. Reg.	Zhou (WHENet) reproduced	Eff.Net-B0	300W-LP	Biwi+	6.42	7.26	7.74	4.26
	Yang (FSA) reproduced	Custom	300W-LP	Biwi+	5.15	**6.10**	6.20	3.15
	Ruiz (Hopenet) reproduced	ResNet50	300W-LP	Biwi+	4.83	6.50	4.89	3.13
	RCRw [20]	ResNet18	300W-LP	Biwi+	5.58±.08	9.25±.21	4.52±.06	2.97±.03
	RCRw (proposed)	ResNet18	300W-LP	Biwi+	**4.54**±.06	6.42±.12	**4.44**±.09	**2.75**±.03
Domain Adaptation	Kuhnke (PADACO) [16]	ResNet18	SynHead++	Biwi+	4.13	**4.51**	4.11	3.78
	RCRw [20]	ResNet18	SynHead++	Biwi+	4.01±.03	5.54±.13	**3.78**±.10	**2.71**±.03
	RCRw (proposed)	ResNet18	SynHead++	Biwi+	**3.85**±.02	4.79±.07	3.92±.02	2.85±.04

our average and pitch performance but also slightly deteriorated pitch and roll accuracy. Nevertheless, the additional gain is so significant that this deterioration can be accepted. A detailed analysis of the effects of different components of our method can be found in Section 4.4.

Looking at Table 2, pitch error is higher than roll or yaw for all reported results. We suspect that pitch estimation seems to be a harder problem. For our experiments, pitch estimation has gained the least from our method. Presumably, pitch estimation can not benefit from the relative pose labels, as the pitch angle is constant for all our augmentations. In contrast, roll benefits the most from our relative pose labels. Probably for this reason, we outperform almost all other work in terms of roll error.

4.4 Ablation Studies

In this section we analyze the different components of our framework in relation to head pose estimation performance. Even though, we are mainly interested in the synthetic-to-real scenario, we include experiments with 300W-LP, as this database is highly used in the HPE community.

We first establish a number of baselines in the first section, **"Supervised Only"**, of Table 3. These experiments are trained only with the supervised loss on a source dataset and evaluated on the Biwi+ dataset. The experiments only differ in the use of different augmentation combinations and the selected source dataset.

Experiments trained on SynHead++ reveal the effects of augmentations during training on synthetic data. It is clearly visible that augmentations help to improve target performance. Interestingly, rotation and flipping augmentations only give small or no performance boost. This can be explained by the structure of the datasets. SynHead++ already covers all poses found in the test set Biwi+. Any additional pose augmentations to SynHead++ will likely make the training set more different to the test set. Color, scale, translation and blur augmentations create images that might look more similar to the real-world test set. However, augmentations are not sufficient to reach the performance of related work on the Biwi dataset, which suggests that additional methods might increase performance. These findings demonstrate two of our key assumptions. First, training on a synthetic image dataset does not provide automatically good results for a real-world image dataset. Secondly, the tested augmentations improve the performance but are not sufficient to force the network to learn features that generalize well to real-world images.

Experiments trained on 300W-LP show that augmentation also increase the performance if an "in-the-wild" training set such as 300W-LP is used. The performance gain from augmentation is smaller than for SynHead++. We think this is explained by a smaller domain gap between 300W-LP and Biwi+, because both contain real-world images. Comparing the absolute performance on Biwi+, training with 300W-LP and full augmentations is inferior to training on SynHead++ with full augmentations. This again can be explained by the structure of the datasets. 300W-LP does not include all poses of Biwi+. So even if we have very similar (real-world) image features, some poses can simply not be learned from 300W-LP as they can only be found in Biwi+ or SynHead++.

Consistency regularization uses our proposed consistency framework. Again different augmentations are evaluated. Using only pose-preserving augmentations for consistency regularization (CR), already improves the results to baselines. Adding rotation and flipping further increases the performance. Best performance for yaw and roll is gained

TABLE 3
Ablation on different baseline and consistency regularization settings for our method. Head pose estimation results are for experiments tested on the Biwi+ dataset [16]. Experimental results are grouped in blocks describing the training method. We report mean and standard deviation of the average absolute angular errors in degree and mean absolute error (MAE) over all angles for 10 training runs. Best results in bold. Augmentations for experiments: rotation (R), flip (F), color distortion (C), scaling (S), translation (T), Gaussian blur (B). CR = Consistency Regularization, RCR = Relative Pose Consistency Regularization, w = Weighted loss.

Experiment	Method	Training Set	Target BN	MAE	Pitch	Yaw	Roll
Supervised Only	Baseline (no aug.)	SynHead++		5.36±.28	5.99±.35	5.60±.60	4.42±.35
	Baseline C·S·T·B			4.72±.11	5.71±.18	4.70±.15	3.75±.08
	Baseline C·S·T·B·F			**4.65**±.09	**5.65**±.16	**4.64**±.17	3.65±.09
	Baseline C·S·T·B·R			4.78±.13	5.71±.20	4.81±.24	3.81±.13
	Baseline C·S·T·B·R·F			**4.65**±.08	5.68±.16	4.69±.16	**3.59**±.06
	Baseline (no aug.)	300W-LP		5.26±.08	7.23±.18	**4.95**±.15	3.58±.10
	Baseline C·S·T·B·R·F			**4.90**±.13	**6.26**±.20	5.00±.22	**3.45**±.12
Consistency Regularization	CR C·S·T·B (no rot/flip)	SynHead++	Yes	4.27±.04	5.67±.06	4.00±.05	3.13±.03
	CR C·S·T·B (no rot/flip)		No	4.31±.04	5.50±.08	4.13±.04	3.31±.02
	RCR C·S·T·B·R (no flip)	SynHead++	Yes	4.14±.04	5.47±.09	4.13±.05	2.84±.03
	RCR C·S·T·B·R (no flip)		No	**3.90**±.02	**4.90**±.04	3.96±.03	2.84±.02
	RCR C·S·T·B·R·F (full)	SynHead++	Yes	4.15±.03	5.87±.07	**3.80**±.04	**2.78**±.02
	RCR C·S·T·B·R·F (full)		No	3.98±.03	5.04±.06	4.12±.04	2.79±.03
Weighted Consistency Regularization	CRw C·S·T·B (no rot/flip)	SynHead++	Yes	4.11±.03	5.24±.07	4.00±.05	3.08±.02
	CRw C·S·T·B (no rot/flip)		No	4.12±.02	5.13±.07	4.04±.04	3.20±.03
	RCRw C·S·T·B·R (no flip)	SynHead++	Yes	4.03±.04	5.22±.07	4.06±.04	2.80±.03
	RCRw C·S·T·B·R (no flip)		No	**3.85**±.02	**4.79**±.07	3.92±.02	2.85±.04
	RCRw C·S·T·B·R·F (full)	SynHead++	Yes	4.01±.03	5.54±.13	**3.78**±.10	**2.71**±.03
	RCRw C·S·T·B·R·F (full)		No	3.89±.02	4.81±.05	4.10±.03	2.76±.02
	RCRw C·S·T·B·R·F (full)	300W-LP	Yes	5.58±.08	9.25±.21	4.52±.06	2.97±.03
	RCRw C·S·T·B·R·F (full)		No	**4.54**±.06	**6.42**±.12	**4.44**±.09	**2.75**±.03

when using the full augmentation scheme with relative pose (RCR). However, best overall performance is gained when using all augmentations without flipping. In the RCR setting, using ”no target BN” boosts the overall performance and is beneficial for pitch estimation. Although a general increase in performance can be observed, yaw and roll estimation is slightly deteriorated by using ”no target BN”. We theorize that this is caused by the distribution differences between Biwi+ and SynHead++. Compared to pitch, the yaw and roll distributions are more similar between the datasets, and therefore the estimation might even benefit from an implicit alignment by a shared batch norm.

Weighted consistency regularization includes the proposed weighting of angles during training.

Compared to CR and RCR, regardless of augmentation scheme, weighting gives a slight performance boost. The observations for applying ”no target BN” are the same as for RCR. Again, best overall performance is gained when using all augmentations without flipping and ”no target BN”.

We also applied the RCRw with full augmentations to 300W-LP. Interestingly, the gain for ”no target BN” is very high. We conjecture that this effect is created by a high difference in pose labels between 300W-LP and Biwi+. The different poses also create very different feature distributions, which causes detrimental effects if batch normalization statistics are averaged from source and target dataset. Possibly for the same reason, the overall gain of using consistency regularization is lower compared to the SynHead++ experiments.

4.5 Influence of Face Detection on Performance

A phenomenon evident from the upper part of Table 2 is that the community has not settled on a standardized evaluation protocol for head pose estimation. Even though all of these works report results on the same dataset, the dataset preprocessing and splitting is often different. In an attempt to improve comparability of results, we try to recreate results on Biwi+ for related work. Luckily, code and models for the works of Ruiz [17], Yang [18] and Zhou [27] are available online. However, we can not simply feed the Biwi+ data to the models, because every method uses different image preprocessing. As suggested in previous work [25], [26], how the face is cropped before pose estimation is a crucial factor for performance. Furthermore, the models are trained to recognize poses that come from crops similar to those used during training, which might depend on the used face detector. We therefore conduct a small study on preprocessing and ask two questions. What role plays face detection for HPE performance? How important is a predictor-specific bounding box, i.e., is a bounding box of one detector replaceable by another? While there are certainly more differences (see 3 for more), here we focus on face detection and transferability of detections (bounding boxes), two steps that have not been analyzed before.

In Figure 4 we visualize the bounding boxes that are used for cropping in different methods. Note that our methods use the crops from Biwi+ [16], which use a CNN based face detector from Dlib [56] and manual labeling. Hopenet [17] uses Dockerface [55] a Faster R-CNN based detector inspired by [59]. FSA [18] uses MTCNN [57] and WHENet [27] utilizes a YOLOv3 [58] model trained to detect faces. Using the aforementioned detectors on the Biwi dataset, we found that face detection provides three sources of errors: finding the wrong person in the image; finding no person in the image; finding a bad crop of the face, e.g., not all parts of the face are visible. All of these cases happen on

TABLE 4
Reproduction of head pose estimation results for methods trained on 300W-LP [19] evaluated on differently processed versions of Biwi [7]. The differences are the number of used images and the bounding boxes used for cropping the images. We report the average absolute angular errors in degree and mean absolute error (MAE). Results selected for comparison in Table 2 are in bold. cleaned = In case of multiple detections, correct face is selected. transformed = Box scale and translation is changed to be similar to the associated work. unchanged = Boxes are used directly.

Method	Processing	Face Detector	Used Biwi Images	MAE	Pitch	Yaw	Roll
Ruiz (Hopenet) [17]	Reported results	Dockerface [55]		4.89	6.61	4.81	3.27
Ruiz (Hopenet)	Dockerface boxes, cleaned	Dockerface	15666	4.83	6.53	4.72	3.22
Ruiz (Hopenet)	Biwi+ boxes, transformed	Dlib [56], manual	15678 (14739, 939)	**4.83**	**6.49**	**4.89**	**3.13**
Ruiz (Hopenet)	Biwi+ boxes, unchanged	Dlib, manual	15678	5.47	6.36	6.39	3.66
Yang (FSA) [18]	Reported results	MTCNN [57]		4.00	4.96	4.27	2.76
Yang (FSA)	MTCNN boxes, cleaned like [18]	MTCNN	13219	4.00	4.96	4.26	2.76
Yang (FSA)	Biwi+ boxes, transformed, MTCNN subset	Dlib, manual	13219	4.07	5.00	4.39	2.82
Yang (FSA)	Biwi+ boxes, unchanged, MTCNN subset	Dlib, manual	13218	3.91	4.78	4.29	2.66
Yang (FSA)	Biwi+ boxes, transformed	Dlib, manual	15678	5.50	6.64	6.37	3.48
Yang (FSA)	Biwi+ boxes, unchanged	Dlib, manual	15678	**5.15**	**6.10**	**6.20**	**3.15**
Zhou (WHENet) [27]	Reported results	YOLOv3 [58]		3.81	4.39	3.99	3.06
Zhou (WHENet)	YOLOv3 boxes, cleaned	YOLOv3	15636	7.62	8.77	9.80	4.30
Zhou (WHENet)	Biwi+ boxes, transformed, YOLOv3 subset	Dlib, manual	15636	7.22	8.14	9.38	4.14
Zhou (WHENet)	Biwi+ boxes, unchanged	Dlib, manual	15678	**6.41**	**7.21**	**8.16**	**3.84**

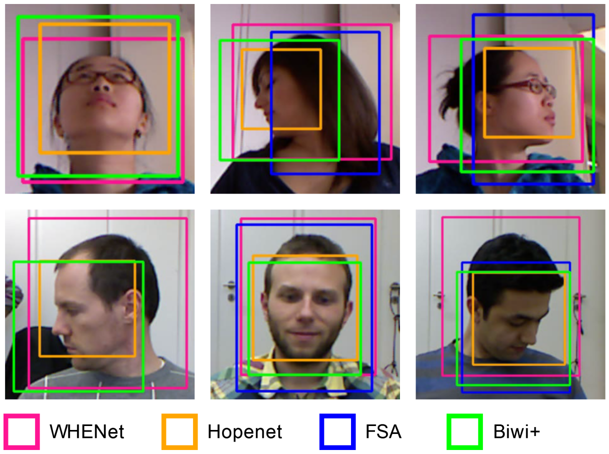

Fig. 4. Visualization of different bounding boxes used for cropping images of the Biwi dataset [7] before pose estimation. Depending on method and image, no face crop is found (pictures in left column), possibly resulting in skipping these images during evaluation.

Biwi and the latter two cases can be seen in Figure 4. To avoid detecting the wrong person, detections are usually selected by size or by distance to image center. We did this in our reproduction experiments for all evaluated face detectors except for MTCNN, where the selection procedure is provided by the authors code [18].

In Table 4 we show our results of recreating related work. ”Used Biwi Images” reports the number of images with face detections for each face detector, which is also the the number of images used for our evaluation. One can see that this number ranges from 13219 (MTCNN) to 15678 (Dlib and manual detections). From our experiments it seems likely that only images with detected faces are used for evaluation in related work. This makes reported results on Biwi variants highly incomparable. Not only the number of compared images differ, e.g., ≈16% of images omitted for FSA, but especially faces with large head rotations and therefore difficult samples are usually the ones not found by face detectors. Similar results[3] can be found for methods like [29] that utilize FSA's data processing code.

As a consequence, we tried to use the boxes provided by Biwi+ for all methods, to gain equally sized test sets. As it would be unfair to use a crop that is dissimilar from the one a network is trained on, we calculated a transformation from Biwi+ box to the box associated with the method in question. The transformation minimizes the mean difference in box translation and box scale between the intersection of detections found by two detectors. Although straightforward, this transformation works very well for FSA and Hopenet, as performance is nearly identical to the original boxes. This shows evidence that the predicted face position from a face detector has only small influence on the final performance. That is, we can interchange face detectors and get similar results. Surprisingly, Biwi+ boxes work even better than the ones created by MTCNN for FSA. Unfortunately, we could not recreate the results reported for WHENet. Another conclusion of Table 4 is the fact that changing the face detection method can change the evaluation dataset and therefore the performance from state of the art to irrelevant.

In summary, the role of face detector seems less important as long as the same subset of images is used during evaluation. Furthermore, this section showed that a much stricter evaluation protocol should be used to compare head pose estimation results. There are even additional differences to the Biwi+ evaluation protocol, which we neglected here, a full analysis is available on 3. Even though, discussed in literature [16], [25], [26] as an important factor, by using a non-standardized preprocessing, many works indirectly assume preprocessing to have negligible influence for performance comparison. On the contrary, we believe the contribution of preprocessing might even conceal the effects of novel methods on observed performances. This calls for a more unified evaluation protocol. We make the code to recreate the results of Table 4 publicly available[3]. There we also investigate additional factors, such as different rotation representations, and their effects on performance.

[3] https://github.com/kuhnkeF/headposeplus

5 Conclusions and Future Work

We propose relative pose consistency, a new approach to improve deep head pose estimation performance with semi-supervised learning. Our method allows pose-altering augmentations, like rotation, to be incorporated into a consistency regularization framework. In addition, we introduce two extensions, a weighting scheme and a batch processing scheme, to improve performance. We evaluate our method in two scenarios: domain-adaptation and cross-dataset evaluation. For domain-adaptation, our method outperforms previous work by 7%. To enable direct comparisons for the cross-dataset evaluation, we reproduced results from related work. Thereby, we uncover that inconsistent preprocessing seems to prevent comparability of head pose estimation results and suspect that the effects of data preprocessing can conceal methodological contributions. This raises the question of whether stricter evaluation protocols are needed. In this context, our method improves the state of the art for the cross-dataset evaluation by 6%. Furthermore, our approach trained only on labels from synthetic data outperforms previous work trained on real-world images (300W-LP) by 20%. We thereby demonstrate that state-of-the-art performance on real-world images can be achieved when using only labels from synthetic training data. Ultimately, however, there is still a gap to methods trained with real-world datasets that are very similar to the target domain.

In future work, our framework could also be combined with other methods. Among the many consistency regularization frameworks we based our work on the simple Π-Model [42]. Therefore, another direction could be to extend our framework to consistency frameworks like self-ensembling with teacher student models [45]. Lastly, the concept of relative pose consistency could be applied to other pose estimation tasks such as hand or body pose estimation, or scale and translation estimation methods like [60]. An example of another pose estimation application has been proposed for gaze estimation in [61].

References

[1] G. Borghi, M. Venturelli, R. Vezzani, and R. Cucchiara, "Poseidon: Face-from-depth for driver pose estimation," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2017, pp. 4661–4670.

[2] A. Recasens, A. Khosla, C. Vondrick, and A. Torralba, "Where are they looking?" in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015.

[3] X. Zhang, Y. Sugano, and A. Bulling, "Revisiting data normalization for appearance-based gaze estimation," in *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*, 2018, pp. 1–9.

[4] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker, "Towards large-pose face frontalization in the wild," in *Proceedings of the IEEE Inter. Conf. on computer vision*, 2017, pp. 3990–3999.

[5] I. Masi, S. Rawls, G. Medioni, and P. Natarajan, "Pose-aware face recognition in the wild," in *Proceedings of the IEEE Conf. on computer vision and pattern recognition*, 2016, pp. 4838–4846.

[6] S. Li and W. Deng, "Deep facial expression recognition: A survey," *IEEE transactions on affective computing*, 2020.

[7] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool, "Random forests for real time 3d face analysis," *Int. Journal of Computer Vision*, vol. 101, no. 3, pp. 437–458, February 2013.

[8] I. Martinikorena, R. Cabeza, A. Villanueva, and S. Porta, "Introducing i2head database," in *Proceedings of the 7th Workshop on Pervasive Eye Tracking and Mobile Eye-Based Interaction*, 2018, pp. 1–7.

[9] A. Schwarz, M. Haurilet, M. Martinez, and R. Stiefelhagen, "Driveahead-a large-scale driver head pose dataset," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 1–10.

[10] M. Selim, A. Firintepe, A. Pagani, and D. Stricker, "Autopose: Large-scale automotive driver head pose and gaze dataset with deep head orientation baseline," in *Inter. Conf. on Computer Vision Theory and Applications (VISAPP)*, 2020.

[11] X. Liu, W. Liang, Y. Wang, S. Li, and M. Pei, "3d head pose estimation with convolutional neural network trained on synthetic images," in *IEEE Int. Conf. on Image Processing*, 2016, pp. 1289–1293.

[12] S. Lathuilière, R. Juge, P. Mesejo, R. Muñoz-Salinas, and R. Horaud, "Deep mixture of linear inverse regressions applied to head-pose estimation," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2017, pp. 4817–4825.

[13] A. Larumbe, M. Ariz, J. J. Bengoechea, R. Segura, R. Cabeza, and A. Villanueva, "Improved strategies for hpe employing learning-by-synthesis approaches," in *2017 IEEE Inter. Conf. on Computer Vision Workshops (ICCVW)*, 2017, pp. 1545–1554.

[14] J. Gu, X. Yang, S. De Mello, and J. Kautz, "Dynamic facial analysis: From bayesian filtering to recurrent neural networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2017, pp. 1531–1540.

[15] Y. Wang, W. Liang, J. Shen, Y. Jia, and L.-F. Yu, "A deep coarse-to-fine network for head pose estimation from synthetic data," *Pattern Recognition*, vol. 94, pp. 196–206, 2019.

[16] F. Kuhnke and J. Ostermann, "Deep head pose estimation using synthetic images and partial adversarial domain adaption for continuous label spaces," in *Proceedings of the IEEE/CVF Inter. Conf. on Computer Vision*, 2019, pp. 10 164–10 173.

[17] N. Ruiz, E. Chong, and J. M. Rehg, "Fine-grained head pose estimation without keypoints," in *IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, June 2018.

[18] T.-Y. Yang, Y.-T. Chen, Y.-Y. Lin, and Y.-Y. Chuang, "Fsa-net: Learning fine-grained structure aggregation for head pose estimation from a single image," in *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2019, pp. 1087–1096.

[19] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, "Face alignment across large poses: A 3d solution," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 146–155.

[20] F. Kuhnke, S. Ihler, and J. Ostermann, "Relative pose consistency for semi-supervised head pose estimation," in *2021 16th IEEE Inter. Conf. on Automatic Face and Gesture Recognition (FG 2021)*, 2021, pp. 01–08.

[21] Y. Ganin and V. S. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. of the 32nd Inter. Conf. on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, vol. 37. JMLR.org, 2015, pp. 1180–1189.

[22] G. French, M. Mackiewicz, and M. Fisher, "Self-ensembling for domain adaptation," in *6th Inter. Conf. on Learning Representations, ICLR 2018, Vancouver, BC, Canada*, 2018.

[23] B. Ahn, J. Park, and I. S. Kweon, "Real-time head orientation from a monocular camera using deep neural network," in *Asian Conf. on Computer Vision*. Springer, 2014, pp. 82–96.

[24] M. Patacchiola and A. Cangelosi, "Head pose estimation in the wild using convolutional neural networks and adaptive gradient methods," *Pattern Recognition*, vol. 71, pp. 132–143, 2017.

[25] M. Shao, Z. Sun, M. Ozay, and T. Okatani, "Improving head pose estimation with a combined loss and bounding box margin adjustment," in *The 14th IEEE Inter. Conf. on Automatic Face and Gesture Recognition (FG2019)*, 2019.

[26] S. Lathuilière, P. Mesejo, X. Alameda-Pineda, and R. Horaud, "A comprehensive analysis of deep regression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 9, pp. 2065–2081, 2020.

[27] Y. Zhou and J. Gregson, "Whenet: Real-time fine-grained estimation for wide range head pose," in *BMVC*, 2020.

[28] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Inter. Conf. on Machine Learning*. PMLR, 2019, pp. 6105–6114.

[29] T. Hempel, A. A. Abdelrahman, and A. Al-Hamadi, "6d rotation representation for unconstrained head pose estimation," in *2022 IEEE Inter. Conf. on Image Processing (ICIP)*, 2022, pp. 2496–2500.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[32] R. Ranjan, V. M. Patel, and R. Chellappa, "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *CoRR*, vol. abs/1603.01249, 2016. [Online]. Available: http://arxiv.org/abs/1603.01249

[33] A. Kumar, A. Alavi, and R. Chellappa, "Kepler: keypoint and pose estimation of unconstrained faces by learning efficient h-cnn regressors," in *12th IEEE Int. Conf. on Automatic Face & Gesture Recognition*, 2017, pp. 258–265.

[34] F.-J. Chang, A. Tuan Tran, T. Hassner, I. Masi, R. Nevatia, and G. Medioni, "Faceposenet: Making a case for landmark-free face alignment," in *IEEE Int. Conf. on Computer Vision*, 2017, pp. 1599–1608.

[35] R. Ranjan, S. Sankaranarayanan, C. D. Castillo, and R. Chellappa, "An all-in-one convolutional neural network for face analysis," in *12th IEEE Int. Conf. on Automatic Face & Gesture Recognition*, 2017, pp. 17–24.

[36] R. Valle, J. M. Buenaposada, and L. Baumela, "Multi-task head pose estimation in-the-wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[37] S. K. Mustikovela, V. Jampani, S. D. Mello, S. Liu, U. Iqbal, C. Rother, and J. Kautz, "Self-supervised viewpoint learning from image collections," in *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2020, pp. 3971–3981.

[38] T. Nguyen-Phuoc, C. Li, L. Theis, C. Richardt, and Y.-L. Yang, "Hologan: Unsupervised learning of 3d representations from natural images," in *Proc. of the IEEE/CVF Inter. Conf. on Computer Vision*, 2019, pp. 7588–7597.

[39] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Inter. Conf. on machine learning*. PMLR, 2020, pp. 1597–1607.

[40] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[41] J. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. Á. Pires, Z. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent - A new approach to self-supervised learning," in *Advances in Neural Information Processing Systems 33: NeurIPS, December 6-12*, 2020.

[42] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *5th Inter. Conf. on Learning Representations, ICLR, Toulon, France, April 24-26*. OpenReview.net, 2017.

[43] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[44] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, (ICML)*, vol. 3, 2013.

[45] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems*, 2017, pp. 1195–1204.

[46] B. Athiwaratkun, M. Finzi, P. Izmailov, and A. G. Wilson, "There are many consistent explanations of unlabeled data: Why you should average," in *7th Inter. Conf. on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9*, 2019.

[47] A. Oliver, A. Odena, C. Raffel, E. D. Cubuk, and I. J. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," in *Advances in Neural Information Processing Systems 31: NeurIPS, December 3-8, Montréal, Canada*, 2018, pp. 3239–3250.

[48] J. Choi, M. Jeong, T. Kim, and C. Kim, "Pseudo-labeling curriculum for unsupervised domain adaptation," in *30th British Machine Vision Conf., BMVC, Cardiff, UK, September 9-12*. BMVA Press, 2019, p. 67.

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[50] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, "Revisiting batch normalization for practical domain adaptation," in *5th Inter. Conf. on Learning Representations, ICLR, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017.

[51] W. Chang, T. You, S. Seo, S. Kwak, and B. Han, "Domain-specific batch normalization for unsupervised domain adaptation," in *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20*. Computer Vision Foundation / IEEE, 2019, pp. 7354–7362.

[52] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "300 faces in-the-wild challenge: The first facial landmark localization challenge," in *2013 IEEE Inter. Conf. on Computer Vision Workshops*, 2013, pp. 397–403.

[53] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd Inter. Conf. on Learning Representations, ICLR, San Diego, CA, USA, May 7-9*, Y. Bengio and Y. LeCun, Eds., 2015.

[54] H. Joo, T. Simon, X. Li, H. Liu, L. Tan, L. Gui, S. Banerjee, T. Godisart, B. C. Nabbe, I. A. Matthews, T. Kanade, S. Nobuhara, and Y. Sheikh, "Panoptic studio: A massively multiview system for social interaction capture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 1, pp. 190–204, 2019.

[55] N. Ruiz and J. M. Rehg, "Dockerface: an easy to install and use faster r-cnn face detector in a docker container," *CoRR*, vol. abs/1708.04370, 2017. [Online]. Available: http://arxiv.org/abs/1708.04370

[56] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1755–1758, 2009.

[57] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, 2016.

[58] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[59] H. Jiang and E. G. Learned-Miller, "Face detection with the faster R-CNN," in *12th IEEE Inter. Conf. on Automatic Face & Gesture Recognition, FG 2017, Washington, DC, USA, May 30 - June 3, 2017*. IEEE Computer Society, 2017, pp. 650–657.

[60] V. Albiero, X. Chen, X. Yin, G. Pang, and T. Hassner, "img2pose: Face alignment and detection via 6dof, face pose estimation," in *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR, virtual, June 19-25*. IEEE/CVF, 2021, pp. 7617–7627.

[61] Y. Bao, Y. Liu, H. Wang, and F. Lu, "Generalizing gaze estimation with rotation consistency," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition, CVPR*, June 2022, pp. 4207–4216.

Felix Kuhnke Felix Kuhnke studied Media Technology at the Technische Universität Ilmenau and received his M.Sc. in 2014. In 2015, he joined the Institut für Informationsverarbeitung at the Leibniz University Hannover where he is currently working as a research assistant toward his Ph.D. degree. His research interests include analysis and synthesis of human faces, machine learning, domain adaptation, virtual avatars and facial interaction.

Jörn Ostermann studied electrical engineering and communications engineering at the University of Hannover and Imperial College London. He received the Dipl.-Ing. and Dr.-Ing. degrees from the University of Hannover in 1988 and 1994, respectively. In 1994, he joined AT&T Bell Labs. From 1996 to 2003, he was with AT&T Labs – Research. Since 2003, he is a Full Professor and Head of the Institut für Informationsverarbeitung at Leibniz University Hannover, Germany. From 2008 until 2020, Jörn Ostermann was the Chair of the Requirements Group of MPEG (ISO/IEC JTC1 SC29 WG11). Since then, he is Convenor of the MPEG Technical Coordination Advisory Group. Jörn Ostermann received several international awards and is a Fellow of the IEEE. He published more than 200 research papers and book chapters. He is a coauthor of a graduate-level text book on video communications. He holds more than 30 patents. His current research interests are video coding and streaming, computer vision, 3D modeling, face animation, and computer–human interfaces.