Deep Head Pose Estimation Using Synthetic Images and Partial Adversarial Domain Adaption for Continuous Label Spaces

Felix Kuhnke, Jörn Ostermann
Institut für Informationsverarbeitung
Leibniz University Hannover, Germany
kuhnke@tnt.uni-hannover.de

Abstract

Head pose estimation aims at predicting an accurate pose from an image. Current approaches rely on supervised deep learning, which typically requires large amounts of labeled data. Manual or sensor-based annotations of head poses are prone to errors. A solution is to generate synthetic training data by rendering 3D face models. However, the differences (domain gap) between rendered (source-domain) and real-world (target-domain) images can cause low performance. Advances in visual domain adaptation allow reducing the influence of domain differences using adversarial neural networks, which match the feature spaces between domains by enforcing domain-invariant features. While previous work on visual domain adaptation generally assumes discrete and shared label spaces, these assumptions are both invalid for pose estimation tasks. We are the first to present domain adaptation for head pose estimation with a focus on partially shared and continuous label spaces. More precisely, we adapt the predominant weighting approaches to continuous label spaces by applying a weighted resampling of the source domain during training. To evaluate our approach, we revise and extend existing datasets resulting in a new benchmark for visual domain adaption. Our experiments show that our method improves the accuracy of head pose estimation for real-world images despite using only labels from synthetic images.

1. Introduction

Knowing the pose of the human head in an image provides important information in human-computer interaction. Head pose estimation (HPE) can be used to estimate the focus of attention, a key indicator of human behavior. Estimating attention can be useful in driver assistance systems or to analyze social interaction. Head pose information can also be used to produce better face alignments for pose-invariant face or expression recognition.

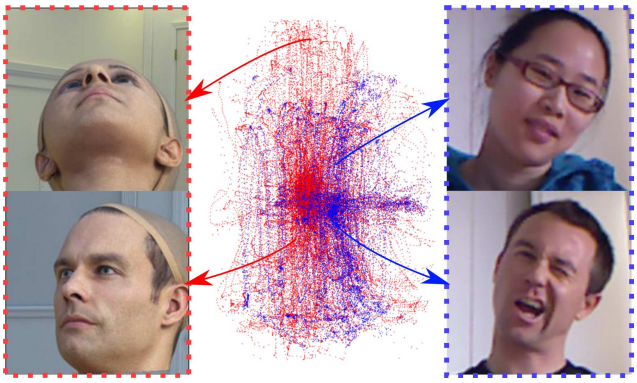

Figure 1. Exemplary continuous label space of two head pose datasets [15, 10]: Synthetically rendered (red) and real-world (blue). Note the difference in distribution shape and density. Images from source and target domain are shown on the left and right, respectively. Our goal is to transfer knowledge from source to target domain in an unsupervised manner.

HPE is commonly formulated as a regression problem, where the task is to predict the continuous orientation in 3D space (*e.g.*, Euler angles). Deep learning approaches have become the state of the art in head pose estimation outperforming most traditional approaches. Producing enough accurately labeled training data, required for deep learning, is a very challenging task. Recording real-world head images with pose measurements comes with a number of challenges. Measurements can be based on sensor data like depth images [10], or inertial measurement unit (IMU) sensors [3], which are both prone to sensor noise. The Biwi dataset [10], a common benchmark for HPE, has an average error of 1 degree [15]. Another approach based on manually labeled keypoints yields similarly inaccurate results due to unknown 3D model and camera parameters. Rendering synthetic face images provides inexpensive and virtually unlimited quantities of accurately labeled data. However, training solely on synthetic data (source) can cause poor performance when testing on real-world data (target) due to

mismatch or shift of underlying data distributions (domain gap).

Recently, there has been great interest in visual domain adaptation (DA) for deep learning [33], which tries to close the domain gap by learning domain-invariant features. Typical DA scenarios are classification tasks with discrete and shared label spaces, *i.e*. both target and source data share the identical set of class labels. For regression problems with continuous label spaces, this assumption of fully shared (identical) label sets does not hold. As illustrated in Figure 1, the label distributions are not necessarily identical, and the target labels only form a subset of the source label set. It is therefore not possible to directly apply current DA methods to HPE. Partial domain adaptation (PDA) tries to resolve these issues for discrete label spaces by estimating the differences between the label set distributions [6]. However, available PDA methods can not be applied directly to HPE because they do not consider continuous label spaces. To the best of our knowledge, neither DA nor PDA has been applied to head pose estimation nor regression tasks.

Our goal is to improve performance for real-world HPE using both labeled data from the synthetic source domain and unlabeled data from the real-world target domain. To exploit the advantages of synthetic image data for HPE tasks, we extend the concept of partial adversarial domain adaptation [6] to regression problems and continuous label spaces. Our method considers the density and shape of label distributions between domains to counteract misalignment of label spaces. Furthermore, we are able to simplify the prevalent weighted loss functions by using a weighted random sampler which provides a straightforward and more efficient solution for partial domain adaptation. Finally, we introduce a novel benchmark for PDA with continuous label spaces by revising and extending available datasets.

While our research is motivated by accurate head pose estimation, our contributions are threefold:

- We bring together the unconnected topics of head pose estimation and adversarial domain adaptation and compare current deep HPE methods in the context of synthetic data and domain adaptation.
- State-of-the-art HPE results using our novel approach for partial adversarial domain adaptation.
- A benchmark for PDA with continuous label spaces as a novel challenge for the visual domain adaptation community.

2. Related Work

In the following, we will first review recent deep learning-based HPE methods and the use of synthetic data for HPE and subsequently review related works for visual domain adaptation methods focusing on (partial) adversarial DA methods.

2.1. Deep Learning-based Head Pose Estimation

Vision-based head pose estimation can be categorized into two approaches. One approach is to detect geometric facial features (*e.g*., landmarks) and use a reference 3D head model to estimate the pose from these features. The other approach is to use the complete facial appearance to estimate the pose, either by a model of facial appearance or directly learn the relation from image to pose. A survey on classical methods is given in [23]. In this paper, we will focus on deep learning-based head pose estimation directly from a single monocular RGB image.

Anh *et al*. [1] were among the first to present a deep learning-based approach for HPE. Using a convolutional neural network (CNN), they directly regressed the head pose information. Patacchiola and Cangelosi [25] evaluated different CNN architectures and adaptive gradient methods for head pose estimation. Several networks have been presented that perform multiple facial analysis tasks [20, 27, 28, 7] like landmark localization, pose estimation, gender recognition, and other tasks. For example, Chang *et al*. [7] predicted facial keypoints and head pose jointly using a ResNet architecture [16]. However, multi-analysis approaches only coarsely evaluated pose estimation performance. The performance difference between using facial landmarks for pose estimation and direct regression was investigated by Ruiz *et al*. [29]. They introduced a novel loss function for deep HPE. In their experiments, they outperformed landmark-based pose estimation approaches. In contrast to our work, the aforementioned works do not use synthetic training data. Ruiz *et al*. also train on a synthetically expanded dataset by utilizing the 300W-LP dataset [35]. However, 300W-LP includes augmentations of real photographs (warped versions of these pictures) but does not contain images of rendered 3D face models.

Using images of rendered 3D face models provides a solution to obtain high amounts of accurately labeled data. The synthetic face pose dataset SynHead was introduced by Gu *et al*. [15]. In their work, they focused on improving head pose prediction performance for temporal sequences by using recurrent neural networks. They trained and evaluated their method on the SynHead dataset. Furthermore, they reported the performance when a network trained with synthetic data is fine-tuned on real-world data. In contrast, we do not use any temporal information and perform single-frame predictions. In addition, our goal is not to fine-tune on real-world data but to use an unsupervised approach that does not require any labels for the target domain. Due to specific characteristics of the SynHead dataset, it is difficult to use for HPE and DA benchmarking (see Section 4). Liu *et al*. [22] created a synthetic head pose dataset to train a CNN for HPE. They evaluated their model trained exclusively on synthetic data on a real-world dataset. Assuming that their synthetic data is close enough to real-world

data, they did not apply any domain adaptation. To date, their synthetic dataset is not publicly available. While both works [15, 22] use synthetic training data, either no or only supervised transfer learning (fine-tuning) is used to overcome domain mismatch. This is different to our approach where we explicitly tackle domain mismatch using partial adversarial domain adaptation.

2.2. Partial Adversarial Domain Adaptation

In our review, we will focus on adversarial domain adaptation and more detailed on partial adversarial techniques as these techniques form the basis of our method. The interested reader is referred to two recent surveys [9, 33] that extensively summarize the current state of the art of (deep) visual domain adaption.

The seminal work of Ganin *et al*. [12, 13] introduced the concept of a domain adversarial neural network (DANN). DANN works by matching the distributions of features extracted from different domains by making them indistinguishable for a discriminative classifier (also called domain adversary). One has to note the very similar concept of generative adversarial networks (GANs) described by Goodfellow *et al*. [14]. Besides, the applications of both methods are quite different.

Numerous works build upon DANN to approach domain adaptation [31, 32]. However, these works assume identical label spaces, which means that for every sample of source data there exists target data with the same label. In a realistic scenario, where large amounts of source data and only unlabeled target data is available, this assumption is inadequate. Aligning the source and target feature distributions (*e.g*., using DANN) will also align the label spaces, causing negative transfer as target features are matched to unequal source labels.

To overcome this problem, Cao *et al*. [6] proposed a partial adversarial domain adaptation (PADA) network. PADA mitigates the effect of mismatch between label spaces by downweighting the data of source classes not expected in the target labels. PADA is improved in follow up works [5, 26] where additional domain adversaries are added for every category in source label space. A similar concept was proposed by Zhang *et al*. [34], where an additional domain classifier is added to the network to identify source samples from the outlier classes. Another approach was presented by Chen *et al*. [8]. They propose to learn a class weighting ratio to match the label distributions. While these methods consider a partially shared label space, they all assume a discrete label space. The task is always classification where some classes do not exist in the target domain. In our case, we cannot relate to fixed class labels but have continuous labels of head poses which also do not allow category-wise extensions [5, 26].

Interestingly, domain adaptation has not been used for HPE at all. Despite the recent works in PDA, it is unclear how to transfer these methods for HPE. Furthermore, we could not find any regression task or dataset related to visual domain adaptation. In this work, we show the first approach to apply PDA to HPE and introduce the novel problem paradigm of continuous label spaces (regression) to visual domain adaptation.

3. Method

In this section, we introduce our novel method for partial domain adaptation for continuous label spaces. Our solution and experiments are specifically developed to solve our head pose estimation task, but might also be applied to other regression tasks. Our method is inspired by previous (partial) domain adversarial methods, which are based on adversarial methods and we start by reintroducing the required notations and concepts. In the typical domain adaptation scenario, data is available from the 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 the target domain $\mathcal{D}_t = \{(x_i^t)\}_{i=1}^{n_t}$, only data is available. In classical DA it is assumed that the source-domain label space $\mathcal{C}_s$ and target-domain label space $\mathcal{C}_t$ are shared. In contrast to DA, in PDA $\mathcal{C}_t$ is only a subset of $\mathcal{C}_s$ ($\mathcal{C}_t \subset \mathcal{C}_s$). Source data with labels in $\mathcal{C}_s \setminus \mathcal{C}_t$ are referred to as source outliers.

3.1. Partial Domain Adversarial Networks

In their simplest form DANN consist of three subnetworks. The design is illustrated in Figure 2. In our case, a **domain discriminator** D is trained to distinguish the source domain from the target domain samples. The **feature extractor** F is trained to extract features that simultaneously minimize the task loss and further maximize the discriminator loss in order to create features indistinguishable to D. The **pose regressor** R is trained to fulfill the actual task (head pose estimation) leading to the following functional [12]:

$$\begin{aligned} E(\theta_D, \theta_F, \theta_R) = \mathcal{L}_y(R(F(X_s)), Y_s) \\ - \lambda \mathcal{L}_d(D(F(X_s \cup X_t)), L_s \cup L_t), \end{aligned} \tag{1}$$

where $\mathcal{L}_y$ is the task loss (pose prediction error) and $\mathcal{L}_d$ is the domain classification loss weighted by λ. λ is typically increased from 0 to λ_{max} during training. θ denotes parameters of D, F and R. L_s and L_t are labels describing the domain origin. $\mathcal{L}_d$ is the cross-entropy loss and L_s and L_t are $\vec{1}$ and $\vec{0}$, respectively.

The following minimax optimization will deliver saddle points of Eq. (1) to learn the networks' parameters $\hat{\theta}$ that fulfill the domain adaptation goals:

$$\begin{aligned} (\hat{\theta}_F, \hat{\theta}_R) &= \arg\min_{\theta_F, \theta_R} E(\theta_F, \theta_R, \hat{\theta}_D) \\ \hat{\theta}_D &= \arg\max_{\theta_D} E(\hat{\theta}_F, \hat{\theta}_R, \theta_D). \end{aligned} \tag{2}$$

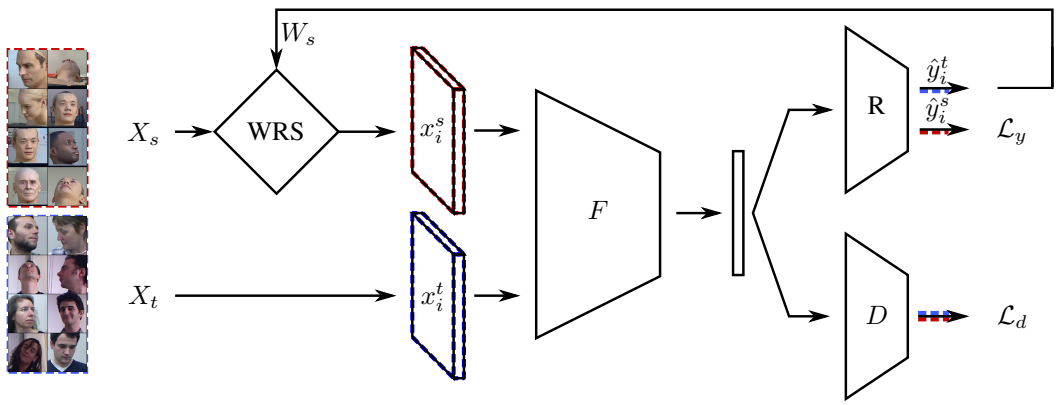

Figure 2. Proposed architecture for domain-adapted head pose estimation: The feature extraction network F is trained to extract domain-invariant features from source and target domain samples (x_i^s, x_t^s) using domain-adversarial training. Domain feedback is provided by the domain discriminator D. A pose regressor R estimates the head pose for samples from both domains. Pose estimates from target samples $\hat{y}_i^t$ are feed back to generate sampling weights W_s. Instead of sampling directly from the source data X_s a weighted random sampler (WRS) selects source samples with similar labels to the estimated target labels. This enforces a similar label distribution of source and target domain samples during training.

The minimax optimization can be solved iteratively similar to GANs [14] or using a gradient reversal layer [12]. Different possible manifestations are unified and presented as a general framework by Tzeng *et al*. [32].

The original DANN framework does not consider partial domain adaptation. In PDA, the general goal is to reduce the negative influence of source outliers during training. This is usually done using a weighting scheme, to down-weight the contribution of source outliers to the loss functions.

The average target label predictions $\hat{Y}_t$ over all target samples are commonly used to produce a class-dimensional weight vector [5, 6, 26]. These class weights are then used to weight the contribution of all losses calculated with source data. That is, task loss and domain classification loss of source samples are weighted down for classes that are rarely predicted for target samples.

Similarly, Zhang *et al*. [34] use the output of the domain discriminator as the likelihood of the sample coming from the source distribution. They assume that high likelihoods indicate samples from the source outliers, as no target samples should have similar features. Subsequently, they use the domain discriminator predictions to generate weights used during training with source samples.

We tried to produce weights with a discriminator [34]. However, we found that in our case, the domain discriminator output was insufficient to indicate source outliers. Furthermore, we can not use class weights because our label space is continuous.

3.2. Extension to Continuous Label Spaces

There is currently no partial domain adaptation technique for continuous label spaces. To avoid negative transfer from source outliers we need to control the influence of source outliers during training. Instead of using target label predictions to generate source *class weights* [5, 6, 26], we propose to use them to generate source *sample weights*.

First, we will describe a straightforward adaptation of **PADA-like** [6] methods from class weighting to sample weighting. Secondly, we will take some time to revise the weighted loss scheme to a more efficient resampling procedure, using a **weighted random sampler**. Thirdly, we will introduce a new weighting scheme for balanced resampling of the source data: Partial Adversarial Domain Adaptation for COntinuous label spaces (**PADACO**).

PADA-like: To create weights for source samples, we propose to measure the distance of target predictions to source labels in label space. In our setting the label space is $\mathbb{R}^3$ consisting of the three rotation angles pitch, yaw, and roll. We compute the distance between rotations with the mean squared error, which is also our task loss function $\mathcal{L}_y$. To obtain weights $w_i^s \in W_s$ for every source sample, we adapt the minimum distance for every y_i^s to $\hat{Y}_t$. We use a distance threshold t to exclude samples that are too far away from the target predictions. The weights are calculated by:

$$w_i^s = \begin{cases} 0, \text{ if } \min_{\hat{y}_i^t \in \hat{Y}_t} \mathcal{L}_y(\hat{y}_i^t, y_i^s) \geq t \\ t - \min_{\hat{y}_i^t \in \hat{Y}_t} \mathcal{L}_y(\hat{y}_i^t, y_i^s), \text{ otherwise.} \end{cases} \tag{3}$$

Equivalent to Cao *et al*. [6], we normalize the weights by dividing with $\max(W_s)$. These weights are applied to the loss functions to weight down the losses from source outliers [5, 6, 26, 34]. We will further refer to our adaptation of weighting methods like PADA to continuous label spaces as PADA-like.

Weighted random sampler: Current methods for PDA apply weights for every processed data sample to multiple loss functions during training [5, 6, 26, 34]. Generally, $\mathcal{C}_s$ is larger than $\mathcal{C}_t$, leading to many (near) zero-weighted source samples pushed through the network without any benefit. The larger the source outlier space, the more time and energy is wasted. Therefore, we propose to use weights not for weighting after the forward pass but for sample selection prior to the forward pass. Using a weighted random sampler (WRS) depicted in Figure 2 we can select appropriate samples from X_s, resulting in a simpler and more efficient training scheme. WRS uses source weights W_s as probabilities of a multinomial probability distribution to resample the data.

Another benefit of resampling compared to weighted loss functions is the interplay with batch normalization (batch norm) [18]. Batch norm has been found beneficial for domain adaptation [11, 21]. In preliminary experiments we found that domain-wise batch norm [11] provides a considerable performance boost. Using weighted loss functions however, will not change the mini-batch statistics, as even zero-weighted samples are used in batch norm calculations. While batch means and standard deviations can also be calculated with weights, these would have to be explicitly transferred to all batch norm layers. Furthermore, a weighted batch norm is not available in any modern deep learning framework. Using the weighted resampling strategy lets us use the default batch norm method without any changes. Lastly, this simple but effective change of weighting to sampling strategy can be readily applied to other existing weight-based PDA methods.

PADACO: Using weights or resampling based on weights reduces the influence of source outliers during training. However, despite differences in their shape, we found that source and target label distributions also differ in density. In other words, the ratio of samples with the same or similar labels can be imbalanced between source and target data. To avoid misalignment of label spaces, we need to balance the contributions of source samples during training. Therefore, we combine sample weights and consider the label space distribution densities in our PADACO method.

With the WRS approach, source data can be resampled to account for data imbalance without much effort, but, compared to PADA-like, the weighting (calculation of W_s) needs to be adapted. Instead of calculating a weight for every source sample, we propose to assign a fixed amount of source samples to every target sample. Using nearest neighbor search for every target label prediction on the source labels, we can select the N_n nearest source samples for training. A balancing is already given when using this nearest neighbor approach. As a result, every target sample is assigned to a fixed number N_n of source samples and the same ratio (1:N_n) of similar labels from source and target is provided during training.

To compute sampling weights W_s for every source sample, we first initialize all weights with zero. We evaluate the target dataset to obtain the current target label predictions $\hat{Y}_t$ and find N_n nearest source samples for every target prediction. The weight of a source sample is incremented by 1 for every time it is assigned to a target label prediction. In other words, to account for multiple assignments to the same source samples, we count the number of times a source sample is found as a neighbor to a target sample to form W_s. To create sampling probabilities for the sampler, we divide the weights W_s by the sum of all weights.

For efficient nearest neighbor search even with many data points, space partitioning (*e.g.*, a k-d tree [2]) can be used. As the search strategy is changed, we do not compare all source labels to all target predictions as in Eq. (3), but only search neighbors for all target predictions. This strategy will also improve efficiency because the amount of target samples n_t is typically much smaller than n_s.

During the development, we also examined other ideas. We tried to apply additional thresholding to dismiss neighbors too far away from target labels. However, we found that this does not improve the results and only adds an additional parameter to the method. We also tried to iteratively update the weights during training, to allow the weights to change during adversarial training. While this approach can converge in some cases, we found it to be highly unstable. Despite these findings, we think that stabilizing iterative weight updates is a promising direction for future work.

Our final training procedure is described in Algorithm 1.

Algorithm 1: Training procedure

Input: labeled source samples X_s, Y_s
unlabeled target samples X_t
parameter λ_{max}, N_n
Output: $\hat{\theta}_F, \hat{\theta}_R$
Stage-1:
$\hat{\theta}_F, \hat{\theta}_R \leftarrow$ pre-train F and R on X_s with Y_s
$\hat{\theta}_D \leftarrow$ random initialization
Stage-2:
$\hat{Y}_t \leftarrow$ evaluate target data $R(F(X_t))$
$W_s \leftarrow$ calculate weights using N_n, Y_s, and $\hat{Y}_t$
while $\lambda < \lambda_{max}$ **do**
  $b_s \leftarrow$ sample source batch with weighted sampling from X_s using W_s
  $b_t \leftarrow$ sample target batch from X_t
  $\hat{\theta}_F, \hat{\theta}_R \leftarrow$ train F and R with b_s
  $\hat{\theta}_F, \hat{\theta}_D \leftarrow$ train F and D with b_s and b_t using adversarial training [12]
  $\lambda \leftarrow$ update λ according to a schedule

4. SynHead++, SynBiwi+, Biwi+

For validating our method is is not possible to directly utilize existing benchmarks due to reasons we will discuss in this section. We therefore introduce three extensions[1] to existing datasets [10, 15]. Our goal is the provide source and target datasets for the task of visual domain adaptation with continuous label spaces (*e.g.*, pose estimation).

As a real-world, target-domain dataset, we choose the Biwi Kinect Head Pose Database (Biwi) [10] containing 24 sequences of 20 different subjects (14 men, 6 women, 4 people with glasses) recorded with a kinect sensor. Our source-domain datasets are based on SynHead [15], a synthetic head pose dataset of 10 rendered 3D head models in various poses. The original SynHead already includes smoothed head motion tracks of all 24 Biwi sequences. However, SynHead was rendered using the Euler angles provided by Biwi but with a different sequence of rotation axes. This rotation order (dissimilar to the Biwi order) causes that several SynHead images and Biwi images with the same label show different head rotations. In extreme cases, SynHead images show no part of the face at all.

An issue in current HPE research are inconsistent face crops. As an essential pre-processing step, the crop of the original image (based on a face bounding box) used for further processing plays an important role in HPE performance. Typically, comparing this step is neglected in the HPE community, and different face detectors are used throughout experiments.

To overcome these issues and to evaluate and compare the tasks of partial and non-partial domain adaptation, we extend and revise Biwi [10] and SynHead [15] to:

- SynBiwi+: A shared label space dataset ($\mathcal{C}_t = \mathcal{C}_s$)
- SynHead++: A subset label space dataset ($\mathcal{C}_t \subset \mathcal{C}_s$)
- Biwi+: A target test set for HPE and domain adaptation tasks with SynBiwi+ and SynHead++

For all SynHead images with visible faces, we recomputed the intended Biwi angle representation leading to SynHead+. We rotate available SynHead+ images to produce images with rotations as close as possible to the Biwi dataset leading to SynBiwi+. For every image in the Biwi dataset, SynBiwi+ has 10 corresponding images containing the 10 synthetic head models of SynHead. As we only generate images by rotating original images, we do not get perfect alignment. The mean average Euler angle error between Biwi and SynBiwi+ is 0.15°, which we think is sufficient for the envisioned experiments. Finally, for partial domain adaptation experiments where the source dataset should be a proper superset of the target dataset, we create SynHead++ which is the union of SynHead+ and SynBiwi+.

To further improve reproducibility, we provide bounding boxes for the new datasets and the original Biwi dataset which we then denote by a plus sign (Biwi+). We evaluated three available face detectors [4, 17, 19] to produce bounding boxes. However, all detectors fail for extreme head rotations on both datasets. Further, multiple detections are sometimes produced on ears or persons in the background. Based on the detections of [19], we manually corrected all bounding boxes and added missing boxes manually to the datasets. Exemplary images of the datasets are shown in Figure 1.

[1]The labels and code to recreate the datasets are available at `http://www.tnt.uni-hannover.de/project/headposeplus`.

5. Experiments

In the following, we will analyze different levels of domain knowledge transfer used for head pose estimation. We compare the traditional supervised methods to our new DA and PDA experiments and further analyze the effects of different weighting schemes for PDA.

5.1. Implementation Details

For all our experiments, the feature extractor F is ResNet18 as provided by PyTorch [24]. The domain discriminator D is a fully connected (fc) layer network with two layers (512 neurons each) connected as Input-fcLayer-BatchNorm-LeakyReLU-fcLayer-Output. The regression network R is an fc layer with 512 neurons and 3 output values for estimation of Euler angles.

We add random backgrounds to all synthetic face images using randomly cropped images from the backgrounds folder of the original dataset [15]. We do not use common data augmentations such as random crops, flips or color adjustments. All images are cropped to the bounding boxes described in Section 4 and rescaled to match the input of the feature extractor F. Inspired by [11, 21], we process mini-batches of source and target data separately. This forces batch normalization to use different normalization statistics for each domain during training. For all experiments, we use stochastic gradient descent with momentum 0.9, Nesterov, a batch size of 200, and a learning rate schedule with base learning rate 0.03 for D, F, and R. The learning rate is slowly decreased after the first third of training. We set the threshold for PADA-like to $t = 3.5$ and the number of nearest neighbors for PADACO to $N_n = 10$. To analyze only the impact of weight calculation methods, the PADA-like experiment also uses the WRS.

To create baseline models (Stage-1, see Alg. 1), which we will later use in Stage-2 of our training, we use the pre-trained ResNet18 as F, and further train F for 20 epochs with the dataset required for the following domain adaptation experiments (SynBiwi+ or SynHead++). For validation, 3% of data is held out. Finally, we select the epoch

Experiment	Method	Network	Training set	Test set	MAE	Pitch	Yaw	Roll
Intra domain	Anh [1]	Custom CNN	Biwi*	Biwi*	2.93	3.4	2.8	2.6
	Liu [22]	Custom CNN	Biwi$\diamond$	Biwi$\diamond$	5.93	6.0	6.1	5.7
	Ruiz [29]	ResNet50	Biwi†	Biwi†	3.23	3.39	3.29	3.00
	Gu [15]	VGG16 [30]	Biwi†	Biwi†	3.66	4.03	3.91	3.03
Inter domain	Ruiz [29]	ResNet50	300W-LP [35]	Biwi†	4.90	6.61	4.81	3.27
	Liu [22]	Custom CNN	*unavailable*	Biwi	3.73	4.3	4.5	2.4
Inter domain	BaselineDA	ResNet18	SynBiwi+	Biwi+	4.58	4.99	4.85	3.89
Domain adaptation	**DANN** [12]	ResNet18	SynBiwi+	Biwi+	**3.34**	**3.56**	**3.43**	**3.03**
	PADACO (proposed)	ResNet18	SynBiwi+	Biwi+	4.04	4.47	4.11	3.56
Inter domain	BaselinePDA	ResNet18	SynHead++	Biwi+	4.53	4.97	4.61	3.97
Partial DA	DANN [12]	ResNet18	SynHead++	Biwi+	6.05	8.08	6.17	3.91
	PADA-like	ResNet18	SynHead++	Biwi+	6.41	8.14	6.86	4.22
	PADACO (proposed)	ResNet18	SynHead++	Biwi+	**4.13**	**4.51**	**4.11**	**3.78**

Table 1. Head pose estimation results on variants of the Biwi dataset. Biwi variants: *Random split (86% and 14% images), †Split by sequence (16 and 8 sequences), $\diamond$ Split by subject (18 and 2 subjects). SynHead++, SynBiwi+ and Biwi+ are our novel benchmark datasets for head pose estimation and domain adaptation. Experimental results are grouped in blocks describing the use of data from different domains during training and testing. Our proposed method achieves the best results for the challenging task of partial domain adaptation.

with lowest validation error as a baseline starting point (see Table 1) for the following domain adaptation experiments.

For DA and PDA experiments, during the first third of the training, λ is set to 0 to train the discriminator. Then λ is scheduled from 0 to $\lambda_{max} = 0.2$. On reaching λ_{max} training is stopped after 5 epochs on SynHead++ (PDA experiments) or 16 on SynBiwi+ (DA experiment).

5.2. Overview and Results

We conducted experiments for HPE in the settings of domain adaptation and partial domain adaptation using the proposed datasets. All results are sorted by experiment type in Table 1. The experiment type describes the use of data from different domains during training and testing. In the intra-domain setting, only data from one domain is used. Inter domain describes the setting where training and testing data are from different domains, but no domain adaptation techniques are applied. These techniques are evaluated in the domain adaptation and partial DA experiments. The domain adaptation experiments are our control experiments where we synthetically enforce that source and target domain share a nearly identical label space. Contrarily, partial DA experiments do not assume these constraints and can be seen as a realistic scenario for real-world applications. In the evaluation of partial DA, we will illustrate the effects of using different source weighting schemes. We report intra- and inter-domain results from the literature as a comparison to the novel non-partial and partial DA results. Furthermore, we trained two inter-domain baseline models on the proposed datasets. The performance of head pose estimation is usually measured with the mean absolute error (MAE) of the Euler angles. We report MAE and absolute error for every rotation angle (pitch, yaw, and roll) in degree.

Intra Domain Intra-domain results show the current state of the art for monocular deep HPE methods trained and evaluated on the Biwi dataset. Due to different training and test set splits the results should not be compared to each other but serve as an overview of possible intra-domain results.

Inter Domain and Baselines Inter-domain results are more related to the domain adaptation task. Comparing the inter-domain to the intra-domain results of Ruiz *et al*. [29], we can conclude that there exists a domain mismatch between the source (training) and target (test) dataset. An exception is made by Liu *et al*. [22] as they outperform their intra-domain results. One reason could be similar statistics between the Biwi dataset and their synthetic training set, which shares the same head pose ranges with Biwi [22].

Our inter-domain baselines outperform the inter-domain method of Ruiz *et al*. [29] using a smaller network architecture. Direct comparison of methods should be handled with care due to differences in experimental setups.

Domain Adaptation To compare the differences between the performance of methods on partially shared and identical label spaces, we evaluate DANN [12] and PADACO on our shared label space dataset SynBiwi+. Based on the BaselineDA model, we apply the DANN and PADACO method with parameters as described in Section 5.1. DANN yields impressive results for head pose estimation compared

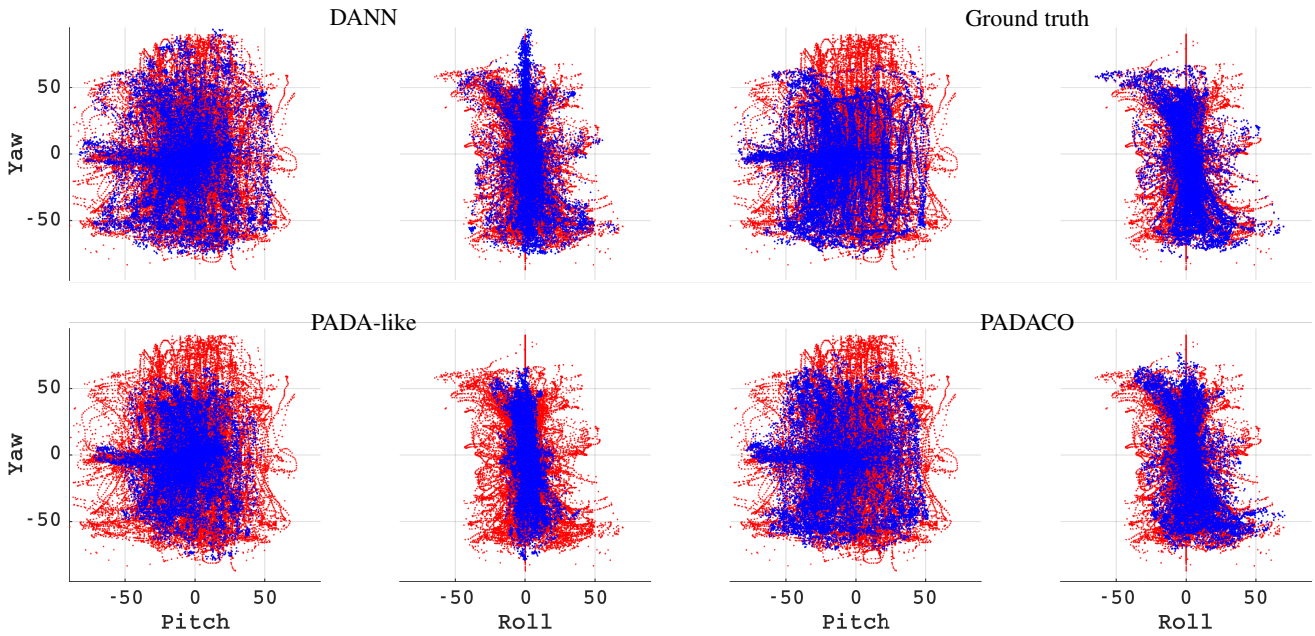

Figure 3. Label space visualization after training with different weighting schemes: In addition to ground truth labels, for every PDA experiment we show source labels Y_s (red) and the predicted target labels $\hat{Y}_t$ (blue). The 3D label space of rotations is visualized by 2D projections on yaw/pitch and yaw/roll (angles in degree). The different distributions reveal the effects of applying the different weighting schemes. DANN [12] expands $\hat{Y}_t$ into Y_s, PADA-like collapses $\hat{Y}_t$ to the higher density regions of Y_s and PADACO (proposed) keeps the overall shape of $\hat{Y}_t$ similar to the ground truth.

to methods trained on inter-domain data and even methods trained on Biwi (intra domain) directly. The improvement of mean absolute error (MAE) is over 1° as can be seen in Table 1. This result encourages the search for similar performing PDA methods and further validates our assumption that DA is a feasible approach for HPE. While PADACO improves the result compared to the baseline by 12% (0.54°), it does not reach the performance of DANN. However, in contrast to PADACO, DANN requires a prior assumption on the label distribution. The partial domain adaptation results will show that DANN fails if this assumption does not hold.

Partial Domain Adaptation For PDA we evaluate DANN, PADA-like, and PADACO. The results show the expected, DANN fails to work in the case of non-identical label spaces. Instead, the MAE is increased by nearly 1.5°. Fig. 3 shows the distribution of label predictions after training. We can clearly see that DANN produces negative transfer by aligning the label spaces. In our framework, DANN is identical to setting all the weights W_s to 1.

Despite using a weighting procedure, the PADA-like approach produces worse results compared to DANN. Comparing to the ground truth in Figure 3, we can see a contraction. We believe this is caused by the imbalance of weighted source and target samples as the higher density regions in source label space attract the target samples during training.

Compared to the others, our novel approach PADACO does not diverge and even decreases the error on the target domain by nearly 10%. The balanced resampling of source samples seems to avoid negative transfer by avoiding a matching of the target to the dissimilar source label space distribution.

6. Conclusion

We proposed a novel unsupervised domain adaptation technique to improve deep head pose estimation performance. We extended recent works on partial domain adaptation to the previously neglected regression tasks where labels are not discrete classes but reside in a continuous label space. Using a balanced resampling of source data and partial adversarial domain adaptation, we lowered the head pose estimation error by nearly 10%. Our approach can be applied to other regression tasks such as hand or body pose estimation to improve results when training on data from another domain (*e.g.*, synthetic data). With our results for partial domain adaption, a promising research direction was established. We will try to extend our work in further studies. In this regard, we are looking forward to others proposing solutions using the novel domain adaptation benchmark[1] introduced in this paper.

References

[1] Byungtae Ahn, Jaesik Park, and In So Kweon. Real-time head orientation from a monocular camera using deep neural network. In *Asian Conf. on Computer Vision*, pages 82–96. Springer, 2014.

[2] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[3] Guido Borghi, Marco Venturelli, Roberto Vezzani, and Rita Cucchiara. Poseidon: Face-from-depth for driver pose estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 5494–5503, 2017.

[4] Gary Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[5] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Partial transfer learning with selective adversarial networks. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2724–2732, 2018.

[6] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *IEEE Proc. European Conf. on Computer Vision*, pages 135–150, 2018.

[7] Feng-Ju Chang, Anh Tuan Tran, Tal Hassner, Iacopo Masi, Ram Nevatia, and Gerard Medioni. Faceposenet: Making a case for landmark-free face alignment. In *IEEE Int. Conf. on Computer Vision*, pages 1599–1608, 2017.

[8] Qingchao Chen, Yang Liu, Zhaowen Wang, Ian Wassell, and Kevin Chetty. Re-weighted adversarial adaptation network for unsupervised domain adaptation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 7976–7985, 2018.

[9] Gabriela Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*, 2017.

[10] Gabriele Fanelli, Matthias Dantone, Juergen Gall, Andrea Fossati, and Luc Van Gool. Random forests for real time 3d face analysis. *Int. Journal of Computer Vision*, 101(3):437–458, February 2013.

[11] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for domain adaptation. *arXiv preprint arXiv:1706.05208*, 2017.

[12] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.

[13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[15] Jinwei Gu, Xiaodong Yang, Shalini De Mello, and Jan Kautz. Dynamic facial analysis: From bayesian filtering to recurrent neural network. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1531–1540, 2017.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[17] Peiyun Hu and Deva Ramanan. Finding tiny faces. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 951–959, July 2017.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[19] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10(Jul):1755–1758, 2009.

[20] Amit Kumar, Azadeh Alavi, and Rama 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*, pages 258–265, 2017.

[21] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

[22] Xiabing Liu, Wei Liang, Yumeng Wang, Shuyang Li, and Mingtao Pei. 3d head pose estimation with convolutional neural network trained on synthetic images. In *IEEE Int. Conf. on Image Processing*, pages 1289–1293, 2016.

[23] Erik Murphy-Chutorian and Mohan Manubhai Trivedi. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):607–626, 2009.

[24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS Workshop*, 2017.

[25] Massimiliano Patacchiola and Angelo Cangelosi. Head pose estimation in the wild using convolutional neural networks and adaptive gradient methods. *Pattern Recognition*, 71:132–143, 2017.

[26] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *Thirty-Second AAAI Conf. on Artificial Intelligence*, 2018.

[27] Rajeev Ranjan, Vishal M. Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *CoRR*, abs/1603.01249, 2016.

[28] Rajeev Ranjan, Swami Sankaranarayanan, Carlos D. Castillo, and Rama Chellappa. An all-in-one convolutional neural network for face analysis. In *12th IEEE Int. Conf. on Automatic Face & Gesture Recognition*, pages 17–24, 2017.

[29] Nataniel Ruiz, Eunji Chong, and James M. Rehg. Fine-grained head pose estimation without keypoints. In *IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, June 2018.

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

[31] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *Proc. IEEE Int. Conf. on Computer Vision*, pages 4068–4076, 2015.

[32] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

[33] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[34] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 8156–8164, 2018.

[35] Xiangyu Zhu, Zhen Lei, Xiaoming Liu, Hailin Shi, and Stan Z. Li. Face alignment across large poses: A 3d solution. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 146–155, 2016.