From Synthetic to Real: Unsupervised Domain Adaptation for Animal Pose Estimation

Chen Li Gim Hee Lee
Department of Computer Science, National University of Singapore
{lic, gimhee.lee}@comp.nus.edu.sg

Abstract

Animal pose estimation is an important field that has received increasing attention in the recent years. The main challenge for this task is the lack of labeled data. Existing works circumvent this problem with pseudo labels generated from data of other easily accessible domains such as synthetic data. However, these pseudo labels are noisy even with consistency check or confidence-based filtering due to the domain shift in the data. To solve this problem, we design a multi-scale domain adaptation module (MDAM) to reduce the domain gap between the synthetic and real data. We further introduce an online coarse-to-fine pseudo label updating strategy. Specifically, we propose a self-distillation module in an inner coarse-update loop and a mean-teacher in an outer fine-update loop to generate new pseudo labels that gradually replace the old ones. Consequently, our model is able to learn from the old pseudo labels at the early stage, and gradually switch to the new pseudo labels to prevent overfitting in the later stage. We evaluate our approach on the TigDog and VisDA 2019 datasets, where we outperform existing approaches by a large margin. We also demonstrate the generalization ability of our model by testing extensively on both unseen domains and unseen animal categories. Our code is available at the project website[1].

1. Introduction

Animal pose estimation has received increasing attention over the last few years because of many potential applications in zoology, biology and aquaculture. Despite the great success of applying deep neural networks to human pose estimation, the lack of well-labeled animal pose data makes it infeasible to directly leverage on the powerful deep learning approaches. Existing works overcome this problem by transferring knowledge from other more accessible domains such as synthetic animal data [23, 5, 46, 47, 48] or human data [6]. The advantage of synthetic data is that it is low cost and convenient to generate a large scale of data with accurate ground truth. Moreover, the domain gap between synthetic and real animals is more manageable than that between other domains such as human and animals. This is evident from the results of [6], where sufficient labeled data in the real animal domain is needed for the network to work despite the use of sophisticated domain adaptation techniques.

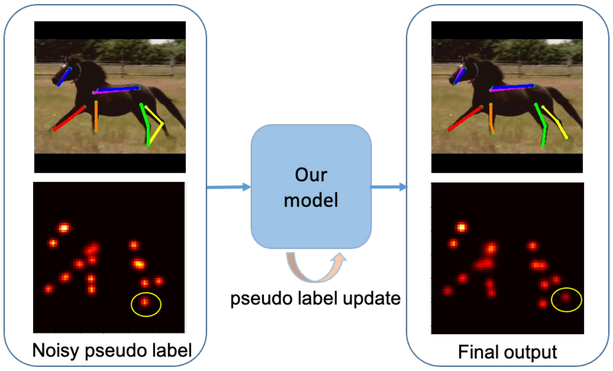

Figure 1. Our method takes in noisy pseudo labels (*e.g.* hind hoof on left image) generated from model trained with labeled synthetic data and outputs the correct animal pose on real images.

The domain gap between synthetic and real animals mainly comes from the differences in texture and background, and the limited pose variance of synthetic data. To solve the domain shift problem, existing works first generate pseudo labels with a model trained on synthetic data, and then gradually incorporate more pseudo labels into the training according to the confidence score. However, these pseudo labels are inaccurate even with refinement techniques such as confidence-based filtering [6] or geometry-based consistency check [23]. Fig. 1 shows an example where a model trained on synthetic animals gives wrong predictions (*e.g.* the hind hoof) with high confidence (marked in yellow circle in the heatmap). This kind of noisy pseudo labels cannot be filtered out based on the confidence score and will lead to degraded performance when used

[1] https://github.com/chaneyddtt/UDA-Animal-Pose

naively for training.

In this paper, we propose a novel approach to learn from synthetic animal data. We design a multi-scale domain adaptation module (MDAM) to reduce the domain gap. Our MDAM consists of a pose estimation module and a domain classifier. We first train the pose estimation module with the synthetic data [23] to generate an initial set of pseudo labels for the real animal images. We then train our MDAM on the synthetic labels and the pseudo labels. However, the accuracy of MDAM is limited by the presence of noise in the pseudo labels. To alleviate this problem, we introduce an online coarse-to-fine pseudo label updating strategy. Specifically, we propose a self-distillation module in the inner coarse-update loop and a mean-teacher [31] in the outer fine-update loop to generate better pseudo labels that gradually replace the old noisy ones.

We design our pseudo label updating strategy according to the *memorization effect* [3, 42] of deep networks, which states that deep networks learn from clean samples at the early stage before eventually memorizing (*i.e*. overfits to) the noisy ones. To avoid the memorization effect, we rely more on the initial pseudo labels at the early stage when the self-distillation module and mean-teacher are still at their infancy in training. Our coarse-to-fine pseudo label updating strategy gradually replaces the noisy initial labels when the self-distillation module and mean-teacher gained enough competency to generate more reliable pseudo labels. Consequently, we are able to supervise our network with more accurate pseudo labels and prevent overfitting at the same time. As illustrated in Fig. 1, our model can successfully locate the joint (hind hoof on the right image) although the initial pseudo label is not accurate.

We validate our approach on the TigDog Dataset [10], where we outperform existing unsupervised domain adaptation techniques by a large margin. We also demonstrate the generalization capacity of our approach by directly testing on the Visual Domain Adaptation Challenge dataset (VisDA2019), the Zebra dataset [46] and the Animal-Pose dataset [6]. Experimental results show that our approach can generalize well to both unseen domains and unseen animal categories. Our main contributions are as follows:

- We design an unsupervised domain adaptation pipeline for animal pose estimation, which consists of a multi-scale domain adaptation module, a self-distillation module and a mean-teacher network.
- We propose an online coarse-to-fine pseudo label updating strategy to alleviate the negative effect of unreliable pseudo labels.
- Our approach achieves state-of-the-art results on the TigDog dataset and the VisDA2019 dataset, and can also generalize well to unseen domains and unseen animal categories.

2. Related Work

Human Pose estimation. Human pose estimation has been an active research field for decades. One of the most popular early approaches is the pictorial structure [9, 2, 29] which uses a tree structure to model the spatial relationships among body parts. These methods do not perform well in complex scenarios because of the limited representation capabilities. Recently, deep learning based approaches [28, 24, 8, 39, 35, 7, 38, 26] have achieved significant progress due to the availability of large scale training data such as the MPII dataset [1] and the COCO keypoint detection dataset [21]. Existing works can be divided into two categories. The first category [7, 38, 26] adopts a single stage backbone network, typically ResNet [15], to generate deep features, and then upsampling or deconvolution is applied to generate heatmaps with higher spatial resolution. The second category [24, 8, 39, 35] is based on a multi-stage architecture where the generated results from the previous stage are refined step by step. In this paper, we adopt the single stage approach as our basic structure so that we can directly apply domain adaptation to the output of the backbone network.

Animal Pose Estimation. Animal pose estimation is relatively under-explored compared to human pose estimation mainly due to the lack of labeled data. To solve this problem, Mu *et al*. [23] use synthetic animal data generated from CAD models to train their model, which is then used to generate pseudo labels for the unlabeled real animal images. Subsequently, the generated pseudo labels are gradually incorporated into training based on three consistency check criteria. Cao *et al*. [6] propose a cross-domain adaptation scheme to learn a shared feature space between human and animal images such that their network can learn from existing human pose datasets. They also select pseudo labels into the training based on the confidence score. In contrast to [23] which does not need any labels for real animal images, [6] needs part of the real animal images to be labeled in their dataset to facilitate a successful transfer. Similar to [23], we focus on unsupervised domain adaptation from synthetic animal data. Instead of gradually incorporating pseudo labels into training, we conduct an online coarse-to-fine pseudo label update to alleviate the negative effect of noisy pseudo labels.

In addition, there are also several works focusing on 3D animal pose and shape estimation [48, 47, 46, 5, 18, 4, **?**]. [48] builds a statistical 3D shape model SMAL by learning from scans of toy animals. To recover more detailed 3D shape of animals, [47] regularizes the deformation of the mesh from SMAL to constrain the final shape. [46] trains a neural network on a digitally generated dataset to predict 3D pose, shape and texture for the SMAL model.

Unsupervised Domain Adaptation. Unsupervised domain adaptation aims to learn a model from a labeled source domain that can perform well on an unlabeled target domain. One mainstream approach is based on adversarial learning [11, 16, 33, 36], where a feature extractor tries to learn domain-invariant features in order to fool a domain discriminator. The alignment with adversarial learning can facilitate the transfer of labels from the source to the target domain. In addition to feature level alignment, other works also try to reduce the domain shift in the input [16] or output level [32, 40]. In this work, we apply a domain classifier to the feature maps of multiple scales such that both global and local features can be aligned.

Learning from Noisy Data Learning from noisy labels is an important research topic especially for the deep learning community. This is because deep learning algorithms rely heavily on large scale labeled training data that is costly to collect. To reduce the negative effect of noisy labels, some approaches focus on training noise robust models by designing robust losses [12, 34, 44] or by correcting the loss with a transition matrix [27, 13, 37]. Sample selection based approaches [22, 17, 14, 41] attempt to select possibly clean samples in each iteration for training. One of the most representative methods is Co-Teaching [14, 41], which trains on all samples at the beginning and gradually drops the samples with large loss values. This *small-loss* trick , which is based on the *memorization effect* [3, 42] of deep networks, has also adopted by other works [17, 30] to select more reliable labels. Given the noisy pseudo labels, we also conduct sample selection similar to the Co-Teaching. Moreover, we gradually update the pseudo labels with the knowledge from a self-distillation module and a teacher network.

3. Our Method

We propose an unsupervised domain adaptation approach for animal pose estimation. The labeled source domain $\mathcal{S}$ consists of synthetic animal images $I_\mathcal{S}$ and the corresponding pose labels $Y_\mathcal{S}$ generated from CAD models, and the unlabeled target domain $\mathcal{T}$ consists of in-the-wild animal images $I_\mathcal{T}$ without pose labels. The goal is to learn a pose estimation model that can adapt well to the unlabeled target domain. To this end, we design a student-teacher network as shown in Fig. 2. The student and teacher networks share the same architecture: a basic pose estimation module (PEM), a self-distillation module (SDM) and a domain classifier (DC). We first pretrain the PEM on $I_\mathcal{S}$ and use it to generate pseudo labels for $I_\mathcal{T}$. However, these pseudo labels are noisy due to the domain gap between the synthetic and real images, and can hurt the performance when used naively in training. To alleviate this negative effect, we propose an online coarse-to-fine pseudo label updating strategy with the self-distillation module and teacher network.

3.1. Multi-scale Domain Adaptation Module

Our MDAM consists of a pose estimation module and a domain classifier D. The pose estimation module follows an encoder-decoder architecture, where the encoder is the feature extractor G and the decoder is the pose estimator P. Given a pair of images $(I_\mathcal{S}, I_\mathcal{T}) \in \mathbb{R}^{H\times W\times 3}$ from the source and target domains, we feed them into the pose estimation module to get the corresponding feature maps $(F_\mathcal{S}, F_\mathcal{T})$ and heatmaps $(\hat{H}_\mathcal{S}, \hat{H}_\mathcal{T})$:

$$\begin{aligned} F_\mathcal{S} = G(I_\mathcal{S}), \quad \hat{H}_\mathcal{S} = P(F_\mathcal{S}), \\ F_\mathcal{T} = G(I_\mathcal{T}), \quad \hat{H}_\mathcal{T} = P(F_\mathcal{T}). \end{aligned} \tag{1}$$

Similar to human pose estimation [24], we define the animal pose estimation loss in the source domain as the mean-square error (MSE) between the estimated and ground truth heatmaps:

$$\mathcal{L}_\mathcal{S} = \frac{1}{\mathcal{N}} \sum_{i,j,c} \|\hat{H}_\mathcal{S}(i,j,c) - H_\mathcal{S}(i,j,c)\|^2, \tag{2}$$

where $\mathcal{N} = h_o \times w_o \times K$, $H_\mathcal{S}$ represents the ground truth heatmaps with resolution $h_o \times w_o$, and K represents the total number of joints.

We use the pseudo labels $\tilde{H}_\mathcal{T}$ for the target domain since the ground truth for the target domain is not available:

$$\mathcal{L}_\mathcal{T} = \frac{1}{\mathcal{N}} \sum_{i,j,c} \|\hat{H}_\mathcal{T}(i,j,c) - \tilde{H}_\mathcal{T}(i,j,c)\|^2. \tag{3}$$

Note that these pseudo labels $\tilde{H}_\mathcal{T}$ and their corresponding confidence scores $C_\mathcal{T}$ are generated from our pose estimation module pretrained on the source domain data following the training procedure from [23].

To bridge the domain gap between the source and target domains, we apply a domain classifer D [11, 16, 33] to the output of the feature extractor G. The domain classifier attempts to classify the real target data from the synthetic source data using a cross-entropy loss $\mathcal{L}_d$:

$$\mathcal{L}_d = -\log(1 - D(F_\mathcal{T})) - \log(D(F_\mathcal{S})), \tag{4}$$

while the feature extractor tries to fool the domain classifier by maximizing $\mathcal{L}_d$, *i.e.* minimizing:

$$\mathcal{L}_{\text{adv}} = -\mathcal{L}_d. \tag{5}$$

We use a gradient reversal layer [11] for optimization.

We apply the domain classifier to the feature maps at multiple scales given that both local (*e.g.* a small batch around a joint) and global information (*e.g.* the relationship between different joints) are important for joint detection. Specifically, we concatenate the intermediate outputs of the pose estimator and feed them into the domain classifier, as shown in the right part of Fig. 2.

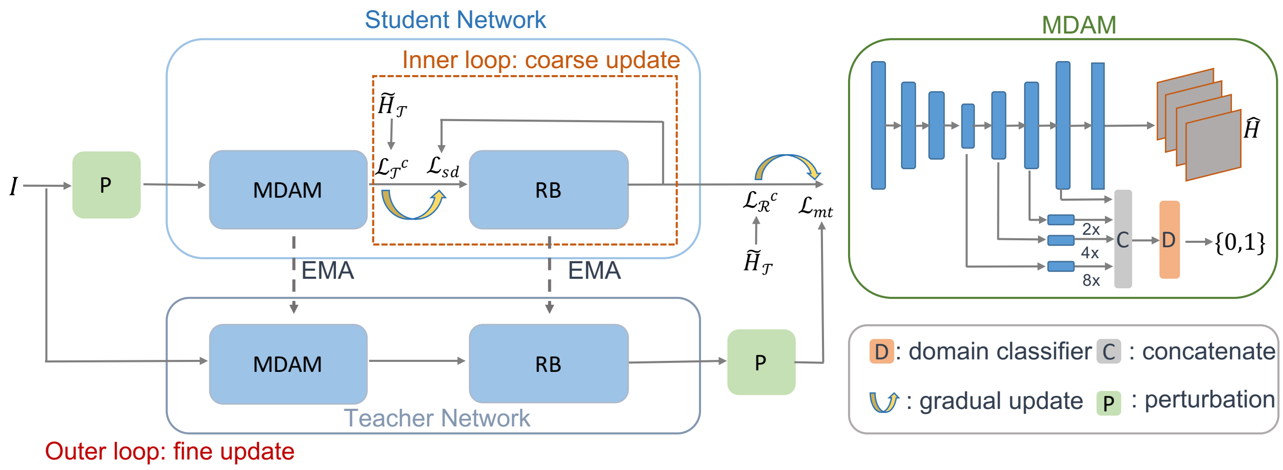

Figure 2. Our network is a student-teacher architecture, where the student network consists of a multi-scale domain adaptation module (MDAM), a refinement block (RB) and a self-feedback loop. We conduct online coarse-to-fine pseudo label update through the inner loop and the outer loop, respectively.

3.2. Coarse-to-Fine Pseudo Label Update

The pseudo labels we use in Eq. 3 are noisy although we filter the samples based on the consistency-check criteria described in [23]. To circumvent this problem, we propose the coarse-to-fine pseudo label updating strategy to gradually replace the noisy pseudo labels with more accurate ones. As shown Fig. 2, our coarse-to-fine pseudo label updating strategy consists of two nested loops.

Inner coarse-update loop: As shown in Fig. 2, the inner loop consists of the self-distillation module: a refinement block (RB) and a self-feedback loop. The input to the refinement block is the output of MDAM $\hat{H}_{\mathcal{T}}$, and we denote its output as $\mathcal{R}_{\mathcal{T}}$. The output of MDAM is supervised by the output of the refinement block via the self-feedback loop with a self-distillation loss:

$$\mathcal{L}_{\text{sd}} = \frac{1}{\mathcal{N}} \sum_{i,j,c} \|\hat{H}_{\mathcal{T}}(i,j,c) - \mathcal{R}_{\mathcal{T}}(i,j,c)\|^2. \tag{6}$$

We also supervise the output of MDAM $\hat{H}_{\mathcal{T}}$ concurrently with the noisy pseudo labels $\tilde{H}_{\mathcal{T}}$, *i.e.*

$$\begin{aligned} \mathbf{L}_{\mathcal{T}} &= \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \mathcal{L}^{c}_{\mathcal{T}}, \quad \text{where } \mathcal{C} = \{c \mid \mathcal{L}^{c}_{\mathcal{T}} < l_{\text{th}}\}, \\ \mathcal{L}^{c}_{\mathcal{T}} &= \frac{1}{\mathcal{M}} \sum_{i,j} \|\hat{H}_{\mathcal{T}}(i,j,c) - \tilde{H}_{\mathcal{T}}(i,j,c)\|^2. \end{aligned} \tag{7}$$

$\mathcal{M} = h_0 \times w_0$, and in contrast to Eq. 3, $\mathcal{L}^{c}_{\mathcal{T}} \in \mathbb{R}^k$ do not sum over c, *i.e.* $\mathcal{L}^{c}_{\mathcal{T}}$ is the loss term per joint. $\mathcal{C}$ is the set of joint indices with a loss value $\mathcal{L}^{c}_{\mathcal{T}}$ smaller than the threshold l_{th}, which dynamically decreases as the training proceeds. This means that we start the training with all the pseudo labels and gradually drop those with large loss values. The intuition is that the clean samples tend to exhibit smaller losses than noisy ones before the network eventually overfit to the noisy ones [3, 42]. On the other hand, we assign a gradually increasing weight to $\mathcal{L}_{\text{sd}}$ in the total loss. This results in a net effect of gradually replacing the initial noisy pseudo labels with better pseudo labels produced by the refinement block $\mathcal{R}_{\mathcal{T}}$ at the later stage of training.

Outer fine-update loop: As shown in Fig. 2, the outer loop is a student-teacher architecture. The student network consists of the multi-scale domain adaptation module and the self-distillation module. The teacher network has an identical architecture with the student network with the exception of the self-feedback loop in the self-distillation module. Furthermore, we follow the mean-teacher [31] paradigm to update the teacher model θ' with the exponential moving average (EMA) of the student model θ:

$$\theta'_t = \alpha \times \theta'_{t-1} + (1 - \alpha) \times \theta_t, \tag{8}$$

where t denotes the training step and α denotes a smoothing coefficient. The output of the teacher network is used to supervise the student network, *i.e.* the output of the refinement block $\mathcal{R}_{\mathcal{T}}$. We apply a random perturbation $\mathcal{P}$ to the input of the student network, and we denote the output of the teacher network as $T_{\mathcal{T}}$. The random perturbation $\mathcal{P}$ is concurrently applied to the output of the teacher network, *i.e.* $\mathcal{P}T_{\mathcal{T}}$. We then enforce the self-consistency loss on the student-teacher network:

$$\mathcal{L}_{\text{mt}} = \frac{1}{\mathcal{N}} \sum_{i,j,c} \|\mathcal{R}_{\mathcal{T}}(i,j,c) - \mathcal{P}T_{\mathcal{T}}(i,j,c)\|^2. \tag{9}$$

$\mathcal{P}$ is generated from random image rotation, flipping, occlusion, and Gaussion noise. Note that we only apply perturbations that will affect the final output to the teacher network, *i.e*. random rotation and flipping. Similar to the self-distillation module, we also concurrently supervise the output of the refinement block $\mathcal{R}_\mathcal{T}$ with the noisy pseudo labels $\tilde{H}_\mathcal{T}$ via the following loss function:

$$\begin{aligned} \mathbf{L}_\mathcal{R} &= \frac{1}{|\mathcal{C}|}\sum_{c\in\mathcal{C}} \mathcal{L}^c_\mathcal{R}, \quad \text{where } \mathcal{C} = \{c \mid \mathcal{L}^c_\mathcal{R} < l_{\text{th}}\}, \\ \mathcal{L}^c_\mathcal{R} &= \frac{1}{\mathcal{M}}\sum_{i,j} \|\mathcal{R}_\mathcal{T}(i,j,c) - \tilde{H}_\mathcal{T}(i,j,c)\|^2. \end{aligned} \tag{10}$$

We use the dynamic threshold l_{th} to gradually drop the large loss terms. Similar to the noisy pseudo label on MDAM loss in Eq. 7, $\mathcal{L}^c_\mathcal{R}$ does not sum over c.

It is shown in [25] that the teacher network is able to provide more stable learning signal than the pseudo labels since it is a temporal ensemble of networks. Therefore, we also add a gradually increasing weight term to $\mathcal{L}_{\text{mt}}$ in the total loss. This means that the outputs of the teacher network are taken to be better pseudo labels to replace the old noisy ones at the later stage of training, and thus preventing overfitting to the noisy pseudo labels.

Remarks: Note that we place the self-distillation module in the inner loop for coarse update since self-distillation is based on the self-feedback loop with a softer regulatory strength compared to the mean-teacher based on self-consistency. It is beneficial to do the softer self-distillation before the stronger outer loop fine updates by the mean-teacher in the nested loops. The softer regulations from self-distillation prevents the mean-teacher from making drastic replacement of the initial noisy pseudo labels too quickly in the training. Consequently, this allows the network to avoid the *memorization effect* [3, 42], and therefore benefit from the noisy pseudo labels at the early stage and then the better pseudo labels at the later stage of training.

3.3. MixUp Regularizer

We further adopt the recently proposed MixUp [43] to enhance the robustness of our network to the noisy pseudo labels. Specifically, MixUp reduces the negative effect of noisy pseudo labels by combining pseudo labels with the ground truth labels. Given a pair of images $(I_\mathcal{S}, I_\mathcal{T})$ from the source and target domains, and the corresponding ground truth and pseudo label heatmaps $(H_\mathcal{S}, \tilde{H}_\mathcal{T})$, we perform MixUp to construct virtual training examples by :

$$\begin{aligned} &\lambda \sim \text{Beta}(\alpha,\alpha), \quad \lambda' = \max(\lambda, 1-\lambda), \\ &I'_\mathcal{S} = \lambda' I_\mathcal{S} + (1-\lambda') I_\mathcal{T}, \\ &H'_\mathcal{S} = \lambda' H_\mathcal{S} + (1-\lambda') \tilde{H}_\mathcal{T}. \end{aligned} \tag{11}$$

$\text{Beta}(\alpha,\alpha)$ is the Beta distribution, where we set both hyperparameters to be α. λ is the parameter to determine the weight of the MixUp from the source and target domains. $I'_\mathcal{S}$ and $H'_\mathcal{S}$ are the input image and label heatmap in the source domain after MixUp. We take the maximum value of $(\lambda, 1-\lambda)$ such that $I'_\mathcal{S}$ is closer to $I_\mathcal{S}$ than to $I_\mathcal{T}$. This is to ensure that the domain label for $I'_\mathcal{S}$ is unchanged after applying MixUp. We also generate virtual example for $I_\mathcal{T}$ by simply changing the max(.,.) to the min(.,.) operator.

3.4. Optimization

The overall objective function to train our network can be expressed as:

$$\mathcal{L} = \mathcal{L}_\mathcal{S} + \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{inner}} + \mathcal{L}_{\text{outer}}, \tag{12}$$

where

$$\begin{aligned} \mathcal{L}_{\text{inner}} &= \lambda_{\text{sd}}\mathcal{L}_{\text{sd}} + \lambda_\mathcal{T}\mathbf{L}_\mathcal{T}, \\ \mathcal{L}_{\text{outer}} &= \lambda_{\text{mt}}\mathcal{L}_{\text{mt}} + \lambda_\mathcal{R}\mathbf{L}_\mathcal{R}. \end{aligned}$$

$\mathcal{L}_\mathcal{S}$ is the fully supervised loss in the source domain (*c.f*. Eq. 2) and $\mathcal{L}_{\text{adv}}$ represents the adversarial loss (*c.f*. Eq. 5). $\mathcal{L}_{\text{inner}}$ consists of the two loss terms in the inner loop: 1) the self-distillation loss $\mathcal{L}_{\text{sd}}$ (*c.f*. Eq. 6) and 2) noisy pseudo labels on MDAM loss $\mathbf{L}_\mathcal{T}$ (*c.f*. Eq. 7). $\mathcal{L}_{\text{outer}}$ is the two loss terms in the outer loop: 1) mean-teacher loss $\mathcal{L}_{\text{mt}}$ (*c.f*. Eq. 9) and 2) noisy pseudo labels on the refinement block loss $\mathbf{L}_\mathcal{R}$ (*c.f*. Eq. 10). Furthermore, the domain classifier concurrently minimizes $\mathcal{L}_d$ (*c.f*. Eq. 4), and the adversarial training is implemented with the gradient reversal layer.

λ_{adv}, λ_{sd}, $\lambda_\mathcal{T}$, λ_{mt}, $\lambda_\mathcal{R}$ are the weights to balance all losses. As mentioned in the previous section, we gradually increase λ_{mt} and λ_{sd} from 0 to their maximum value at the first 10 epochs of training by using a sigmoid-shape function $e^{-5(1-x)^2}$ [31], where $x \in [0,1]$. At the same time, we also decrease the $\lambda_\mathcal{T}$ and $\lambda_\mathcal{R}$ at each epoch until to the minimum value. Note that $\lambda_\mathcal{T}$ and $\lambda_\mathcal{R}$ are responsible for balancing the losses, and play no role in removing the noisy pseudo labels in the training. The dynamic threshold l_{th} in $\mathbf{L}_\mathcal{T}$ and $\mathbf{L}_\mathcal{R}$ is responsible for removing noisy pseudo labels. We determine l_{th} using Algorithm 1, where it is dynamically set to the value of the α_N^{th} smallest value of $\mathcal{L}^c_\mathcal{T}$ or $\mathcal{L}^c_\mathcal{R}$. α_N is the cut-off index, which we initialize to K and gradually decrease it during training.

```
Algorithm 1: Compute Dynamic Threshold $l_{\text{th}}$
Input  : Loss $\mathcal{L}^c_{\text{y}}$ of $K$ joints $\{\mathcal{L}^1_{\text{y}}, \dots, \mathcal{L}^K_{\text{y}}\}$, where
         $y = \mathcal{T}$ (c.f. Eq. 7) or $y = \mathcal{R}$ (c.f. Eq. 10);
         Cut-off index $\alpha_N$
Output: Dynamic threshold $l_{\text{th}}$
// get indices of $\mathcal{L}^c_{\text{y}}$ in ascending order
$\{\text{idx}_1, \dots, \text{idx}_K\} \leftarrow \text{sort\_ascending}(\{\mathcal{L}^1_{\text{y}}, \dots, \mathcal{L}^K_{\text{y}}\})$ ;
// get value of $\mathcal{L}^c_{\text{y}}$ at $c = \text{idx}_{\alpha_N}$
$l_{\text{th}} \leftarrow \mathcal{L}^{\text{idx}_{\alpha_N}}_{\text{y}}$ ;
```

4. Experiments

We use Resnet [15] as our feature extractor G, followed by several deconvolutional layers as the pose estimator P. As in [7], the intermediate feature maps of the pose estimation module are upsampled and then concatenated. The output is fed into both the domain classifier and the refinement block. The domain classifier has a fully-convolutional architecture, which consists of six convolutional layers with leaky Relu as the activation function. The refinement block has one bottleneck block followed by one convolutional layer. We first pretrain the pose estimation module on the synthetic dataset for 100 epochs, and then use it to generate pseudo labels for real images. Both synthetic and real data are used to train the whole network for 80 epochs. The learning rate starts at 0.00025 and is decreased using the polynomial decay with power of 0.9 [32]. Our model is optimized with Adam [19] with default parameters in Pytorch. More training details are included in the supplementary materials.

4.1. Datasets

We train our network with images and pose annotations $\{I_{\mathcal{S}}, H_{\mathcal{S}}\}$ for horse and tiger from the Synthetic Animal dataset [23] and real images $I_{\mathcal{T}}$ from the TigDog dataset[10], and test our model on the test split of the TigDog dataset. We test the generalization capacity of our model on the VisDA2019 dataset, which contains the same animal categories as the TigDog dataset. Moreover, we also test our model on unseen animal categories in the Zebra dataset [46] and the Animal-Pose dataset [6].

Synthetic Animal Dataset: The dataset contains images for five animal categories, including horse, tiger, sheep hound and elephant, with 10,000 images for each animal category. The texture of animals are randomly genrated from the COCO dataset or from the original CAD models.

TigDog Dataset: The dataset provides keypoint annotations for horse and tiger, where the images are taken from YouTube (for horse) and National Geographic documentaries (for tiger). There are 19 keypoints in the dataset, including eyes, chin, shoulders, legs, hip and neck. We only use the images from this dataset for training and evaluate on 18 keypoints that do not include neck as in [23].

VisDA2019 Dataset: The dataset is designed for multi-source domain adaptation and semi-supervised domain adaptation on image classification task. There are in total six domains, including real, sketch, clipart, painting, infograph and quickdraw. [23] manually annotates the keypoints for horse and tiger from the sketch, painting and clipart domains. We use this dataset to test the generalization capacity of our approach to unseen domains.

Zebra and Animal-Pose Datasets: The Zebra dataset contains images of Gravy's zebra, which are collected in Kenya with pre-computed bounding boxes. The Animal-Pose dataset contains annotations for five animal categories: dog, cat, horse, sheep and cow. We use these two datasets to test the generalization capacity of our model on unseen animals from unseen domains.

4.2. Results on the TigDog Dataset

The Percentage of Correct Keypoints (PCK), which reports the percentage of detections that fall within a normalized distance, is used as the evaluation metric following [23]. We train a unified model on all animal categories instead of training one model for each animal category as in [23]. We believe that this is more practical in the real setting. The PCK@0.05 accuracy of our approach, and the existing unsupervised domain adaptation approaches taken from [23] are shown in Tab. 1. 'Real' represents model trained with the real animal pose data and 'Syn' represents model trained only with synthetic data. As can be seen from Tab. 1, our model outperforms existing unsupervised domain techniques by a large margin. For horse category, our approach improves the state-of-the-art CC-SSL by 12.34%, and even outperforms the model trained with real data. For tiger category, we also achieve the best performance among other UDA techniques with an improvement of 5.64% compared to CC-SSL. We did not outperform the supervised model for tiger. The reason is that tigers generally live in forests, where occlusion by surrounding floras happens frequently. However, this kind of occlusion do not occur in the synthetic data, and thus making it very challenging for our model to adapt to the severe occlusion scenario. This also explains why all UDA methods in Tab. 1 show better performance for horse, which lives in the grasslands with lesser occlusions.

4.3. Generalization to Unseen Domains

We test the generalization capacity of our model by directly applying it to the unseen domains in the VisDA2019 dataset. The PCK@0.05 accuracy of our approach for horse and tiger under sketch, painting and clipart domains are shown in Tab. 2. Following [23], we evaluate our model under two settings: 1) The Visible Kpts Accuracy represents accuracy for only visible joints, and 2) the Full Keypoints Accuracy represents accuracy for all joints including self-occluded joints. Both CC-SSL and our approach outperform the model trained on real images, which demonstrates the importance of learning from other domains. Furthermore, our approach also outperforms CC-SSL by a large margin, especially for horse under the painting domain (80.05 *vs*. 73.71, 78.42 *vs*. 71.56) and for tiger under all domains. We also show some qualitative results for horse and tiger in each domain in the first row of Fig. 3.

	Horse Accuracy								Tiger Accuracy							
	Eye	Chin	Shoulder	Hip	Elbow	Knee	Hooves	Mean	Eye	Chin	Shoulder	Hip	Elbow	Knee	Hooves	Mean
Real	79.04	89.71	71.38	91.78	82.85	80.80	72.76	78.98	96.77	93.68	65.90	94.99	67.64	80.25	81.72	81.99
Syn	46.08	53.86	20.46	32.53	20.20	24.20	17.45	25.33	23.45	27.88	14.26	52.99	17.32	16.27	19.29	21.17
Cycgan [45]	70.73	84.46	56.97	69.30	52.94	49.91	35.95	51.86	71.80	62.49	29.77	61.22	36.16	37.48	40.59	46.47
BDL [20]	74.37	86.53	64.43	75.65	63.04	60.18	51.96	62.33	77.46	65.28	36.23	62.33	35.81	45.95	54.39	52.26
Cycada [16]	67.57	84.77	56.92	76.75	55.47	48.72	43.08	55.57	75.17	69.64	35.04	65.41	38.40	42.89	48.90	51.48
CC-SSL [23]	84.60	90.26	69.69	**85.89**	68.58	68.73	61.33	70.77	96.75	90.46	44.84	77.61	**55.82**	42.85	64.55	64.14
Ours	**91.05**	**93.37**	**77.35**	80.67	**73.63**	**81.83**	**73.67**	**79.50**	**97.01**	**91.18**	**46.63**	**78.08**	50.86	**61.54**	**70.84**	**67.76**

Table 1. PCK@0.05 accuracy for the TigDog dataset. 'Real' and 'Syn' represent models trained with the labeled real or synthetic dataset, respectively. All other approaches are trained with the labeled synthetic dataset and the unlabeled real dataset. (Best results in bold)

	Horse						Tiger					
	Visible Kpts Accuracy			Full Kpts Accuracy			Visible Kpts Accuracy			Full Kpts Accuracy		
	Sketch	Painting	Clipart	Sketch	Painting	Clipart	Sketch	Painting	Clipart	Sketch	Painting	Clipart
Real	65.37	64.45	64.43	61.28	58.19	60.49	48.10	61.48	53.36	46.23	53.14	50.92
CC-SSL [23]	72.29	73.71	73.47	70.31	71.56	72.24	53.34	55.78	59.34	52.64	48.42	54.66
Ours	**76.65**	**80.05**	**75.45**	**73.74**	**78.42**	**73.61**	**60.85**	**61.54**	**65.12**	**59.58**	**56.09**	**60.66**

Table 2. PCK@0.05 accuracy for the VisDA2019 dataset. (Best results in bold)

4.4. Generalization to Unseen Animals from Unseen Domains

We further test the generalization capacity of our model in a more challenging scenario, where our model is directly applied to unseen animal categories from unseen domains. Note that our model is trained only with the horse and tiger categories, and we test on both the Zebra and Animal-Pose datasets.

The Zebra dataset contains images of Gravy's zebra collected in Kenya, and 28 keypoints are provided with each image . We only test on the 18 keypoints that are described in the TigDog dataset. The PCK@0.05 accuracy of our proposed approach is shown in Tab. 3. Zebra3D represents the approach used in [46] for 3D zebra pose estimation. This model is trained on a synthetic zebra dataset, which is generated from zebra models with appearance taking from real zebra images. We compare with their results without the post optimization process. The results of CC-SSL are obtained by running their publicly available checkpoint. As they train one model for each animal category, we use the one that gives better accuracy on this dataset. We can see that our approach outperforms CC-SSL with an improvement of 14.3%. Our approach also achieves comparable results to Zebra3D although our model has not been trained on the zebra category. Note that the accuracy of our approach and CC-SSL for joint hip is zero because the joint locations for hip are defined differently for the Synthetic Animal dataset (which is used to train our model) and the Zebra dataset. This is another reason why our approach and CC-SSL are not as good as Zebra3D.

We also test on the 1,000 images from the Animal-Pose dataset, with 200 images for each animal category. All animal categories in this dataset are unseen except for horse. We show our results in Tab. 4, where the results for CC-SSL are from the checkpoint that gives better average accuracy. We can see that our approach can generalize well to unseen animal categories such as sheep and cow, with an accuracy close to horse. The performance of our model for dog and cat is not as good as that for sheep and cow. We attribute this to two reasons: 1) The shape and size of dogs and cats are very different from horses (or tigers), especially for cats with much smaller size. 2) Dog and cat are always in a sit or prone pose, which is not the case for horse or tiger living in the wild environment. We show some failed examples in Fig. 3 for illustration (the last three examples in the last row). We also show qualitative results for each animal category in Fig. 3. We can see that our model successfully estimates some challenging poses, such as the jumping horse, lying down cat and running dog.

4.5. Ablation Study

We conduct ablation study on the TigDog dataset and the results are shown in Tab. 5. We use the multi-scale domain adaptation module as our backbone architecture and train it with only the pseudo labels (mdam+pl) or the supervision from the teacher network (mdam+mt). We also compare with CC-SSL [23], where the authors train the model and update the pseudo label in an iterative way. We can see that our backbone MDAM outperforms CC-SSL because we explicitly enforce the network to learn domain invariant features by applying a domain classifier. The MDAM trained with the teacher network is not as good as the one trained with pseudo labels, and this suggests that the teacher network alone cannot provide enough supervision. The performance is improved by adding the outer fine-update loop (mdam+mt+outlp), where we gradually update the pseudo labels with the teacher network. This demonstrates the importance of our progressive updating strategy, which helps

	Eye	Chin	Shoulder	Hip	Elbow	Knee	Hooves	Mean
Zebra3D*[46]	-	-	-	-	-	-	-	59.5
CC-SSL [23]	60.06	82.29	**30.30**	0	32.45	65.13	61.97	50.07
Ours	**65.33**	**87.50**	23.74	0	**45.32**	**76.02**	**69.77**	**57.23**

Table 3. PCK@0.05 accuracy for the Zebra dataset. * denotes approaches trained with the zebra category. (Best results in bold)

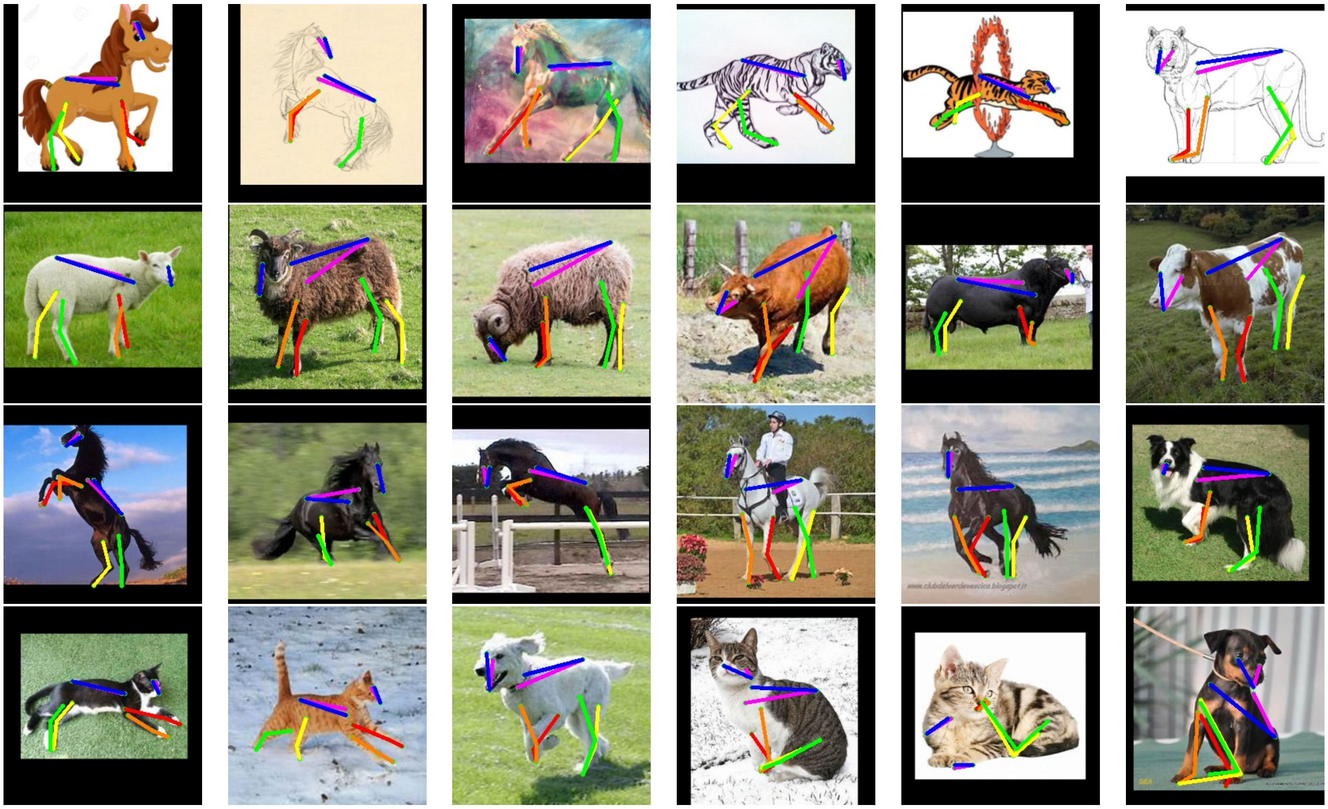

Figure 3. Qualitative results for the VisDA2019 dataset (the first row) and the Animal-Pose dataset (the last three rows).

	Horse	Dog	Cat	Sheep	Cow	Mean
CC-SSL [23]	65.35	30.27	15.05	52.39	63.71	47.60
Ours	**72.84**	**42.48**	**27.65**	**59.51**	**71.31**	**56.77**

Table 4. PCK@0.05 accuracy for the Animal-Pose dataset. All animal category are unseen except for horse.

the network learn from pseudo labels at the early stage and then from the more accurate teacher network. Moreover, the performance is further improved by adding the inner coarse-update loop (mdam+mt+outerlp+inlp). This shows the efficiency of updating the pseudo labels in a coarse-to-fine manner. Finally, our model is further enhanced with the MixUp regularizer (full model).

	Horse	Tiger	Mean
CC-SSL [23]	70.77	64.14	67.52
mdam + pl	74.42	64.90	69.69
mdam + mt	74.74	62.62	68.70
mdam + mt + outlp	78.38	67.15	72.70
mdam + mt + outlp + inlp	78.53	68.01	73.25
full model	79.50	67.76	73.66

Table 5. Ablation study for each component of our network.

5. Conclusion

We propose an approach for unsupervised domain adaptation on animal pose estimation. A multi-scale domain adaptation module is designed to transfer knowledge from the synthetic source domain to the real target domain. In addition, a coarse-to-fine pseudo label updating strategy is further introduced to gradually replace noisy pseudo labels with more accurate ones during training. As a result, we enable our network to benefit from the noisy pseudo labels at the early stage, and the updated labels at the later stage without suffering from the "memorization effect". Extensive experiments on several benchmark datasets show the effectiveness of our approach.

Acknowledgement

This work is supported by the Tier 2 grant MOE-T2EP20120-0011 from the Singapore Ministry of Education.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[2] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *2009 IEEE conference on computer vision and pattern recognition*, pages 1014–1021, 2009.

[3] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In *Proceedings of Machine Learning Research*, pages 233–242, 2017.

[4] Benjamin Biggs, Oliver Boyne, James Charles, Andrew Fitzgibbon, and Roberto Cipolla. Who left the dogs out?: 3D animal reconstruction with expectation maximization in the loop. In *ECCV*, 2020.

[5] Benjamin Biggs, Thomas Roddick, Andrew Fitzgibbon, and Roberto Cipolla. Creatures great and smal: Recovering the shape and motion of animals from video. In *Asian Conference on Computer Vision*, pages 3–19, 2018.

[6] Jinkun Cao, Hongyang Tang, Hao-Shu Fang, Xiaoyong Shen, Cewu Lu, and Yu-Wing Tai. Cross-domain adaptation for animal pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9498–9507, 2019.

[7] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7103–7112, 2018.

[8] Xiao Chu, Wei Yang, Wanli Ouyang, Cheng Ma, Alan L Yuille, and Xiaogang Wang. Multi-context attention for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1831–1840, 2017.

[9] Matthias Dantone, Juergen Gall, Christian Leistner, and Luc Van Gool. Human pose estimation using body parts dependent joint regressors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3041–3048, 2013.

[10] Luca Del Pero, Susanna Ricco, Rahul Sukthankar, and Vittorio Ferrari. Articulated motion discovery using pairs of trajectories. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2151–2160, 2015.

[11] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189, 2015.

[12] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. *arXiv preprint arXiv:1712.09482*, 2017.

[13] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *International Conference on Learning Representations*, 2017.

[14] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pages 8527–8537, 2018.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998, 2018.

[17] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313, 2018.

[18] Angjoo Kanazawa, Shahar Kovalsky, Ronen Basri, and David Jacobs. Learning 3d deformation of animals from 2d images. In *Computer Graphics Forum*, pages 365–374, 2016.

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[20] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6936–6945, 2019.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755, 2014.

[22] Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". In *Advances in Neural Information Processing Systems*, pages 960–970, 2017.

[23] Jiteng Mu, Weichao Qiu, Gregory D Hager, and Alan L Yuille. Learning from synthetic animals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12386–12395, 2020.

[24] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499, 2016.

[25] Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. Self: Learning to filter noisy labels with self-ensembling. 2020.

[26] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4903–4911, 2017.

[27] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017.

[28] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4929–4937, 2016.

[29] Benjamin Sapp, Chris Jordan, and Ben Taskar. Adaptive pose priors for pictorial structures. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 422–429, 2010.

[30] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *International Conference on Machine Learning*, pages 5907–5915, 2019.

[31] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017.

[32] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7472–7481, 2018.

[33] Riccardo Volpi, Pietro Morerio, Silvio Savarese, and Vittorio Murino. Adversarial feature augmentation for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5495–5504, 2018.

[34] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 322–330, 2019.

[35] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[36] Yuan Wu, Diana Inkpen, and Ahmed El-Roby. Dual mixup regularized learning for adversarial domain adaptation. In *European Conference on Computer Vision*, pages 540–555, 2020.

[37] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? In *Advances in Neural Information Processing Systems*, pages 6838–6849, 2019.

[38] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 466–481, 2018.

[39] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation. In *proceedings of the IEEE international conference on computer vision*, pages 1281–1290, 2017.

[40] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li, and Xiaogang Wang. 3d human pose estimation in the wild by adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5255–5264, 2018.

[41] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173, 2019.

[42] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

[43] Hongyi Zhang, M. Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. 2018.

[44] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, pages 8778–8788, 2018.

[45] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[46] Silvia Zuffi, Angjoo Kanazawa, Tanya Berger-Wolf, and Michael Black. Three-d safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5358–5367, 2019.

[47] Silvia Zuffi, Angjoo Kanazawa, and Michael J Black. Lions and tigers and bears: Capturing non-rigid, 3d, articulated shape from images. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3955–3963, 2018.

[48] Silvia Zuffi, Angjoo Kanazawa, David W. Jacobs, and Michael J. Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.