Supplementary Material
SyDog: A Synthetic Dog Dataset for Improved 2D Pose Estimation

Moira Shooter Charles Malleson Adrian Hilton
University of Surrey
Stag Hill, University Campus, Guildford GU2 7XH
{m.shooter,charles.malleson,a.hilton}@surrey.ac.uk

1. Data Generation

Figure 1 shows some example images from the SyDog dataset.

2. Training Set Up

We used a GeForce RTX 2080 Ti for training. The code was implemented with Pytorch Lightning [1]. In this paper we focus on the usage of synthetic data and not the architecture design therefore we trained a 2-Stacked Hourglass network with 2 blocks (2HG), an 8-Stacked Hourglass network with 1 block (8HG) and a pre-trained Mask R-CNN model with a ResNet50 as a backbone from the TorchVision library. For 2HG and 8HG RMSprop was used as an optimiser with a learning rate set to 1×10^{-3} and for the Mask R-CNN we also used an RMSprop but with a learning rate set to 1×10^{-5}. The batch size was set to 32 for the stacked hourglass networks and 16 for the Mask R-CNN, although when the networks were trained with the mixed dataset, which consists of both real and synthetic samples, the batch size was set to 8, 4 of which were real samples and 4 of which were synthetic samples. We applied early stopping to our networks, the networks stop training when the validation does not improve for 10 epochs. The model is saved when there is an improvement in the validation loss. Originally the loss function for the stacked hourglass network would be the mean squared error between the ground truth x_i and all the heatmaps generated by the network y_i but because we only care about the visible keypoints, we modified the loss function by multiplying the keypoints' ground truth visibility v_i with the squared error such that only the visible keypoints contribute to the loss function.

$$MSE_{masked} = \frac{1}{n}\sum_{i=1}^{n} v_i(y_i - x_i)^2, v_i = 0, 1 \quad (1)$$

For the Mask R-CNN's loss function we sum the classification, regression and keypoint loss which our returned by the Mask R-CNN during training.

Learning rate	PCK ↑ (%)	MPJPE ↓ (%)
1×10^{-5}	**50.77**	**20.03**
1×10^{-6}	46.58	21.17
1×10^{-7}	44.87	22.53
1×10^{-8}	41.32	22.53
1×10^{-9}	39.68	23.74

Table 1: Pose estimation results from the Mask R-CNN on the StanfordExtra test dataset when fine-tuned with a smaller learning rate. The performance is evaluated using the percentage of correct keypoints (PCK) with a threshold set to 0.1 and the mean per joint per error (MPJPE) which are both normalised w.r.t. the length of the ground truth bounding box diagonal.

In order to train the stacked hourglass network we represented the joints as 2D heatmaps. The ground truth heatmaps are produced by generating 2D Gaussians with a standard deviation (std) of 3 pixels centered at the joint's location. The stacked hourglass network takes 256x256 RGB-images as input and returns 25 heatmaps of size 64x64. Because the StanfordExtra dataset contains different sized images, it was necessary to resize them to 256x256; there was no need to resize the synthetic images as they were generated to be size 256x256. To train the Mask R-CNN the joints were represented as a list containing the joint's coordinates and visibility. The Mask R-CNN takes 256x256 images as input and returns the predicted bounding boxes, labels, scores of each prediction and the locations of the predicted keypoints. Before feeding the data to the networks, we made sure that each pixel had the same similar data distrubtion by normalizing the data. The StanfordExtra dataset was normalised with a mean=[0.4822, 0.4621, 0.3972] and a std=[0.2220, 0.2172, 0.2167]; while the SyntheticDog dataset was normalised with a mean=[0.6528, 0.4980, 0.5418] and a std=[0.1827, 0.1970, 0.1946].

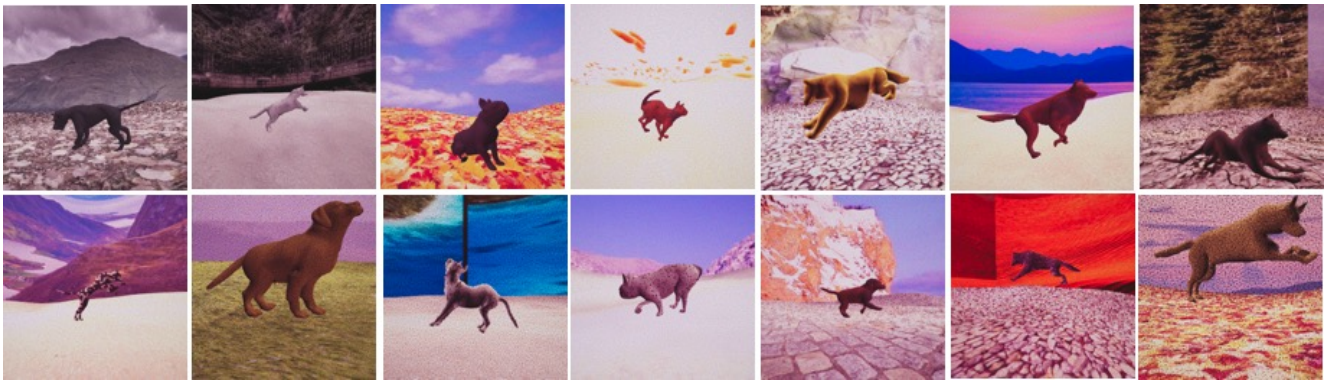
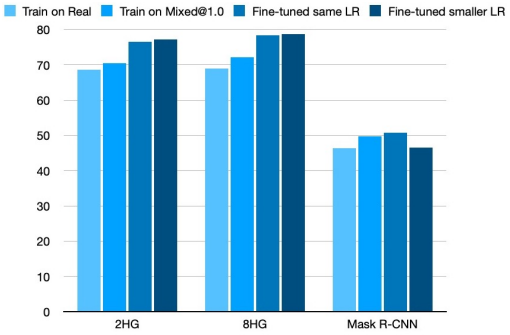
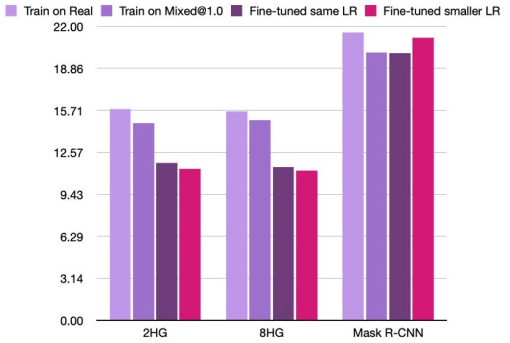


Figure 1: Examples of the SyDog dataset. The data is made varied using different lightning conditions (post-process effects), environments, dog's appearance and camera viewing points.



(a) Bar graph of the percentage of correct keypoints (PCK) between various pose estimation models and training data.



(b) Bar graph of the mean per joint per error (MPJPE) between various pose estimation models and training data.

Figure 2: Quantitative comparison on StanfordExtra test dataset for the 2HG, 8HG and Mask R-CNN trained purely on real data, fine-tuned with real data and trained with the mixed dataset.

3. Experiments and Results

Table 1 shows the pose estimation results from the Mask R-CNN when fine-tuned with smaller learning rates.

References

[1] WA Falcon and .al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019. 1

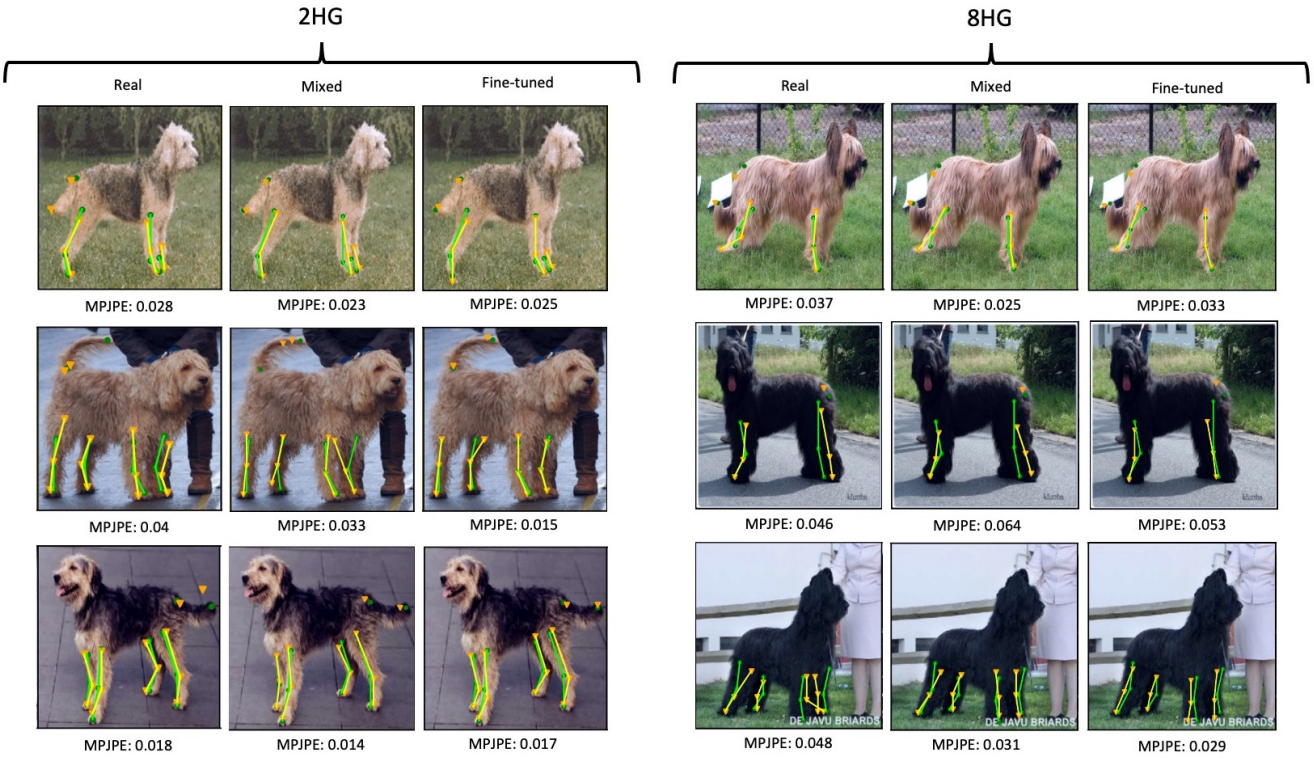


Figure 3: Qualitative comparison on StanfordExtra between the N-stacked hourglass networks trained purely on real data, fine-tuned with real data and trained with the mixed dataset. The ground truth (green) and predicted (yellow) pose with the mean per joint per error (MPJPE) are displayed.