Exploring end-to-end Human Pose Estimation via integrative argmax approximation ("soft-argmax")

Bouchiat Kouroche
kbouchiat@student.ethz.ch

Braun Jona
jonbraun@student.ethz.ch

Kaufmann Andreas
ankaufmann@student.ethz.ch

ABSTRACT

In this project we study the task of 3D human pose estimation from a single RGB image with deep networks. We have to predict the location of 17 joints of the human body, using the two datasets H36M [2, 6] and MPII [1]. For evaluation the metric called Mean per Joint Position Error (MPJPE) in milimeters is used. This work uses the idea of "Integral Human Pose Regression" by Sun et al. [8] as a baseline. We here show how the approach by Sun et al. works and that the application of various augmentations, the Pyramid Pooling Module (PPM) and other modifications lead to an accuracy of **37.30** millimeters on the public leader board of ETH.

1 INTRODUCTION

Human pose estimation is an active sub-field of Computer Vision (CV). It sets its focus on applying CV methods to the problem of detecting human posture. In particular the task can be reduced to localization of human joints with 2D or 3D coordinates given a 2D input image. It is often regarded as an important first step in machine perception of people in images. Convolutional Neural Networks (CNNs) have shown to be very effective in solving many CV tasks, this task is no exception. Nevertheless, detecting 3D joint positions is quite challenging since human postures are difficult to capture with simple algebraic rules and joints can be partially covered or even fully occluded. Insufficient data and, in the case of 3D detection, depth ambiguity are other difficulties that researchers have faced.

One high-performing method which relies on CNNs, consists of generating a heat map for each joint, and detecting joint position by taking the point with the maximum likelihood in the heat map. This method can be extended to 3D by generating 3-dimensional likelihood heat maps where the depth of a joint is encoded through the depth of the most likely point in the output tensor. CNNs can also be used in "direct regression" where the outputs are designed to directly extract 2D or 3D coordinates of each joint, removing the in-between step of detecting joint positions in the heat map. Unfortunately this method has shown to be much less effective in solving 2D human pose estimation. Our method uses a hybrid of both, the heat map and regression methods and is very closely related to the work of Sun et al. in "*Integral Human Pose Regression*" [8].

Our approach is end-to-end in that the network uses heat maps as an intermediate step but outputs regressed 3d-coordinates for each joint. "Argmax" is not a differentiable operator so we cannot simply attach an argmax to the head of the network and flow gradient through it. Instead we rely on so-called "soft-argmax" which attempts to approximate argmax by taking a weighted average of all possible joint coordinates given the joint heat maps. We focused on enhancing the method proposed by Sun et al., by attempting to augment the dataset, making structural changes to the network and also exploring a stochastic approach to localization. Our best model achieved a public accuracy of 37.30 millimeters (MPJPE) on the leaderboard, nearly five millimeters below the hard baseline of 42.20.

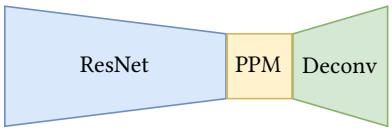

Figure 1: Illustration of our proposed human pose model.

2 METHOD

Our original model, illustrated in figure 1, is very similar to that of Sun et al. [8]. Ours comprises a backbone, a head network and a bottleneck block, which lies between the backbone and the head network. The main difference is the additional bottleneck block, where we employ the Pyramid Pooling Module (PPM), introduced by the PSPNet [9]. Firstly, we use a ResNet [5] backbone, to generate low-level features which are passed through the PPM bottleneck. The bottleneck aggregates global features, using pyramid pooling, which are then upscaled using three transposed convolutional layers in the head network to generate the intermediary joint heat maps. Then a 1x1 convolution is used to output a heat map for each joint.

Finally we apply the "soft-argmax" operation to localize the joints $\mathbf{J}$ in the heat maps. Using similar notation to the one in the Sun et al. paper [8], we define the output x coordinate for the k-th joint as:

$$\mathbf{J}_k^x = \sum_{x=1}^{W} x \cdot (\tilde{\mathbf{V}}_k)_x \tag{1}$$

$\tilde{\mathbf{V}}_k^x$ is the vector containing the likelihood at each x coordinate and is computed as follows:

$$\tilde{\mathbf{V}}_k^x = \sum_{z=1}^{D} \sum_{y=1}^{H} (\tilde{\mathbf{H}}_k)_{x,y,z} \tag{2}$$

We apply the same procedure symmetrically to detect the y and z coordinates. The resolution on depth, height and width, used in equations 1 and 2, are denoted as D, H and W. $\mathbf{H}_k$ is the heat map output of the network and $\tilde{\mathbf{H}}_k$ corresponds to the softmax-ed heat map tensor for the k-th joint:

$$(\tilde{\mathbf{H}}_k)_{x,y,z} = \frac{e^{(\mathbf{H}_k)_{x,y,z}}}{\int_{\mathbf{q}\in\Omega} e^{(\mathbf{H}_k)\mathbf{q}}} \tag{3}$$

This "soft-argmax" is differentiable and allows to directly apply an L1 regression loss to the detected joint coordinates, enabling

end-to-end training. We can also combine this with L2 heat map loss by using a Gaussian blob around the joint position as a ground-truth for the heat maps. However, we find that this decreases model performance when training with both the Human 3.6M [6] and MPII [1] datasets.

Zhao et al. introduced the Pyramid Pooling Module (PPM) in the Pyramid Scene Parsing Network [9]. It proved to be a valuable global contextual prior to extend the effective receptive field of the model. In pyramid pooling we apply global average pooling with different scales using the feature maps produced by the ResNet. The feature maps are pooled into four different bins of size 1, 2, 3 and 6. Then a 1x1 convolution reduces the depth dimension of the bins and bilinear upsampling scales them up to the feature maps input size. Finally the ResNet feature maps and the bins are concatenated and serve as the output of the module. Our hypothesis for the Pyramid Pooling Module is that adding additional global context information by concatenating the pyramid pooled output with the original feature maps of the ResNet can only benefit the model, since we do neither remove nor adjust a layer but only add information along the depth dimension. We argue that increasing the receptive field with the PPM helps the model to understand the global context of the individual joints and hence results in more accurate predictions.

Our best achieving model uses the ResNet-152, pre-trained on the ImageNet[3] dataset. The bottleneck is based on a PPM as described in section 3. Upsampling is performed by transposed convolution using a scale factor of 2, followed by a 2D convolution using 256 4x4 filters and a batch normalization and a Rectified Linear Unit (ReLU) step. We repeat this three times which results in heat maps 8x larger than the features obtained from the PPM. We have a final 1x1 convolution which outputs 17 joint heat map tensors, each with a depth of 72 layers.

The datasets were augmented using random flips and color shifting as described in section 3. We also enabled synthetic randomized image patch occlusion with objects from the Pascal VOC dataset [4].

We used the Adam optimizer with an initial learning rate of 0.0001 ($\beta_1 = 0.9, \beta_2 = 0.999$). The learning rate is scheduled using a multi-step scheduler which scales by a factor of $\gamma = 0.1$ at "milestone" epochs 40 and 60, which we found to be suitable for learning rate reduction. We trained using image patches of size 288x384 with a batch size of 14 (limited by the available GPU resources). The best scoring model obtaining the public score of 37.30 on the ETH leaderboard, it was trained for 70 epochs.

3 EVALUATION

In this section we evaluate our proposed method with the MPJPE metric in milimeter on the validation split of the Human3.6M [2, 6] dataset (contains subject S8). The training is performed on the training split of the Human3.6M dataset (contains subjects S1, S5, S6, S7) and the training split of the MPII 2D [1] dataset. We use the following model and hyperparameters as a baseline for all the experiments: The backbone is the ResNet-50 [5], pretrained on the ImageNet [3]. The head network uses the standart implementation of [8]. For the depth of the heatmaps at the output of the head, we use 64 for each joint. The initial learning rate is always 0.0001, as an optimizer we use Adam. The images are cropped to size 256x256. The batch-size is set to 24.

This baseline achieves a score of **53.30** with the MPJPE metric.

3.1 Experiment 1: Bottleneck block

To analyse the effect of a bottleneck block we use a variety of different structures for that block and evaluate its performance with respect to the model without the block. In this experiments, we applied the augmentation "Horizontal Flip" on the Human3.6M data and increased the depth dimension of the heatmaps to 72.

We compare four different configurations of the PPM which vary in the number of bins used.

- PPM 1: bins = [1, 2, 3, 6] (equal to the PSPNet [9])
- PPM 2: bins = [1, 6, 12, 18]
- PPM 3: bins = [1, 2, 3, 6, 12, 18]
- PPM 4: bins = [1, 2, 3, 6, 12, 18, 24]

Increasing the number of bins reduces the depth dimension of each bin after the 1x1 convolution to keep the computational complexity low. Following this reasoning we expect diminishing returns if we use to many bins.

method	MPJPE	best epoch
Baseline + Horizontal Flip	50.65	55
ResNet-50, PPM, 3xDeconv	**48.46**	41
ResNet-50, PPM 2, 3xDeconv	48.90	58
ResNet-50, PPM 3, 3xDeconv	49.75	41
ResNet-50, PPM 4, 3xDeconv	50.99	41

Table 1: Pyramid Pooling

Table 1 shows that applying the standard form of PPM results in an improvement of **2.2mm** in MPJPE. PPM 2 and PPM 3 also achieve a slight improvement, whereas PPM 4 scores worse compared to the baseline. The result of PPM 4 shows that increasing the number of bins to much, leads to a lower accuracy.

3.2 Experiment 2: Data Augmentations

The Human3.6m dataset was created out of 3.6 Million video frames by 11 professional actors. The images are highly correlated, reasons therefore are the usage of video frames, the small number of actors and the surrounding of the actors, which was always the same room. Therefore the model overfits with high probability. To compensate for the highly correlated data, we evaluate the effect of augmentations. The augmentations are only applied on the Human3.6m dataset (not to the MPII dataset) due to the aforementioned reasons. We use the basic model architecture as described at the beginning of the evaluation section.

We applied each of the following augmentations separately to the base model and evaluate their contribution to the MPJPE score.

- Horizontal flip: The image and ground truth are flipped horizontally with probability 0.5.
- Color shift: We shift all pixel values per channel by a random factor between 0.8 and 1.2.
- The Visual Object Classes (VOC) [4] dataset is used to measure the effect of occluding the training images with objects. Occluding 75 % of the training images has shown to work

well. An image gets occluded with 1-8 objects, object sizes are scaled with a factor of 0.2 to 1.0 to its original size and they are placed at different locations on the training image. These parameters are chosen randomly, uniformely distributed.

augmentation	MPJPE	best epoch
Baseline	53.30	37
Horizontal Flip	**50.64**	37
Color Shift	**49.98**	37
Zoom	62.29	28
Rotation	53.99	31
VOC	**47.98**	55

Table 2: Effect of different data augmentations

From the results in table 2 we see that image occlusion improves the MPJPE the most, by **5.3mm** (compared to the baseline). Horizontal Flip improves the MPJPE by **2.7mm** and Color Shift improves it by **3.3mm**

3.3 Final model

In this section we combine the best results to our final model. In addition to the experiments, we also evaluated on the size of the ResNet and the imagesize. Using ResNet 152 instead of ResNet 50 led to an increase in accuracy of 4.3mm and using the image patches around the center of the person of size 288x384, instead of 256x256, improved the MPJPE by 0.4mm. Therefore, we use a ResNet-152 as our backbone, the PPM as our bottleneck block and 3xDeconv as our head network with a depth dimension of 72 for the heatmap outputs. The image size is set to 288x384 and we apply the augmentations Horizontal Flip, Color Shift, as well as Image Occlusion. The batch size is reduced to 14 such that we do not run out of GPU memory.

In table 3 we report the best achieved MPJPE score on the validation split (subject S8) of Human3.6m as well as the score achieved on the public leader board of ETH which uses a split of subjects S9 and S11.

train dataset	MPJPE (S8)	MPJPE public (S9,S11)
H36m-train, MPII-train	42.64	pending
H36m-trainval, MPII-trainval	–	37.30

Table 3: Our final model results.

4 DISCUSSION

In our empirical evaluation we show that the Pyramid Pooling Module as a bottleneck, appropriate image augmentation, a deeper backbone and larger images lead to a higher accuracy. We further show that combining these methods to our final model results in a significantly higher score than every other method applied separately. However we see multiple ways in how to enhance our final model for even better results.

In-depth experiments. All the experiments in section 3 were executed separately, thus we do not know what the effect of combining two or more of the findings would be. We think an improvement of the experiments would be to combine e.g. PPM with different sizes of the ResNet to find out what its contribution to accuracy is in the final model.

Pyramid Pooling Module. In addition, it would be interesting to lift the restriction of the PPM, to reduce the depth dimension of the bins after the 1x1 convolution and see if that would change the outcome of the experiments. We probably could observe the opposite, that using 7 or more bins decreases the MPJPE more then using only 4 bins.

Bottleneck Block - BotNet. Finally, we would like to discuss an experiment that surprisingly did not lead to an improvement of the MPJPE.
In this experiment we evaluate the recent improvement of the ResNet using Multi-Head Self-Attention (MHSA) called Bottleneck Transformer Network (BoT-Net) [7]. We use the BoT-Net-50 architecture where only the fifth ResNet-50 layer, comprised of 3x3 convolution layers, is replaced with three BoT-Net blocks. A BoT-Net block is made up of three layers. The first is a simple 1x1 convolution, followed by MHSA and the third is again a 1x1 convolution.

Evaluating on the BoT-Net-50 we found that the MPJPE score on the validation subject S8 of the Human3.6m dataset is not improving compared to the ResNet-50. We further observe that PPM again helps to improve the result, compared to the plain BoT-Net. Taking a closer look at the training loss and having in mind that we only used Horizontal Flip as an augmentation, we propose the hypothesis that the BoT-Net is overfitting to the training data, which then leads to worse results on the validation set. One needs to conduct further studies using appropriate data augmentation to verify the effectiveness of the BoT-Net in this scenario.

5 CONCLUSION

In this report we advocate to increase the receptive field of the integral human pose model and empirically demonstrate that using the Pyramid Pooling Module as a bottleneck layer between a backbone and a head network leads to an increase in accuracy. As a second part, we demonstrate that image augmentation such as Occlusion, Horizontal Flips and Color Shift, contribute a lot to model generalization. Combining these methods with integral regression loss, that allows end to end training using jointly 2D and 3D ground truth, the accuracy of **37.30** millimeters on the public ETH leaderboard can be achieved.

A APPENDIX

A.1 Resnet Backbones

probably not enough space, put in one sentence in the section Final model?

backbone	MPJPE	epoch
resnet50	52.61088791540997	..
resnet101	52.23755333589003	42
resnet152	**48.303517520069434**	34

Table 4: Compare of different backbones

Conclusion: The larger the resnet, the better the MPJPE score

A.2 Different image sizes

probably not enough space, put in one sentence in the section Final model?

Img. size	Val MPJPE	epoch
256x256	48.303517520069434	34
288 x 384	**47.90724288592864**	41

Table 5: Different image sizes

Conclusion: Slight improvement but less then stated in paper

A.3 Augmentations - failed

These augmentations did not improve the score.

- Zoom: We randomly zoom in or out by a random factor between 0 and 25%.
- Rotate: We rotate the image with probability 0.6 left or right by a random factor of 0-30 degrees.

augmentation	MPJPE	epoch
No Augmentation	53.29591	37
Rotation	53.98505	31
Zoom	62.29041	28

Table 6: Effect of different data augmentations

We assume that the augmentation "Zoom" can lead to joints not being part of the image patch and that leads to a worse result in the reported score. For "Rotation" we argue that rotating an image which captures human actions leads to unnatural viewing angles, which causes the result not to improve.

A.4 Atrous Spatial Pyramid Pooling

One of them was Atrous Spatial Pyramid Pooling (ASPP) ... Do not report ASPP, because it does not improve significantly? If we need more text we could do it.

bottleneck	MPJPE	epoch
-	**50.64760**	55
ASPP	50.67867	51
ASPP 2	49.50291	44
ASPP 3	50.32215	41

Table 7: ASPP Compare

REFERENCES

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2014. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 3686–3693. https://doi.org/10.1109/CVPR.2014.471

[2] Cristian Sminchisescu Catalin Ionescu, Fuxin Li. 2011. Latent Structured Models for Human Pose Estimation. In *International Conference on Computer Vision*.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2012. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[6] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. 2014. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 7 (2014), 1325–1339. https://doi.org/10.1109/TPAMI.2013.248

[7] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. 2021. Bottleneck Transformers for Visual Recognition. arXiv:cs.CV/2101.11605

[8] Xiao Sun, Bin Xiao, Shuang Liang, and Yichen Wei. 2017. Integral Human Pose Regression. *CoRR* abs/1711.08229 (2017). arXiv:1711.08229 http://arxiv.org/abs/1711.08229

[9] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. 2017. Pyramid Scene Parsing Network. arXiv:cs.CV/1612.01105

ACRONYMS

BoT-Net Bottleneck Transformer Network. 3

CNN Convolutional Neural Network. 1
CV Computer Vision. 1

ETH Eidgenössische Technische Hochschule. 1–3

GPU Graphics Processing Unit. 2, 3

MHSA Multi-Head Self-Attention. 3
MPJPE Mean per Joint Position Error. 1–3

PPM Pyramid Pooling Module. 1–3

ReLU Rectified Linear Unit. 2

VOC Visual Object Classes. 2