Machine Perception Report

Fatjon Zogaj
fzogaj@student.ethz.ch

Patrik Okanovic
pokanovic@student.ethz.ch

Rafael Sterzinger
rsterzinger@student.ethz.ch

ABSTRACT

This paper compares a variety of different architectures and techniques for the challenge of monocular 3D human pose estimation challenge on the H36M [9] dataset. We propose to make use of standard augmentation and occlusion [16], volumetric heatmap regression [15] in combination with a pretrained ResNext101 [17] backbone for the highest score. Human joints follow the underlying structure of a graph which is why we also examine Graph Convolutional Network [19] which we combine with evolutionary training data [10]. Our best ResNext101 backed model is able to achieve state of art.

1 INTRODUCTION

Human pose estimation (HPE) from a single monocular image is an open challenge garnering significant attention in computer vision [11] as it offers a wide variety of useful applications in real-life. For instance, HPE is utilized in human tracking, human-computer interaction, sports motion analysis, and self-driving cars [4]. Estimating the full-body pose has been extensively studied in both 2D and 3D [14]. Thanks to the availability of large-scale 2D annotated human pose datasets and the emergence of deep neural networks achieving very low error rates [3, 12, 18], the 2D human pose estimation problem can be considered as nearly solved. In contrast, the 3D human pose estimation remains an open problem. This is partially because of the ambiguity of recovering the 3D information from single images and also due to the lack of large-scale 3D pose annotated datasets. Methods for solving the 3D human pose estimation can roughly be divided into two categories: i) learning end-to-end networks that recover 2D input images to 3D poses directly, ii) extracting 2D human poses from input images and then lifting 2D poses to 3D space [6].

2 METHOD

This section describes the methods used in our final model which includes preprocessing steps, augmentations, changes of representation, parameters, and the architecture of the model.

2.1 Data augmentation

Despite the recent success of deep models, many of them still require a large amount of data to perform well. Our provided dataset is limited in size, therefore we perform data augmentation as follows. The input pictures are normalized and patched to a size of 256 x 256. Later, we increased this value to 288 x 384 to allow for an even more accurate joint location prediction. Standard data augmentation includes rotation ±30 degrees, horizontal flipping, scaling by a factor between 0.75 and 1.25, and color scaling between 0.8 and 1.2. Overall, frames are rotated with a probability of 60% and flipped with a probability of 50%.

2.2 Data occlusion

Furthermore, we use synthetic occlusion to make the network robust to occluded joints, since erasing objects or simply putting them in the picture has been shown to improve results in image classification, and object detection in general [16]. Using objects from the Pascal VOC [5] dataset we filter out persons, segments with area below 500px and segments labeled as *difficult* or *truncated*. Finally, between 1 and 8 objects from the filtered VOC dataset are added to random locations of the frame.

2.3 Backbone Model

The ever-increasing amount of available data has given rise to huge network architectures, and the effective use of transfer learning. As such we have experimented with various ResNet [8] and ResNext models and settled on the ResNext101 [17] architecture as it achieved the best training scores and times during our runs. Instead of the last pooling and fully connected layers, we stack 3 deconvolution layers and a final convolutional layer on top of the ResNext101 backbone. This is described in more detail in the next section.

During training time we have tried freezing different components of the architecture and settled on freezing the first 7 layers, leaving the last sequential block of the ResNext101 backbone and our added head open for training.

2.4 Heatmap Regression

Another important development in the task of HPE was the introduction of heatmap representations as an alternative to directly regressing the joint locations. In this representation each location in the heatmap represents a probability of the joint being at a certain position [15]. Given such a heatmap representation $\mathbf{H}_k$ for the k^{th} joint, the position p of the joint $\mathbf{J}_k$ is estimated by the maximum likelihood $\mathbf{J}_k = \underset{\mathbf{p}}{\text{argmax}}\, \mathbf{H}_k(\mathbf{p})$. In our case, such a heatmap representation is learnt by a shallow head which has been described in the previous section. This head network first upsamples the extracted features to a certain resolution (64 x 64 by default) and then produces K heatmaps. For almost all of our experiments we used 64 heatmaps per joint, however, as this value directly corresponds to the granularity of the z-dimension, we later increased this value to 72.

This approach yields good performance in HPE but has some drawbacks, mainly the usage of the *argmax* operator that makes the calculation of the maximum likelihood non-differentiable and, thus hinders the possibility of training a neural network end-to-end. Sun et al. [15] alleviate these problems by modifying the joint calculation to "taking-expectation" instead of "taking-maximum". With this, the above equation in the discrete case can be formulated as $\mathbf{J}_k = \sum_{p_x=1}^{D} \sum_{p_y=1}^{H} \sum_{p_x=1}^{W} \mathbf{p} \cdot \mathbf{H}_k(\mathbf{p})$ which is also known as the *soft argmax*. This formulation can further be modified which then allows to calculate the expected joint position in each dimension

separately. A benefit from this is the additional option to train on 2D (MPII) and 3D (H36M) data simultaneously and end-to-end.

Given the fact that the first [16] and second [13] place of the 2018 ECCV PoseTrack Challenge made use of this heatmap representation with the *soft argmax*, we employed this technique as well.

3 OTHER APPROACHES

In the following section we list different techniques which performed worse than the ones included in our best model.

3.1 Graph Neural Networks

As the task of 2D human pose estimation has been extensively studied, achieving very low error rates [3, 12, 18], we also take a look at an architecture for 3D-joint regression based on 2D-joint input. A human skeleton follows the underlying structure of a graph which is why a Graph Convolutional Network (GCN) offers itself well to model this. Making use of Zhao et al.'s Semantic GCN (SemGCN) model allows us to capture semantic information in regard to global and local joint relations [19].

As the original SemGCN model utilizes joint grouping (maxpool: 16 joints $\rightarrow$ 8 joints), we adapt it for our use case of 17 joints. We have looked at a variety of ways on how to incorporate the missing joint (Neck/Nose). These options include i) simply concatenating the missing joint to the other grouped ones, ii) grouping the missing joint with itself, iii) grouping the missing joint with other related joints and iv) removing grouping altogether. We also played around with reordering the joints to put more emphasis on one over another as max-pooling makes use of locality.

Two different training approaches offer themselves well for this. In the first one, the training of the SemGCN can be separated into two parts. We first train the SemGCN to predict the z-axis based on 2D ground truth input data. Afterwards we use this pre-trained model and stack it on top of our also pre-trained backbone, dropping the third dimension of the backbone output to feed it into the SemGCN. The second approach simply stacks the untrained SemGCN on top of the backbone and trains it together from scratch.

On top of that we differentiate between the case of freezing the backbone model completely, and only freezing its last layer and respective head.

3.2 Evolutionary Training Data

In order to train the mapping of joints from 2D to 3D we have used an evolutionary technique for augmenting the dataset as proposed in [10]. We represent the joints of the human 3D skeleton by a set of bones organized hierarchically in a kinematic tree. This way the dependence of adjacent joints with tree edges is captured.

The overall algorithm can be seen in 1.

With algorithm 1 we create 2D-3D pairs for teaching the model to map from 2D to 3D. Crossover is implemented as a random exchange of sub-trees of bones. The mutation operator perturbates the local orientation of one bone vector for a certain degree. Mutation can easily make an invalid pose, therefore we use natural selection to remove invalid poses. Natural selection is implemented using a function which indicates which pose is invalid and gives that pose a value of $-\infty$ making it unable to be selected into the new

Algorithm 1: Data evolution

input: Initial population of 3D joints $D_{old} = \{\mathbf{p}\}_{i=1}^{N}$, noise σ, number of generations G

$D_{new} = D_{old}$

for *i=1:G* **do**

  Parents=Sample(D_{new})

  Children=NaturalSelection(Mutation(Crossover(Parents)))

  $D_{new} = D_{new} \cup Children$

return D_{new}

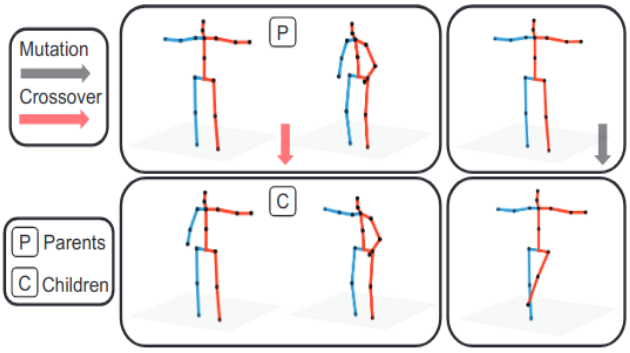

Figure 1: Examples of evolutionary operation. Crossover and mutation take two and one random samples respectively to synthesize novel human skeletons. In this example the right arms are selected for crossover while the left leg is mutated., source [10]

generation. This function uses the prior angle constraints defined from [2]. Both mutation and crossover can be seen in Fig. 1

3.3 Miscellaneous

Although ensemble methods were not allowed, using dropout layer were an exception. Hence, we have experimented using a dropout layer after the heatmaps and the feature extraction part, however it did not improve our score compared to our best model.

Before evaluating the SemGCN architecture to lift our 2D joint predictions to 3D, we explored the idea of utilizing a de-noising autoencoder as a refinement step to our 3D joint predictions. We borrowed this idea from the authors Ghosh et al. [7] which use this component to adjust unfeasible poses from human motion predictions. After some testing of this idea, we learned about the SemGCN architecture which seemed more promising and hence stopped further exploration.

Both, the de-noising autoencoder and the SemGCN, focus more or less on capturing the overall human pose composition. This idea was already explored in a different way by Sun et al. [14] where they proposed a loss function which incorporates information of the skeleton. Instead of calculating the loss as $\sum_{k=1}^{K} ||\tilde{\mathbf{J}}_k - \mathbf{J}_k||_1$ which only considers the joint position they proposed a loss which incorporates bone lengths. Given a bone, $\mathbf{B}_k = \mathbf{J}_{parent(k)} - \mathbf{J}_k$, the loss is formulated as $\sum_{k=1}^{K} ||\tilde{\mathbf{B}}_k - \mathbf{B}_k||_1$. Even tough this should e.g.

avoid unrealistic poses, incorporating this loss did not improve our score.

4 EVALUATION

We evaluate models for 10 to 30+ hours, starting with a learning rate of 0.0001 and decaying it once the loss seems to have converge. For our final model we trained for 65 epochs on both the MPII and the H36M dataset simultaneously with a batch size of 16. Here, the learning rate was decayed by 0.1 after 55, 60, and 65 epochs.

Table 1: Ablation study on the progression of the Mean Per Joint Position Error (MPJPE) metric. The score denotes the performance on the test set. Values with $*$ denote the performance on the validation set. Data augmentation (A), Bone Loss (B), Evolved Data (D), Extra Data (E), Remove Joint Grouping (G), Higher Resolution (H), Integral Heatmap Regression (I), Data occlusion (O), Ground Truth Pre-Training (P), 3D-Regression (R), SemGCN Head (S), Last Layer Unfrozen (U), 3D SemGCN Input (3)

Backbone	Extensions	MPJPE$_{test}$
AlexNet	R	141.50
AlexNet	O	84.50
ResNet50	I	72.15
ResNet50	I,O	63.01
ResNext101	I,O,A	54.14
ResNext101	I, O, A, S, 3	57.43
ResNet152	I,O,A,H,E	50.94
ResNext101	I,O,A,H,E	46.48
ResNext101	I,O,A,B	54.60*
SemGCN		66.87*
SemGCN	P	60.08*
SemGCN	P, U	54.12*
SemGCN	P, U, G	46.05*
SemGCN	P, U, G, D	45.68*

Synthetic occlusion. Using the given sample code with the AlexNet as the backbone of the model synthetic occlusion gives an impressive improvement to the score lowering it by 40.28%.

Model backbone. A stronger backbone plays a significant role when it comes to improving the score. Using the ResNet50 with integral heatmap regression improved the score from the initial model by 49.01%. Adding synthetic occlusion to that gives a score which has 12.69% lower MPJPE on the test set. Furthermore, using an even more powerful backbone such as ResNext101 with integral heatmap regression, synthetic occlusion and standard augmentation improved the model to a score of 54.14 MPJPE. One can compare using ResNext101 to ResNet152 with identical extensions as shown in 1 to see that ResNext101 improves the score by 3.91%.

Larger image size/depth resolution/dataset. These three extension were added simultaneously and, thus it is difficult to tell which one impacted the score by how much. However, is was clear that these extensions have a positive impact [15]. In our case, we could improve our performance on the MPJPE from 54.14 to 46.48 which

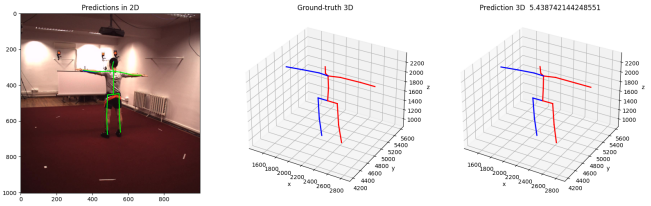

Figure 2: Visualization of a random sample of the H36M Dataset. Left: Input picture with overlaid 2D ground truth and prediction; Middle: ground truth; Right: Our prediction

is a decrease by around 14%. Regarding the changes to the image size, the resolution was increased from 256 x 256 to 288 x 384. For the depth resolution/the amount of heatmaps produced per joint, the value was increased from 64 to 72. As we wanted to utilize all the resources available to us, the final model was trained on the training and validation set of both the MPII and the H36M.

SemGCN. In regards to the SemGCN model, only the scores of the z-axis matter as the other 2D input is the ground truth. Thus, the following error scores for the SemGCN are not directly comparable to the others. Our first experiments showed that when not pre-training the SemGCN model on ground truth data, it is able to achieve a validation score of 66.87. This can be brought down to 60.08 by incorporating pre-training. As a next step we then analyzed the effect of unfreezing the last layer of the backbone which achieves a score of 54.12. Unfreezing the SemGCN without any pre-training on the other hand got an error of around 65.5. This could be explained due to the fact that we stop training the model if it seems to perform badly early on, which is the case for non-pre-trained models as these take longer to train. Like previously mentioned, we compare different ways of adding the missing 17th joint. Surprisingly the grouping methods even in combination with joint reordering perform similarly bad with scores around 60 using pre-trained models. Removing the grouping altogether seems to prove most useful, being able to get a validation error of 46.05.

5 DISCUSSION

In section 2 and 3 we showcased that we explored a wide variety of approaches in order to solve the task of HPE. In comparison to section 2, section 3 presented approaches which performed worse than our best model. We think that the main reason for this was the inability of our backbone to predict close to the 2D ground truth which hindered the presented methods to show their full potential.

6 CONCLUSION

In this paper we compare a variety of different architectures and techniques for the task of 3D human pose prediction on the H36M dataset [9]. We provide an extensive benchmark study analyzing direct image-to-3D-joint prediction as well as image-to-2D-joint prediction with a final 3D regression.

Our best model makes use of augmentation and occlusion, volumetric heatmap regression in combination with a pretrained ResNext101 backbone and is able to beat the state of the art in the monocular, single frame H36M benchmark [1, 10].

REFERENCES

[1] [n. d.]. Papers With Code Monocular 3D Human Pose Estimation on Human3.6M. https://paperswithcode.com/sota/monocular-3d-human-pose-estimation-on-human3. Accessed: 2021-06-20.

[2] Ijaz Akhter and Michael J. Black. 2015. Pose-Conditioned Joint Angle Limits for 3D Human Pose Reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[3] Adrian Bulat, Jean Kossaifi, Georgios Tzimiropoulos, and Maja Pantic. 2020. Toward fast and accurate human pose estimation via soft-gated skip connections. *CoRR* abs/2002.11098 (2020). arXiv:2002.11098 https://arxiv.org/abs/2002.11098

[4] Yucheng Chen, Yingli Tian, and Mingyi He. 2020. Monocular Human Pose Estimation: A Survey of Deep Learning-based Methods. *Computer Vision and Image Understanding* 192 (March 2020), 102897. https://doi.org/10.1016/j.cviu.2019.102897 arXiv: 2006.01423.

[5] M. Everingham, L. Gool, Christopher K. I. Williams, J. Winn, and Andrew Zisserman. 2009. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision* 88 (2009), 303–338.

[6] Haoshu Fang, Yuanlu Xu, Wenguan Wang, Xiaobai Liu, and Song-Chun Zhu. 2017. Learning Knowledge-guided Pose Grammar Machine for 3D Human Pose Estimation. *CoRR* abs/1710.06513 (2017). arXiv:1710.06513 http://arxiv.org/abs/1710.06513

[7] Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. 2017. Learning Human Motion Models for Long-term Predictions. *arXiv:1704.02827 [cs]* (Dec. 2017). http://arxiv.org/abs/1704.02827 arXiv: 1704.02827.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *CoRR* abs/1512.03385 (2015). arXiv:1512.03385 http://arxiv.org/abs/1512.03385

[9] 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* (2014).

[10] Shichao Li, Lei Ke, Kevin Pratama, Yu-Wing Tai, Chi-Keung Tang, and Kwang-Ting Cheng. 2020. Cascaded Deep Monocular 3D Human Pose Estimation With Evolutionary Training Data. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[11] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G. Derpanis, and Kostas Daniilidis. 2016. Coarse-to-Fine Volumetric Prediction for Single-Image 3D Human Pose. *CoRR* abs/1611.07828 (2016). arXiv:1611.07828 http://arxiv.org/abs/1611.07828

[12] Zhihui Su, Ming Ye, Guohui Zhang, Lei Dai, and Jianda Sheng. 2019. Improvement Multi-Stage Model for Human Pose Estimation. *CoRR* abs/1902.07837 (2019). arXiv:1902.07837 http://arxiv.org/abs/1902.07837

[13] Xiao Sun, Chuankang Li, and Stephen Lin. 2018. An Integral Pose Regression System for the ECCV2018 PoseTrack Challenge. *arXiv:1809.06079 [cs]* (Sept. 2018). http://arxiv.org/abs/1809.06079 arXiv: 1809.06079.

[14] Xiao Sun, Jiaxiang Shang, Shuang Liang, and Yichen Wei. 2017. Compositional Human Pose Regression. *arXiv:1704.00159 [cs]* (Aug. 2017). http://arxiv.org/abs/1704.00159 arXiv: 1704.00159.

[15] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. 2018. Integral Human Pose Regression. *arXiv:1711.08229 [cs]* (Sept. 2018). http://arxiv.org/abs/1711.08229 arXiv: 1711.08229.

[16] István Sárándi, Timm Linder, Kai O. Arras, and Bastian Leibe. 2018. Synthetic Occlusion Augmentation with Volumetric Heatmaps for the 2018 ECCV PoseTrack Challenge on 3D Human Pose Estimation. *arXiv:1809.04987 [cs]* (Nov. 2018). http://arxiv.org/abs/1809.04987 arXiv: 1809.04987.

[17] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2016. Aggregated Residual Transformations for Deep Neural Networks. *CoRR* abs/1611.05431 (2016). arXiv:1611.05431 http://arxiv.org/abs/1611.05431

[18] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. 2020. TransPose: Towards Explainable Human Pose Estimation by Transformer. *CoRR* abs/2012.14214 (2020). arXiv:2012.14214 https://arxiv.org/abs/2012.14214

[19] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N. Metaxas. 2019. Semantic Graph Convolutional Networks for 3D Human Pose Regression. *CoRR* abs/1904.03345 (2019). arXiv:1904.03345 http://arxiv.org/abs/1904.03345