3D Human Pose Estimation

Amr Eltelt*
Diploma of Advanced Study in Data Science
aeltelt@student.ethz.ch

Pirmin Philipp Ebner*
Diploma of Advanced Study in Data Science
ebnerp@student.ethz.ch

ABSTRACT

We address the 3D Human Pose Estimation by combining a Resnet model and a linear model. The accuracy on the mean per joint position error (MPJPE) in millimeter on the reduced version of "Human3.6mm" is 112.1 of our method using the Resnet model to predict 2d pose from images and then use a linear model to predict 3d pose from 2d pose.

1 INTRODUCTION

Detecting 3D human poses is of great importance in many areas including robotics, human computer interaction and autonomous driving. In this project we study the task of 3D human pose estimation from a single RGB image with deep networks.

A reduced version of "Human3.6m" dataset [8] is used for training and validation. This dataset contains images of human performing various actions, as well as the ground-truth 2D and 3D pose annotations.

Given an image - a 2-dimensional representation - of a human being, 3D pose estimation is the task of producing a 3-dimensional figure that matches the spatial position of the depicted person. In order to go from an image to a 3D pose, an algorithm has to be invariant to a number of factors, including background scenes, lighting, clothing shape and texture, skin color and image imperfections, among others.

We explore the power of decoupling 3D pose estimation into the well studied problems of 2D pose estimation [12, 16], and 3D pose estimation from 2D joint detections. Separating pose estimation into these two problems gives us the possibility of exploiting existing 2D pose estimation systems, which already provide invariance to the previously mentioned factors.

In our project, we used either the ResNet model [1] or the method of Newell et. al. [12] applying the stacked hourglass for predicting the 2D pose from a given 2D image. The method uses features which are processed across all scales and consolidated to best capture the various spatial relationships associated with the body. Afterwards, we used the system of Martinez et. al. [9] that given 2D joint locations predicts 3D positions. The network is based on a simple, deep, multilayer neural network with batch normalization [7], dropout [15] and Rectified Linear Units (RELUs) [11], as well as residual connections [3].

2 METHODS

To arrive at the final architecture we went through a number of iterations. We found that image augmentation (brightness, contrast, hue, rotation and flipping) improved our model and we reached the easy baseline. However, the training error for these models stayed too high. We had high bias.

*Both students equally contributed to this project.

Therefore, we decided to use a highly complex model in combination to predict first 2D pose and afterwards the 3D pose. We used the given Resnet model to predict 2D pose from the images and combine this with a linear model proposed by Martinez et. al. [9] to predict the 3D pose afterwards. The new model helped us to reach the hard baseline.

In the paper by Martinez et. al. [9] they used the stacked hourglass model to predict the 2D pose from the images. In the next step we decided to implemented the stacked hourglass model by Newell et. al. [12]. The model is based on heatmaps to precissly predict the 2D pose.

2.1 Predicting 2D pose

Estimating 2D human pose is hard because people appear in a wide range of poses and have varying body shape. They wear varied clothing and the articulation results in significant self occlusion. We have tested several state-of-the-art methods to address these problems.

2.1.1 *ResNet50 Model [1].*

Residual neural networks (ResNet) is a kind of neural network that builds on constructs known from pyramidal cells in the cerebral cortex. It is done by utilizing skip connections, or short-cuts to jump over some layers. The model is implemented with double- or triple- layer skips that contain nonlinearities (ReLu) [11] and batch normalization in between [7]. An additional weight matrix is used to learn the skip weights.

Model Training. We trained the stacked hourglass on the provided 2D images and 2D pose ground truth training set. No subjects are excluded from train set for validation, since we would like to include as much examples as possible for training. We do data augmentation that includes rotation (+/- 30 degrees), add occlusion [14] and randomly change brightness, contrast and applied hue. We trained the network for 5 epochs using the momentum optimizer with a starting learning rate of 0.003 and batches of size 8.

The code was given by the TA and integrated to the provided project skeleton.

2.1.2 *Stacked Hourglass Model [12].*

The network captures and consolidates information across all scales of the image. The design is similar to hourglass based on our visualization of the steps of pooling and subsequent up-sampling used to get the final output of the network. This produce pixel-wise outputs, the hourglass network pools down to a very low resolution, then upsamples and combines features across multiple resolutions. The hourglass module before stacking is related to conv-deconv and encoder-decoder architectures and multiple hourglasses are stacked end- to-end, feeding the output of one as input into the next. The key to this approach is the prediction of intermediate heatmaps upon which we can apply a loss. One hourglass is set-up following:

- Convolutional and max pooling layers are used to process features down to a very low resolution.
- At each max pooling step, the network branches off and applies more convolutions at the original pre-pooled resolution.
- After reaching the lowest resolution, the network begins the top-down sequence of upsampling and combination of features across scales.
- Nearest neighbor upsampling of the lower resolution followed by an elementwise addition of the two sets of features are done to bring together information across two adjacent resolutions.
- The topology of the hourglass is symmetric, so for every layer present on the way down there is a corresponding layer going up.

After reaching the output resolution of the network, two consecutive rounds of 1x1 convolutions are applied to produce the final network predictions. The output of the network is a set of heatmaps where for a given heatmap the network predicts the probability of a joint's presence at each and every pixel.

Model Training. We trained the stacked hourglass on the provided 2D images and 2D pose ground truth training set. No subjects are excluded from train set for validation, since we would like to include as much examples as possible for training. We do data augmentation that includes rotation (+/- 30 degrees), add occlusion [14] and randomly change brightness, contrast and applied hue. We trained the network for 5 epochs using the rmsprop optimization with a starting learning rate of $2.5e^{-4}$ and batches of size 8.

The code is referenced from the Tensorflow implementation [4, 6] and integrated to the provided project skeleton.

2.2 Predicting 3D pose

Recent work show that inferring 3D joints from ground truth 2D projections can be solved with a surprisingly low error rate - 30% lower than state of the art - on the largest existing 3D pose dataset [9].

2.2.1 *Linear Model.*

For 3D pose estimation a linear model is used as proposed by Martinez et. al. [9]. The main contribution of this method is the intuitive network design, fast computation, and good performance. All of which, makes it an attractive approach to deploy for our task. Interestingly, the approach shows that predicting a relatively accurate 3D key points from 2D detections, which carry less information, is a solvable problem in contradiction to what is perceived by previous work [10] [13]. The linear model consists of 6 linear layers:

- 1 layer applied directly to the input to increase dimensionality to 1024
- 4 layers, each with 1024 units and followed by blocks of batch normalization, max-norm constraint, ReLu activation, and dropout 0.5. Each two layers are wrapped in a residual connection
- 1 layer that produces outputs of size 3n

The system learns a function $f^*: R^{2n} \rightarrow R^{3n}$ that minimizes prediction error over a dataset N poses:

$$f^* = \min_f \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(f(x_i) - y_i) \tag{1}$$

In [9], the model is trained and tested on either the input 2D detections obtained from the pre-trained stacked hourglass network [12] or from a calculated 2D projection. The output is the Human3.6M 3D pose ground truth. The later setting achieves their best performance.

Model Training.

We train the linear model on the provided 2D and 3D pose ground truth training set. No subjects are excluded from train set for validation, since we would like to include as much examples as possible for training. Standard normalization is applied to the 2D inputs and 3D outputs by subtracting the mean and dividing by the standard deviation. Similar to [9], we train the network for 200 epochs using adam optimizer with a starting learning rate of 0.001 and batches of size 64. The weights are initialized using Kaiming [2].

The code is referenced from the Tensorflow implementation [5] and integrated to the provided project skeleton.

3 EXPERIMENTS

We focus our numerical evaluation on Human3.6M datasets [8] for 3D human pose estimation. The evaluation metric for this project is the mean per joint position error (MPJPE) in milimeter. This is the euclidean distances between predicted joint positions and ground-truth positions, averaged over all joints and all samples, after alignment of the root (central hip) joint. Typically, training and testing is carried out independently in each action.

We modified the models to evaluate the different performance. We didn't modify the ResNet model but only applied different data augmentation (brightness, contrast, hue, rotation and flipping) to reduce the bias. On the other side, we evaluated two variants of the stacked hourglass model:

- Variant 1 (Stacked HG V1): Using a joint detection heatmaps of size 64 x 64 as the highest resolution of the hourglass
- Variant 2 (Stacked HG V2): Using a joint detection heatmaps of size 128 x 128 to have a more fine-grained detections, hopefully moving our system closer to its performance when trained on ground truth.

Beside that, also two variants of the linear model were evaluated:

- Variant 1 (Linear V1): The linear model hyperparameters are set as specified in 2.2.1.
- Variant 2 (Linear V2): The linear model without batch normalization, max-norm constraint, dropout, and residual connection.

4 RESULTS

For the different settings we have experimented, we use the provided test set of the reduced version of "Human3.6mm" to predict 2D pose and then the 3D pose. The test score is the reported public score given on the project submission page.

We summarize the results in Table 1. The score is the 3D pose predictions accuracy on the mean per joint position error (MPJPE)

in millimeter. Our selected setting is the combination of ResNet model and linear model with the default parameters as specified in [9] which scored 112.1, while highest score achieved is with linear model without setting default parameters which scored 110.4.

In all cases, surprisingly the linear model performed slightly better in variant 2 than variant 1, opposed to [9], with 2-3mm difference. However, settings parameters such as batch normalization combined with dropout and residual connections, improves the generalization of the model.

Further tests were performed. We increased the bi-linear layers (From 4 layer total to 6 layers), this improved accuracy by 2 mm. 2D pose augmentation by flipping, this gave an accuracy decrease of 10 mm. Normalization of the 2D pose improved results in general.

Model	Test accuracy (mm)
Martinez et. al. [9]	67.5
Augmentation + ResNet 3D	155.7
Stacked HG V1 + Linear V1	121.9
Stacked HG V2 + Linear V1	155.8
Stacked HG V1 + Linear V2	113.4
Stacked HG V2 + Linear V2	145.5
ResNet 2D + Linear V1	112.1
ResNet 2D + Linear V2	**110.4**

Table 1: Comparison of the baseline model.

5 DISCUSSION AND CONCLUSION

We coupled a state-of-the-art 3D detector with a simple, fast and lightweight deep neural network to achieve accurate results in the task of image-to-3D human pose estimation. However, we couldn't achieve the performance of the linear model [9] because the selected training and test set of their work is different than in our case. Especially, we trained our model on different poses compared to the test set. Martinez et. al. [9] concluded that their method performs poorly when there is no examples in the training that match with the test and this is in our case.

We preferred to select the model setting of ResNet and Linear variant 1 which scored 112.1 rather than the one with linear variant 2 which has a higher score of 110.4, because the residual connection, max-norm, dropout in variant 1 has a better performance in terms of robustness and generalization. Moreover, the difference in accuracy between the two settings is low.

Further Investigations:

(1) Combining 2D predictions from stacked Hourgalss and 2D or 3D predictions from Resnet to train and test the linear model to predict 3D pose. The workflow can be explained as follows:
   (a) Split the train images into set1 and set2. We can select examples from each subject and actions equally distributed between sets.
   (b) Train both models on set1 (Images vs 2D or 3D)
   (c) Predict using set2: $2D_{s2s}$ (Stacked HG) and $2D_{s2r}$ or $3D_{s2r}$ (Resnet)
   (d) Predict using test set: $2D_{ts}$ and $2D_{tr}$ or $3D_{tr}$
   (e) Train linear model with the output of (1c) vs 3D gt
   (f) Predict 3D pose with the output of (1d)

(2) Using the ResNet model to predict 3D pose and use the linear model to fine tuning the output:
   (a) Train the ResNet model with the ground-truth 3D training data
   (b) Run trained ResNet model using the training data.
   (c) Use the predicted 3D pose to train the linear model with the ground-truth 3D to fine tuning the ResNet output.

ACKNOWLEDGMENTS

We would like to thank the ETH AIT Lab for organizing this challenge and writing the skeleton code. Many thanks to Xu Chen for the fruitful discussions.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun. 2015. Deep Residual Learning for Image Recognition. *axXiv:1512.03385v1 [cs.CV] 10 Dec 2015* (2015).
[2] K. He, X. Zhang, S. Ren, and J. Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *ICCV* (2015).
[3] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. *CVPR* (2016).
[4] https://github.com/500swapnil/stacked hourglass. 2018. A Tensorflow implementation of Stacked Hourglass Network for Keypoint Detection. (2018).
[5] https://github.com/una-dinosauria/3d-pose baseline. 2017. A simple yet effective baseline for 3d human pose estimation. *ICCV* (2017).
[6] https://github.com/wbenbihi/hourglasstensorlfow. 2017. Stacked Hourglass model: TensorFlow implementation. (2017).
[7] S. Ioffe and C. Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML* (2015).
[8] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. 2014. Human 3.6mm: Large scale datasets and predictive methods for 3d human sensing in natural environments. *TPAMI* (2014).
[9] Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little. 2017. A simple yet effective baseline for 3d human pose estimation. *ICCV* (2017).
[10] F. Moreno-Noguer. 2017. 3d human pose estimation from a single image via distance matrix regression. *CVPR* (2017).
[11] V. Nair and Hinton G. E. 2010. Rectified linear units improve restricted Boltzmann machines. *ICML* (2010).
[12] Alejandro Newell, Kaiyu Yang, and Jia Deng. 2016. Stacked Hourglass Networks for Human Pose Estimation. *ECCV* (2016).
[13] G. Pavlakos, X. Zhou, and K. G. Derpanis, K.and Daniilidis. 2017. Coarse-to-fine volumetric prediction for single-image 3D human pose. *CVPR* (2017).
[14] Istvan Sarandi, 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:1809.04987v3 [cs.CV] 6 Nov 2018* (2018).
[15] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* (2014).
[16] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. 2016. Convolutional pose machines. *CVPR* (2016).