3D Human Pose Estimation Using Part Affinity Field

By

Zixu Zhao

Senior Thesis in Computer Engineering

University of Illinois at Urbana-Champaign

Advisor: Professor Thomas S. Huang

Spring, 2018

Abstract

Nowadays, following the success of deep learning in Computer Vision field, many researches are underway to produce state-of-the-art technologies that can predict 3D human poses given raw image pixels. These end-to-end systems create possibilities for future studies such as human pose or gait recognition, and their practical values in industry are beyond imagination.

This thesis proposes an end-to-end system that predicts human joint locations in 3D space using only the raw image pixels as inputs. While the most state-of-the-art method believes that lifting joint locations from camera space to 3D space can be done in a simple and effective way only using 2D joint locations as inputs [16], our proposed system is even more effective and accurate with the help of part affinity fields.

Keyword: Computer Vision; Pose Estimation; Neural Network

To my parents, for their love and support.

Acknowledgements

The author of this report would like to express the deepest gratitude to the Department of Electrical and Computer Engineering for providing this fabulous research opportunity. This research is supervised by Professor Thomas S. Huang, and thanks are due to him for his kind guidance and sharp foresight.

The author would like to acknowledge the great support and guidance received from Mr. Ding Liu, who is a Ph.D. student in Image Formation and Processing Group at Beckman Institute, for the thorough discussions on the technical problems and the outline of this research.

This thesis is submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Engineering. The contents of this report only reflect the points of view of the author, who's also responsible for the facts and the accuracy of the data presented herein.

Table of Contents

1. INTRODUCTION

1.1 BACKROUND

3D human pose estimation is a task of predicting three dimensional human keypoint locations from its two-dimensional figure, which is usually an image, or video. Human beings can perceive spatial arrangements from two dimensional depictions of humans. This ability, however, is quite challenging for computers. Luckily, this challenge has become more and more possible to be solved with recent developments in human recognition technologies, and estimating 3D human pose given the raw image pixels has become a topic that received strong attention from Computer Vision community. It is deeply believed that tackling 3D human pose problem is meaningful for many applications such as Human-Computer Interactions (HCI), 3D gaming, Virtual Reality, and sport performance analysis. In any of those applications, computers must operate closely with users, and in some scenarios, such as 3D gaming, computers must always analyse human body poses during the interaction period.

Early endeavours to explore this field have ended up with quite a few successful prototypes. For example, in gaming field, Microsoft Kinect has been one of the most successful ones. The Kinect toolkit analyses human pose in real-time and integrates full-body controls to games. These early achievements mainly used methods like depth imaging, human silhouette detection, feature detection like SIFT or edge direction histograms, which showed that human pose algorithm can be invariant to many factors such as different backgrounds, lighting conditions, human body size and clothing styles and colors. However, these methods also face several challenges. Human pose estimation using depth camera (such as Microsoft Kinect sensor) are very inaccurate on the objects that are beyond 10 meters away from the camera, since RGB-D camera can barely catch anything precisely if the objects are that far. Also, edge detection or interest points methods are sensitive to edginess and sudden color changes during feature engineering process and thus have poor accuracy on complex scenes (such as complex clothing textures or background scenes).

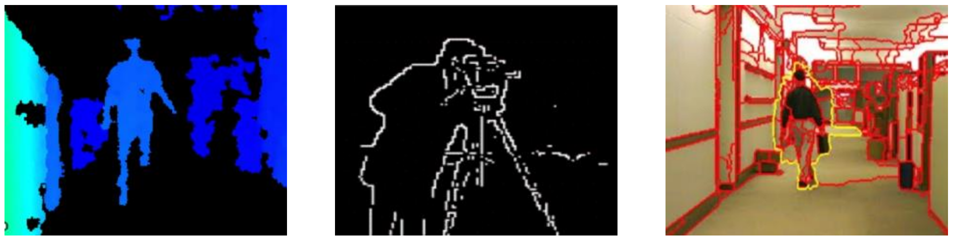


Fig. 1. (a) Human silhouette detection using RGB-D camera is inaccurate at long distances (b) edge detection methods are hard to separate the cameraman from the objects nearby (c) human detection using image segmentation is hard to separate out the man from complex backgrounds

There are many early methods to approach human pose detection. Image segmentation has been widely used in many different fields and are quite successful [1][2][3], but when it comes to human segmentation, it's efficiency is challenged due to complex environments. There are also depth image approaches [4][5] which are quite successful but suffer from real-life constraints (RGB-D cameras are expensive). Figure 1 shows some possible challenges with these early methods. It was not until recently that deep learning approaches started to outperform on 2d pose estimation datasets. Nowadays researchers have started to explore end-to-end deep architectures that are capable of inferring 3D keypoint locations from raw image pixels [6][7]. While there are some methods that achieve this using synthetic data to train the system [8][9], this thesis uses an existing 2D pose estimation systems from [10] to produce 2D joint locations and part affinity fields (PAF), and then aims to estimate 3D pose as a separate training stage.

2. LITERATURE REVIEW

This review focuses on surveying previous works on 2D human joint detection and 3D pose estimation.

Previous works that focused on human joints on individuals like [11][12][13], majorly employed a top-down strategy which directly uses person detectors and estimates the pose individually for each detected person. Unlike many of the other bottom-up approaches like [14][15], these top-down methods depend heavily on the reliability of person detectors: if detectors fail to locate some people in hard scenarios, it’s impossible to estimate their poses. However, bottom-up approaches which commonly labelled all body parts globally and then tried to associate them to individuals, suffer heavily from running time issues, and sometimes even take minutes to process one image. Fortunately, a recent work [10] presents an efficient method with the most state-of-the-art accuracy on human 2D pose estimation benchmarks, which uses bottom-up approach to detect body parts via part affinity fields (PAF). PAF in practice is a set of vector fields that represent the orientation of limbs, it’s calculated by subtracting a body part’s location from another body part’s location, assuming both joints belong to a limb. The thesis uses this technology to predict 2D joint locations, with proper modifications to produce 3D PAF (in the original work PAF is only a 2D vector field).

Meanwhile, several previous works [6][7] have explored to infer 3D pose directly from raw image pixels, while others [8][9] tried to decouple 3D pose problems into 2D joint estimation and use the detected 2D joint locations as the inputs. The benefit of the latter is that it depends on the already well studied 2D pose estimation architectures which are proved to be invariant to other factors such as noise, background scenes, light conditions and so on. Not long ago, [16] has shown that a very simple linear network can solve 3D space task with incredibly low error rate, with just 2D joint locations as inputs. This means that understanding spatial arrangements for computers with depth ambiguity is easier than expected, provided that 2D pose estimation brings as little error as possible. However, this approach gives up all the global contextual cues from human body in the image, and keeps only the 2D joint locations as inputs, and it also suffers from early commitment: if the body part detectors perform badly, 3D pose estimator will have bad accuracy.

In this thesis, we will use PAF to help estimate 3D human pose. PAF encodes important contextual cues of human limbs and provides a sense of direction in 3D space to 3D pose estimation stage.

3. THE PROPOSED METHOD

3.1 PROBLEM DEFINITION

The system was trained and tested on Human 3.6M dataset, which is the largest 3D pose estimation benchmark available. The dataset contains 3.6 million single-person images and are divided into 7 subsets $(S_1, S_2, S_3, \dots, S_7)$. Each subset contains human poses in the following scenarios, as shown in figure 2. The dataset is based on poses made by 11 professional actors, and Figure 2 only includes one of them.

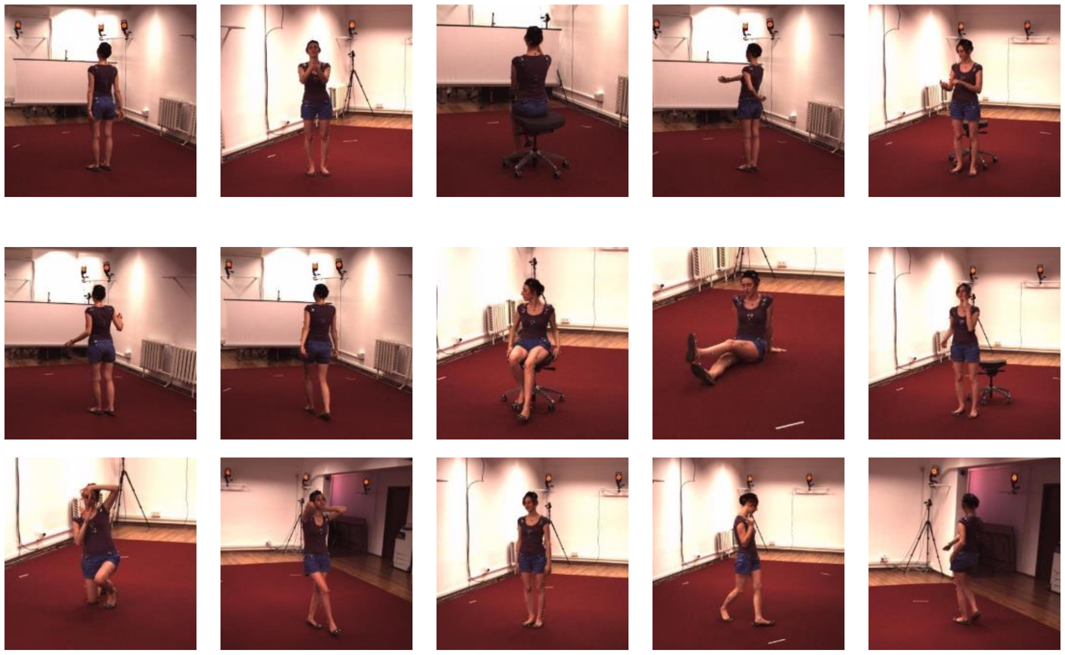


Fig. 2. Human poses in each subset (a) – (o) directions, Discussion, eating, greeting, phoning, posing, purchases, sitting, sitting down, smoking, taking photo, waiting, walk together, walking, and walking dog

$S_1, \dots, S_5$ are used as training sets, and S_6 and S_7 are used as validation sets.

In our proposed system, we break 3D pose estimation problems into two parts. The first part is to produce joint confidence maps and part affinity fields (PAF), and the second part will estimate joints in 3D space by utilizing the outputs from part one.

The dataset provides each subset its corresponding label L_i, which is a 3D point location of 32 selected human keypoints in the dataset. The label needs to be further processed in our system. We select 17 of those keypoints as the targeted joints to train and used the camera parameters corresponding to each pose to project its 3D joint locations to camera space. The camera space joint locations will be used as label L_i^1 for 2D joint detection. We also define 16 pairs of joints as human limbs, and from L_i, we use point subtraction to get 16 3D limb vectors as label L_i^2 to produce PAF. Both L_i^1 and L_i^2 are used as labels to train the system in the first part. After this 2D joint detection stage is over, we use L_i as the label for the next stage training: 3D pose estimation.

3.2 DATA PREPROCESSING

Human 3.6M provides the 3D joint locations in world coordinates and camera parameters for each image, and we can project all 3D points to camera space. To prepare the groundtruth value for the network, we still have to preprocess them further.

Each human keypoint has a 2D joint confidence map **J** associated with it. The confidence map has the same size as the original image with each pixel having a floating-point value from 0 to 1. This value represents a possibility that a joint occurs at this pixel location. To prepare the groundtruth value for the system, the joint location is labelled with value 1, the surrounding neighbourhood pixels are assigned values according to their locations in a Gaussian heatmap with the joint location with value 1 as the peak. More formally, the joint location neighbourhood for a specific joint *i* has value defined as below:

$$J_i(p) = \exp(-\frac{\|p-q_i\|_2^2}{\sigma^2}) \tag{1}$$

Where *p* is the pixel location at the confidence map, q_i is the joint location for joint *i*, and σ is a user-defined parameter to control the spread of the region of interests of a joint. Figure 3 below shows a color-coded Gaussian heatmap of human left elbow. Yellow means high confidence of joint location, and blue means low confidence.

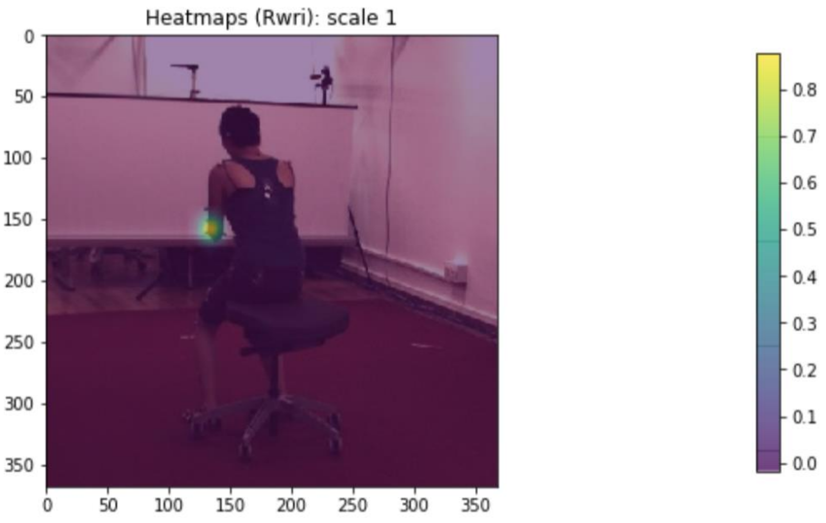


Fig. 3. Color-coded heatmap

Each human limb also has a vector field **V**, which we refer to as PAF in our system, that preserves the orientation information of that specific limb. PAF is a three-dimensional vector field and each dimension has the same size as the original image. For each pixel location belonging to an area covered by a human limb, there's a normalized 3D vector that encodes the direction from one end of the limb to the other end by saving the x, y, z components of this limb vector to the three-dimensional matrix. More formally, we define a unit vector ***v*** in the direction of limb connecting joints *i* and *j*:

$$\boldsymbol{v} = \frac{Q_i - Q_j}{\|Q_i - Q_j\|_2} \tag{2}$$

Where Q_i and Q_j are the two joint locations in 3D space.

We then define the groundtruth PAF V_{ij} for limb *ij*:

$$V_{ij}(p) = \boldsymbol{v}, \quad \text{for } p \text{ in area covered by limb} \tag{3}$$

For visualization purposes, the pixels in the area shaded in light blue should be considered as parts of the limb.

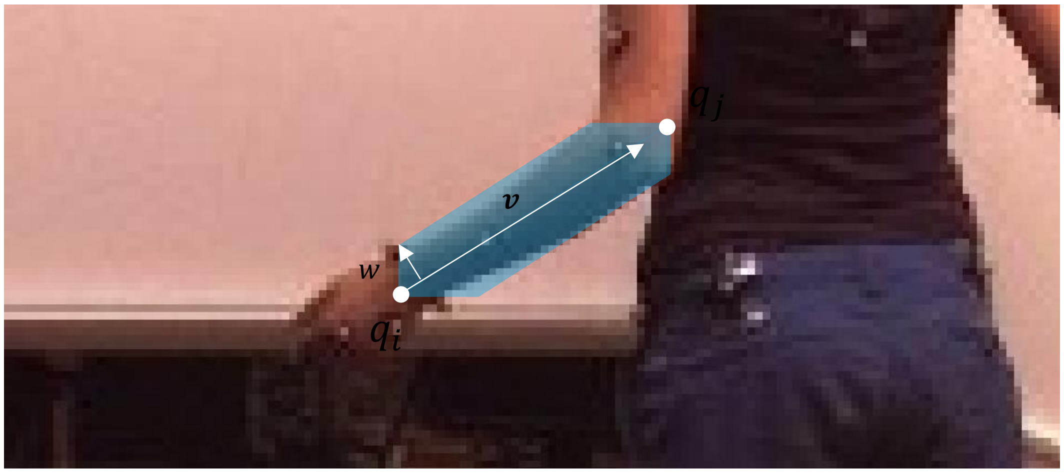


Fig. 3. an example limb with its limb area and limb vector

3.3 NETWORK DESIGN IN PART ONE

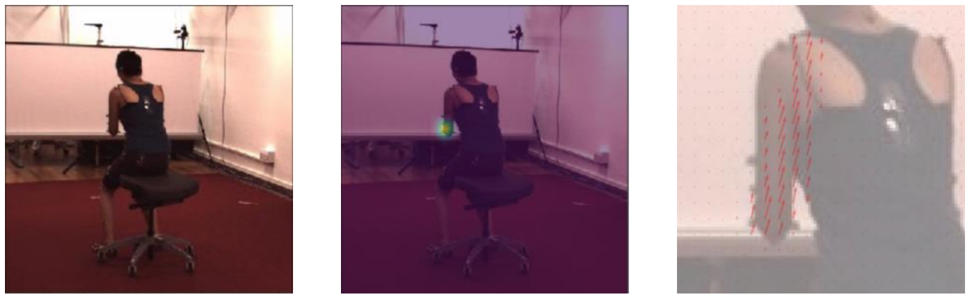


Fig. 4. General pipeline of our method (a) input image (b) joint confidence map of human left elbow (c) part affinity field of human left arm

As is illustrated in Figure 4, the system takes an image of size $w \times h$ and goes through the first network to produce a 2D confidence map **J** for each keypoint and a 3D vector field **V** for each limb. For each input image, the system will produce joint confidence map $J_1, J_2, \ldots, J_N$ for N joints (shown in Figure 4 (b)), where $J_n \in \mathbb{R}^{w \times h}$, for n $\in \{1 \ldots \mathrm{N}\}$, and 3D vector fields $V_1, V_2, \ldots, V_M$ for M limbs (shown in Figure 4 (c)), where $V_m \in \mathbb{R}^{w \times h \times 3}$, for m $\in \{1 \ldots \mathrm{M}\}$. Each image location has its degree of association with body parts encoded in V_m as a 3D vector, and the *x*, *y*, and *z* components of V_m are saved in three matrices of size $w \times h$.

A very detailed picture of our CNN architecture visualized with the help of Netscope Caffe Visualizer [17] is presented in Figure 5. Our network is a two-branched multi-staged CNN. It's multi-staged because we use an iterative prediction system following the architecture proposed by Wei et al. [18]. The input image first goes through the first 10 layers of VGG-19 [19] as an initial feature engineering process. The generated feature map **M** is then used as input to the first stage, which produces the first set of joint confidence maps $J_1^1, J_2^1, \ldots, J_N^1$ for N predefined joints and the first set of PAFs $V_1^1, V_2^1, \ldots, V_M^1$ for M predefined limbs. It's two-branched because the first branch is used to predict human keypoints and the second branch is used to produce PAFs.

The CNN structure starting from the next stage are all the same for each stage. The network takes the feature map **M**, joint confidence maps $\sum_i^N J_i^t$, and PAFs $\sum_i^M V_i^t$ from stage t (input concatenation labelled as concat_stage in orange in Figure 4.) and produces new joint confidence maps and PAFs simultaneously, which will be one part of the next stage's inputs.

The reason for having this iterative nature in this CNN structure is that CNN on each stage provides a more abstract feature map based on its input and refines the final predictions over each successive stage. Although the loss may eventually converge on each stage, the next stage's input is based on the previous stage's refinement, so multi-staged structure can further reduce the loss on the final stage.

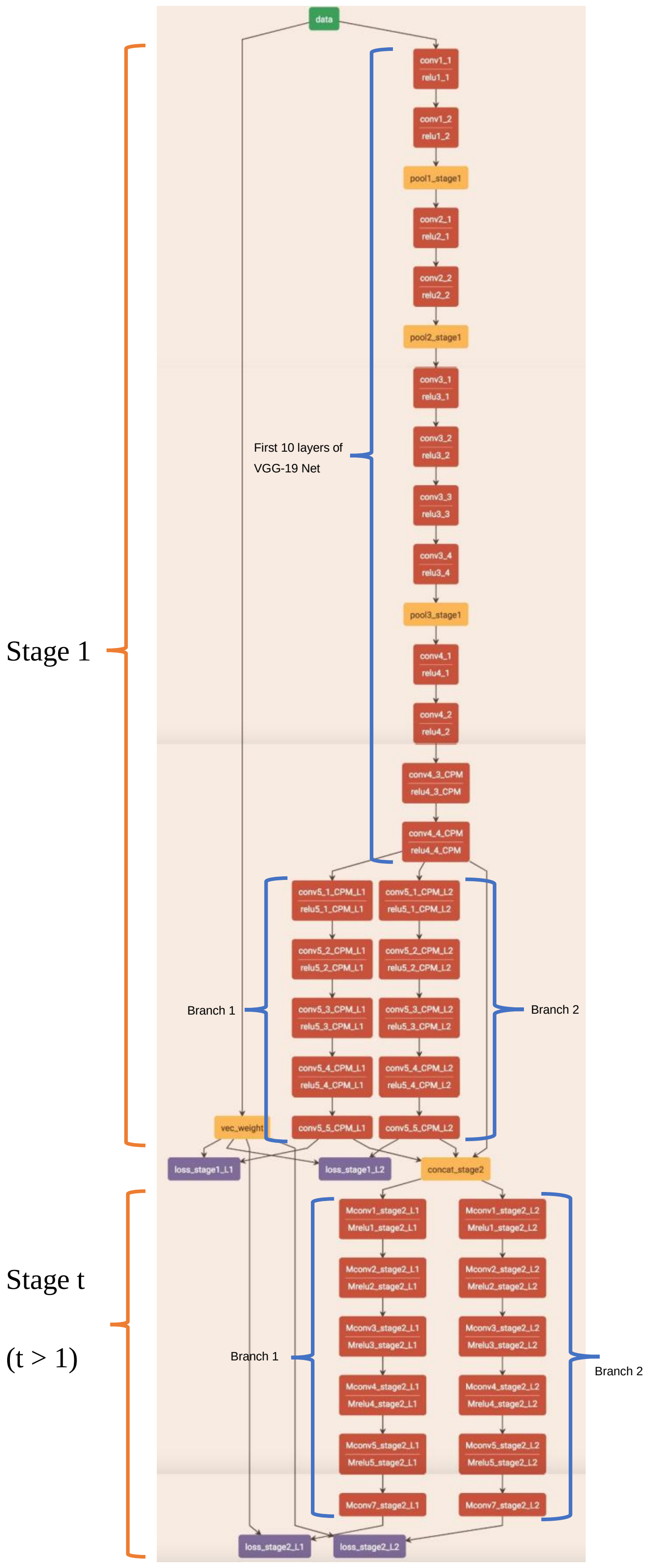


Figure 5. two-branched multi-stage CNN structure visualization. Stage 1 is an initialization stage that partially uses VGG-19 net to generate a set of features based on the input image. Then branch 1 produces the initial joint confidence maps and branch 2 produces part affinity field. Stages after stage 1 have the same structure. It is up to the users to define how many stages they want.

To guide the network, we define two loss functions (labelled as loss_stage in purple in Figure 5.) at the end of each stage. The loss functions for both confidence maps and PAFs are Euclidean Loss between groundtruth values and predictions.

The loss function for branch one which produces joint confidence maps $\sum_i^N J_i^t$ at stage *t* is formally defined here:

$$\ell(J^t) = \sum_i^N \sum p \left\| \widehat{J_i^t(p)} - J_i^t(p) \right\|_2^2 \tag{4}$$

where $\widehat{J_i^t(p)}$ is the predicted set of joint confidence maps at stage *t*, and $J_i^t(p)$ is the groundtruth value at stage *t*. The loss function sums over all pixel locations *p* at the confidence maps, and the total loss is the sum of all confidence maps for all the **N** joints.

The loss function for branch one which produces joint confidence maps $\sum_i^N J_i^t$ at stage *t* is formally define here:

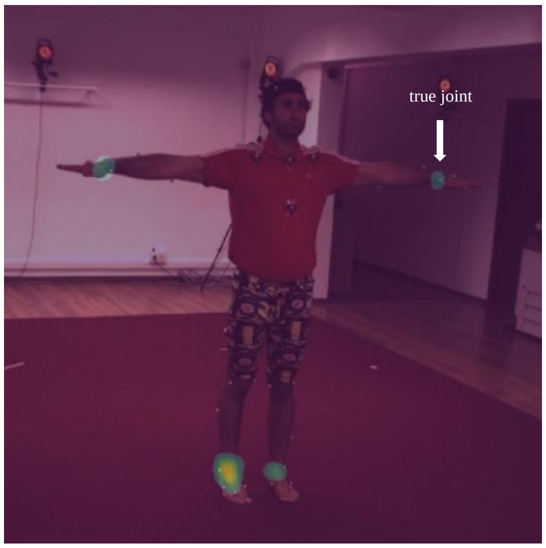
$$\ell(V^t) = \sum_i^M \sum p \left\| \widehat{V_i^t(p)} - V_i^t(p) \right\|_2^2 \tag{5}$$

where $\widehat{V_i^t(p)}$ is the predicted set of joint confidence maps at stage *t*, and $V_i^t(p)$ is the groundtruth value at stage *t*. The loss function should sum all the PAFs for all **M** limbs.

The objective at each stage is to minimize the total loss $\ell(J^t) + \ell(V^t)$ at each stage.

3.4 JOINT DETECTION

Given a set of joint confidence maps, how can we localize each joint? In other words, in a joint confidence map, where each pixel represents the belief of the presence of a certain joint, how to localize the joint as close to the groundtruth as possible? The basic assumption is that any joint must be a peak (in practice a peak can be found by non-maximum suppression), since the groundtruth joint location was made as the peak of joint Gaussian heatmap. We first find all the peaks in a joint heatmap. To recognize the best peak among all the candidates, a naïve method people prone to do is just to keep the joint that has the highest confidence in the joint confidence map and discard all other joint candidates, but in practice the true joint might not always be the one that has the highest confidence. Consider an example shown in the figure on the left, which is a joint confidence map superimposed on the original image. It shows four strong joint candidates. However, the true joint on the top right is not the one with the highest confidence, other joint candidates no matter how large the confidence is, are false positives.



Fortunately, part affinity field has come to help. Multiple joint candidates introduce a large set of possible limbs. We connect the two possible joints for each limb and score the limb candidates by measuring the line integral over the limbs' corresponding PAFs. More formally, between the pixel locations q_1 and q_2 for predicted joints J_1 and J_2 on the three-dimensional part affinity field matrix, we sample **X** uniformly-spaced values and each sampled value is a 3D limb vector. We now only use the *x*, *y* components of these values as 2D vectors to project onto the 2D unit vector from q_1 to q_2. The resulted projection for all the **X** sampled vectors will be added together as a score for this limb candidate. The equation for limb score is thus simply an integral:

$$S_{ij} = \sum_{u=0}^{\boldsymbol{X}} V_{ij,\, p_u}(x, y) \cdot \frac{Q_i - Q_j}{\|Q_i - Q_j\|_2} \tag{6}$$

where $V_{ij,\,p_u}$ is the PAF for limb ij on the uth pixel location, and Q_i is the pixel location at joint i.

Human body pose construction can be seen as a tree in which we use all the nodes to represent joints and all edges to represent limbs. Each edge is associated with its limb score. Since we only consider single person scenarios in our proposed system, there are two extra constraints to consider. First, although each node might have many candidate nodes, the tree has exactly N nodes to represent N joints. Second, once a candidate node is selected to represent an internal node for an edge, it must also be used in adjacent edges, as shown in Figure 6.

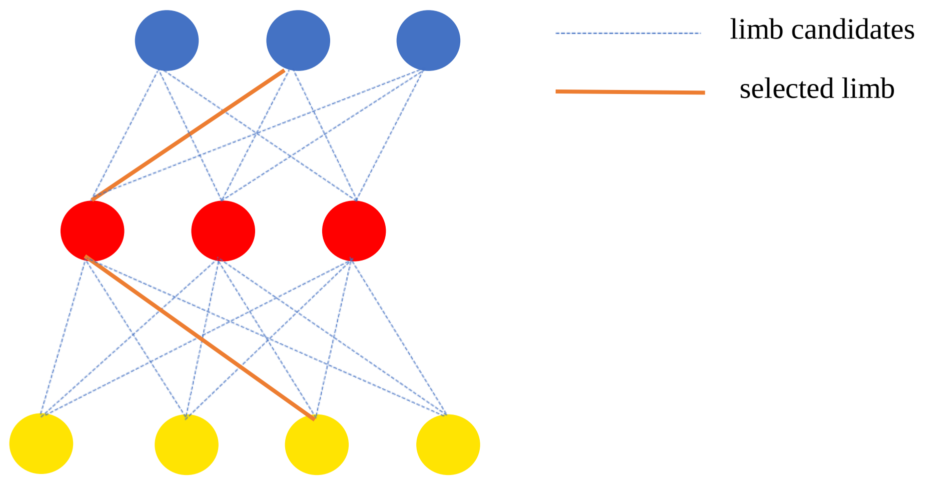


Fig. 6. Circles shaded in three colors represent three sets of joint candidates, and only one candidate is chosen from each set as part of the global solution.

Once a node is chosen as an internal node in a tree, like the left red node in the graph, it must be the sole representative of its set. Its adjacent set will also select a representative to connect to it. The best solution should include N-1 edges that together make the global limb scores the largest.

This detection algorithm might seem very slow since we must try all the combinations of joint candidate and select the one that produces the largest global limb score. If each joint has u candidates, the runtime for this algorithm is $O(u^N)$.

N is constant, so the runtime is polynomial. This performance is actually as efficient as some other state-of-art approaches [10][20].

When all the joints are successfully detected, we can then select the corresponding limbs. This will help to prepare limb vector fields on the next stage.

3.5 DATA PREPARATION FOR PART TWO

In joint detection, each joint candidate set selects only one best representative that makes the global solution optimal. The output data after joint detection is a 2D joint location matrix, the size of it is N by 2. To prepare the input data for part two, we flatten this matrix into a 1D array and concatenate it with PAFs.

To prepare PAF matrix, we use the same **X** sampled 3D vectors for each limb, and only select the sample values whose 2D vectors are on the same direction as the unit limb vector. The mean of x, y, z components of the selected sample values will be the final 3D limb vector. We get the limb vectors for all N – 1 limbs and flatten this matrix into a 1D array of size 3N – 3 to concatenate with joint location array. This final 1D joint location and limb vector array will be the input for stage two network.

3.6 NETWORK DESIGN IN PART TWO

The proposed system greatly benefits from many of the recent researches on learning-based 2D to 3D joint prediction. For example, [10] presents a deep convolutional neural network approach using stacked hourglass architecture [21], and other works prefer to use raw images as inputs. A common belief the authors of these approaches hold is that using 2D detections to predict 3D points is a difficult task because 2D detections provide less information than raw images or 2D joint probability distribution. However, thanks to Martinez *et al.* [16], they showed a very simple but effective baseline that surprisingly achieved good scores on Human3.6M benchmark using only 2D point locations. In our system, the network is based on the architecture proposed in [16], but instead of only using 2D detections, we use 3D part affinity fields, which keeps the global contextual cues.

The architecture in part two is simpler than part one, mostly because part two is dealing with low-dimensional inputs. The input to this network is the joint location and limb vector array from section 3.4 joint detection. Our system requires very small input size for each human pose, so we use linear layers, and we can even store the entire dataset in the GPU which dramatically reduces the training time.

We also use one linear layer to preprocess the input and another one before the output, to make sure they all have the right sizes. An illustration of our linear layer is given in equation 7 and figure 7:

$$(w_1, w_2, \dots, w_n) \begin{Bmatrix} x_1 \\ x_1 \\ \dots \\ x_n \end{Bmatrix} + b \tag{7}$$

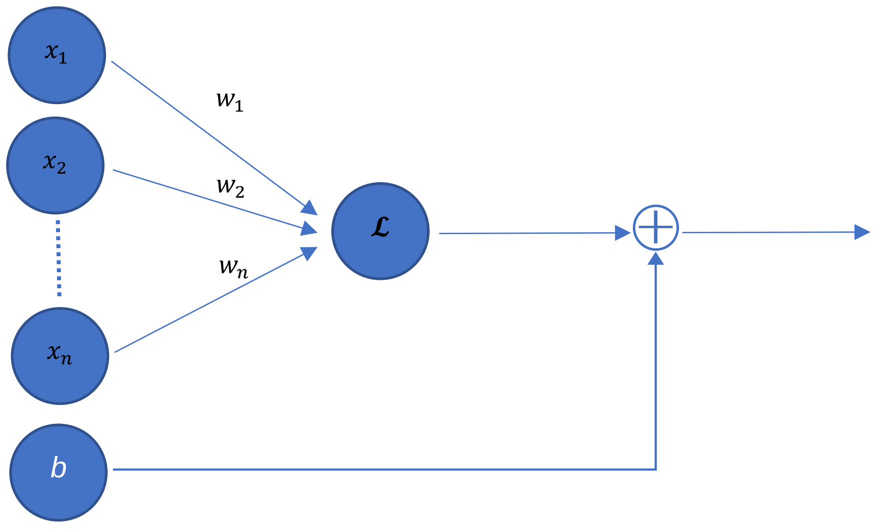


Fig. 7. $\{x_1, x_2, \dots, x_n\}$ represents the input layer; $\{w_1, w_2, \dots, w_n\}$ corresponds to weights in linear neural network; b is an extra value to add to the system

In our system, depending on the number of frames (**F**) in the dataset, the input is a matrix of size **F** by 5N – 5. The first linear layer sets the input dimension to 5N – 5 by *m*, where *m* is a parameter to be fine-tuned. Another linear layer before the output will make sure the size of the output is **F** by 3N – 3.

Our network mainly includes a bilinear layer, after each linear layer we perform batch normalization [22], RELU [23], dropout [24], as well as residual connections. The workflow is presented in figure 8.

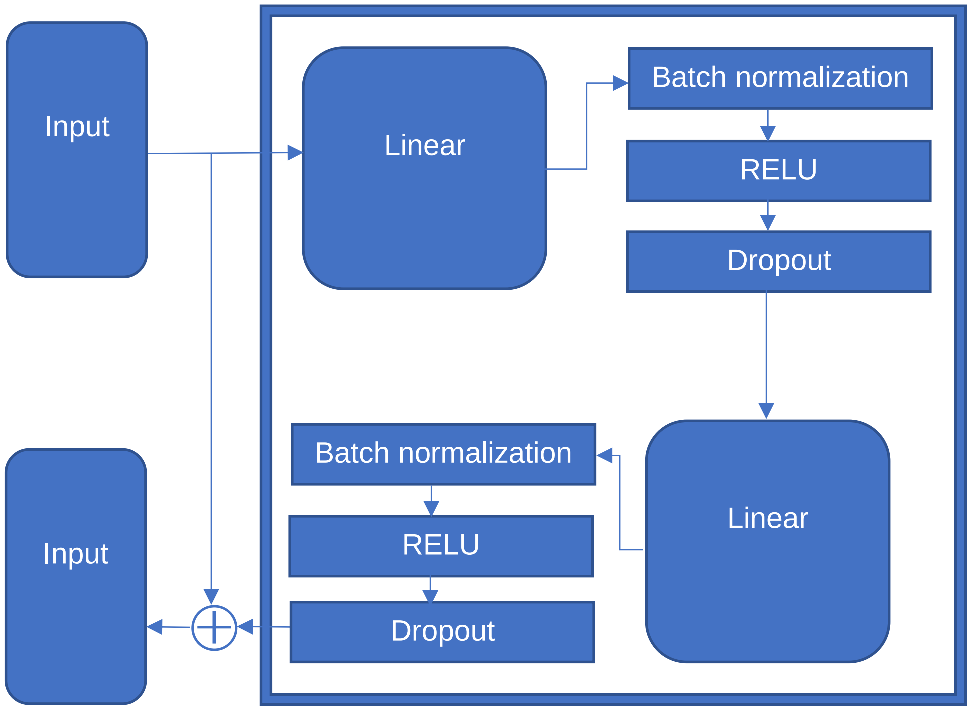


Fig. 8. The baseline of our approach

I reality, the quality of joint and limb detector might be not ideal due to noisy raw images, the small size of training data or other constraints. The noises on the input data is likely to cause covariate shift, which decreases the robustness during test-time. Therefore, we use batch normalization to make data comparable across features. We also added residual connections since it’s proved to improve performance of the system [25].

We use dropout to reduce dangers on overfitting. RELU is a common choice to add non-linearities to the network.

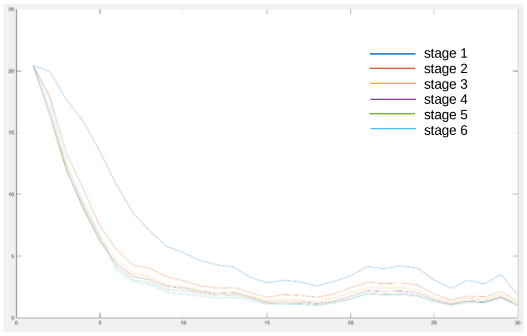
4. EXPERIMENT RESULTS

4.1 EXPERIMENT DATASET

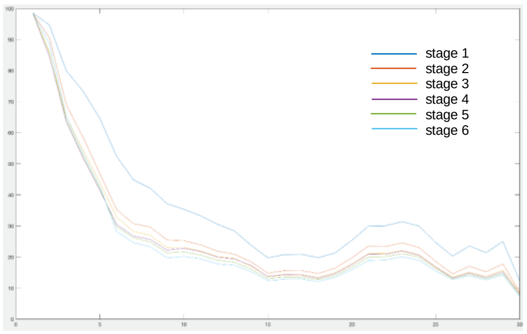
The dataset we used for the experiment includes 410,000 single-person pose images divided into 9 subsets each covering 15 poses. 310,000 images are from subsets $(S_1, S_2, S_3, \dots, S_7)$ used as training set, and the other 100,000 images from subsets (S_8, S_9) are used as validation sets. We select 17 keypoints among a total of 32 keypoints for each person in Human3.6M dataset to prepare groundtruth and use original images as inputs to the network.

4.2 PERFORMANCE OF NETWORK IN PART ONE

To obtain joint locations and limb vectors, we first run the two-branched multi-stage CNN in part one. As is explained in section 3.3, a multi-staged CNN structure has an iterative process that refines the final predictions over each successive stage. This is also verified during the training process.



(a)



(b)

Fig. 9. Training loss

(a) Euclidean loss of keypoints vs number of iterations (in 10,000)

(b) Euclidean loss of limb vectors vs number of iterations (in 10,000)

As shown in figure 9, both keypoint prediction and limb vector prediction have the lowest training loss at stage 6.

After the training process is over, we run the joint detection algorithm introduced in section 3.4, and the output (shown in figure 10) is remarkably accurate.

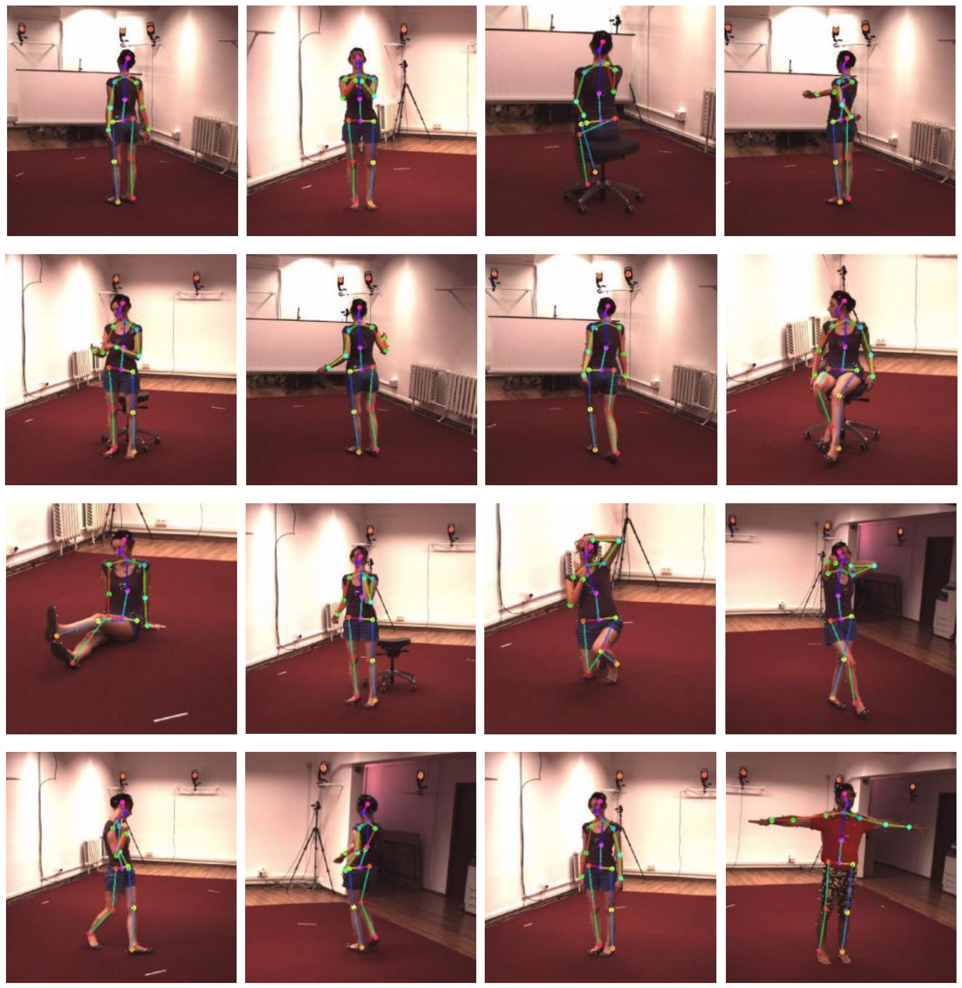


Fig. 10. 2D human detection (a) – (o) correspond to figure 2 (p) an image from validation sets

4.3 PERFORMANCE OF NETWORK IN PART TWO

We use the results from joint detection in part one and pre-process them according to section 3.5. We evaluate the performance of our system by comparing it with other state-of-the-art methods. The detailed performance comparison is shown in table 1.

Table 1. Classifier's accuracy comparisons of the manifold dimensions

other methods	Direc t.	Discuss	Eating	Greet	Phone	Photo	Pose	Purch.
Tekin *et al.* [26]	102.4	147.2	88.8	125.3	118.0	182.7	112.4	129.2
Zhou *et al.* [27]	87.4	109.3	87.1	103.2	116.2	143.3	106.9	99.8
Du *et al.* [28]	85.1	112.7	104.9	122.1	139.1	135.9	105.9	166.2
Park *et al.* [29]	100.3	116.2	90.0	116.5	115.3	149.5	117.6	106.9
Zhou *et al.* [30]	91.8	102.4	96.7	98.8	113.4	125.2	90.0	93.8
Pavlakos *et al.* [31]	67.4	71.9	66.7	69.1	72.0	77.0	65.0	68.3
Martinez *et al.* [16]	51.8	56.2	58.1	59.0	69.5	78.4	55.2	58.1
Ours	31.47	36.19	32.07	34.26	36.76	46.21	32.87	35.52

(a)

other methods	Sit	SitD	Smoke	Wait	WalkD	Walk	WalkT	avg
Tekin *et al.* [26]	138.9	224.9	118.4	138.8	126.3	55.1	65.8	125.0
Zhou *et al.* [27]	124.5	199.2	107.4	118.1	114.2	79.4	97.7	113.0
Du *et al.* [28]	117.5	226.9	120.0	117.7	137.4	99.3	106.5	126.5
Park *et al.* [29]	137.2	190.8	105.8	125.1	131.9	62.6	96.2	117.3
Zhou *et al.* [30]	132.2	159.0	107.0	94.4	126.0	79.0	99.0	107.3
Pavlakos *et al.* [31]	83.7	96.5	71.7	65.8	74.9	59.1	63.2	71.9
Martinez *et al.* [16]	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9
Ours	40.24	43.86	35.96	33.34	39.56	31.05	40.4	36.02

(b)

(a) and (b) in table 1 show the performance of our system on all the poses. The overall performance in all cases has less error than all current state-of-the-art methods.

5. CONCLUSION

This senior thesis research introduces a novel end-to-end baseline for estimating 3D human poses from raw images using a two-staged deep learning model. The proposed model has simple logic, compact structure, and strong effectiveness. It’s been trained and tested on diverse dataset, and experiments have shown strong indications of beating all existing state-of-the-art methods. However, more tests and thorough comparisons must be made between our method and other approaches, and my future work is to fine-tune the system and explore the robustness of this system on poor quality inputs.

References

[1] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*. Springer, Cham, 2015.

[2] Tutumluer, Erol, et al. *Field Evaluation of Ballast Fouling Conditions Using Machine Vision*. No. Safety-27. 2017.

[3] Zhao, Zixu. "Matlab Implementation of Machine Vision Algorithm on Ballast Degradation Evaluation." *arXiv preprint arXiv:1804.08835* (2018).

[4] Stone, Erik E., and Marjorie Skubic. "Fall detection in homes of older adults using the Microsoft Kinect." *IEEE journal of biomedical and health informatics*19.1 (2015): 290-301.

[5] Xia, Lu, Chia-Chih Chen, and Jake K. Aggarwal. "Human detection using depth information by kinect." *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*. IEEE, 2011.

[6] Pavlakos, Georgios, et al. "Coarse-to-fine volumetric prediction for single-image 3D human pose." *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017.

[7] Tekin, Bugra, et al. "Structured prediction of 3d human pose with deep neural networks." *arXiv preprint arXiv:1605.05180*(2016).

[8] Rogez, Grégory, and Cordelia Schmid. "Mocap-guided data augmentation for 3d pose estimation in the wild." *Advances in Neural Information Processing Systems*. 2016.

[9] Varol, Gül, et al. "Learning from synthetic humans." *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*. 2017.

[10] Cao, Zhe, et al. "Realtime multi-person 2d pose estimation using part affinity fields." *CVPR*. Vol. 1. No. 2. 2017.

[11] Pishchulin, Leonid, et al. "Articulated people detection and pose estimation: Reshaping the future." *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012.

[12] Gkioxari, Georgia, et al. "Using k-poselets for detecting people and localizing their keypoints." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

[13] Sun, Min, and Silvio Savarese. "Articulated part-based model for joint object detection and pose estimation." *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011.

[14] Pishchulin, Leonid, et al. "Deepcut: Joint subset partition and labeling for multi person pose estimation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[15] Insafutdinov, Eldar, et al. "Deepercut: A deeper, stronger, and faster multi-person pose estimation model." *European Conference on Computer Vision*. Springer, Cham, 2016.

[16] Martinez, Julieta, et al. "A simple yet effective baseline for 3d human pose estimation." *IEEE International Conference on Computer Vision*. Vol. 206. 2017.

[17] “Netscope.” *Quick Start - Netscope,* ethereon.github.io/netscope/quickstart.html.

[18] Wei, Shih-En, et al. "Convolutional pose machines." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[19] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

[20] Yang, Yi, and Deva Ramanan. "Articulated human detection with flexible mixtures of parts." *IEEE transactions on pattern analysis and machine intelligence* 35.12 (2013): 2878-2890.

[21] Newell, Alejandro, Kaiyu Yang, and Jia Deng. "Stacked hourglass networks for human pose estimation." *European Conference on Computer Vision*. Springer, Cham, 2016.

[22] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *arXiv preprint arXiv:1502.03167* (2015).

[23] Nair, Vinod, and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines." *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010.

[24] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1), 2014.

[25] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[26] Tekin, Bugra, et al. "Direct prediction of 3d body poses from motion compensated sequences." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[27] Zhou, Xiaowei, et al. "Sparseness meets deepness: 3D human pose estimation from monocular video." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[28] Du, Yu, et al. "Marker-less 3d human motion capture with monocular image sequence and height-maps." *European Conference on Computer Vision*. Springer, Cham, 2016.

[29] Park, Sungheon, Jihye Hwang, and Nojun Kwak. "3D human pose estimation using convolutional neural networks with 2D pose information." *European Conference on Computer Vision*. Springer, Cham, 2016.

[30] Zhou, Xingyi, et al. "Deep kinematic pose regression." *European Conference on Computer Vision*. Springer, Cham, 2016.

[31] Pavlakos, Georgios, et al. "Coarse-to-fine volumetric prediction for single-image 3D human pose." *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017.