3D Human Pose Estimation Using Convolutional Neural Networks with 2D Pose Information

3 authors, including:

Sungheon Park
Seoul National University
14 PUBLICATIONS   **318** CITATIONS

Nojun Kwak
Seoul National University
185 PUBLICATIONS   **3,751** CITATIONS

Some of the authors of this publication are also working on these related projects:

Knowledge Distillation View project

3D Human Pose Estimation Using Convolutional Neural Networks with 2D Pose Information

Sungheon Park, Jihye Hwang, and Nojun Kwak

Graduate School of Convergence Science and Technology,
Seoul National University, Seoul, Korea
`sungheonpark@snu.ac.kr, hjh881120@gmail.com, nojunk@snu.ac.kr`

Abstract. While there has been a success in 2D human pose estimation with convolutional neural networks (CNNs), 3D human pose estimation has not been thoroughly studied. In this paper, we tackle the 3D human pose estimation task with end-to-end learning using CNNs. Relative 3D positions between one joint and the other joints are learned via CNNs. The proposed method improves the performance of CNN with two novel ideas. First, we added 2D pose information to estimate a 3D pose from an image by concatenating 2D pose estimation result with the features from an image. Second, we have found that more accurate 3D poses are obtained by combining information on relative positions with respect to multiple joints, instead of just one root joint. Experimental results show that the proposed method achieves comparable performance to the state-of-the-art methods on Human 3.6m dataset.

Keywords: human pose estimation, convolutional neural network, 2D-3D joint optimization

1   Introduction

Both 2D and 3D human pose recovery from images are important tasks since the retrieved pose information can be used to other applications such as action recognition, crowd behavior analysis, markerless motion capture and so on. However, human pose estimation is a challenging task due to the dynamic variations of a human body. Various skin colors and clothes also make the estimation difficult. Especially, pose estimation from a single image requires a model that is robust to occlusion and viewpoint variations.

Recently, 2D human pose estimation achieved a great success with convolutional neural networks (CNNs) [1,2,3]. Strong representation power and the ability to disentangle underlying factors of variation are characteristics of CNNs that enable learning discriminative features automatically [4] and show superior performance to the methods based on hand-crafted features. On the other hands, 3D human pose estimation using CNNs has not been studied thoroughly compared to the 2D cases. Estimating a 3D human pose from a single image is more challenging than 2D cases due to the lack of depth information. However, CNN can be a powerful framework for learning discriminative image features and

estimating 3D poses from them. In the case where the target object is fixed such as human body, it is able to learn useful features directly from images without keypoint matching step in the typical 3D reconstruction tasks.

Though recent algorithms that are based on CNNs for 3D human pose estimation have been proposed [5,6,7], they do not make use of 2D pose information which can provide additional information for 3D pose estimation. From 2D pose information, undesirable 3D joint positions which generate unnatural human pose may be discarded. Therefore, if the information that contains the 2D position of each joint in the input image is used, the results of 3D pose estimation can be improved.

In this paper, we propose a simple yet powerful 3D human pose estimation framework based on the regression of joint positions using CNNs. We introduce two strategies to improve the regression results from the baseline CNNs. Firstly, not only the image features but also 2D joint classification results are used as input features for 3D pose estimation. This scheme successfully incorporates the correlation between 2D and 3D poses. Secondly, rather than estimating relative positions with respect to only one root joint, we estimated the relative 3D positions with respect to multiple joints. This scheme effectively reduces the error of the joints that are far from the root joint. Experimental results validate the proposed framework significantly improves the baseline method and achieves comparable performance to the state-of-the-art methods on Human 3.6m dataset [8] without utilizing the temporal information.

The rest of the paper is organized as follows. Related works are reviewed in Section 2. The structure of CNNs used in this paper and two key ideas of our method, 1) the integration of 2D joint classification results into 3D pose estimation and 2) multiple 3D pose regression from various root nodes, are explained in Section 3. Details of implementation and training procedures are explained in Section 4. Experimental results are illustrated in Section 5, and finally conclusions are made in Section 6.

2   Related Work

Human pose estimation has been a fundamental task since early computer vision literature, and numerous researches have been conducted on both 2D and 3D human pose estimation. In this section, we will cover both 2D and 3D human pose estimation methods focusing on the CNN-based methods.

Early works for 2D human pose estimation which are based on deformable parts model [9], pictorial structure [10,11,12], or poselets [13] train the relationship between body appearance and body joints using hand-crafted features. Recently proposed CNN based methods drastically improve the performance over the previous hand-crafted feature based methods. DeepPose [1] used CNN-based structure to regress joint locations with multiple iterations. Firstly, it predicts an initial pose using holistic view and refine the currently predicted pose using relevant parts of the image. Xiaochuan et al. [14] integrated both the local part appearance and the holistic view of an image using dual-source CNN. Convolu-

tional pose machine [3] is a systematic approach to improve prediction of each stage. Each stage operates a CNN which accepts both the original image and confidence maps from preceding stages as an input. The performance is improved by combining the joint prediction results from the previous step with features from CNN. Joao et al. [2] proposed a self-correcting method by a top-down feedback. It iteratively learns a human pose using a self-correcting CNN model which gradually improves the initial result by feeding back error predictions. Xiao et al. [15] proposed an end-to-end learning system which captures the relationships among feature maps of joints. Geometrical transform kernels are introduced to learn features and their relationship jointly.

Similar to the 2D case, early stage of 3D human pose estimation is also based on the low-level features such as local shape context [16] or segmentation results [17]. With the extracted features, 3D pose estimation is formulated as a regression problem using relevance vector machines [16], structured SVMs [17], or random forest classifiers [18]. Recently, CNNs have drew a lot of attentions also for the 3D human pose estimation tasks. Since search space in 3D is much larger than 2D image space, 3D human pose estimation is often formulated as a regression problem rather than a classification task. Li and Chan [5] firstly used CNNs to learn 3D human pose directly from input images. Relative 3D position to the parent joint is learned by CNNs via regression. They also used 2D part detectors of each joints in a sliding window fashion. They found that loss function which combines 2D joint classification and 3D joint regression helps to improve the 3D pose estimation results. Li et al. [6] improved the performance of 3D pose estimation by integrating a structured learning framework into CNNs. Recently, Tekin et al. [7] proposed a structured prediction framework which learns 3D pose representations using an auto-encoder. Temporal information from video sequences also helps to predict more accurate pose estimation result. Zhou et al. [19] used the result of 2D pose estimation to reconstruct a 3D pose. They represented a 3D pose as a weighted sum of shape bases similar to typical non-rigid structure from motion, and they designed an EM-algorithm which formulates the 3D pose as a latent variable when 2D pose estimation results are available. The method achieved the state-of-the-art performance for 3D human pose estimation when combined with 2D pose predictions learned from CNN. Tekin et al. [20] used multiple consecutive frames to build a spatio-temporal features, and the features are fed to a deep neural network regressor to estimate the 3D pose.

The method proposed in this paper aims to provide an end-to-end learning framework to estimate 3D structure of a human body from a single image. Similar to [5], 3D and 2D pose information are jointly learned in a single CNN. Unlike the previous works, we directly propagate the 2D classification results to the 3D pose regressors inside the CNNs. Using additional information such as 2D classification results and the relative distance from multiple joints, we improve the performance of 3D human pose estimation over the baseline method.
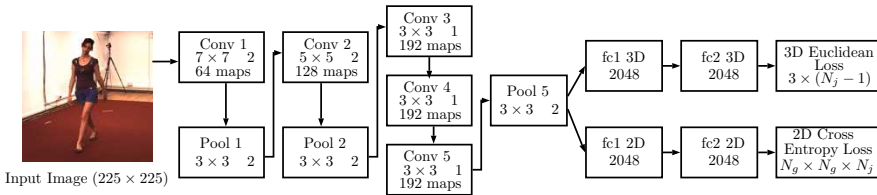
Fig. 1. The baseline structure of CNN used in this paper. Convolutional and pooling layers are shared for both 2D and 3D losses, and the losses are attached to different fully connected layers.

3   3D-2D Joint Estimation of Human Body Using CNN

The task of 3D human pose estimation is defined as predicting the 3D joint positions of a human body. Specifically, we estimate the relative 3D position of each joint with respect to the root joint. The number of joints N_j is set to 17 in this paper according to the dataset used in the experiment. The key idea of our method is to train CNN which performs 3D pose estimation using both image features from the input image and 2D pose information retrieved from the same CNN. In other words, the proposed CNN is trained for both 2D joint classification and 3D joint regression tasks simultaneously. Details of each part is explained in the following subsections.

3.1   Structure of the Baseline CNN

The CNN used in this experiment consists of five convolutional layers, three pooling layers, two parallel sets of two fully connected layers, and loss layers for 2D and 3D pose estimation tasks. The CNN accepts a 225×225 sized image as an input. The sizes and the numbers of filters as well as the strides are specified in Figure 1. The filter sizes of convolutional and pooling layers are the same as those of ZFnet [21], but we reduced the number of feature maps to make the network smaller.

Joint optimization using both 3D and 2D information helps CNN to learn more meaningful features than the optimization using 3D regression alone. Li et al. [5] trained a CNN both for 2D joint detection task and for 3D pose regression task. Since both tasks share the same convolutional layers, features that are useful for estimating both 2D and 3D positions of joints in an image are learned in convolutional layers. Following the idea, we also used both 2D and 3D loss functions in the CNN. Convolutional layers are shared, and the feature maps after the last pooling layer are connected to two different fully connected layers, each of which is connected to 2D loss function and 3D loss function respectively (See Figure 1).

We formulated 2D pose estimation as a classification problem. For the 2D classification task, we divided an input image into $N_g \times N_g$ grids and treat each grid as a separate class, which results in N_g^2 classes per joint. The ground truth

label is assigned in accordance with the ground truth position of each joint. When the ground truth joint position is near the boundary of a grid, zero-one labeling that is typically used for multi-class classification may give unprecise information. Therefore, we used a soft label which assigns non-zero probability to the four nearest neighbor grids from the ground truth joint position. The target probability for the ith grid g_i of the jth joint is inversely proportional to the distance from the ground truth position, i.e.,

$$\hat{p}_j(g_i) = \frac{d^{-1}(\hat{\mathbf{y}}_j, \mathbf{c}_i)I(g_i)}{\sum_{k=1}^{N_g^2} d^{-1}(\hat{\mathbf{y}}_j, \mathbf{c}_k)I(g_k)}, \tag{1}$$

where $d^{-1}(\mathbf{x}, \mathbf{y})$ is the inverse of the Euclidean distance between the point $\mathbf{x}$ and $\mathbf{y}$ in the 2D pixel space, $\hat{\mathbf{y}}_j$ is the ground truth position of the jth joint in the image, and $\mathbf{c}_i$ is the center of the grid g_i. $I(g_i)$ is an indicator function that is equal to 1 if the grid g_i is one of the four nearest neighbors, i.e.,

$$\mathbf{I}(g_i) = \begin{cases} 1 & \text{if } d(\hat{\mathbf{y}}_j, \mathbf{c}_i) < w_g \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where w_g is the width of a grid. Hence, higher probability is assigned to the grid closer to the ground truth joint position, and $\hat{p}_j(g_i)$ is normalized so that the sum of the class probabilities is equal to 1. Finally, the objective of the 2D classification task for the jth joint is to minimize the following cross entropy loss function.

$$\mathcal{L}_{2D}(j) = -\sum_{i=1}^{N_g^2} \hat{p}_j(g_i) \log p_j(g_i), \tag{3}$$

where $p_j(g_i)$ is the probability that comes from the softmax output of the CNN.

On the other hand, estimating 3D position of joints is formulated as a regression task. Since the search space is much larger than the 2D case, it is undesirable to solve 3D pose estimation as a classification task. The 3D loss function is designed as a square of the Euclidean distance between the prediction and the ground truth. We estimate 3D position of each joint relative to the root node. Hence, the loss function for the jth joint when the root node is the rth joint becomes

$$\mathcal{L}_{3D}(j, r) = \left\| \mathbf{R_j} - (\hat{\mathbf{J}}_\mathbf{j} - \hat{\mathbf{J}}_\mathbf{r}) \right\|^2, \tag{4}$$

where $\mathbf{R_j}$ is the predicted relative 3D position of the jth joint from the root node, $\hat{\mathbf{J}}_\mathbf{j}$ is the ground truth 3D position of the jth joint, and $\hat{\mathbf{J}}_\mathbf{r}$ is that of the root node. The overall cost function of the CNN combines (3) and (4) with weights, i.e.,

$$\mathcal{L}_{all} = \lambda_{2D} \sum_{j=1}^{N_j} \mathcal{L}_{2D}(j) + \lambda_{3D} \sum_{j \neq r}^{N_j} \mathcal{L}_{3D}(j, r). \tag{5}$$
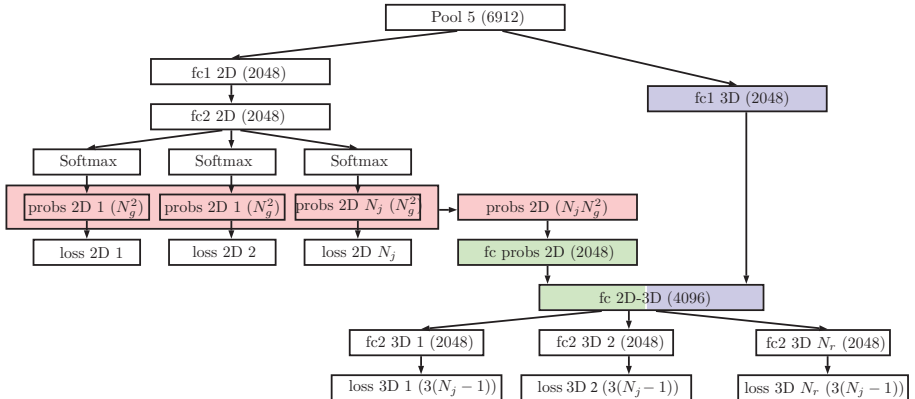
Fig. 2. Structure of fully connected layers and loss functions in the proposed CNN. The numbers in parentheses indicate the dimensions of the corresponding output feature vectors.

3.2   3D Joint Regression with 2D Classification Features

In the baseline architecture in Figure 1, 2D and 3D losses are separated with different fully connected layers. Though convolutional layers learn features relevant to both 2D and 3D pose estimation thanks to the shared convolutional layers, the probability distribution that comes from 2D classification may give more stable and meaningful information in estimating 3D pose. The joint locations in an image are usually a strong cue for guessing 3D pose. To exploit 2D classification result as a feature for the 3D pose estimation, we concatenate the outputs of softmax in the 2D classification task with the outputs of the fully connected layers in the 3D loss part. The proposed structure after the last pooling layer is shown in Figure 2. First, the 2D classification result is concatenated (*probs 2D* layer in Figure 2) and passes the fully connected layer (*fc probs 2D*). Then, the feature vectors from 2D and 3D part are concatenated (*fc 2D-3D*), which is used for 3D pose estimation task. Note that the error from the *fc probs 2D* layer is not back-propagated to the *probs 2D* layer to ensure that layers used for the 2D classification are trained only by the 2D loss part. The idea of using 2D classification result as an input for another task is similar to [3], which repeatedly uses the 2D classification result as an input by concatenating it with feature maps from CNN. Unlike [3], we simply vectorized the softmax result to produce $N_g \times N_g \times N_j$ feature vector rather than convolving the probability map with features in the convolutional layers.

The proposed framework can be trained end-to-end via back-propagation algorithm. Because 2D classification will give an inaccurate prediction in the early stage of training, it is possible that 3D regression may be disturbed by the classification result. However, we empirically found that 3D loss converges successfully, and the performance of 3D pose estimation improves as well, as explained in Section 5.

(a)   (b)

Fig. 3. Visualization of joints to be estimated (Red and green dots). (a) Baseline method predicts relative position of the joints with respect to one root node (Green dot). (b) For multiple pose regression, the positions of joints are estimated with respect to multiple root nodes (Green dots).

3.3 Multiple 3D Pose Regression from Different Root Nodes

In the baseline architecture, we predicted the relative 3D position of each joint with respect to only one root node which is around the position of the hip. When joints such as wrists or ankles are far from the root node, the accuracy of regression may be degraded. Li et al. [5] designed a 3D regression loss to estimate the relative position between each joint and its parent joint. However, errors may be accumulated when intermediate joint produces inaccurate result in this scheme. As an alternative solution, we estimate the relative position over multiple joints. We denote the number of selected root nodes as N_r. For the experiments in this paper, we set $N_r = 6$ and selected six joints so that most joints can either be the root node or their neighbor nodes. The selected joints are visualized in Figure 3(b). Therefore, there are six 3D regression losses in the network, which is illustrated in Figure 2. Then, the overall loss becomes

$$\mathcal{L}_{all} = \lambda_{2D} \sum_{j=1}^{N_j} \mathcal{L}_{2D}(j) + \lambda_{3D} \sum_{r \in \mathbf{R}} \sum_{j \neq r}^{N_j} \mathcal{L}_{3D}(j, r), \qquad (6)$$

where $\mathbf{R}$ is the set containing the joint indices that are used as root nodes. When the 3D losses share the same fully connected layers, the trained model outputs the same pose estimation results across all joints. To break this symmetry, we put the fully connected layers for each 3D losses (*fc2 3D* layers in Figure 2).

At the test time, all the pose estimation results are translated so that the mean of each pose becomes zero. Final prediction is generated by averaging the translated results. In other words, the 3D position of the jth joint $\mathbf{X}_j$ is calculated as

$$\mathbf{X}_j = \frac{\sum_{r \in \mathbf{R}} \mathbf{X}_j^{(r)}}{N_r}, \qquad (7)$$

where $\mathbf{X}_j^{(r)}$ is the predicted 3D position of the jth joint when the rth joint is set to a root node.

4   Implementation Details

The proposed method is implemented using Caffe framework [22]. Batch normalization [23] is applied to all convolutional and fully connected layers. Also, drpoout [24] is applied to every fully connected layers with drop probability of 0.3. Stochastic gradient descent of batch size 128 is used for optimization. Initial learning rate is set to 0.01, and it is decreased by a factor of 0.5 for every 4 epochs. The optimization is finished after 28 epochs. The momentum and the weight decay parameters are set to 0.9 and 0.001 respectively. The weighting parameter λ_{2D} and λ_{3D} are initially set to 0.1 and 0.5 respectively. λ_{2D} is decreased to 0.01 after 16 epochs because we believe that 2D pose information plays an important role in learning informative features especially in the early stage of training.

Input images are cropped using the segmentation information provided with the dataset so that a person is located around the center of an image. The cropped image is resized to 250×250. We randomly cropped the resized image into an image of 225×225 size, then it is fed into the CNN as an input image. During the test time, only the center crop is evaluated for the pose prediction. Data augmentation based on the principal component analysis of training images [25] is also applied. N_g is set to 16, so the input image is divided into 256 square grids for 2D loss calculation. N_r is set to 6, and the position of the root nodes are illustrated in Figure 3(b).

For the ground truth 3D pose that is used in the training step, we firstly translated the joints to make the shape to be zero mean. Then, we scaled the 3D shape so that the Frobenius norm of the 3D shape becomes 1. Since different person has different height and size, we believe that the normalization helps to reduce ambiguity of scale and to predict scale-invariant poses. During the testing phase, scale should be recovered to evaluate the performance of the algorithm. Similar to [19], we infer the scale using the training data. The lengths of all connected joints from the training set are averaged. The scale of the result from the test data is determined so that the length of connected joints in the estimated shape is equal to the pre-calculated average length. Since the lengths for arms and legs from the estimated shape often have a large variation, we only used the length of joints in the torso which is stable in most cases.

5   Experimental Results

We used Human 3.6m dataset [8] to evaluate our method and compared the proposed method with the other 3D human pose estimation algorithms. The dataset provides 3D human pose information acquired by a motion capture system with synchronized RGB images. It consists of 15 different sequences which

Table 1. Quantitative results on Human 3.6m dataset. The best and the second best methods for each sequence are marked as (1) and (2) respectively.

	Directions	Discussion	Eating	Greeting	Phoning	Photo
LinKDE [8]	132.71	183.55	132.37	164.39	162.12	205.94
Li and Chan [5]	-	148.79	104.01	127.17	-	189.08
Li et al. [6]	-	136.88	96.94	124.74	-	168.68
Tekin et al. [7]	-	129.06	91.43	121.68	-	162.17
Tekin et al. [20]	102.41	147.72	$88.83^{(2)}$	125.28	118.02	182.73
Zhou et al. [19]	$87.36^{(1)}$	$109.31^{(1)}$	$87.05^{(1)}$	$103.16^{(1)}$	$116.18^{(2)}$	$143.32^{(1)}$
Our method	$100.34^{(2)}$	$116.19^{(2)}$	89.96	$116.49^{(2)}$	$115.34^{(1)}$	$149.55^{(2)}$

	Posing	Purchases	Sitting	Sitting Down	Smoking	Waiting
LinKDE [8]	150.61	171.31	151.57	243.03	162.14	170.69
Li and Chan [5]	-	-	-	-	-	-
Li et al. [6]	-	-	-	-	-	-
Tekin et al. [7]	-	-	-	-	-	-
Tekin et al. [20]	$112.38^{(2)}$	129.17	138.89	224.90	118.42	138.75
Zhou et al. [19]	$106.88^{(1)}$	$99.78^{(1)}$	$124.52^{(1)}$	$199.23^{(2)}$	$107.42^{(2)}$	$118.09^{(1)}$
Our method	117.57	$106.94^{(2)}$	$137.21^{(2)}$	$190.82^{(1)}$	$105.78^{(1)}$	$125.12^{(2)}$

	Walk Dog	Walking	Walk Together	Average
LinKDE [8]	177.13	96.60	127.88	162.14
Li and Chan [5]	146.59	77.60	-	-
Li et al. [6]	132.17	69.97	-	-
Tekin et al. [7]	130.53	65.75	-	-
Tekin et al. [20]	$126.29^{(2)}$	$55.07^{(1)}$	$65.76^{(1)}$	124.97
Zhou et al. [19]	$114.23^{(1)}$	79.39	97.70	$113.01^{(1)}$
Our method	131.90	$62.64^{(2)}$	$96.18^{(2)}$	$117.34^{(2)}$

contain specific actions such as discussion, eating, walking, and so on. There are 7 different persons who perform all 15 actions. We trained and tested each action individually. Following the previous works on the dataset [5,19], we used 5 subjects (S1, S5, S6, S7, S8) as a training set, and 2 subjects (S9, S11) as a test set. The training and the testing procedures are conducted on a single PC with a Titan X GPU. Training procedure takes 7-10 hours for one action sequence depending on the number of training images. For the evaluation metric, we used the mean per joint position error (MPJPE).

First, we compared the performance of our method with the conventional methods on Human 3.6m dataset. Table 1 shows the MPJPE of our method and the previous works. The smallest and the second smallest errors for each sequence are marked. Our method achieves the best performance in 3 sequences and shows the second best performance in 9 sequences. Note that the methods of [20] and [19] make use of temporal information from multiple frames. Meanwhile, our method produce a 3D pose from a single image. Our method is also beneficial against [20] and [19] in terms of running time and the simplicity of the algorithm

Table 2. Comparison of our method with the baseline.

	Discussion	Eating	Greeting	Phoning	Photo	Walking
Baseline CNN	125.45	95.21	120.69	119.66	153.76	72.55
Multi-reg	122.71	94.67	119.70	119.25	153.54	71.19
2D-cls	118.19	91.39	118.19	115.84	149.97	64.27
Multi-reg+2D-cls	**116.19**	**89.96**	**116.49**	**115.34**	**149.55**	**62.64**

(a)   (b)
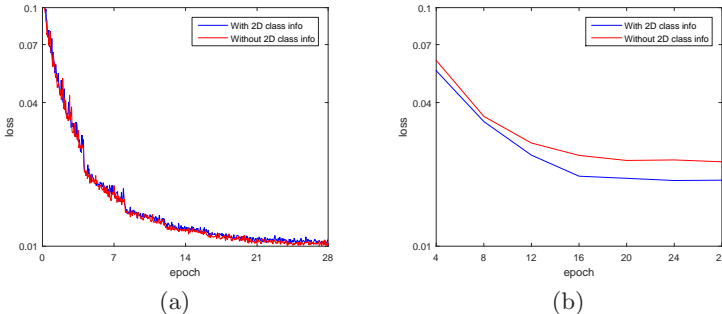
Fig. 4. The 3D losses of Walking sequence with and without 2D classification result integration. (a) Losses for training data. (b) Losses for test data.

since the estimation is done by a forward pass of the CNN and simple averaging. Moreover, from Table 1, it is justified that our method outperforms the CNN based methods that predict 3D pose from a single image [5,6,7].

Next, we measured the effect of our contribution, 1) the integration of 2D classification results and 2) regression from multiple root nodes, by comparing their performance with the baseline CNN. Note that the 2D classification loss is also used in the baseline CNN. The difference of the baseline CNN is that 2D classification results are not propagated to the 3D loss part, i.e., *probs 2D*, *fc probs 2D* and *fc 2D-3D* layers in Figure 2 are deleted in the baseline CNN. The results are shown in Table 2. Multiple regression from different root nodes and the integration of 2D classification results are denoted as *Multi-reg* and *2D-cls* respectively. Both modifications improve the result over the baseline CNN in all tested sequences. 2D classification integration showed larger error reduction rate than the multiple regression strategy, which proves that the 2D classification information is indeed a useful feature for 3D pose estimation. Multiple regression can be considered as an ensemble of different estimation results, which improves the overall performance. It can be found that the error reduction rate for the case that both 2D classification result integration and multiple regression are applied is slightly bigger than the sum of the reduction rates when they are individually applied in most sequences. Since each 3D pose regressor takes advantage of 2D classification feature, there is a synergy effect between the two schemes.

We also analyzed the effect of integrating 2D classification result in terms of 3D losses. Training losses are measured every 50 iterations and testing losses are

measured every 4 epochs. The results on the Walking sequence are illustrated in Figure 4. For the training data, loss is slightly smaller when 2D classification information is not used (Figure 4(a)). However, test loss is much lower when 2D classification information is used(Figure 4(b)). This indicates that 2D classification information impose generalization power and reduce overfitting for CNN regressor. Since the 2D joint probabilities provide more abstracted and subject-independent information compared to the features obtained from an image, the CNN model is able to learn representations that are robust to variability of subjects in the image.

Finally, we illustrated qualitative results of our method in Figure 5. Input images, ground truth poses, and the estimation results with and without 2D classification information are visualized. Different colors are used to distinguish the left and right sides of human bodies. It can be found that 2D pose estimation results help reducing the error of 3D pose estimation. While the CNN which does not use 2D classification information gives poor results, the estimated results are much more improved when 2D classification information is used for 3D pose estimation.

6   Conclusions

In this paper, we propose novel strategies which improve the performance of the CNN that estimates 3D human pose. By reusing 2D joint classification result, the relationship between 2D pose and 3D pose is implicitly learned during the training phase. Moreover, multiple regression results with different root nodes gives an effect of ensemble learning. When both strategies are combined, 3D pose estimation results are significantly improved and showed comparable performance to the state-of-the-art methods without exploiting any temporal information of video sequences.

We expect that the performance can be further improved by incorporating temporal information to the CNN by applying the concepts of recurrent neural network or 3D convolution [26]. Also, efficient aligning method for multiple regression results may boost the accuracy of pose estimation.

References

1. Toshev, A., Szegedy, C.: Deeppose: Human pose estimation via deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014) 1653–1660
2. Carreira, J., Agrawal, P., Fragkiadaki, K., Malik, J.: Human pose estimation with iterative error feedback. arXiv preprint arXiv:1507.06550 (2015)
3. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. arXiv preprint arXiv:1602.00134 (2016)
4. Bengio, Y.: Learning deep architectures for ai. Foundations and trends® in Machine Learning **2**(1) (2009) 1–127
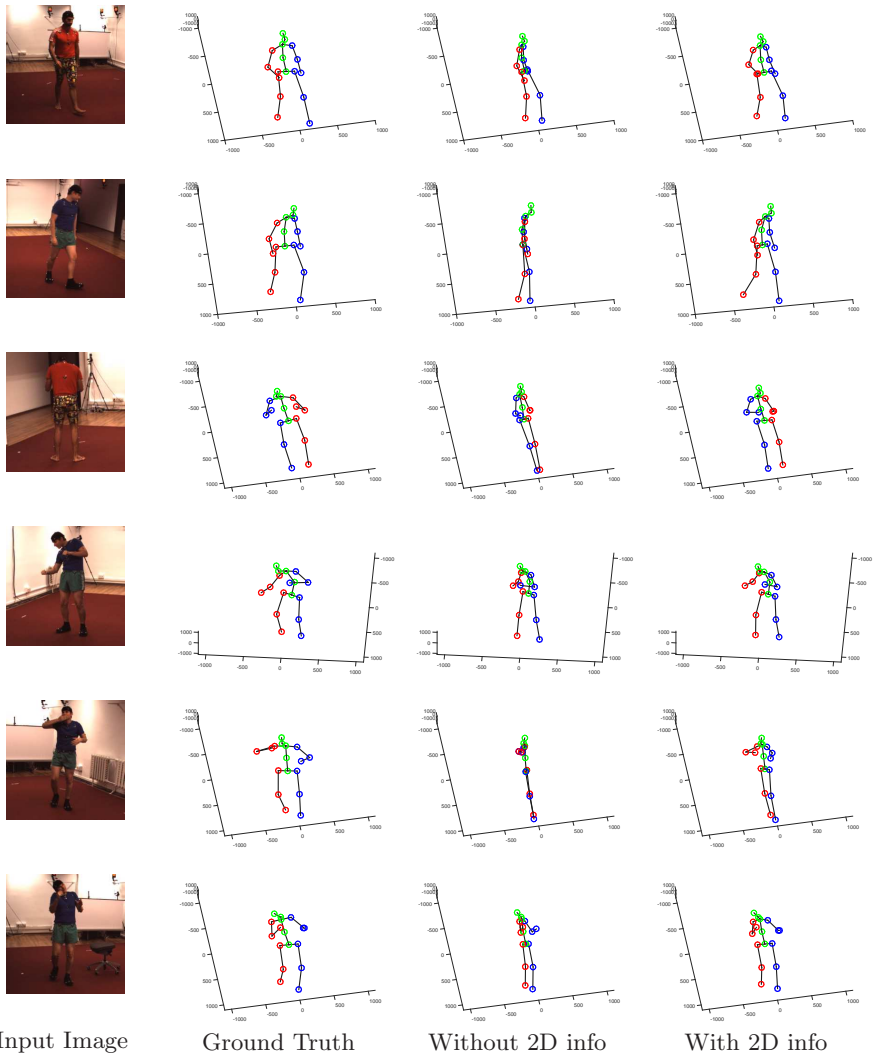
Fig. 5. Qualitative results of our method on Human 3.6m test dataset. The estimation results are compared with the results from the baseline method. First column : input images. Second column : Ground truth 3D position. Third column : Pose estimation result without 2D classification information integration. Fourth column : Pose estimation result with 2D classification information integration.

5. Li, S., Chan, A.B.: 3d human pose estimation from monocular images with deep convolutional neural network. In: Asian Conference on Computer Vision, Springer (2014) 332–347

6. Li, S., Zhang, W., Chan, A.B.: Maximum-margin structured learning with deep networks for 3d human pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 2848–2856

7. Tekin, B., Katircioglu, I., Salzmann, M., Lepetit, V., Fua, P.: Structured prediction of 3d human pose with deep neural networks. arXiv preprint arXiv:1605.05180 (2016)

8. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: 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) (JUL 2014) 1325–1339

9. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. International Journal of Computer Vision **61**(1) (2005) 55–79

10. Andriluka, M., Roth, S., Schiele, B.: Pictorial structures revisited: People detection and articulated pose estimation. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE (2009) 1014–1021

11. Dantone, M., Gall, J., Leistner, C., Van Gool, L.: Human pose estimation using body parts dependent joint regressors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2013) 3041–3048

12. Yang, Y., Ramanan, D.: Articulated pose estimation with flexible mixtures-of-parts. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE (2011) 1385–1392

13. Bourdev, L., Malik, J.: Poselets: Body part detectors trained using 3d human pose annotations. In: 2009 IEEE 12th International Conference on Computer Vision, IEEE (2009) 1365–1372

14. Fan, X., Zheng, K., Lin, Y., Wang, S.: Combining local appearance and holistic view: Dual-source deep neural networks for human pose estimation. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2015) 1347–1355

15. Chu, X., Ouyang, W., Li, H., Wang, X.: Structured feature learning for pose estimation. (June 2016)

16. Agarwal, A., Triggs, B.: Recovering 3d human pose from monocular images. IEEE transactions on pattern analysis and machine intelligence **28**(1) (2006) 44–58

17. Ionescu, C., Li, F., Sminchisescu, C.: Latent structured models for human pose estimation. In: 2011 International Conference on Computer Vision, IEEE (2011) 2220–2227

18. Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., Moore, R.: Real-time human pose recognition in parts from single depth images. Communications of the ACM **56**(1) (2013) 116–124

19. Zhou, X., Zhu, M., Leonardos, S., Derpanis, K.G., Daniilidis, K.: Sparseness meets deepness: 3d human pose estimation from monocular video. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2016)

20. Tekin, B., Rozantsev, A., Lepetit, V., Fua, P.: Direct prediction of 3d body poses from motion compensated sequences. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2016)

21. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European Conference on Computer Vision, Springer (2014) 818–833

22. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadar-rama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)

23. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of The 32nd International Conference on Machine Learning. (2015) 448–456

24. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**(1) (2014) 1929–1958

25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105

26. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: 2015 IEEE International Conference on Computer Vision (ICCV), IEEE (2015) 4489–4497