Filling the Gaps: Predicting Missing Joints of Human Poses Using Denoising Autoencoders

Nicolò Carissimi[1,2(✉)], Paolo Rota[1,3], Cigdem Beyan[1], and Vittorio Murino[1,4]

[1] Istituto Italiano di Tecnologia, Genoa, Italy
{nicolo.carissimi,paolo.rota,cigdem.beyan,vittorio.murino}@iit.it
[2] Università degli Studi di Genova, Genoa, Italy
[3] Università degli Studi di Trento, Trento, Italy
[4] Università degli Studi di Verona, Verona, Italy

Abstract. State of the art pose estimators are able to deal with different challenges present in real-world scenarios, such as varying body appearance, lighting conditions and rare body poses. However, when body parts are severely occluded by objects or other people, the resulting poses might be incomplete, negatively affecting applications where estimating a full body pose is important (e.g. gesture and pose-based behavior analysis). In this work, we propose a method for predicting the missing joints from incomplete human poses. In our model we consider missing joints as noise in the input and we use an autoencoder-based solution to enhance the pose prediction. The method can be easily combined with existing pipelines and, by using only 2D coordinates as input data, the resulting model is small and fast to train, yet powerful enough to learn a robust representation of the low dimensional domain. Finally, results show improved predictions over existing pose estimation algorithms.

Keywords: Human pose estimation · Generative methods

1 Introduction

2D human pose estimation is a well-known research topic that has been studied for years by the computer vision community. The problem consists in finding the location of body parts in an image depicting one or more persons.

Early works [6,11,23] used handcrafted features to detect local body parts and graphical models to infer global pose estimates. These works have shown promising results but they were not good enough to be applied in real-world scenarios. Recently, the creation of huge annotated datasets [4,20] and the use of deep learning techniques [28,29] led to significant improvements, not only in specific 'ad-hoc' datasets but also in generalizing to wider and more unconstrained scenarios.

L. Leal-Taixé and S. Roth (Eds.): ECCV 2018 Workshops, LNCS 11130, pp. 364–379, 2019.
https://doi.org/10.1007/978-3-030-11012-3_29

Even though existing methods produce high precision results, the problem of human pose estimation still remains challenging. For instance, real world images present several complexities, such as body appearance variability due to different clothing and lighting, uncommon body poses and crowded scenarios which introduce occlusion problems. Specifically, when parts of the body are severely occluded, the resulting missing visual information might lead to the generation of incomplete or wrong body poses. We show an example in Fig. 1, where the image is taken from Salsa [3], a well known dataset for human behavior analysis and social signal processing; the scene represents a typical indoor crowded scenario, with challenging lighting conditions and people occluding each other. The resulting estimated body joints and poses (Fig. 1, generated by the tool OpenPose [7], one of the state of the art and most famous pose estimators) present several missing joints (typically wrists, ankles or entire legs) and some wrong limbs. When human poses are used as features for social interaction analysis or as inputs for other tasks, the missing information might severely harm the final accuracy of the entire system. Another example is when 2D poses are used for (bottom-up) 3D pose estimation: Fig. 2 shows how the missing joints dramatically harm the final 3D reconstruction. Therefore, the need of a method that is able to estimate the whole set of joints correctly is of paramount importance.

In this paper, we tackle this problem by proposing an algorithm that estimates the position of the missing joints and completes partial human poses generated by pose estimation algorithms. We cast the task as a denoising problem, where the corrupted signal is represented by the partial human pose, and the resulting uncorrupted signal is the full reconstructed pose. Inspired by the architecture of the denoising variational autoencoders [26,30] we propose a network that generates complete poses from the noisy or missing 2D joints coordinates generated by any human pose estimator. Our model does not require RGB data, is simple, lightweight and fast to train. Despite its simplicity, we show, both quantitatively and qualitatively, improved predictions. Additionally, by ditching image data, our model is also able to estimate joints positions which are outside the camera view, and thus not detectable by standard 2D pose estimation algorithms. Finally, our algorithm is easily pluggable in any existing architecture and can be used as a pose-based feature extractor for high-level human behavior understanding in the wild.

The rest of the paper is organized as follows. In Sect. 2, we review the related work in human pose estimation. Section 3 describes our proposed model. In Sect. 4, we show quantitative and qualitative results. Finally, we present concluding remarks in Sect. 5, together with a discussion about how the proposed method can be utilized for higher level computer vision tasks.

2 Related Work

Pose Estimation. For many years, human pose estimation algorithms focused on single person pose estimation (SPPE). Recently, interest has shifted to "in the wild" multi-person pose estimation (MPPE), which presents a different set

Fig. 1. Examples of incomplete and wrong 2D poses estimated on a frame from Salsa [3] using the tool OpenPose [7].

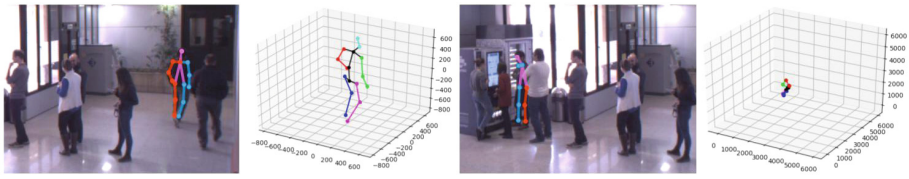

Fig. 2. Effect of complete and incomplete 2D poses on bottom-up 3D pose estimation (3D poses estimated using [27]). A complete 2D pose (a) leads to the estimation of a plausible 3D pose (b), while an incomplete 2D pose (c) (with missing arms) leads to a wrong 3D pose estimation (d).

of challenges, including detecting an unknown number of people, possibly highly dense scenarios with severe occlusions and complex body interactions. In this work, given that we focus on multi-person scenarios, we mainly review previous MPPE works. These approaches can be categorized into two families: *top-down* and *bottom-up* approaches [7].

Top-down approaches require person detectors on the whole image and use SPPE algorithms to infer poses on the single detections. Having a body centered in the input image is a strong prior that enables SPPE algorithms to use both local and global information for the final predictions [22], resulting in robust predictions. Unfortunately, one of the biggest disadvantages of these approaches is that the final predictions are heavily influenced by the initial person detector results, which, in very crowded scenarios, can lead to poor detections, bounding boxes not centered on a single person or that cut parts of the body. Iqbal and Gall [15] use Faster R-CNN [25] as person detector, and a convolutional pose machine [32] as an SPPE algorithm to generate candidate joints. The resulting bounding boxes might show multiple people; thus, in order to associate each joint to the

correct person and remove the wrong candidates, inference is performed locally on a fully connected graph for each person using integer linear programming. Fang et al. [10] tackle the problems of bounding box error, i.e. bounding boxes which are not perfectly centered on a person, and of redundant detections, i.e. when multiple bounding boxes are generated for a single person. The problem of the bounding box error is attenuated by a network which generates higher quality bounding boxes from existing ones, based on spatial transformer networks [16]. Finally, poses generated by redundant detections are merged into a single, most confident one by using a novel technique called parametric pose non-maximum suppression. Chen et al. [8] focus on detecting "hard to predict" joints, i.e. joints which are occluded, invisible or in front of complex backgrounds. They devise a two stage network, where, during training, the first stage predicts the visible joints, while the second stage focuses on the hard joints by selecting the top M losses and backpropagating only their gradient.

Bottom-up approaches, on the other end, take the whole image as an input, without any prior information on person location and number. Their pipeline is usually composed of two stages (which can be sequential or parallel), where the first one predicts candidate locations for all joint types, and the second one groups the joints into individual human poses. These algorithms do not need to rely on the performance of person detectors and the computational time does not depend on the number of detections. However, as a downside, the lack of prior information, such as person location and scale, makes the resulting poses less precise. Pishchulin et al. [24] propose a bottom-up method which jointly detects a set of candidate body joints and associates them to single people by casting the problem as a graph partitioning one, where nodes are joint detections and edges are scores based on spatial offsets and appearance features. The partitioning is solved via integer linear programming on a fully connected graph of all the detected joints, which requires a long computational time. Insafutdinov et al. [14] propose an improved approach over [24], by using better part detectors, improved regressors for the score generation, and a faster partitioning algorithm based on incremental optimization, although it still requires several minutes of computational time. Cao et al. [7] focus on improving computational time by proposing a real-time approach which jointly learns to detect and associate joints. Extending the work from [32], the proposed network has two branches: one for joints and one for part affinity fields (PAFs) prediction, where a PAF can be seen as encoding the probability of a pixel to belong to the limb between two predicted joints. Finally, a bipartite graph matching solver is used to get the final estimated poses. In contrast to the previously mentioned multi-stage pipelines, Newell et al. [21] propose a novel approach where the network is taught to simultaneously detect and group joints, without the need of additional aggregation steps. For each joint, the network outputs a corresponding embedding vector, and these vectors are then used to cluster joints into final human poses.

Our proposed method *does not estimate poses* and it differs from all the existing works in two fundamental ways: *(i)* its primary goal is to correct existing predictions, by regressing missing joints in an incomplete input pose, estimating

a most likely pose given the actual detection; *(ii)* the proposed method uses only 2D coordinates as input, not RGB data; even if we discard rich information, we show that partial 2D coordinates provide enough information for robust predictions.

Data Denoising and Restoration. The problem of dealing with missing or incomplete data in machine learning arises in many applications. Recent strategies make use of generative models to impute missing or corrupted data. Advances in computer vision using deep generative models have found applications in image/video processing, such as denoising, restoration, super-resolution and inpainting. In [17], Jain and Seung train convolutional layers in an unsupervised way for the task of denoising images, by reducing the error between the original (input) and the reconstructed images, the latter obtained by applying gaussian noise to the former. Xu et al. [34] state that image degradation can be modeled as a translation-invariant convolution operation and that image restoration can be achieved with the inverse process, i.e. deconvolution; thus, a deep convolutional neural network (CNN) is used to learn the deconvolution operation in a data driven way, without the need to have any prior knowledge of the cause of image degradation. Xie et al. [33] propose the use of stacked denoising autoencoders with layer wise training for image denoising and inpainting.

Inspired by these methods, we cast the problem of missing joints prediction as a denoising and restoration problem.

3 Methodology

The choice of our model is motivated by two main reasons. In the first place, the model should be able to predict missing information and, second, has to deal with low dimensional data. Occlusions and degraded visual data might cause a pose detector to miss some types and number of joints in an unpredictable way. The resulting partial human pose can, thus, be seen as a noisy, stochastically corrupted version of the original data which is the complete human pose in our case. The model must, then, be able to learn a robust representation of the data even when parts of the data are missing. Unlike RGB images, which are composed by hundreds, or thousands of pixels, our domain data are small vectors of a few concatenated 2D coordinates (see Sect. 3.2), therefore we choose a model which is simple, yet powerful enough to learn a robust representation of this low dimensional domain data. Auto-encoders, as seen in previous works [26,30], are a powerful tool for learning representations of complex data distributions, and their denoising variant [30] is specifically designed to deal with incomplete input data.

We now proceed with a short review of the theory behind auto-encoders, denoising auto-encoders and one of their most recent variants, variational auto-encoders (Sect. 3.1). Finally, we describe in detail the architecture of the network we use (Sect. 3.2).

3.1 Auto-Encoders

Auto-encoders have been introduced long ago by Rumelhart et al. [26] they consist in an unsupervised learning model and have been used for different purposes such as dimensionality reduction, feature extraction [30], pre-training of deep nets [5,31], data generation and reconstruction [13]. Concretely, an auto-encoder is a type of multi-layer neural network trained to map the input to a different representation of it, so that the input can be reconstructed from that representation. The simplest form of an auto-encoder has a single hidden layer which maps (*encodes*) an input $\mathbf{x}$ to its new representation $\mathbf{y}$

$$\mathbf{y} = s(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

where s is a (usually non-linear) activation function, while W and b are, respectively, the weights and bias of the layer. The encoded input $\mathbf{y}$ is, then, mapped back (*decoded*) to a reconstruction $\mathbf{x_r}$ of the input

$$\mathbf{x_r} = s(\mathbf{W}'\mathbf{y} + \mathbf{b}'). \tag{2}$$

Training is performed by minimizing a loss function

$$L(\mathbf{x}, \mathbf{x_r}) \tag{3}$$

which, in our case, is the mean squared error MSE, calculated between the reconstructed input $\mathbf{x_r}$ and the target output, which is the input $\mathbf{x}$ itself. If the dimension of $\mathbf{y}$ is smaller than the dimension of $\mathbf{x}$, the auto-encoder is called undercomplete; on the other end, if the dimension of $\mathbf{y}$ is larger, the auto-encoder is called overcomplete.

If the dimension of the hidden units is larger than the original input, the auto-encoder might learn the identity function; however, there are different techniques to avoid this occurrence. One of these techniques introduces randomness during training: the network is fed with a stochastically corrupted version of the input $\mathbf{x_c}$, while the target output remains the original uncorrupted input $\mathbf{x}$. This training approach has been introduced by Vincent et al. in [30] and the resulting models are called *denoising auto-encoders*. Their original purpose was to make the learned representation more robust to partial corruption of the input, but they present an additional useful property, i.e. the ability to reconstruct missing data from the input, which is well suited for our problem of missing joints prediction.

One downside of standard auto-encoders is that they tend to map similar input samples to latent vectors which might be very close to each other, resulting in almost identical reconstructions. This behavior is acceptable when input data represents classes (e.g. images of numbers or letters). On the contrary, preserving small input differences in the reconstruction is very important when

dealing with human poses, which do not form a clustered space, but a continuous, smooth domain. Variational auto-encoders [19] can learn such a continuous representation by design, making them more suited for our problem. Similarly to classic auto-encoders, they have the same encoder/decoder structure (where the encoder maps the input to a latent representation, and the decoder reconstructs the input from such representation). The main difference is that the latent variables are not a "compressed representation" of the domain data itself, but they encode the parameters (i.e. the mean μ and standard deviation σ) of a *distribution* (typically an n-dimensional Gaussian one) modeling the input data. In order to force this, another term is added to the loss (see Eq. 3), i.e. the Kullback-Leibler divergence (D_{KL}) [2], which measures the divergence between two probability distributions and has the form

$$D_{KL}(P\|Q) = -\sum_i P_i log(Q_i/P_i) \tag{4}$$

where P is the encoded n-dimensional Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ and Q is the target standard normal distribution.

3.2 Modeling the Human Pose

Since the chosen reconstruction loss needs a complete human pose as the target output, we need to select full human poses from the dataset as training data. Each pose is represented by a set of n 2D locations, where n is the number of joints (see Sects. 4.1 and 4.2). The concatenation of these joints produces a vector of $n * 2$ elements (the x and y coordinates) which is the input of the network. Since the coordinates of the annotated joints are labeled in the image space, poses which are very similar to each other might appear in different parts of the image, resulting in input vectors with very different values. We, thus, normalize them using the following procedure: first we find the center of the torso (C_T) by averaging the coordinates of the neck and at least one of the shoulders and hip joints; the pose is then translated to the obtained 2D point and finally scaled by the distance between the neck and C_T. At testing time, this normalization technique requires an incomplete pose to have all the aforementioned joints, negatively affecting the number of poses that is processed by the network (see Sect. 4.4). Given that we are using a denoising auto-encoder, the training data must also be *corrupted*. We do this by adding noise to the previously normalized poses, randomly masking a small number of joints.

Figure 3 shows the overall architecture: since the input vector is small compared to the space of the data we want to reconstruct, we choose to implement an overcomplete auto-encoder. The encoder and decoder are composed by 2 hidden layers, each one gradually encoding (and decoding) more robust features. Layers μ and σ represent, respectively, the mean and standard deviation of the distribution we want to learn, while the final layer of the encoder represents a sample of it.

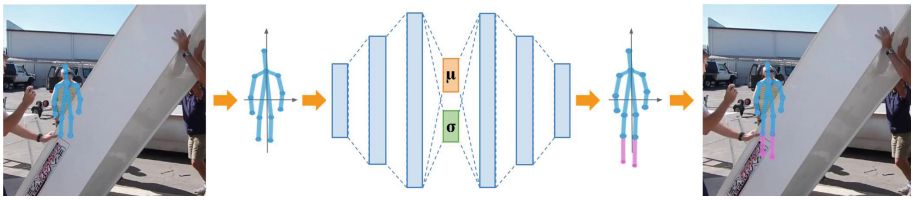

Fig. 3. The pipeline of our method. Given an RGB image, a human pose prediction algorithm is used to generate one or more poses. The incomplete ones are, then, normalized and fed to the auto-encoder, which outputs the corresponding full human poses.

4 Experiments

In this section, we report quantitative and qualitative results of our method, evaluated on two datasets, MPII Human Pose [4] and Microsoft's COCO Keypoint Detection [20], which are the most famous and widely datasets for multi-person pose estimation.

4.1 MPII Human Pose Dataset

The MPII Human Pose dataset [4] consists of around 25000 images and a total of 40000 annotated human poses. The training set is composed of 28000 of these poses, while the test set is composed of 11000 poses. Images contain people engaged in numerous activities and a variety of contexts, with a high variable of articulated poses and camera perspectives. People can be fully visible, severely occluded or partially out of the camera field of view. A full pose is composed of 16 landmarks, each one corresponding to the location, in image coordinates, of a body joint (head, neck, thorax and left and right shoulders, elbows, wrists, hips, knees and ankles).

4.2 COCO Keypoint Detection Dataset

The Microsoft's COCO Keypoint Detection dataset [20] is a subset of the whole COCO dataset, focused on the localization of person keypoints. The training and validation sets contain, respectively, around 260000 and 11000 annotated human poses. Unlike MPII, a full pose is composed of 17 joints, corresponding to nose and left and right shoulders, elbows, wrists, hips, knees, ankles, eyes and ears.

4.3 Experimental Settings

Our model takes a pose as the input and generates its reconstruction. If the pose is incomplete, i.e. with one or more missing joints, a prediction of the corresponding full pose is generated as output. As described in Sect. 3.2, the loss function

needs a fully annotated pose; thus, we need to select a subset of the training data containing only complete poses. For the MPII dataset, this results in a total of, approximately, 20000 samples; we, then, use our own split (85%/15%) on the obtained data for training and validation purposes, and augment the remaining training data following a standard procedure for single pose estimation algorithms [9,22], obtaining a total of approximately 500000 training samples. In particular, we perform data augmentation by flipping and rotating the original poses ($\pm 30°$). We then normalize each pose, mask a random number of joints (from 0 up to 5, which roughly corresponds to 35% of the total number of joints) and feed the obtained data to the network.

The COCO dataset, on the contrary, has only a few thousands of complete poses, which, even after data augmentation, would not be enough for training purposes. Therefore, we decide to use the dataset only for testing. Since COCO and MPII have different annotated joint types, we feed the network (trained on MPII) with only the joints that are common between the two datasets (i.e. left and right shoulders, elbows, wrist, hips, knees and ankles) and set to zero the missing ones (head, neck and thorax).

The encoder is composed of 2 fully connected hidden layers, with 64 and 128 hidden units. Symmetrically, the decoder is composed of 2 hidden layers, with 128 and 64 hidden units, and an output layer with the same dimension as the input one. As in [12,19], we use 20 latent dimensions. Every fully connected layer has ReLu non-linearities. The loss function is the sum of MSE (between the uncorrupted input and the reconstructed pose) and the D_{KL} (Sect. 3.1). The network is implemented using TensorFlow [1] and trained with the Adam optimizer [18] with a learning rate of 1e−3.

4.4 Quantitative Analysis

In this section we show quantitative results of the proposed pipeline on the datasets described in Sects. 4.1 and 4.2. For the generation of the input poses, we use the bottom-up multi-person pose estimator OpenPose [7,32] and its matlab implementation, without modifying its preset parameters. Although OpenPose is not the best performing method on MPII and COCO anymore, and it's less precise in predicting complete human poses compared to other top-down approaches, we found it to be the more robust when tested on real-life datasets not strictly related to pose estimation and on which it wasn't trained on (such as Salsa [3]). Figure 4 shows a comparison between poses generated by OpenPose and those generated by the state of the art top-down approach called Regional Multi-Person Pose Estimation (RMPE) [10]. In (a), OpenPose (left) produces a complete and better estimation of the pose, compared to RMPE (right). In (b), OpenPose (left) cannot predict the head, the wrists and the right ankle, while RMPE (center, right) predicts all joints; however, RMPE generates two poses for the same person, due to redundant detections, and their quality is worse than the OpenPose one. Clearly, the underlying person detector is an important factor in the final performance of a top-down pose estimation algorithm. Also, top-down approaches learn not just local information (i.e. joints appearance)

but also global information (i.e. joints relative location and appearance) and this information might be harder to generalize to unseen data.

Fig. 4. Comparison between OpenPose and Regional Multi-Person Pose Estimation (RMPE) [10]. (a) and (b) left show OpenPose predictions, while (a) right and (b) center, right show RMPE predictions. Orange joints have a confidence score below 0.2. (Color figure online)

We compare OpenPose's results with the results generated by our method using two metrics, the Miss Rate (MR) and the Percentage of Correct Keypoints (PCKh). MR is computed as

$$\#joints_{missed}/\#joints_{gt} \tag{5}$$

where $\#joints_{missed}$ is the number of missed (annotated) joints and $\#joints_{gt}$ is the number of all (annotated) joints. PCKh is a standard metric in pose estimation introduced in [4] for evaluation on the MPII dataset, where a keypoint is considered as correctly predicted if its distance from the ground truth is less than a fixed threshold (specified as a fraction of the person's head size). The corresponding ground truth is assigned to each pose according to the highest PCKh.

While MR quantifies how many joints are failed to be predicted, PCKh quantifies the actual "quality" of the predictions.

We do not perform a comparison using the standard mean Average Precision (mAP) metric, which is commonly used in MPII for multi-person pose estimation, because it penalizes joints with no ground truth correspondence as false positives.

Table 1 shows that our method outperforms OpenPose in terms of number of missing joints. As can be seen, the highest missing rate differences correspond to joints which are body extremes (i.e. wrists and ankles) and thus more prone to be occluded. Even though our method is supposed to predict all missing joints, the missing rate is not 0 because it relies on the detection of the subjects by the baseline human pose estimator.

The quality of the predictions generated by our method can be seen in Table 2, where its PCKh is better than the OpenPose one, especially (as for the missing rate) for those joints which are frequently occluded. The highest difference in PCKh can be seen when computed over joints labeled as "occluded" only. Results on head and neck are omitted because they are never occluded.

One advantage of our method is that, by using 2D coordinates as input domain, it can be easily applied to different datasets it has not been trained on: Tables 3 and 4 show, respectively, the Missing Rate and the PCKh computed on COCO.

Table 1. Joints Missing Rate on the MPII dataset

Method (all joints)	Ankle	Knee	Hip	Wrist	Elbow	Shoulder	Neck	Head	Average MR
OpenPose [7]	0.072	0.040	0.021	0.066	0.037	0.019	0.011	0.019	0.039
Our method	**0.020**	**0.015**	**0.016**	**0.012**	**0.011**	**0.010**	**0.011**	**0.011**	**0.014**

Table 2. PCKh@0.5 on the MPII dataset, computed on all joints and only on joints labeled as occluded

Method (all joints)	Ankle	Knee	Hip	Wrist	Elbow	Shoulder	Neck	Head	Average PCKh
OpenPose [7]	79.87	87.17	93.0	79.15	89.03	95.97	97.71	96.11	88.73
Our method	**80.93**	**87.44**	**93.06**	**80.38**	**89.92**	**96.41**	**97.75**	**96.53**	**89.33**
Method (occluded joints)	Ankle	Knee	Hip	Wrist	Elbow	Shoulder			Average PCKh
OpenPose [7]	59.07	73.26	87.71	57.06	76.71	91.23			74.47
Our method	**61.18**	**73.83**	**87.80**	**60.78**	**78.70**	**92.32**			**75.77**

Finally, we report the computational time for training and testing. The analysis is performed on a laptop with 16 GB of RAM and an NVIDIA GeForce GTX 960M with 4 GB of RAM. Training requires only 3 h, while reconstruction of a single pose requires, on average, 0.88 ms. This shows that our method can be easily combined with any existing pose estimation architecture without significantly affecting the overall computational time.

Table 3. Joints Missing Rate on the COCO dataset

Method (all joints)	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Average MR
OpenPose [7]	0.0021	0.0104	0.0214	0.0371	0.0752	0.0539	0.0333
Our method	**0.0021**	**0.0032**	**0.0068**	**0.0150**	**0.0093**	**0.0052**	**0.0069**

Table 4. PCKh@0.5 trained on MPII and tested on COCO, computed on all joints

Method (all joints)	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Average PCKh
OpenPose [7]	80.22	54.91	63.71	39.48	35.74	23.60	49.61
Our method	**80.07**	**56.25**	**65.44**	**41.25**	**41.40**	**28.43**	**52.14**

4.5 Qualitative Analysis

In this section we show qualitative results of our predictions. In Fig. 5 (images taken from MPII), the top row shows (in blue), predictions obtained from OpenPose, while the bottom row shows the corresponding complete poses generated by our model (the predicted missing joints are in magenta). In column (a) and (b) ankles are missing from sitting poses and our model is able to predict a plausible locations of them. In column (c) a man is standing but both ankles are completely occluded by a foreground object and missing from the pose generated by OpenPose; however, our model is able to predict their position and produce a plausible complete standing pose. In column (c) the right arm (elbow and wrist) is missing; our model generates the missing joints in a spatial configuration which is similar to the joints of the visible left arm. The last column shows an extreme case where the number of missing joints is very high, thus providing little context for the final prediction: a man is standing, with raised arms and head occluded by a foreground object. Although our model generates arms which are completely lowered, the resulting pose is still a plausible human pose.

Figure 6 shows more examples of predictions obtained from OpenPose (top row) and the corresponding complete poses generated by our method (bottom row). In column (a), not just an ankle but the entire left leg (knee and ankle) is missing; the predicted complete pose closely resembles the sitting person pictured in the image. In column (b), the right wrist is not detected and both arms are raised, but our prediction is very close to the real wrist. Ankles (columns (b), (c) and (d)), are outside the camera field of view; however our model is able to predict a full pose even when RGB information is missing.

Finally, Fig. 7 shows predictions on frames from Salsa (another dataset our model was not trained on), where it can be seen that our method can generate

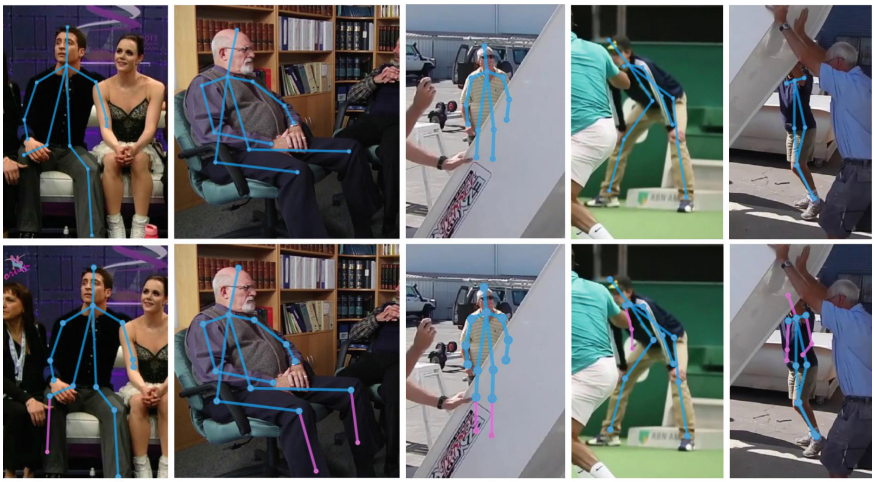

Fig. 5. Examples of predictions obtained from OpenPose (top row, in blue) and the corresponding complete poses generated by our method (bottom row, in magenta) on MPII. (Color figure online)

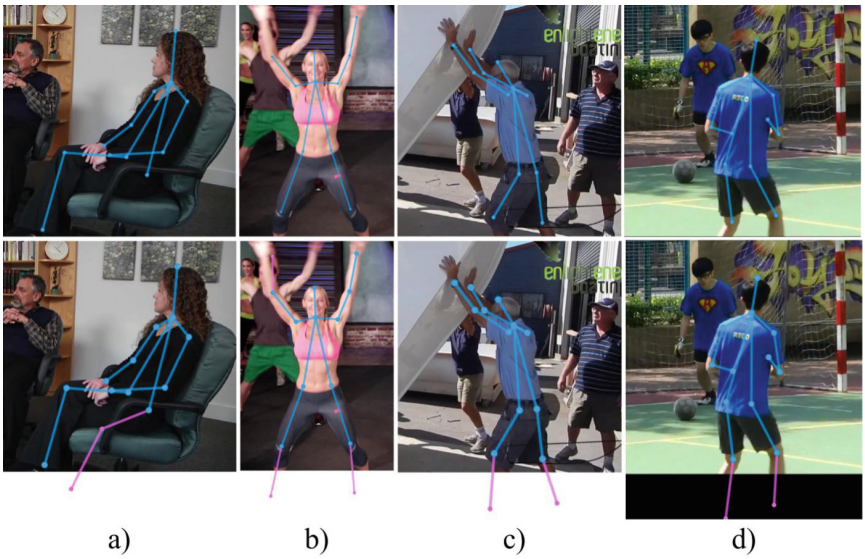

Fig. 6. More examples of predictions obtained from OpenPose (top row, in blue) and the missing joints predicted by our method (bottom row, magenta) on MPII. As can be seen, the model is also capable of predicting joints which are outside of the camera field-of-view. (Color figure online)

Fig. 7. Examples of predictions on Salsa. Top row: OpenPose results (in blue). Bottom row: missing joints predicted by our method (in magenta). (Color figure online)

plausible human poses even when half of the body is missing (see columns (c) and (d), with completely occluded legs and arms).

5 Conclusions and Future Work

In this paper, we presented a method for the prediction of missing joints from incomplete input poses. We approached the task as a denoising problem and showed that a simple model leads to a satisfactory boost in performance. We reported quantitative and qualitative results on several datasets and showed increased prediction performance over a well-known multi-person pose estimation algorithm and the ability to predict joints locations even when entire limbs are occluded.

Although state of the art pose estimators are able to cope with different challenging scenarios, occlusions are still a problem and can lead to missing joints predictions. The resulting incomplete poses might negatively affect the performance of applications based on 2D poses, such as behavior analysis based on body pose and gestures or bottom-up 3D pose estimation algorithms which lift 2D coordinates to 3D. Our method can, then, be an aid in these contexts, providing "complete" information. Future work will explore in detail the effects of our method in the context of human behavior analysis.

References

1. Abadi, M., et al.: Tensorflow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016), pp. 265–283 (2016). https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

2. Akaike, H.: Information theory and an extension of the maximum likelihood principle. In: Parzen, E., Tanabe, K., Kitagawa, G. (eds.) Selected Papers of Hirotugu Akaike. Springer, New York (1998). https://doi.org/10.1007/978-1-4612-1694-0_15
3. Alameda-Pineda, X., et al.: SALSA: a novel dataset for multimodal group behavior analysis. PAMI **38**, 1707–1720 (2016)
4. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: new benchmark and state of the art analysis. In: CVPR (2014)
5. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: NIPS (2007)
6. Bourdev, L., Malik, J.: Poselets: Body part detectors trained using 3D human pose annotations. In: ICCV (2009)
7. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: CVPR (2017)
8. Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded pyramid network for multi-person pose estimation. arXiv preprint arXiv:1711.07319 (2017)
9. Chen, Y., Shen, C., Wei, X.S., Liu, L., Yang, J.: Adversarial PoseNet: a structure-aware convolutional network for human pose estimation. In: CVPR (2017)
10. Fang, H.S., Xie, S., Tai, Y.W., Lu, C.: RMPE: regional multi-person pose estimation. In: CVPR (2017)
11. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: CVPR (2008)
12. Feng, W., Kannan, A., Gkioxari, G., Zitnick, C.L.: Learn2Smile: learning non-verbal interaction through observation. In: IROS (2017)
13. Hou, X., Shen, L., Sun, K., Qiu, G.: Deep feature consistent variational autoencoder. In: WACV (2017)
14. Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., Schiele, B.: DeeperCut: a deeper, stronger, and faster multi-person pose estimation model. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 34–50. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_3
15. Iqbal, U., Gall, J.: Multi-person pose estimation with local joint-to-person associations. In: ECCV Workshop (2016). http://arxiv.org/abs/1608.08526
16. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. In: NIPS (2015)
17. Jain, V., Seung, S.: Natural image denoising with convolutional networks. In: Advances in Neural Information Processing Systems, pp. 769–776 (2009)
18. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
19. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
20. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
21. Newell, A., Huang, Z., Deng, J.: Associative embedding: end-to-end learning for joint detection and grouping. In: NIPS (2017)
22. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 483–499. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_29
23. Pishchulin, L., Andriluka, M., Gehler, P., Schiele, B.: Strong appearance and expressive spatial models for human pose estimation. In: ICCV (2013)

24. Pishchulin, L., et al.: DeepCut: Joint subset partition and labeling for multi person pose estimation. In: CVPR (2016)
25. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: NIPS (2015)
26. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533 (1986)
27. Tome, D., Russell, C., Agapito, L.: Lifting from the deep: convolutional 3D pose estimation from a single image. In: CVPR (2017)
28. Tompson, J.J., Jain, A., LeCun, Y., Bregler, C.: Joint training of a convolutional network and a graphical model for human pose estimation. In: NIPS (2014)
29. Toshev, A., Szegedy, C.: DeepPose: human pose estimation via deep neural networks. In: CVPR (2014)
30. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: ICML (2008)
31. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**(Dec), 3371–3408 (2010)
32. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: CVPR (2016)
33. Xie, J., Xu, L., Chen, E.: Image denoising and inpainting with deep neural networks. In: Advances in Neural Information Processing Systems, pp. 341–349 (2012)
34. Xu, L., Ren, J.S., Liu, C., Jia, J.: Deep convolutional neural network for image deconvolution. In: Advances in Neural Information Processing Systems, pp. 1790–1798 (2014)