Primate Tracking Using Convolutional Pose Machines with U-Net Architecture for OpenMonkeyChallenge

Matthew Choi
University of Minnesota
choix709@umn.edu

Reese Kneeland
University of Minnesota
kneel027@umn.edu

Rijul Mahajan
University of Minnesota
mahaj068@umn.edu

Ninh Tran
University of Minnesota
tran1108@umn.edu

Abstract

The OpenMonkeyChallenge [1] (OMC) is a benchmark challenge proposed to apply pose tracking algorithms to Non-Human Primates (NHP) using a new dataset containing over 100,000 annotated images of primates in natural habitats. The annotations contain 17 body landmarks on data covering a wide variety of 26 species of monkeys. Our project outlines an adapted model for this challenge featuring Enhanced CPM with U-Net Architecture, and compares that model against two established baselines, the Simple Baselines for Human Pose Estimation [2], and the OpenPose model using a Convolutional Pose Machine [3]. Our Enhanced CPM model, despite being significantly less efficient than the CPM baseline in terms of training time, was able to extract higher resolution features from our data, and outperformed the baseline by a significant margin.

1. Introduction

The OMC serves to expand a growing sector of computer vision research focused on tracking body landmarks in animals [4]. This type of research provides many useful tools to related fields in ecology, conservation, biomedicine, neuroscience, and psychology. Despite the usefulness of this type of research on primates, NHPs remain an outlier among species regarding this task, as their homogeneous body texture and wide array of body positions pose a challenge for the available pose classification models [5]. Previous research in this field has either been limited to smaller and less expansive datasets, or the datasets used have been in less useful clinical settings, rather than in more natural backgrounds. With this new larger dataset, we aim to beat the previous baseline of classification tools on this new data. Models solving the problem are evaluated on three key metrics: mean per joint position error (MPJPE) [6], probability of correct keypoint (PCK) [7], and average precision (AP) based on object keypoint similarity (OKS) [8]. The entries are then ranked on a leaderboard

2. Related Work

The goal of the OMC is to achieve performance benchmarks on NHPs that are comparable to human pose recognition. When evaluating that goal we first acknowledge that the most successful state-of-the-art pose recognition models that exist today focus on humans, with some advanced models training 2D and 3D pose models simultaneously [9]. Some of the most promising existing methods include convolutional pose machines [3], and CNNs combined with an expressive deformable mixture of parts [10]. There are also top-down models including Simple Baselines for Human Pose Estimation [2], DeepLabCut with ResNet [11], and HRNet-W32 [12]. Other models use adaptive point calculation to reduce computation times [13], and some more specialized models implement video tracking combined with inertial sensors to create a real-time model of active movements [14], or generate full mesh representations of pose movements [15]. When transitioning from the many available human pose recognition models to NHP pose recognition, we focus on the ones extracting simple two-dimensional pose landmarks, among which current research finds Deep Neural Networks are among some of the most effective techniques [16]. Models built from ImageNet training data, which includes more animals and variety in pose structures, tend to perform better on nonhuman subjects [17]. We also observe that some of the most successful advancements in human pose recognition have come from self-supervised learning methods [18], and that a key piece of closing the performance gap between human and NHP pose recognition might be in improving the self-supervised

learning algorithms for this type of problem [19]. Some algorithms such as DeepLabCut, have already been explored for non-human pose estimation in other species of animals and insects [20], but an extensive study of this algorithm has not been conducted on NHPs.

3. Baselines

3.1. Simple Baselines for Human pose Estimation and Tracking

In this section we will explore the Simple Baseline for Human Pose Estimation and Tracking, as published by Bin Xiao et. al. through Microsoft Research in 2018. [2]. We will be looking specifically at the ResNet101 implementation of the model.

High-resolution representation learning plays an essential role in many vision problems, e.g., pose estimation and semantic segmentation. In this paper, the authors attempt to build upon this concept by introducing a simple yet effective modification to the existing process. Unlike previous implementations where only the representation from the high-resolution convolution was augmented, the authors of this paper take it a step further by augmenting the high-resolution representation by aggregating the upsampled representations from all of the parallel convolutions instead. It is trivial to see why this was chosen as the "simple" implementation, as one can safely say that this model provides the best ease of generating heatmaps over both deep and low resolution features.

We implemented a modified version of the Simple Baseline ResNet 101 architecture designed to train a network on the OpenMonkey data. Making the necessary changes to this model proved challenging, as at the beginning of this project we were relatively inexperienced with deep learning frameworks. The entire data loading/processing pipeline had to be reworked, the structure of the network altered to accept the new configuration and number of joints that were being tracked, and the accuracy calculations added to compare to the rest of the tested models. After implementing these modifications, it was successfully trained on the data, producing the accuracy metrics outlined in Table 1 below.

MPJPE	PCK@0.2	PCK@0.5	mAP
0.093	0.851	0.965	0.654

Table 1: Simple Baseline ResNet101 Performance Evaluation

In Figure 1, we can see an example image of predicted points against the ground truth labels, we can see in this example that the points for the tail, hind legs, and eyes struggle the most. This makes sense in regards to the model as these are the points that differ the most from pose tracking on humans. The overall uniform color distribution of the monkeys combines with the more complicated pose structures seem to pose a significant challenge for the Simple Baseline model.

Figure 1: A selection of joint heat map predictions compared to the ground truth values [21]

In Figure 2, it's once again clear that the prediction locations for certain joints are completely incorrect, but also that the generated heatmap prediction area is often not very precise, with the range of prediction areas spread widely over the monkey's body. Our proposed model tackles this problem head on by extracting more precise higher dimensional heatmaps from the image, which will be discussed in a later section.

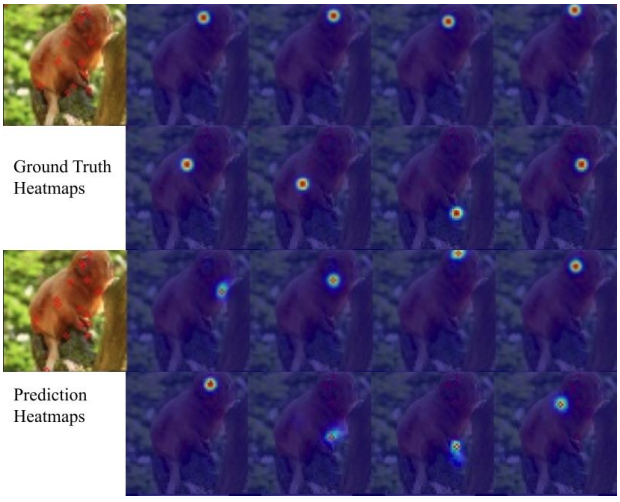


Figure 2: A selection of joint heat map predictions compared to the ground truth values [21]

3.2. Convolutional Pose Machines

A state-of-the-art approach in the trend of using deep learning to improve pose estimation, convolutional pose

machines (CPMs) uses a pose machine framework [22] that utilizes convolutional neural networks to learn spatial models of the relationships between parts [10]. Convolutional pose machines have competitive results on standard datasets including the MPII, LSP, and FLIC datasets.

Classical approaches in pose estimation have traditionally involved the pictorial structures model which expresses the spatial correlations between body parts with a graphical tree model [23]. The authors of CPMs stray away from this approach, stating that this model is prone to occlusion errors, limited in its ability to capture poses with its tree structure, and less flexible. Instead, CPMs builds upon the pose machines model and uses belief maps to develop an expressive non-parametric method that gives probabilities of the locations of the parts of the body at each coordinate of the image with a sequential prediction framework that refines the estimates with each stage [22]. The key difference between CPMs and pose machines is convolutional pose machines recognize the power of convolutional neural networks to learn implicit spatial models of each part of the pose and employs these models to refine the belief maps at each stage of the sequential prediction framework. This results in a powerful model that is capable of encoding complex spatial relationships without priors or a parametric form.

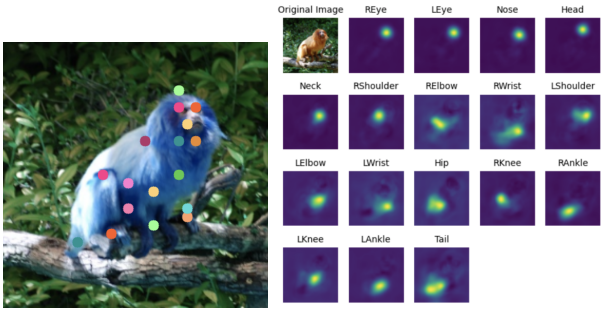
One limitation of CPMs is the vanishing gradient problem. Because CPM is a composition of multiple convolutional neural networks, we deal with a model containing many layers which leads to diminishing strength in weights in the backpropagation. For our paper specifically, we faced issues with our parameters not being optimized enough, and coupled with the fact that we had some limitations on the hardware we were using to train our model. This also led to us not being able to match the author generated model. In the paper, they had an accuracy of about 0.074 MPJPE or 0.761 PCK@0.2, but we were only able to achieve about 0.105 MPJPE or 0.872 PCK@0.2. Figure 3a and 3b show our qualitative result from the CPM implementation.

MPJPE	PCK@0.2	PCK@0.5	mAP
0.105	0.872	0.978	0.590

Table 2: CPM Baseline Performance Evaluation

4. Initial Approach

Our first idea for a proposed method in primate pose estimation addresses the vanishing gradients problem in CPM [10] during training by providing a natural learning objective function that enforces intermediate supervision. Moreover, CPM is a human pose estimation model and we can see that it doesn't perform very well when annotating primates because of it low generalizability. We planned to



(a) Keypoints (b) Feature Heatmaps

Figure 3: CPM Baseline Generated Keypoints and Heatmaps

approach this problem using CPM as the backbone network and modify the training procedure to include self-supervised learning inspired from [21]. The algorithm for the training process was planned to be as follows:

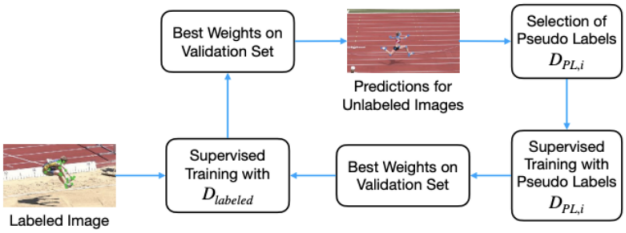


Figure 4: Training procedure for pseudo label training [21]

1. Execute a fully supervised training on a small, in-domain labeled dataset $D_{labeled}$.
2. Use early stopping to select the weights of the epoch with the best score on the validation set.
3. Next, generate pseudo labels which are created from the network itself.
4. Based on the selected weights, pseudo labels are generated for all unlabeled images of the training set, resulting in the pseudo label dataset $D_{PL,1}$ for the first iteration. [21]
5. Afterwards, start the first self-supervised training iteration by training a new neural network on the generated pseudo label dataset $D_{PL,1}$, starting from pretrained weights from the OpenMonkeyChallenge dataset.
6. Again, the best weights according to the validation score are selected.
7. In the next step, fine-tuning based on the selected weights with $D_{labeled}$. The best weights according to

the validation results are determined and used to generate updated pseudo labels $D_{PL,2}$.

8. Then, the next self-supervised training iterations are executed similar to the first one as indicated in Figure 1.

After performing this routine, the network can be further improved by introducing pseudo labels mentioned in Table 1 by selecting the best pseudo labels[21] using the predictions generated with the raw images and add prediction results of a horizontally flipped image and results from randomly chosen augmentations to the prediction set. Predictions with low confidence score are discarded. The mean squared error (MSE) in pixels between the base prediction and augmented predictions is calculated. Now, we select the predictions with the lowest MSE for the pseudo label dataset per keypoint, resulting in an equal number of predictions for each keypoint. As pseudo labels, the base predictions are used instead of the mean over all predictions, as single outliers could shift the mean enormously and the predictions on augmented images are less accurate as they are harder for the model.

The implementation of Enhanced CPM with Self-Supervised Learning was expected to cut down training time significantly and make already existing CPM model more efficient and generalized, however, while implementing the self-supervised learning model, our team observed that the model is too complicated and does not produce the expected result from [21] when trained on the OMC data set. Furthermore, Self-Supervised learning model did not fix the generalizability issue of most proposed models [1]. For the sake of time to produced a plausible results, we switch our implementation from a self-supervised learning model to a Enhanced CPM with U-Net Architecture.

5. Proposed Approach

In the CPM model, there is a module in the architecture that is composed of three successive pairs of a convolutional layer and a pooling layer to learn the convolutional feature map used to predict the belief maps at each stage. Seeing that this feature map is downsampled to a shape eight times smaller than its original dimensions, we believe a lot of information from the high-resolution representations is lost. Taking inspiration from HR-Net [24], we wanted to use a design that would incorporate both high-resolution and low-resolution features in learning important convolutional features for predicting the belief maps.

We utilize U-Net [23] to build upon the CPM model. We modify and extend this architecture such that it works with the OpenMonkeyChallenge training features to obtain more precise predictions. The main idea is to replace the part of the CPM model that learns the convolutional feature map with the U-Net architecture which preserves the original shape of the input.

Therefore, these layers improve the output resolution. High resolution characteristics from the contracting route are merged with the upsampled output to localize. Based on this knowledge, a subsequent convolution layer can learn to create a more exact result. One significant change in our design is that we now have a large number of feature channels in the upsampling section, allowing the network to pass context information to higher resolution layers.

As a result, the expanding path is roughly symmetrical to the contracting path, resulting in a u-shaped structure. The network employs just the valid component of each convolution, i.e., the segmentation map only comprises the pixels for which the whole context is accessible in the input picture. By using an overlap-tile method, this strategy enables for the smooth segmentation of arbitrarily huge pictures (see Figure 2). The missing context is extrapolated by mirroring the input picture to forecast the pixels in the image's border region. This tiling method is necessary for applying the network to huge pictures, as the resolution would otherwise be restricted by GPU memory.

Separating contacting items of the same class is another issue in many cell segmentation tasks. To achieve this, we propose using a weighted loss function, in which the separating background labels between contacting cells are given a high weight in the loss function.

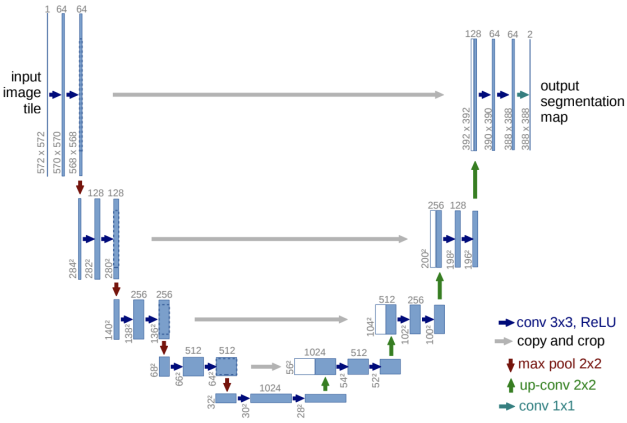


Figure 5: U-net Architecture [23]

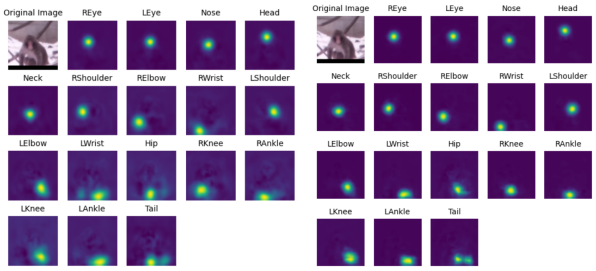
The network design, as shown in Figure 4, consists of a contracting path (left side) and an expansive way (right side) (right side). The convolutional network's contracting route follows the standard architecture. Following [23], our model consists of two 3x3 convolutions (unpadded convolutions) performed twice, each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling.

The number of feature channels doubles with each down-

sampling step. An upsampling of the feature map is followed by a 2x2 convolution ("up-convolution") that halves the number of feature channels, a concatenation with the proportionally cropped feature map from the contracting route, and two 3x3 convolutions, each followed by a ReLU. The loss of boundary pixels in each convolution necessitates cropping. Each 64-component feature vector is mapped to the required number of classes using a 1x1 convolution at the final layer.

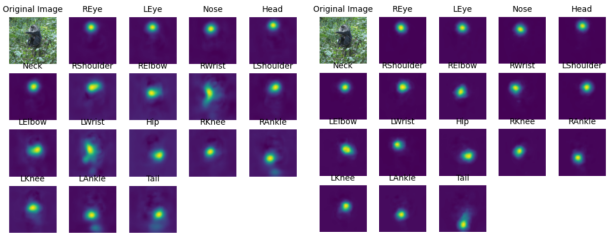
6. Results

6.1. Heatmaps Generated



(a) CPM (b) CPM U-Net

Figure 6: Comparison of Generated Heatmaps Between CPM and CPM + U-Net



(a) CPM (b) CPM + U-Net

Figure 7: Another Comparison of Generated Heatmaps Between CPM and CPM + U-Net

We see that there is a large difference in the level of precision present in the heatmaps that are generated, specifically apparent in the sections like Right Ankle, Tail and Left Wrist in Figure 6, we can see that the CPM + U-Net model has an error that is more localised than the CPM based model. From this result we can conclude that our higher resolution belief maps resulted in a more precise and localized prediction for the pose estimation.

The result appears to corroborate our hypothesis that the predictions are made more accurate due to the larger size of the belief map generated by the CPM + UNet model, being much larger than that of CPM, which also contributes to the computational complexity and runtime.

6.2. Data Accuracy

From the data below, we see that the CPM + U-Net model beats our baselines in every category. Most notably, we see that the mean per joint position error (MPJPE) and the mean average precision (mAP) metrics are quite larger than the baselines. We believe this can be attributed to the shape preserving nature of the CPM + U-Net model as well as being able to incorporate high-resolution and low-resolution representations to predict the belief maps.

Method	MPJPE	PCK 0.2	PCK 0.5	mAP
ResNet101	0.093	0.851	0.965	0.654
CPM	0.105	0.872	0.978	0.590
CPM + U-Net	0.071	0.939	0.991	0.786

Table 3: Performance Evaluation Comparison Between Baselines (ResNet101, CPM) vs Proposed Enhanced CPM with U-net

7. Future Work

Looking forward we want to optimize the Enhanced CPM with U-net architecture to train faster with a more reasonable time on OMC dataset to annotate primate's poses and surpasses performance of other proposed methods based on the criteria mentioned in the OMC. Additional tests on generalizability will be performed and compared with the two established baselines: [2] and [3]. We can try to tune the hyper-parameters better in order to ensure higher accuracy. We also aim to develop better model to integrate high and low resolution features, but without the computational expense. It would be possible for us to improve the network performance with pseudo labels that were earlier mentioned in Table ones by better selecting the ones that we choose to employ in our network.

8. Conclusion

By storing and training on high resolution heatmap our model captures a more precise representation of the extracted image features used to locate the points of the estimated post. Because of this higher resolution representation, it also takes significantly longer to run than our CPM baseline [3], by utilizing U-net, we have to store and train on high resolution heatmaps to get the benefits of the higher precision. The final output generated by our model was trained on an RTX 3080 for 6 days to reach a reasonable convergence, compared to 8-12 hours for our baselines. The extracted features, while more detailed, don't contain fundamentally different information than the baseline, as they

capture a lower definition version of the same belief maps used to generate the predictions. With our model, we do not downsample using the U-Net architecture, so it is 30x times more computationally expensive to train the model, because of this, our model couldn't be trained as thoroughly, but still over-performs our recorded baseline statistics. Overall our approach to the problem of pose tracking on NHP resulted in a more accurate and precise model for calculating pose estimations, but at the cost of a longer training time and wider data pipelines. In the future we would like to take a look at optimizing the upsampling methods to make them more efficient, to allow this architecture to close the gap in training times compared to traditional CPM. We hope the advances made in our testing can help to further research efforts in the field.

9. Group Responsibility

Reese Kneeland: Implementing Simple Baseline, implementing data pipeline and heatmap generation for CPM U-Net model, wrote abstract, introduction, related work, and Simple Baseline sections.

Matthew Choi: Implemented the CPM Baseline model, proposed the CPM U-Net variant architecture, wrote the CPM U-Net model code, wrote the training and prediction modules of the model, contributed to the convolutional pose machines, proposed approach, and heatmaps sections.

Ninh Tran: Attempt to implement Self-Supervised Learning on top of CPM, wrote part of the proposed approach section, review and made edits on limitation, future work, Convolutional Pose Machines sections.

Rijul Mahajan: Background research on baselines and attempted implementation, developed paper and presentation slides structure, contributions to proposed approach, results, future work, related work sections.

10. Code Links and CodaLab Submissions

The finished code for our proposed model can be found at: https://github.com/MattyChoi/PoseEstimation

The CodaLab submissions for our Baseline models were submitted under the name choix709, while the results for our final model were submitted under mahaj068.

References

[1] Y. Yao, A. Mohan, E. Bliss-Moreau, K. Coleman, S. M. Freeman, C. J. Machado, J. Raper, J. Zimmermann, B. Y. Hayden, and H. S. Park, "Openmonkeychallenge: Dataset and benchmark challenges for pose tracking of non-human primates," 09 2021. 1, 4

[2] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," 2018. 1, 2, 5

[3] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," 2016. 1, 5

[4] T. W. Dunn, J. D. Marshall, K. S. Severson, D. E. Aldarondo, D. G. C. Hildebrand, S. N. Chettih, W. L. Wang, A. J. Gellis, D. E. Carlson, D. Aronov, W. A. Freiwald, F. Wang, and B. P. Ölveczky, "Geometric deep learning enables 3d kinematic profiling across species and environments," *Nature Methods*, vol. 18, p. 564–573, 05 2021. 1

[5] P. C. Bala, B. R. Eisenreich, S. B. M. Yoo, B. Y. Hayden, H. S. Park, and J. Zimmermann, "Automated markerless pose estimation in freely moving macaques with openmonkeystudio," *Nature Communications*, vol. 11, 09 2020. 1

[6] K. Iskakov, E. Burkov, V. Lempitsky, and Y. Malkov, "Learnable triangulation of human pose," *arXiv:1905.05754 [cs]*, 05 2019. 1

[7] Z. Cao, G. Hidalgo Martinez, T. Simon, S.-E. Wei, and Y. A. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2019. 1

[8] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, Z. C. Lawrence, and P. Dollár, "Microsoft coco: Common objects in context," 2014. 1

[9] D. Luvizon, D. Picard, and H. Tabia, "2d/3d pose estimation and action recognition using multitask deep learning," 03 2018. 1

[10] W. Yang, W. Ouyang, H. Li, and X. Wang, "End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 06 2016. 1, 3

[11] A. Mathis, P. Mamidanna, K. M. Cury, T. Abe, V. N. Murthy, M. W. Mathis, and M. Bethge, "Deeplabcut: markerless pose estimation of user-defined body parts with deep learning," *Nature Neuroscience*, vol. 21, pp. 1281–1289, 08 2018. 1

[12] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," 2019. 1

[13] Y. Xiao, X. Wang, D. Yu, G. Wang, Q. Zhang, and M. He, "Adaptivepose: Human parts as adaptive points," *arXiv:2112.13635 [cs]*, 12 2021. 1

[14] T. v. Marcard, G. Pons-Moll, and B. Rosenhahn, "Human pose estimation from video and imus," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, p. 1533–1547, 08 2016. 1

[15] D. Xiang, H. Joo, and Y. Sheikh, "Monocular total capture: Posing face, body, and hands in the wild," *arXiv:1812.01598 [cs]*, 12 2018. 1

[16] M. W. Mathis and A. Mathis, "Deep learning tools for the measurement of animal behavior in neuroscience," *Current Opinion in Neurobiology*, vol. 60, pp. 1–11, 02 2020. 1

[17] A. Mathis, T. Biasi, S. Schneider, M. Yüksekgönül, B. Rogers, M. Bethge, and M. W. Mathis, "Pretraining boosts out-of-domain robustness for pose estimation," *arxiv.org*, 09 2019. 1

[18] C. Wan, T. Probst, L. V. Gool, and A. Yao, "Self-supervised 3d hand pose estimation through training by fitting," 2019. 1

[19] T. L. Munea, Y. Z. Jembre, H. T. Weldegebriel, L. Chen, C. Huang, and C. Yang, "The progress of human pose estimation: A survey and taxonomy of models applied in 2d human pose estimation," *IEEE Access*, vol. 8, pp. 133330–133348, 2020. 2

[20] T. Nath, A. Mathis, A. C. Chen, A. Patel, M. Bethge, and M. W. Mathis, "Using deeplabcut for 3d markerless pose estimation across species and behaviors," *Nature Protocols*, vol. 14, pp. 2152–2176, 06 2019. 2

[21] K. Ludwig, S. Scherer, M. Einfalt, and R. Lienhart, "Self-supervised learning for human pose estimation in sports," in *2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 1–6, IEEE, 2021. 2, 3, 4

[22] V. Ramakrishna, D. Munoz, M. Hebert, J. Andrew Bagnell, and Y. Sheikh, "Pose machines: Articulated pose estimation via inference machines," in *European Conference on Computer Vision*, pp. 33–47, Springer, 2014. 3

[23] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *International journal of computer vision*, vol. 61, no. 1, pp. 55–79, 2005. 3, 4

[24] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5693–5703, 2019. 4