# Efficient and Accurate Human Pose Estimation for Video Streams

Fubao Wu

## Abstract

*Human pose estimation has been an active research area in computer vision with wide applications in action recognition, event detection, and human-computer interaction, and so on. With the advancement of deep learning, more and more deep neural networks based pose estimated have been proposed for images and videos. However, the resource computation demand for the deep neural network-based pose estimation has been challenging for real life deployment to detect human poses especially in video streams. Given the input accuracy requirement, we propose a pose estimation framework for video streams to reduce the computation and maintain the accuracy requirement. We leverage the video temporal characteristics to dynamically select a suitable model to detect human poses interchangeably. Two models, which are a full deep neural network model with high accuracy but high computation resources, and a compressed model with low computation resources but a little accuracy loss, are utilized in the framework. We experimentally test on two image datasets and one video dataset, and shows the effectiveness and efficiency of our method, which greatly reduces the computation resources by 48% with just 2% accuracy loss in the video dataset.*

## 1. Introduction

Human pose estimation [2, 23] is to localize a human joint keypoints (e.g., elbow, wrist, etc.) from images or videos, which is a challenging problem. It has broad applications in human action recognition, event detection, human-computer interaction, animation, etc.

As in the Figure 1 shows, the keypoints detected consist of 18 points that are ankles, knees, hips, shoulders, elbows, wrists, necks, torsos, and head tops in this project.

With the advancement of deep learning, there are a lot of improvements on the accuracy of detecting the complex human pose. Recently methods build a neural network on each one whole image/ one frame in the video. The features are extracted from the whole images [23, 25, 5, 9]. For lots of applications and content setting like a person walking , running or dancing, it is too time-consuming or trivial to do human pose with a very complex deep neural network model

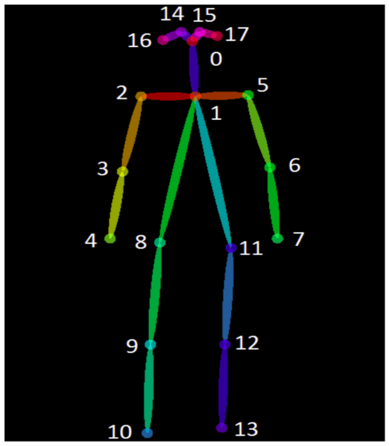


Figure 1: Human pose keypoints used for detection

when there are lots of redundant or repeated information in the video. Existing methods of pose estimation are categorized into two types. The first type is top-down method, which detects human body first with a neural network model and then detects the pose within the human body with another neural network model [26, 5]. The second type is to detect human pose from bottom-up, which directly designs a neural network to detect the pose from scratch [2, 23]. Both methods, though presenting a high accuracy, are time-consuming to train and deploy in real-life applications. It is especially not suitable for the real time application setting.

We propose a new approach to detect human pose considering the temporal characteristics of videos [10] to detect human pose in an efficient and also accurate way. Most time of a video has temporal characteristic that redundant frames exist, or the objects keep relatively no or tidy movements during a period of time. We consider the temporal characteristics and schedule between the expensive full-version neural network model (abbreviated as full model) and its compressed neural network model [12] (which is a simplified model of [12]the original full model, abbreviated as compressed model) to achieve pose estimation. First, we divide a video into multiple periods. In each period, in the beginning segment of the period, we use a controller to decide to select a full model or compressed model. Then the selected model will be applied to the rest time of this period. Then for the next period, we use the same strategy again, which could ensure the accuracy above the certain threshold and also detect human poses as fast as possible.

We use two pose estimation image datasets to train our model and test on another video dataset to verify our method and the system.

The contribution of this work are as follow:

- We propose a dynamic framework for pose estimation efficiently and effectively for video streams.
- We design a compressed neural network model to train and validate for pose estimation.
- We develop a pose estimation system for video streams and experimentally verify the effectiveness and efficiency compared to the baseline models.

## 2. Related Work

In computer vision area, classical computer vision methods for pose estimation focus on modeling the human structure and inter-connectivity among body parts and rely on hand-crafted features [21, 6, 11]. Currently the most effective way to detect human poses is through deep neural network methods [23, 25, 5, 9, 7, 24, 19, 3, 16, 18, 13]. Recent advancement on pose estimation based on deep neural networks are categorized as top-down and bottom-up techniques. Top-down technique [23, 25, 5, 9, 7, 24, 19] is to detect the human body in the images or frames first and then identify its keypoints. Bottom-up technique [3, 16, 18, 13] directly detect the keypoints of a person/multiple persons from an image or frames in videos. Openpose [2] is a bottom-up method that proposes several stages of convolutional neural networks to greedily parse multiple persons' parts and their associations. It is a classical deep neural network model to detect human poses with high accuracy for multiple persons but it is also not efficient to apply it for real life deployment, especially in real-time. Our method is based on the Openpose in which we use it as the full model and propose a compressed model based on it.

There are also some pose estimation methods focused on video detection with temporal and appearance information across frames [17, 22, 8, 14]. [22] considers optical flow and deep convolutional networks as input motion features. Other methods [8, 14] consider using the recurrent neural networks (RNN) for the sequence-to-sequence model for structured output prediction with the temporal characteristics, although the RNN method has a notable accuracy, but it suffers from the expensive computations. Our method explores the temporal characteristics of video frames and considers the redundant and repeated frame information to dynamically change the suitable models to satisfy the accuracy required and also greatly reduce the resource computation for efficient computation.

## 3. Methodology

In a human video content setting, sometimes the persons move fast and behave complicated, sometimes the persons move slowly or with easy pose. Consider the complex model with high accuracy but also high computation resources, it is not efficient to apply this complex model for some video applications. We could use a compressed model for some simple video periods which use simple compressed model with less computing resources but also an acceptable accuracy requirement, and change back to complex full model whenever there is a complex video content settings.

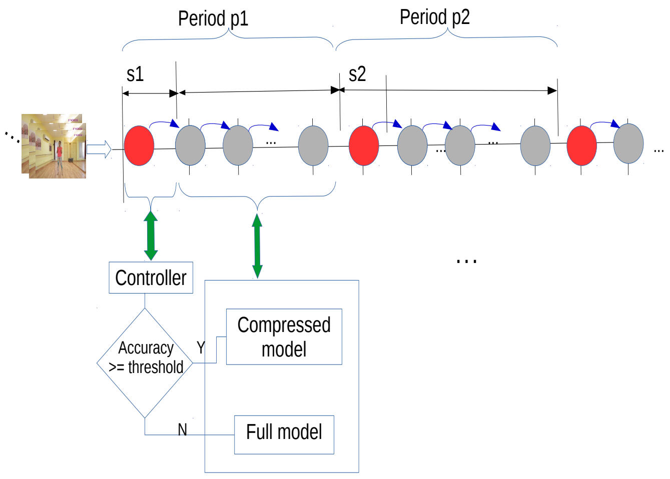


Figure 2: Our proposed pose estimation framework for videos

### 3.1. Proposed Human Pose Estimation Framework

Based on that temporal characteristics, we propose a dynamic architecture to detect human pose interchangeably with two different models.

The framework is shown in Figure 2. Assume the application requirement for the accuracy threshold is $A_{min}$, and the video can be divided into several periods, such as period $p1$ and period $p2$, and so on shown in the figure, in each period, we use a controller to decide to select the full model or compressed model for pose estimation. In each period such as $p1$, we use a small time segment $s1$ in the beginning of the period as the base for the controller. The controller uses the following way to decide which models to use for the rest part of the period. If there is a ground truth, we use the compressed model to detect human poses in the time segment. If the compressed model generates an average accuracy above the accuracy threshold $A_{min}$, then we continue to use this compressed model for the rest of segment. If the compressed model's average accuracy is below the threshold $A_{min}$, we will use a full model to detect the human pose. However, if there is no ground truth for the video, we use the full model as the ground truth, then we detect the human pose in the time segment $s1$ of the period both with full model and compressed model. If the compressed model's average accuracy in $s1$ is better than the threshold $A_{min}$, we use the compressed model to detect the human pose for the rest of the period $s1$, otherwise, we use the full model. If the full model's average accuracy still could not achieve the accuracy threshold in the segment time, we use full model for the rest of the period. Therefore, we could ensure the accuracy and also improve the efficiency. For period $p2$ and other periods in the video, the same strategy is applied.

## 3.2. Full Model

Our full model is based on the classic human pose estimation model Openpose [2]. We brief introduce the model here. It takes an image as input, uses the multi-stage convolutional neural networks to detect body parts, then associate them for multiple persons' pose estimation, and finally out all the persons' poses. As shown in Figure 3, the model takes as input an image, then through several stages of covolutional neural network (CNN) layers and output the 2D locations of joint keypoints for each person in the image. Each stage's output heatmap is connected with the original image features together into the next stage's convolutional neural network output, and so on until the output.

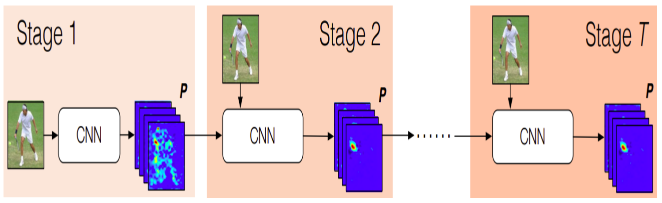


Figure 3: Full model for pose estimation

Each stage of CNN layers simultaneously predict detection confidence maps and affinity fields that encode part-to-part association for multiple persons in the image. As shown in the Figure 4 in the stage $t$, receiving from the input $F$ from the last stage, $t-1$ the stage is split into two branches, the first branch of the stage is to predict confidence maps $S^t$, the second branch of the stage is to predict affinity fields $L^t$. Each branch consists of 5 convolutional blocks and 2 convolution layers with 1x1 filter size. Each convolutional block is composed of 3 convolutional layers with 3x3 filter size. Each stage has the same structure as stage $t$. The total loss of all the stages is the combination of the loss of the confidence map and affinity field at each stage:

$$f = \sum_{t=1}^{T} (f_s^t + f_L^t) \tag{1}$$

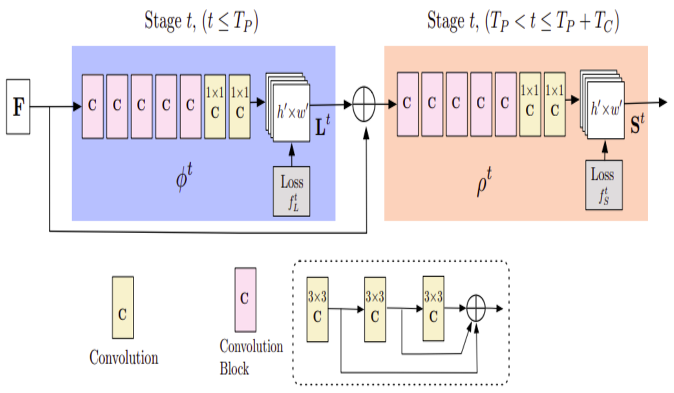


Figure 4: Each-stage convolutional neural network with two branches

Where

$$f_s^t = \sum_{j=1}^{J} \sum_{p} W(p) \cdot \left\| S_j^t(p) - S_j^*(p) \right\|_2^2 \tag{2}$$

$$f_L^t = \sum_{c=1}^{C} \sum_{p} W(p) \cdot \left\| L_c^t(p) - L_c^*(p) \right\|_2^2 \tag{3}$$

$L_2$ Least Square Error is used between the estimated predictions and the ground truth maps and fields. The stage number is a user input parameter, in this project, the full model uses 5 stages.

## 3.3. Compressed Model

Our compressed model is inspired from this to compress the CNN model with fewer model layers and unit size to get a new neural network model, which has much higher efficiency but a little accuracy sacrificed.

We compress the stage, the convolutonal layer and filter size to reduce the full model to a compressed model. To reduce the computation resources and do not sacrifice the accuracy much, we propose a compressed model with 2 stages of the original full model. In each stage, we use 3 convolution blocks and 1 convolutional layer with 1x1 filter size to learn the confidence map and affinity fields. The convolution block is also reduced to 2 convolutions with 3x3 filter size. The $L2$ loss is used as the same as the full model.

## 3.4. Selection between Full Model and Compressed model

There is a trade-off between full model and compressed model. Generally speaking, the full model has high accuracy but high computation resources, the compressed has lower computation but lower accuracy. The selection of segment and period are important parameters that needs to be decided. It is challenging to automatically decide the values by the system. In this project, we just simply set this as input parameter to be decided by users and recommend the best values for value. These parameters could impact the final performance of accuracy and efficiency for human pose estimation. We will show the impact of different parameters on the performance and get the best recommended values for different accuracy threshold requirement.

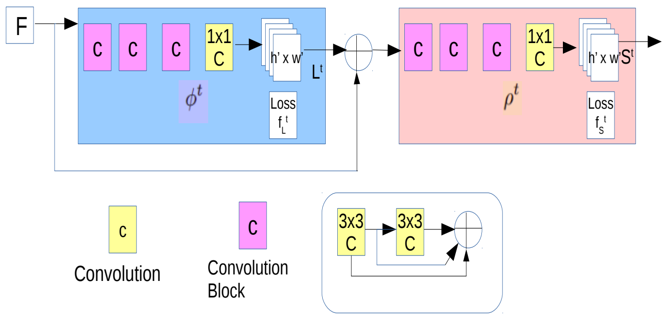


Figure 5: Our proposed compressed model

The loss function

## 4. Experimental Evaluation

### 4.1. Dataset

Table 1: Dataset statistics information

| Dataset | Total Instances | Training+ validation | Test |
|---|---|---|---|
| MPII Pose (images) | 25k | 20k | – |
| COCO Pose (images) | 200k | 150k | – |
| Youtube Pose (videos) | 30 | – | 30 |

We use image datasets from MPII pose dataset [1] and COCO pose dataset [15] with humans. We trained our specialized neural network model on these dataset to make it satisfied a certain accuracy. The pose in the image and frames dataset are already labeled. Then we test our model on the video VGG video dataset [4] to show the testing result and compare with baseline models. The code link is in the google drive: Code.

The dataset statistics is shown in Table 1.

**MPII dataset:** MPII dataset is the state of the art benchmark for evaluation of articulated human pose estimation. The dataset includes around 25K images containing over 40K people with annotated body joints. We use 20k annotated images to train and validate our compressed model in our project.

**COCO pose estimation dataset:** COCO pose estimation dataset are from COCO keypoint challenge dataset, which requires simultaneously detecting people and localizing human keypoints.

The COCO train, validation, and test sets contain more than 200k images and 250,000 person instances labeled with keypoints (the majority of people in COCO at medium and large scales). We use 200k annotated images to train and validate our compressed model in our project.

**VGG pose estimation dataset:** The VGG pose estimation dataset used here is the YouTube Pose dataset, which is a collection of YouTube videos for human (mainly upper) body pose estimation. The videos found on YouTube covering a broad range of activities and people, e.g., dancing, stand-up comedy, how-to, sports, disk jockeys, performing arts and dancing sign language signers. We use 30 videos of them in our project as the test dataset.

### 4.2. Metrics

Given the prediction and ground truth of pose keypoints, the similarity metric between them is calculated based on object keypoint similarity from the COCO standard.

**Object keypoint similarity (OKS)** We measure the quality of human pose estimation result with OKS metrics. We use the definition of commonly used by COCO dataset [20]. It is measured based on the normalized Euclidean distances between each ground truth and detected keypoint shown below.

$$OKS = \frac{\sum_i [e^{-d_i^2/2s^2k_i^2}\delta(v_i > 0)]}{\sum_i [\delta(v_i > 0)]} \quad (4)$$

where $d_i$ are the Euclidean distance between each ground truth and detected keypoint and the vi are the visibility flags of the ground truth (the detector's predicted $v_i$ are not used). Other parameters could be referred to [20]. OKS has the range in [0, 1]. The higher the value, the more accurate of the detection.

Then pose estimation accuracy metric used to measure the accuracy is the average values of OKS at different values $[0.5 : 0.05 : 0.95]$.

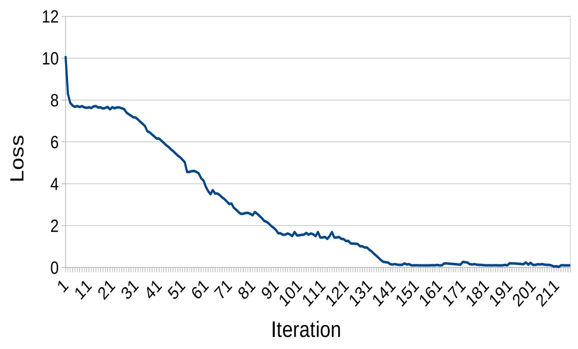
### 4.3. Training and Validation

For the full model, we use the pretrained model publicly available as our model to detect human pose. For the compressed model, we have to train the model from scratch.

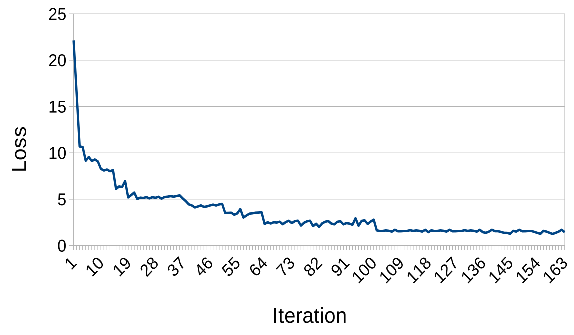
Here we show the validation loss as in Figure 6. The training loss is the average loss of each image in the dataset. It shows it has good loss for the validation loss result during the training process on both image datasets.

### 4.4. Result on the Validation Dataset

We show the performance statistics of full model and compressed model on the validation datasets in Table 2. It shows the full model on the image dataset achieves overall



(a) MPII dataset validation loss



(b) Coco dataset validation loss

Figure 6: Validation Loss for compressed model

61.4% and 75.6% with around 0.25 sec executed on an image on average on a GPU server. The compressed model has dramatically reduced the time to 0.08 sec on the same setting, which is about 3x less of the time on the full model, but with about 9% accuracy loss on the average.

Table 2: Performance result on validation data

| **Model** | **Validation dataset** | **AP (%)** | **Time (s) (average of each image on a GPU server)** |
|---|---|---|---|
| Full model | COCO | 61.4 | 0.265 |
| | MPII | 75.6 | 0.243 |
| Compressed model | COCO | 52.6 | 0.082 |
| | MPII | 65.8 | 0.078 |

## 4.5. Test video dataset result

We show the test result on the 30 video dataset in Table 3. The parameter of time period $p = 4s$, the segment $s = 1$. The user accuracy threshold is set as 0.85.

Table 3: Youtube video detection result

| **Method** | **AP (%)** | **Time (s) average of each frame on a GPU server)** |
|---|---|---|
| Full model | 88.9 | 0.213 |
| Compressed model | 83.6 | 0.083 |
| Our dynamic architecture | 86.7 | 0.136 |

## 4.6. Impact of parameters

There are three parameters involved in our method. The user input accuracy threshold, the period time and the segment time. We evaluate the impact of these parameters on the pose estimation accuracy and the computation resources on 5 videos, and report the average values of the results by varying one parameters and fixing the other parameters. Figure 7 shows the result with different input accuracy thresholds. It shows the monotonically increasing relationship between the accuracy/computation resources with the input accuracy, which there is a good trade-off where the input threshold is around 0.84. Figure 8 shows the result with different input period time. It shows the monotonically decreasing relationship between the accuracy/computation resources with the period time. It is good to choose a trade-off value of period time around 4 to 6 sec. Figure 9 shows the result with different input segment time. It also shows the monotonically decreasing relationship between the accuracy/computation resources with the segment time. It is good to choose a trade-off value of segment time around 1 to 2 sec.

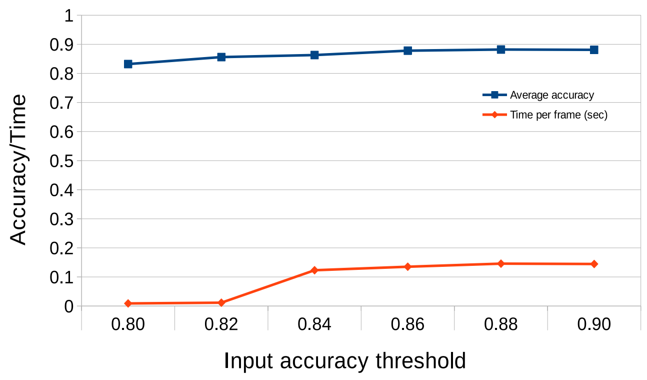


Figure 7: Varying the input accuracy threshold. Period time is set 4 sec, segment time is 1 sec

## 4.7. Pose estimation video demonstration

Here we show two videos' pose estimation results of our proposed method here. The input accuracy threshold is set as 0.85. The period time is 4 sec, the segment time is 1

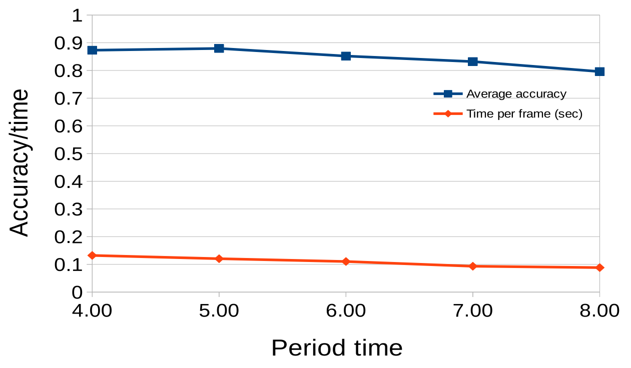


Figure 8: Varying the period time. Input accuracy threshold is set 0.85, segment time is set 1 sec

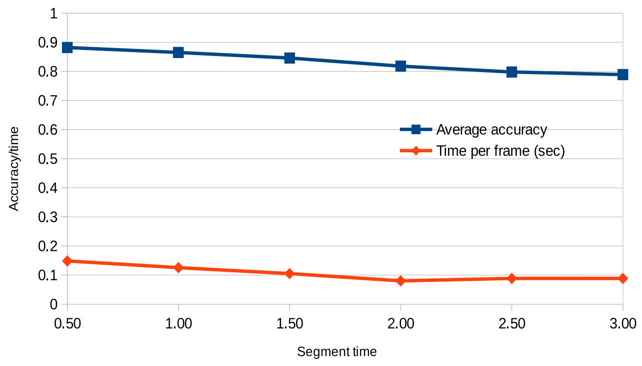


Figure 9: Varying the segment time. Input accuracy threshold is set 0.85, period time is set 4 sec

sec. Figure 10 shows the part of the frames extracted for demonstration in video 1. Figure 11 shows the part of the frames extracted for demonstration in video 2. The yellow lines indicate when the compressed model is used, the color lines indicates when the full model is use. When the human moves faster, the full model is generally used. When the human moves slower, the compressed model is generally used. It shows the effectiveness of our proposed method which satisfies the accuracy threshold requirement with very few computation resources.

## 5. Conclusion

In this project, we propose an effective and efficient method to detect human pose in video by considering the temporal characteristics in video content setting. We have experimentally test the method on two image dataset and one video dataset. It shows the effectiveness and efficiency of our pose estimation method. Compared to the full model of baseline, it greatly reduces the computation resources by 48% with just 2% accuracy loss in the video dataset and almost the real time execution, which is satisfied with the real life deployment. For the future work, the period and segment time is fixed in current work. Therefore, it might be worth to explore the period and segment time values, which are better to be dynamically set by the system according to video content settings.

## References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014. 4

[2] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*, 2018. 1, 2, 3

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017. 2

[4] James Charles, Tomas Pfister, Derek Magee, David Hogg, and Andrew Zisserman. Personalizing human video pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3063–3072, 2016. 4

[5] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7103–7112, 2018. 1, 2

[6] Matthias Dantone, Juergen Gall, Christian Leistner, and Luc Van Gool. Human pose estimation using body parts dependent joint regressors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3041–3048, 2013. 2

[7] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2334–2343, 2017. 2

[8] Georgia Gkioxari, Alexander Toshev, and Navdeep Jaitly. Chained predictions using convolutional neural networks. In *European Conference on Computer Vision*, pages 728–743. Springer, 2016. 2

[9] Shaoli Huang, Mingming Gong, and Dacheng Tao. A coarse-fine network for keypoint localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3028–3037, 2017. 1, 2

[10] Junchen Jiang, Ganesh Ananthanarayanan, Peter Bodik, Siddhartha Sen, and Ion Stoica. Chameleon: scalable adaptation of video analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 253–266. ACM, 2018. 1

[11] Sam Johnson and Mark Everingham. Learning effective human pose estimation from inaccurate annotation. In *CVPR 2011*, pages 1465–1472. IEEE, 2011. 2

[12] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. Noscope: optimizing neural network queries

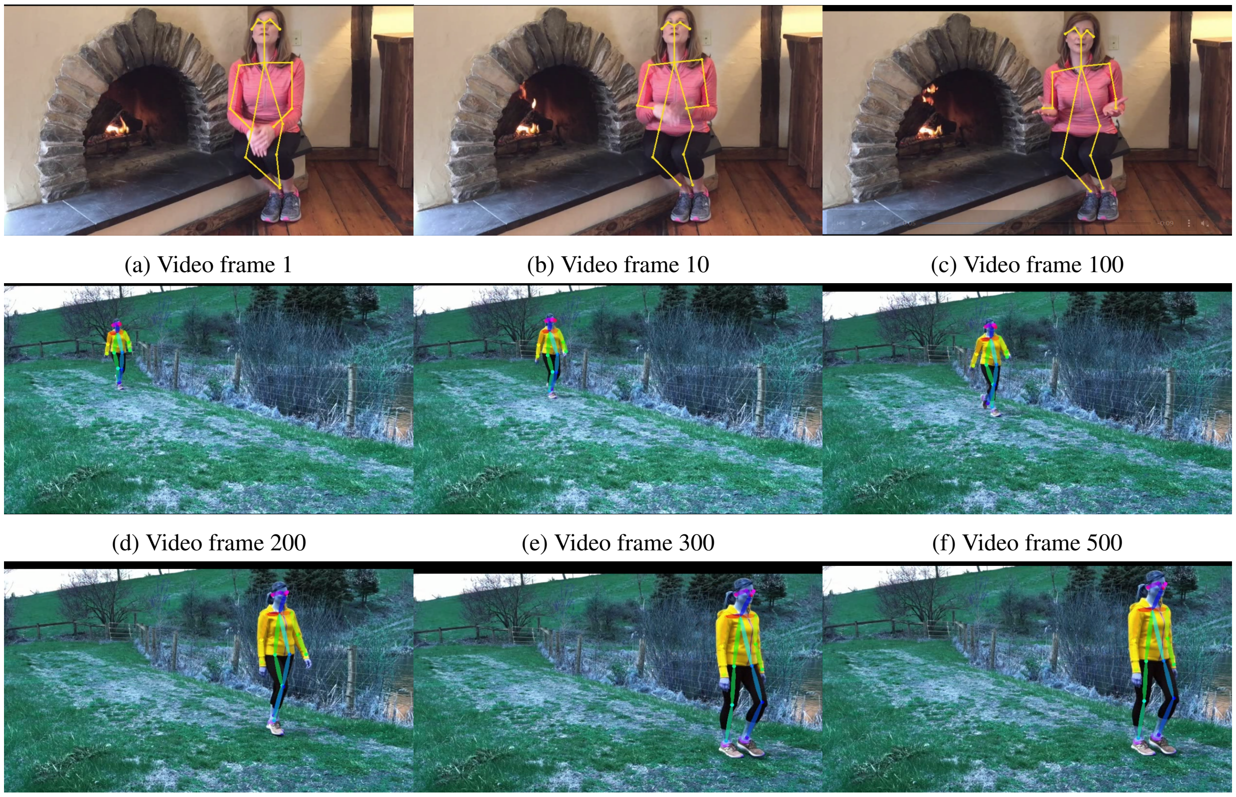

(a) Video frame 1 (b) Video frame 10 (c) Video frame 100
(d) Video frame 200 (e) Video frame 300 (f) Video frame 500
(g) Video frame 700 (h) Video frame 900 (i) Video frame 1000

Figure 10: Demo of frames in video 1 pose estimation

over video at scale. *arXiv preprint arXiv:1703.02529*, 2017. 1

[13] Muhammed Kocabas, Salih Karagoz, and Emre Akbas. Multiposenet: Fast multi-person pose estimation using pose residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 417–433, 2018. 2

[14] Mude Lin, Liang Lin, Xiaodan Liang, Keze Wang, and Hui Cheng. Recurrent 3d pose sequence machines. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 810–819, 2017. 2

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 4

[16] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in neural information processing systems*, pages 2277–2287, 2017. 2

[17] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016. 2

[18] George Papandreou, Tyler Zhu, Liang-Chieh Chen, Spyros Gidaris, Jonathan Tompson, and Kevin Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 269–286, 2018. 2

[19] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4903–4911, 2017. 2

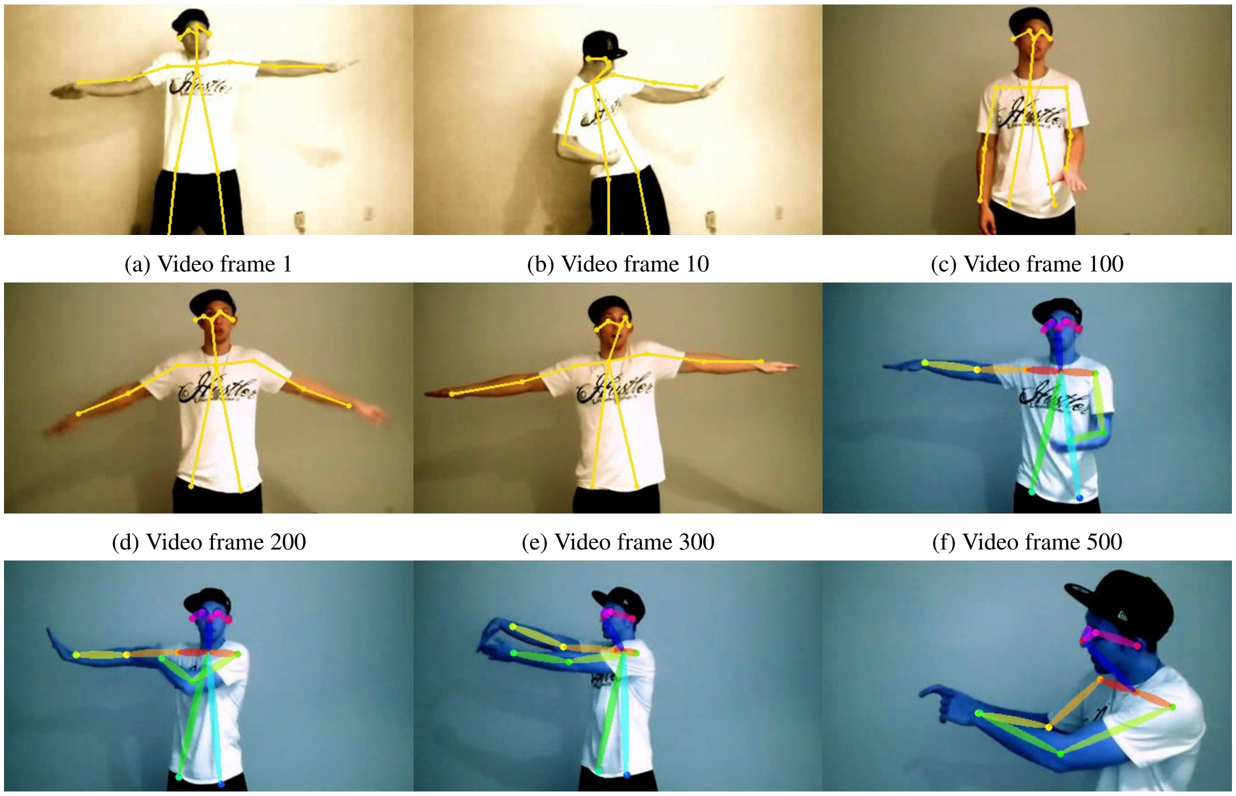
[20] Matteo Ruggero Ronchi and Pietro Perona. Benchmarking and error diagnosis in multi-instance pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 4

[21] Benjamin Sapp, Chris Jordan, and Ben Taskar. Adaptive pose priors for pictorial structures. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 422–429. IEEE, 2010. 2

[22] Jie Song, Limin Wang, Luc Van Gool, and Otmar Hilliges. Thin-slicing network: A deep structured model for pose estimation in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4220–4229, 2017. 2

[23] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *arXiv preprint arXiv:1902.09212*, 2019. 1, 2

[24] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. Integral human pose regression. In *Proceedings of the*


(a) Video frame 1 (b) Video frame 10 (c) Video frame 100

(d) Video frame 200 (e) Video frame 300 (f) Video frame 500

(g) Video frame 700 (h) Video frame 900 (i) Video frame 1000

Figure 11: Demo of frames in video 2 pose estimation

*European Conference on Computer Vision (ECCV)*, pages 529–545, 2018. 2

[25] Fangting Xia, Peng Wang, Xianjie Chen, and Alan L Yuille. Joint multi-person pose estimation and semantic part segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6769–6778, 2017. 1, 2

[26] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 466–481, 2018. 1