arXiv:2006.10204v1 [cs.CV] 17 Jun 2020

BlazePose: On-device Real-time Body Pose tracking

Valentin Bazarevsky Ivan Grishchenko Karthik Raveendran
Tyler Zhu Fan Zhang Matthias Grundmann
Google Research
1600 Amphitheatre Pkwy, Mountain View, CA 94043, USA
{valik, igrishchenko, krav, tylerzhu, zhafang, grundman}@google.com

Abstract

We present BlazePose, a lightweight convolutional neural network architecture for human pose estimation that is tailored for real-time inference on mobile devices. During inference, the network produces 33 body keypoints for a single person and runs at over 30 frames per second on a Pixel 2 phone. This makes it particularly suited to real-time use cases like fitness tracking and sign language recognition. Our main contributions include a novel body pose tracking solution and a lightweight body pose estimation neural network that uses both heatmaps and regression to keypoint coordinates.

1. Introduction

Human body pose estimation from images or video plays a central role in various applications such as health tracking, sign language recognition, and gestural control. This task is challenging due to a wide variety of poses, numerous degrees of freedom, and occlusions. Recent work [10][7] has shown significant progress on pose estimation. The common approach is to produce heatmaps for each joint along with refining offsets for each coordinate. While this choice of heatmaps scales to multiple people with minimal overhead, it makes the model for a single person considerably larger than is suitable for real-time inference on mobile phones. In this paper, we address this particular use case and demonstrate significant speedup of the model with little to no quality degradation.

In contrast to heatmap-based techniques, regression-based approaches, while less computationally demanding and more scalable, attempt to predict the mean coordinate values, often failing to address the underlying ambiguity. Newell *et al*. [9] have shown that the stacked hourglass architecture gives a significant boost to the quality of the prediction, even with a smaller number of parameters. We extend this idea in our work and use an encoder-decoder network architecture to predict heatmaps for all joints, followed by another encoder that regresses directly to the coordinates of all joints. The key insight behind our work is that the heatmap branch can be discarded during inference, making it sufficiently lightweight to run on a mobile phone.

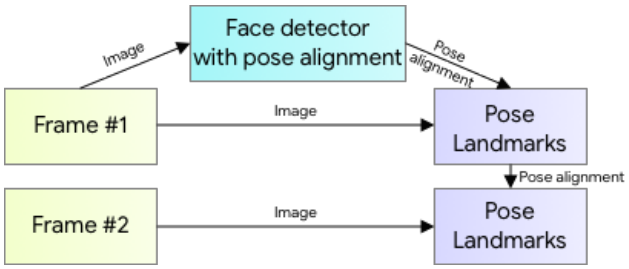

Figure 1. Inference pipeline. See text.

2. Model Architecture and Pipeline Design

2.1. Inference pipeline

During inference, we employ a detector-tracker setup (see Figure 1), which shows excellent real-time performance on a variety of tasks such as hand landmark prediction [3] and dense face landmark prediction [6]. Our pipeline consists of a lightweight body pose detector followed by a pose tracker network. The tracker predicts keypoint coordinates, the presence of the person on the current frame, and the refined region of interest for the current frame. When the tracker indicates that there is no human present, we re-run the detector network on the next frame.

2.2. Person detector

The majority of modern object detection solutions rely on the Non-Maximum Suppression (NMS) algorithm for their last post-processing step. This works well for rigid objects with few degrees of freedom. However, this algorithm breaks down for scenarios that include highly articulated poses like those of humans, *e.g*. people waving or hugging. This is because multiple, ambiguous boxes satisfy the intersection over union (IoU) threshold for the NMS algorithm.

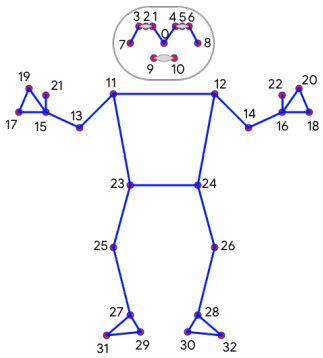

Figure 3. 33 keypoint topology.

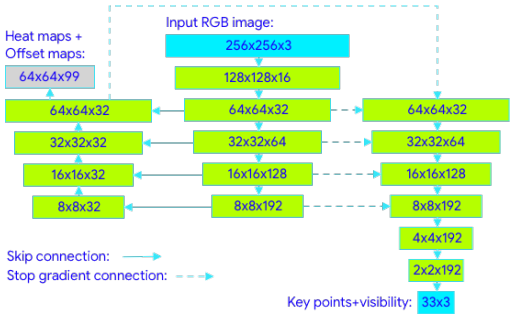

Figure 4. Network architecture. See text for details.

To overcome this limitation, we focus on detecting the bounding box of a relatively rigid body part like the human face or torso. We observed that in many cases, the strongest signal to the neural network about the position of the torso is the person's face (as it has high-contrast features and has fewer variations in appearance). To make such a person detector fast and lightweight, we make the strong, yet for AR applications valid, assumption that the head of the person should always be visible for our single-person use case. As a consequence, we use a fast on-device face detector [2] as a proxy for a person detector. This face detector predicts additional person-specific alignment parameters: the middle point between the person's hips, the size of the circle circumscribing the whole person, and incline (the angle between the lines connecting the two mid-shoulder and mid-hip points).

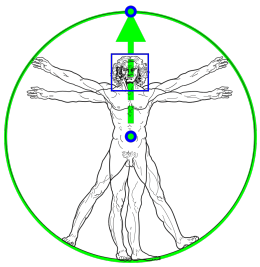

Figure 2. Vitruvian man aligned via our detector vs. face detection bounding box. See text for details.

2.3. Topology

We present a new topology using 33 points on the human body by taking the superset of those used by BlazeFace[2], BlazePalm[3], and Coco[8]. This allows us to be consistent with the respective datasets and inference networks.

In contrast with the OpenPose[4] and Kinect[1] topologies, we use only a minimally sufficient number of keypoints on the face, hands, and feet to estimate rotation, size, and position of the region of interest for the subsequent model. The topology we use is shown in Figure 3. For additional information, please see Appendix A.

2.4. Dataset

Compared to the majority of existing pose estimation solutions that detect keypoints using heatmaps, our tracking-based solution requires an initial pose alignment. We restrict our dataset to those cases where either the whole person is visible, or where hips and shoulders keypoints can be confidently annotated. To ensure the model supports heavy occlusions that are not present in the dataset, we use substantial occlusion-simulating augmentation. Our training dataset consists of 60K images with a single or few people in the scene in common poses and 25K images with a single person in the scene performing fitness exercises. All of these images were annotated by humans.

2.5. Neural network architecture

The pose estimation component of our system predicts the location of all 33 person keypoints, and uses the person alignment proposal provided by the first stage of the pipeline (Section 2.1).

We adopt a combined heatmap, offset, and regression approach, as shown in Figure 4. We use the heatmap and offset loss only in the training stage and remove the corresponding output layers from the model before running the inference. Thus, we effectively use the heatmap to supervise the lightweight embedding, which is then utilized by the regression encoder network. This approach is partially inspired by Stacked Hourglass approach of Newell et al. [9], but in our case, we stack a tiny encoder-decoder heatmap-based network and a subsequent regression encoder network.

We actively utilize skip-connections between all the stages of the network to achieve a balance between high- and low-level features. However, the gradients from the regression encoder are not propagated back to the heatmap-trained features (note the gradient-stopping connections in Figure 4). We have found this to not only improve the heatmap predictions, but also substantially increase the coordinate regression accuracy.

2.6. Alignment and occlusions augmentation

A relevant pose prior is a vital part of the proposed solution. We deliberately limit supported ranges for the angle, scale, and translation during augmentation and data preparation when training. This allows us to lower the network

capacity, making the network faster while requiring fewer computational and thus energy resources on the host device.

Based on either the detection stage or the previous frame keypoints, we align the person so that the point between the hips is located at the center of the square image passed as the neural network input. We estimate rotation as the line L between mid-hip and mid-shoulder points and rotate the image so L is parallel to the y-axis. The scale is estimated so that all the body points fit in a square bounding box circumscribed around the body, as shown in Figure 2. On top of that, we apply 10% scale and shift augmentations to ensure the tracker handles body movements between the frames and distorted alignment.

To support the prediction of invisible points, we simulate occlusions (random rectangles filled with various colors) during training and introduce a per-point visibility classifier that indicates whether a particular point is occluded and if the position prediction is deemed inaccurate. This allows tracking a person constantly even for cases of significant occlusions, like upper body-only or when the majority of person body is out of scene as shown on Figure 5.

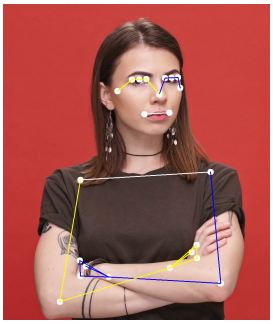

Figure 5. BlazePose results on upper-body case

3. Experiments

To evaluate our model's quality, we chose OpenPose [4] as a baseline. To that end, we manually annotated two in-house datasets of 1000 images, each with 1–2 people in the scene. The first dataset, referred to as AR dataset, consist of a wide variety of human poses in the wild, while the second is comprised of yoga/fitness poses only. For consistency, we only used MS Coco [8] topology with 17 points for evaluation, which is a common subset of both OpenPose and BlazePose. As an evaluation metric, we use the Percent of Correct Points with 20% tolerance (PCK@0.2) (where we assume the point to be detected correctly if the 2D Euclidean error is smaller than 20% of the corresponding person's torso size). To verify the human baseline, we asked two annotators to re-annotate the AR dataset independently and obtained an average PCK@0.2 of 97.2.

We trained two models with different capacities: BlazePose Full (6.9 MFlop, 3.5M Params) and BlazePose Lite (2.7 MFlop, 1.3M Params). Although our models show slightly worse performance than the OpenPose model on the AR dataset, BlazePose Full outperforms OpenPose on Yoga/Fitness use cases. At the same time, BlazePose performs 25–75 times faster on a *single mid-tier phone CPU* compared to OpenPose on a *20 core desktop* CPU[5] depending on the requested quality.

[1]Desktop CPU with 20 cores (Intel i9-7900X)
[2]Pixel 2 Single Core via XNNPACK backend

Model	FPS	AR Dataset, PCK@0.2	Yoga Dataset, PCK@0.2
OpenPose (body only)	0.4[1]	**87.8**	83.4
BlazePose Full	10[2]	84.1	**84.5**
BlazePose Lite	**31**[2]	79.6	77.6

Table 1. BlazePose vs OpenPose

Figure 6. BlazePose results on yoga and fitness poses.

4. Applications

We developed this new, on-device, single person-specific human pose estimation model to enable various performance-demanding use cases such as Sign Language, Yoga/Fitness tracking and AR. This model works in near-realtime on a mobile CPU and can be sped up to super-realtime latency on a mobile GPU. As its 33 keypoint topology is consistent with BlazeFace[2] and BlazePalm[3], it can be a backbone for subsequent hand pose[3] and facial geometry estimation[6] models.

Our approach natively scales to a bigger number of keypoints, 3D support, and additional keypoint attributes, since it is not based on heatmaps/offset maps and therefore does not require an additional full-resolution layer per each new feature type.

References

[1] Azure kinect body tracking joints. https://docs.microsoft.com/en-us/azure/kinect-dk/body-joints. [Online; accessed April 2, 2020]. 2

[2] Valentin Bazarevsky, Yury Kartynnik, Andrey Vakunov, Karthik Raveendran, and Matthias Grundmann. Blazeface: Sub-millisecond neural face detection on mobile gpus. *CoRR*, abs/1907.05047, 2019. 2, 3

[3] Valentin Bazarevsky and Fan Zhang. On-device, real-time hand tracking with mediapipe. https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html. [Online; accessed April 2, 2020]. 1, 2, 3

[4] Z Cao, G Martinez Hidalgo, T Simon, SE Wei, and YA Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 2, 3

[5] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose 1.1.0 benchmark. https://docs.google.com/spreadsheets/d/1-DynFGvoScvfWDA1P4jDInCkbD4lg0IKOYbXgEq0sK0. [Online; accessed March 30, 2020]. 3

[6] Yury Kartynnik, Artsiom Ablavatski, Ivan Grishchenko, and Matthias Grundmann. Real-time facial surface geometry from monocular video on mobile gpus. *CoRR*, abs/1907.06724, 2019. 1, 3

[7] Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. Pifpaf: Composite fields for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11977–11986, 2019. 1

[8] 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. 2, 3

[9] 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. 1, 2

[10] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5693–5703, 2019. 1

Appendix A. BlazePose keypoint names

0. Nose
1. Left eye inner
2. Left eye
3. Left eye outer
4. Right eye inner
5. Right eye
6. Right eye outer
7. Left ear
8. Right ear
9. Mouth left
10. Mouth right
11. Left shoulder
12. Right shoulder
13. Left elbow
14. Right elbow
15. Left wrist
16. Right wrist
17. Left pinky #1 knuckle
18. Right pinky #1 knuckle
19. Left index #1 knuckle
20. Right index #1 knuckle
21. Left thumb #2 knuckle
22. Right thumb #2 knuckle
23. Left hip
24. Right hip
25. Left knee
26. Right knee
27. Left ankle
28. Right ankle
29. Left heel
30. Right heel
31. Left foot index
32. Right foot index