Adapting MobileNets for mobile based upper body pose estimation

Bappaditya Debnath[1], Mary O'Brien[2], Motonori Yamaguchi[3] and Ardhendu Behera[1]
Computer Science[1], Psychology[2], Health and Social Care[3], Edge Hill University, Ormskirk, UK
{debnathb, beheraa}@edgehill.ac.uk

Abstract

Human pose estimation through deep learning has achieved very high accuracy over various difficult poses. However, these are computationally expensive and are often not suitable for mobile based systems. In this paper, we investigate the use of MobileNets, which is well-known to be a light-weight and efficient CNN architecture for mobile and embedded vision applications. We adapt MobileNets for pose estimation inspired by the hourglass network. We introduce a novel split stream architecture at the final two layers of the MobileNets. This approach reduces over-fitting, resulting in improvement in accuracy and reduction in parameter size. We also show that by maintaining part of the original network we are able to improve accuracy by transferring the learned features from ImageNet pre-trained MobileNets. The adapted model is evaluated on the FLIC dataset. Our network out-performed the default MobileNets for pose estimation, as well as achieved performance comparable to the state of the art results while reducing inference time significantly.

1. Introduction

Human body pose estimation or the localization of body joints in monocular RGB images, is a very challenging problem and is an actively researched area in computer vision. This is mainly due to joint occlusions (partial or full), clothing, lighting conditions, variation in body shape, unrestricted viewing angle and complex joint inter-dependencies. It has many potential applications including tracking, robotics and AI, action/activity recognition, human-computer interaction etc. Recent advances in Convolutional Neural Networks (CNNs) have significantly influenced the performance of pose estimation models [1, 14, 15, 28]. Many of these models are adapted from the CNN models, which are focused on image recognition tasks (Fig. 1). Most of these models are complex and require powerful GPUs even for prediction. In many real-world vision applications (e.g. robotics, autonomous vehicles), there is a constraint on resources (e.g. power, memory) and the system is expected to work in real-time without compromising the accuracy. Fig. 1 shows well-known pose estimation and classification models with their respective inference times. For image classification tasks, many light-weight models [18, 7, 3] have been developed for implementation on mobile devices. However, adapting these models for human pose estimation is still in its infancy.

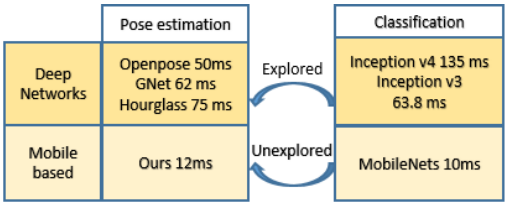


Figure 1. We explore mobile based pose estimation through adaptation of MobileNets. Similar adaptations exist for deep models. GNet [14] and Stacked hourglass [13] inference times are as reported in the paper. Inference times for Inception v3 [23], v4 [24] and OpenPose [1] are from our setup.

In this paper, we investigate adaptation of MobileNets [5] for human pose estimation. MobileNets is a fast and efficient classification model for mobile and embedded systems. State of the art models such as Inception V3 [23] can achieve top-1 accuracy of 84% on Stanford Dogs [9] dataset [19] as compared to MobileNets' 83%. But the number of parameters in MobileNets is $1/6^{th}$ of that of Inception V3. As shown in Fig. 1, well-known pose estimation models take more than 50 ms for single image inference while MobileNets requires only 10 ms. Thus, we intend to explore the area of mobile based pose estimation by adapting MobileNets that have been used widely for classification tasks. Our main contributions are:

1) We adapt MobileNets for pose estimation inspired by the widely used stacked hourglass network [13].

2) We introduce a novel spilt architecture at the final two layers of the MobileNets which reduces over-fitting and increases accuracy.

978-1-5386-9294-3/18/$31.00 ©2018 IEEE

2. Related work

Focus of research on human pose estimation has shifted from classical approaches [17, 30] to deep neural networks since the introduction of AlexNet for pose estimation [26]. Human pose estimation is a regression problem and stacked hourglass network [13] has become the basis of many pose estimation models [14, 29, 8]. The stacked hourglass network goes from high resolution to low and back to high and hence the name hourglass. It also has skip connections that connect the down sampling layers to the up sampling layers. The high output resolution makes it suitable for heatmap regression. Models based on the stacked hourglass network have included hand crafted features to guide their network better [14, 29]. Cao et al. [1] have designed their own network using Part Affinity Fields (PAFs). Pose estimation problem has also been addressed as a body part classification and joint localization problem. The deepcut model [16] uses a partitioning and labelling formulation generated with CNN based part detectors. Use of R-CNN based classification to achieve joint localization has also been explored in [4]. More recently, researchers have explored adaptation of standard CNN based classification architectures for pose estimation. In [28], ResNet-50 is used for progressively regressing the joint coordinates. Papandreou et al. [15] use ResNet architecture with faster R-CNN and regress heatmaps in a novel way of non-maxima suppression. Combination of concepts from pose estimation and classification has also been used together. Ning et al. [14] combine modules from Inception-ResNet [24] within stacked hourglass network along with hand crafted features such as HOG and Hough features for multi-person pose estimation.

Training aforementioned deep networks are expensive in terms of resources (e.g. GPUs) and computational complexities. Transfer learning from pre-trained networks is a way to overcome the aforementioned drawback to some extent. Yosinski et al. [31] explains that generalized features from first few layers of a pre-trained network are transferable. The consensus is that layers towards the end learn composite features specific to a given task that the model is trained for, whereas the layers towards the beginning tend to learn more general features such as edges, corners etc. The trend in image classification is to use ImageNet pre-trained networks and apply transfer learning on the target datasets. This makes training fast and computationally less demanding. In [16], ImageNet pre-trained network is used to fine-tune the pose estimation model. Well-known model such as openpose [1] uses pre-trained VGG-16 [21] model for feature extraction. The VGG-16 has 133 to 144 million parameters. This adds to the computational cost of inference. Therefore, inference using such networks are not suitable for mobile based pose estimation. To reduce the training time, the proposed model takes advantage of the transfer learning by initializing the MobileNets weights from the respective pre-trained model trained on the ImageNet.

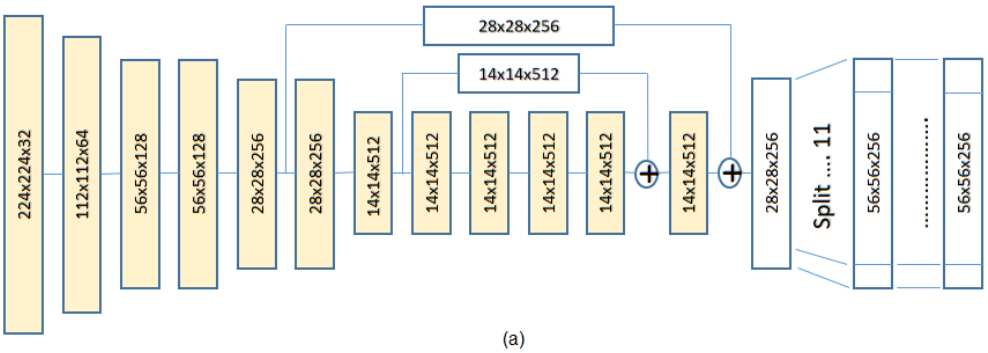
Exploring deep learning models for mobile based systems is an active area of research. The trend is to maximize speed by compromising accuracy. One such model, SqeezeNet [7] matches AlexNet's [11] performance with 50 times less parameters. It reduces the number of parameters by replacing 3x3 filters with 1x1 filters. It also maintains large activation maps by down sampling late in the network. This increases the size of maps and hence accuracy, but the model requires fewer filters and, thus, it is smaller in size. DenseNet [6], shows that if every layer is connected to every successive layer in a feed forward manner, the network achieves similar accuracy with less parameters. To achieve similar accuracy to that of ResNet-152 it requires less than half of the parameters. Xception [3] uses depth-wise separable convolutions for constructing a lighter model. MobileNets uses depth-wise and point-wise separable convolutions for designing an architecture with only 4.3 million parameters. Our research focuses on adapting mobile based classification model for pose estimation and it takes advantage of transfer learning for quick training with less data.

3. Proposed approach

To achieve our objectives, three mobile based classification models DenseNets [6], MobileNets [5] and SqueezeNets [7] are considered. With 0.50 alpha and 160×160 input resolution, MobileNets has 1.32 million parameters against SqueezeNet's 1.25. MobileNets scores 60.2% while SqueezeNets is 57.5% on ImageNet dataset. The best performing MobileNets variation with alpha 1 and input resolution 224×224 gives 70% accuracy. In this variation MobileNets has 4.2 million parameters. MobileNets was preferred over SqueezeNet due to higher accuracy. In our setup, DenseNets 121 takes 63.8 ms while MobileNets takes around 12 ms for the inference of a single image. DenseNets can be scaled from less than 1 to 20 million parameters. For our experiment, we choose a size of DenseNets equivalent to that of MobileNets. A few different DenseNet block size combinations such as 6, 12, 12, 8 with 4.3 million parameters and 6, 12, 12, 16 with 5.7 million parameters were tested. Block size of 6, 12, 24, 16 was also tested with ImageNet initialized weights. Our preliminary experiments showed that DenseNet did not converge as well as MobileNets.

3.1. MobileNets review

MobileNets are based on streamlined architecture that uses depth-wise and pointwise separable convolutions (DPC) [5]. Convolution operation normally filters and then combines inputs and outputs in one step. MobileNets splits filtering and combining into two separate steps. It shows that this results in drastically reduced numbers of param-



Joints	Filter distribution
Left shoulder	25
Left elbow	27
Left wrist	35
Right shoulder	25
Right elbow	27
Right wrist	35
Left hip	27
Right hip	27
Left eye	10
Right eye	10
Nose	8

(b)

Figure 2. (a) Modified MobileNets architecture. First and last layers are normal convolution and rest are depthwise and pointwise separable convolution blocks. Pre-trained lower layers from MobileNets are depicted in yellow. Last two layers are split joint-wise. (b) Joint-wise filter distributions for last two layers

eters making the network faster. The network uses 3x3 depth-wise separable convolutions which results in up to 9 times less computations as compared to standard convolutions at a cost of fractional reduction in accuracy. Suppose D_K is square kernel size, D_F is feature map size, M is the number of input channels and N is the number of output channels. The computational cost for standard convolution would be [5]:

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F \quad (1)$$

But depth-wise separable convolution computes the same operation in two steps and the computational cost comes down to:

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + D_K \cdot D_K \cdot N \cdot D_F \cdot D_F \quad (2)$$

MobileNets lowers the resolution from 224×224 in the input layer to 7×7 in the last convolution layer. The number of filters increases from 32 in the first layer to 1024 in the penultimate layer. The last DPC layer is followed by a global average pooling (GAP) layer whose output is reshaped and fed into a fully connected (FC) layer with output size of 1000. The FC layer is responsible for 24.33% of the parameters.

To adapt MobileNets for pose estimation we considered the following factors. Resolution of 7×7 at the final DPC layer makes heatmap regression difficult. Such supervision requires higher resolution. But, higher resolution with large number of filters (1024) can be a speed bottleneck and may lead to over-fitting. The fully connected layer needs to be removed as it is unsuitable for heatmap regression. We intended to implement these changes while still retaining part of ImageNet pre-trained MobileNets so that the model could benefit from the transfer learning. GAP layer is not used in pose estimation problems which helps in reducing over-fitting. Thus, the goal is to introduce an alternate approach to prevent over-fitting.

3.2. MobileNets modifications

Inspired by the hourglass network [13], we modify the final two DPC layers of mobileNets to increase the resolution through upsampling. If the whole model is changed to reflect an hourglass then pre-trained weights cannot be used and the advantage of transfer learning will be lost which could impact the accuracy. CNNs learn generalized features in the first few layers and class specific features towards the end. Thus, only the final two layers are changed where upsampling is used to increase the filter size from 14×14 to 28×28 and then to 56×56. Increasing the resolution further impacts the speed and, thus, the final output resolution of the model is kept at 56×56. Lower size of heatmap resolution as compared to the input (224×224) does not impact on accuracy as pointed out in [13]. In the proposed model, the final filter resolution is 56×56 as opposed to 7×7 in the original MobileNets. To reduce the impact on speed due to increase in filter size, we reduce the number of filters in the final two layers. These layers have 256 filters each which is $1/4^{th}$ of the 1024 filters in the original MobileNets. For heatmap regression, the last fully connected layer is replaced by a normal convolution layer with 11 filters that correspond to heatmaps for 11 body joints. The two changed DPC layers with the final convolutional layer is shown in white towards the right of Fig 2a. The hourglass network also has side layers (skip connections) which are used to connect features across scales. The two horizontal white boxes depict the skip connections. These are introduced at resolution 28×28 and 14×14. Each skip connection goes through a DPC layer of the same dimension.

3.3. Split Stream

GAP works by enforcing correspondence between confidence map and classes. It inherently prevents over-fitting [12]. GAP is not commonly used by pose estimation models

[29, 14], since in regression problems there are no classes. This layer was thus removed. The FC layer which is responsible for 24% of the parameters is prone to overfitting in the absence of GAP and dropout. None of the standard pose estimation models [13, 29] use dropout for dealing with overfitting as randomly dropping out filters is not suitable for regression. We introduce a novel split stream architecture to deal with over-fitting. The last DPC layer and the final convolution are split into 11 filter groups as shown towards the right side of Fig. 2a. The 11 filter streams correspond to 11 body joints. Filters within each stream are shared and have no connection to filters from different streams. As a result low level features in the lower layers are common but high level features of individual joints are regressed independently. This has two effects: 1) it reduces the number of parameters making the network lighter, 2) it reduces overfitting for pose estimations problems where GAP or dropout is not used. Our experiments show that when split architecture is used the validation error follows the training error more closely than without it. To determine the number of filters needed for each joint, all the joints are first allocated filters equally. Then difficult joints like elbows and wrists are gradually allocated more filters than easier parts like the nose. Over several experiments the optimal filter numbers are obtained. The joint-wise filter distribution is shown in Fig. 2b. In order to improve the detection performance of difficult joints, we experimented by allocating more filters to wrists and elbows but it did not increase accuracy any further.

4. Training details

We use the FLIC [20] dataset for evaluation. It consists of 5003 images out of which 3987 images are for training and 1016 are for testing. 80-20 split in a small dataset indicates good generalization. Images are cropped to loosely fit the person whose annotations are available. Data augmentation is applied in the form of rotation (+/- 30 degrees) and scaling (.75-1.25). For the baseline evaluation of MobileNets for pose estimation, only top softmax layer is removed and the number of classes changed to 22 (2×11 body joints). Mean squared error (MSE) regression loss is applied for supervision. Performance of split stream architecture is also evaluated with model supervised through MSE regression. The final model is supervised with heatmap regression.

Tensorflow is used for implementation. For transfer learning supervision is carried out with the original layers frozen with a learning rate of 0.001 for 50K iterations. The two layers receiving skip connections are also trained. Then the whole model is fine-tuned for 150K iterations, with the learning rate reduced to a 10^{th}. After the training loss plateaus, the learning rate was further reduced by half. The standard practice is to use stochastic gradient descent for optimization but Adam optimizer [10] with default parameters was found to converge the model much faster. While optimizing, the model also keeps track of moving averages of the gradients with a decay of 0.9. This helps to stabilize the training by smoothing the changing of gradients. The model was trained on an Nvidia Quadro M4000 which has an effective memory of 6.7 GB, with a batch size of 16.

Figure 3. Example output from FLIC dataset. Predicted joint positions are marked in Red

Model	Elbows	Wrists
Toshev et al. [26]	92.3	82.0
Tompson et al. [25]	93.1	89.0
Chen et al. [2]	95.3	92.4
Wei et al. [27]	97.6	95.0
Ours	**97.6**	**95.2**
Newell et al. [13]	99.0	97.0

Table 1. FLIC results PCK@0.2

MobileNets	Accuracy	Speed	Parameters	Size
Baseline	96.4	10 ms	4.3m	68 MB
Split	96.9	10 ms	3.3m	52 MB
Final	97.3	12 ms	2.3m	26 MB

Table 2. Comparison of proposed design with baseline

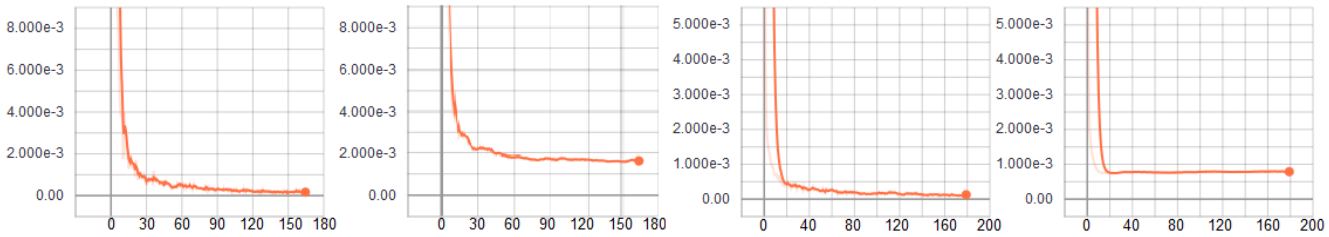


Figure 4. Loss (Y-axis) vs iteration in 1000s (X-axis) curve as generated by tensorboard. From left to right: MobileNets train loss; MobilNets validation loss; proposed model train loss; proposed model validation loss.

5. Evaluation

Evaluation is done using standard Percentage of Correct Keypoints (PCK) metric [27] where correct detection falls within 20% of torso size from the ground truth. For comparison we report wrists and elbow detection rate. These are the most difficult joints and is widely used for the performance comparison on FLIC dataset. Table 1 compares our results with other models and shows competitive results although our model is optimized for both speed and accuracy rather than only accuracy. State of the art result achieved by the stacked hourglass network [13] takes 75 ms for a forward pass on a 12GB Nvidia Titan. Ours takes 12 ms for a forward pass on 8GB Nvidia Quadro. However, the real advantage of using MobileNets based model is that it is optimized for mobile vision applications.

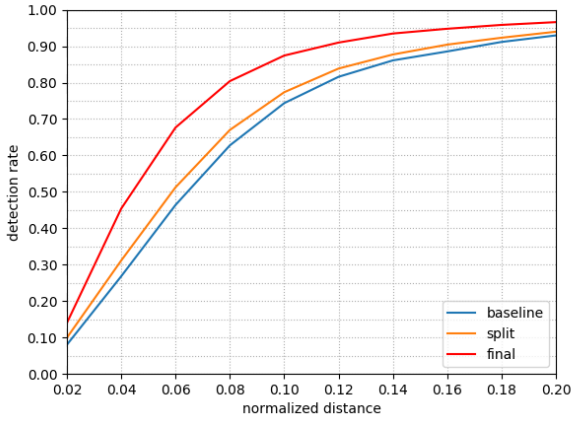


Figure 5. Comparison of elbow and wrist accuracy with baseline across PCK thresholds

Table 2 compares the baseline performance with the split architecture and our final model, which is our novel design that combines the split architecture and the hourglass network. The baseline model is transfer learned from ImageNet pre-trained MobileNets and performed accuracy of 96.4% on the FLIC dataset. If trained from randomly initialized weights the performance is much lower (87%). With the application of the split stream architecture but still regressing with MSE the gain in accuracy is 0.5%. The final accuracy gain with split architecture and hourglass-inspired design is 0.9%. The number of parameters and parameter size of the proposed model is approximately half of MobileNets. The marginal drop in speed is mainly due to the heatmap regression. Fig 5 shows proposed method preforming much better than the baseline at lower PCK thresholds.

MobileNets	Train error	Val error	Accuracy
Baseline	1.57e-4	1.67e-3	96.4
GAP removed	2.4e-05	1.04e-3	95.9
Split	1.52e-4	7.929e-4	96.9

Table 3. Modified MobileNets comparison

6. Discussion

The main novelty lies in the proposed split stream architecture. The network is split into separate groups of filters that do not share weights with other filter groups. This implies that the filter groups at the final two layers take common low level representations of the whole image as input but learn each joint independently of other joints. Table 2 shows that this brings down the number of parameters and parameter size by approximately half which is a big advantage for mobile based applications. Reducing over-fitting is another advantage of this split stream design. It is a well-known fact that larger networks are prone to over-fitting. This, ultimately led to the formation of inception modules [22], which uses reduced connections. Fig. 4 compares the train and validation error of MobileNets and the proposed design. Even though the final training loss is the same for both cases, the validation loss for the proposed model is less than half of the original MobileNets. MobileNets uses global average pooling for preventing over-fitting. When GAP layer is removed, the validation error does not change much but the training error drops by a factor of 6 along with 0.5% reduction in accuracy as shown in the second row of the Table 3. This is a typical sign of over-fitting. When we split the last two layers into separate filter groups for each joint then both accuracy and validation error improves and

the model performs better than the baseline. This shows that the split stream design helps in reducing over-fitting. It is interesting to note that accuracy of each joint is only loosely tied to number of filters allocated.

7. Conclusions

We demonstrate the adaptation of well-known fast and efficient MobileNets for human pose estimation through transfer learning. The network is adapted for heatmap regression inspired by the stacked hourglass network. A novel split architecture is introduced which helps in reducing over-fitting. Our modified MobileNets performs close to state of the art results while being considerably faster. It out performs the baseline considerably across PCK thresholds. We believe this will help advance the field of mobile and embedded vision applications focusing on human pose estimation.

Acknowledgment: This research was partly supported by Edge Hill university's Research Investment Fund (RIF).

References

[1] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, volume 1, page 7, 2017.

[2] X. Chen and A. L. Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *NIPS*, pages 1736–1744, 2014.

[3] F. Chollet. Xception: Deep learning with depthwise separable convolutions. pages 1251–1258, 2017.

[4] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik. R-cnns for pose estimation and action detection. *arXiv preprint arXiv:1406.5212*, 2014.

[5] A. G. Howard et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[6] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. In *IEEE CVPR*, volume 1, pages 4700–4708, 2017.

[7] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[8] L. Ke, M.-C. Chang, H. Qi, and S. Lyu. Multi-scale structure-aware network for human pose estimation. *arXiv preprint arXiv:1803.09894*, 2018.

[9] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization. In *CVPR workshop*, Colorado Springs, CO, June 2011.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[12] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[13] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499. Springer, 2016.

[14] G. Ning, Z. Zhang, and Z. He. Knowledge-guided deep fractal neural networks for human pose estimation. *IEEE Transactions on Multimedia*, 2017.

[15] Papandreou et al. Towards accurate multi-person pose estimation in the wild. In *Proc. CVPR*, pages 4903–4911, 2017.

[16] L. Pishchulin et al. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *Proc. CPVR 2016*, pages 4929–4937.

[17] V. Ramakrishna, D. Munoz, M. Hebert, J. A. Bagnell, and Y. Sheikh. Pose machines: Articulated pose estimation via inference machines. In *ECCV*, pages 33–47. Springer, 2014.

[18] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In *CVPR 2017*, pages 6517–6525. IEEE, 2017.

[19] O. Russakovsky et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

[20] B. Sapp and B. Taskar. Modec: Multimodal decomposable models for human pose estimation. In *In Proc. CVPR*, 2013.

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[22] C. Szegedy et al. Going deeper with convolutions. CVPR, 2015.

[23] C. Szegedy et al. Rethinking the inception architecture for computer vision. In *In Proc. CVPR*, pages 2818–2826, 2016.

[24] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[25] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *In Proc. CVPR*, pages 648–656, 2015.

[26] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *In Proc. CVPR*, pages 1653–1660, 2014.

[27] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Proc. CVPR*, pages 4724–4732, 2016.

[28] X. Xiao and W. Wan. Human pose estimation via improved resnet50. In *ICSSC 2017*, pages 1–5, June 2017.

[29] W. Yang, S. Li, W. Ouyang, H. Li, and X. Wang. Learning feature pyramids for human pose estimation. In *ICCV*, volume 2, 2017.

[30] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR 2011*, pages 1385–1392, June 2011.

[31] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, pages 3320–3328, 2014.