A lightweight 2D Pose Machine with attention enhancement

Luiz Schirmer*, Djalma Lucio*, Alberto Raposo*, Luiz Velho† and Hélio Lopes*
*PUC-Rio - Pontifícia Universidade Católica do Rio de Janeiro
Brazil
†IMPA - Instituto Nacional de Matemática Pura e Aplicada
Brazil

***Abstract*—Pose estimation is a challenging task in computer vision that has many applications, as for example: in motion capture, in medical analysis, in human posture monitoring, and in robotics. In other words, it is a main tool to enable machines do understand human patterns in videos or images. Performing this task in real-time while maintaining accuracy and precision is critical for many of these applications. Several papers propose real time approaches considering deep neural networks for pose estimation. However, in most cases they fail when considering run-time performance or do not achieve the precision needed. In this work, we propose a new model for real-time pose estimation considering attention modules for convolutional neural networks (CNNs). We introduce a two-dimensional relative attention mechanism for feature extraction in pose machines leading to improvements in accuracy. We create a single shot architecture where both operations to infer keypoints and part affinity fields share the information. Also, for each stage, we use tensor decompositions to not only reduce dimensionality, but also to improve performance. This allows us to factorize each convolution and drastically reduce the number of parameters in our network. Our experiments show that, with this factorized approach, it is possible to achieve state-of-art performance in terms of run-time while we have a small reduction in accuracy.**

I. Introduction

Pose estimation is a challenging problem in computer vision with many real applications. Usually, the problem to be addressed involves estimating the 2D human pose, i.e., the anatomical keypoints or body "parts" of persons in images or videos [1]. Inferring the pose of multiple people in images presents a unique set of challenges, considering the number of people appearing in different scales, occlusion and also run-time complexity tends to grow with the number of people in the image [2]. If we find a way to leverage the performance for this task in real-time, it would be hugely beneficial for a large range of applications, such as motion capture for animation, robotics, understanding sign language, and many others.

The state-of-art OpenPose [2] presents an evolution of pose machines with a bottom-up approach for pose detection and a huge reduction in computational cost. However, it does not achieve the desired performance in terms of frames per second in real time, considering both versions of its architecture in limited hardware [1].Thus raising the need for a more efficient architecture or technique to reduce the processing time.

Convolutional Pose Machines are fully convolutional neural networks and there is evidence that a key feature behind the success of these methods is over-parameterization. It could help in finding a good local minimum, however it also leads to a large amount of redundancy [3]. Furthermore, models with a larger number of parameters have increased storage and are computationally intensive. Several papers focus on improving the efficiency of CNNs using tensor decompositions. Most of them consider each layer independently, where the kernel of a convolutional layer can be seen as a 4-dimensional matrix and decomposed in a set of low-rank approximations. On the other hand, we have hand-crafted decomposition methods that use pointwise and depthwise convolutions to improve performance, such as the Mobilenets [4], [5].

Another weakness of convolutional neural networks is that convolution operations consider only local neighborhoods thus missing global information [6]. Recently several papers propose the use of attention modules to leverage this problem, for example, the use of Squeeze-and-Excitation networks [7], Gather-Excite for feature analysis [8], and convolutional block attention modules (CBAM) [9]. They show consistent improvements in the result for image classification on ImageNet [10] and in the COCO dataset [11] across many different models and scales, proving the potential of this approach.

This paper focuses on techniques for leveraging the redundancy in the parameters of Convolutional Pose Machines following tensor decomposition models and introducing a new architecture with attention mechanisms, not only improving performance but also reducing redundancy for pose estimation tasks.

The paper is organized as follow. Section II discuss the related work. Section III describes our new method for 2D pose estimation. Section IV shows our results. Finally, Section V concludes the work and proposes future works.

II. Related Work

In this section, we present some of the works directly related to ours, divided in three categories: Pose Estimation; Tensor Decomposition; and Attention in Neural Networks.

A. Pose estimation

The problem of pose estimation is challenging, due to the enormous variation of color and shape combinations of people and environment in input images. The classical approach employs pictorial models, where each object in an image is modeled as a collection of individual parts in a deformable

configuration [12]. These models, however, presented problems when presented with unusual part configurations and occlusion. Recently, Ramakrishna et al. [13] proposed to use an inference machine framework for pose estimation, named *Pose Machine*. Their method consists on training a series of multi-class predictors for each part. Each predictor consists of multiple stages that receive the input image and the output of the previous stage as input, and results in a confidence map for the presence of a target body-part. Their approach allows for reliable interconnection between body-parts even for unusual configurations (if present in the training data), but still suffers when occlusions are present.

More recently, Wei et al. [14] proposed Convolutional Pose Machines (CPMs), leveraging the power of convolutional deep neural networks for the pose estimation task. This new approach benefits from learning long-range dependencies between image and multi-part cues of the original pose machines, while gaining the ability to learn features directly from the data [14]. CPMs are composed of a sequence of convolutional networks, each producing a bidimensional belief map of the location of each part. One of the issues raised by the authors is the handling of multiple people in close proximity. Since their work was focused on images, they did not consider cases of sequential video frames, where we can exploit temporal coherence to improve accuracy and computational performance.

Cao et al. [15] extended CPMs using Part Affinity Fields (PAFs) to encode the location and orientation of limbs on the image domain. This representation allows for the accurate association between parts even in the presence of multiple people. However, their approach is computationally demanding, where it is not possible to use in low power devices.

Recently, the work of Silva et al. [1] proposed an architecture for pose estimation based on the developments of OpenPose [15]. They factorized the hidden convolution layers of their CPM, where they modeled each convolution kernel as a single high-order tensor, and employ high-order singular value decomposition (HOSVD) to simplify the layers. They also propose a temporal coherence model to extrapolate the results between sequential video frames. Their results indicate a significant computational performance increase while maintaining accuracy and recall measures comparable to OpenPose. However, similar to OpenPose, the use of their model considering low power devices is impractical, due the computational demand and the model size. Also, they achieve a high performance due post processing steps with the use of Optical Flow and Kalman Filters.

Further, Papandreou et al. [16] propose a new botom-up model for pose estimation with a fully-convolutional architecture and a part-induced geometric embedding descriptor which associate semantic person pixels with their corresponding person instance. This model presents outstanding performance in real time considering the pose net, however, it has limitations considering accuracy, where problems as flicking and missing keypoints are not uncommon.

B. Tensor Decomposition

Tensor decomposition, analogously to matrix decomposition, is a way to express a tensor as a product of simpler, usually smaller tensors. There is a rising interest in exploring efficient architectures for Neural Networks, either for use in embedded device applications or their use in ubiquitous computing [4]. Hand-crafted methods can be considered analogous to tensor decomposition, for example, the MobileNet [4], [5] and Xception [17] which decompose convolutions using efficient depthwise and pointwise convolutions. On the other hand, several works have been dedicated to leveraging tensor decompositions to speed up computation or to reduce the number of parameters of convolutional neural networks. Most cases are focused on applying layer-wise decompositions, considering the kernel of each layer. A kernel of a 2D convolutional layer of a neural network can be described as a 4-dimensional tensor, where its dimensions are defined by the number of columns and rows, the number of channels of the input, for example, the RGB channels of an image, and the number of output channels. Kim et al. [18] propose using HOSVD to split a regular convolution into three others, drastically reducing the computation and model size.

1) Tensor Notation: A *tensor* can be seen as a high-dimensional matrix, i.e, with three or more dimensions. The order of a tensor $\mathcal{T}$ is the number of its dimensions [19]. As matrices' rows and columns, tensors have fibers, for example, a matrix column is a mode-1 fiber and a matrix row is a mode-2 fiber [19], [20].

2) Unfolding: Similar to matrix flattening we also have the Unfolding operation, where we stack the fibers of a tensor in a given way to obtain a matrix representation [19], [21]–[23].

3) Tensor Rank: A rank of a tensor $\mathcal{T}$ is the smallest number of rank-one tensors that generate $\mathcal{T}$ by computing their sum [19]. A rank-one tensor is a mode-N tensor where it can be seen as the outer product of N vectors [19], [20]. In other words, each element of $\mathcal{T} \in R^{I_1 \times I_2 \times ... I_N}$ is the product of the corresponding element-wise operation defined by Equation 1.

$$t_{i_1 i_2 ... i_n} = v^{(1)}_{i_1} v^{(2)}_{i_2} ... v^{(N)}_{i_N} \tag{1}$$

4) Mode-n multiplication: A multiplication of a matrix M by a tensor $\mathcal{T}$ is defined by Equation 2.

$$\mathcal{X} = (\mathcal{T} \times M)_{(I_1 \times I_2 \times ... \times I_{n-1} \times J_n \times I_{n+1} \times ... \times I_N)} = \sum_{i_n} t_{i_1 .... i_N} m_{j_n i_n} \tag{2}$$

5) Tensor Decompositon: The SVD decomposition can be generalized to High Order Tensors. It considers the orthonormal spaces associated with the different modes of a tensor [18]. The High Order SVD is defined in Equation 3 as follows:

$$M = C \times U^1 \times U^2 \times U^3 \tag{3}$$

where U^1, U^2, U^3 contains the 1-mode, 2-mode, and 3-mode singular vectors, respectively, related to the column space of M_{mode-1}, M_{mode-2}, and M_{mode-3} matrix unfoldings. C is a core tensor with orthogonality property [24].

C. Attention in Neural Networks

In recent years, there has been an increasing interest by the machine learning community in attention models, especially due to their success in capturing long-distance interactions. It was introduced by for the encoder-decoder in a neural sequence translation model by Bahdanau et. al [25]. In special, self-attentional Transformer architecture achieved state-of-the-art results applied to natural language process tasks as machine translation [26]. This architecture shows to be suitable for capturing long-distance relations between entities.

Considering feed-forward convolutional neural networks, Woo et al. [9] presented an attention module where, given an intermediate feature map, the module tries to infer attention maps along the color channel and spatial dimensions, and after the attention maps are multiplied by the input feature map for adaptive feature refinement.

More recently, Bello et al. [6] proposed 2D self-attention as a replacement for convolutions for image classification tasks, since self-attention captures global behavior more appropriately than convolutional layers, which are inherently local. Their results indicate that even though self-attention layers are successful in replacing convolutional layers for image classification, a combination of both techniques yields better results in a controlled environment [6]. However, this global form attends to all spatial locations of an input, limiting its usage to small inputs which typically require significant down sampling of the original image [27].

On the other hand, Squeeze-and-Excitation Networks (SE-Net) introduce an attention block for CNNs that improves channel interdependencies, not limiting the input size and presenting huge performance boost in accuracy when applied to existing architectures. Also there is almost no extra computational cost when using this kind of architecture. With its simple but effective idea of weight, each channel adaptively allows us to better interpret its outputs compared with other models, which do not have formal proof; for example, the transformer-based model presented by Bello et al. [6].

III. Method

Our method can be seen as an improvement for pose machines. The system takes, as input, a color image of size $w \times h$ and produces the 2D locations of anatomical keypoints for each person in the image considering a bottom up approach. As output we have the set of 2D confidence maps S of body part locations and a set of 2D vector fields L [2]. We developed a new architecture for the data refinement using SE-Net blocks between intermediary layers. With this approach we improve the model accuracy and after, we decompose convolutional layers aiming to reduce redundancy and improve processing time. Figure 1 represents the architecture of our model following the convolutional pose machines [14], where we refine the predictions over successive stages. We built a sequential model for the first 6 convolutional blocks of each stage, adding after each pair of convolutions, attention modules. The outputs of two consecutive convolutional blocks are concatenated, following an approach similar to DenseNet. The last 6 convolutional layers form 2 branches with 3 layers each, where the first is responsible for the 38 maps for part affinity fields and the second, for 18 feature maps for body keypoints. The output of the first branch is used as input to the second in a sequential approach.

After all stages, we have a post-processing stage similar to Openpose [15]. With N candidates for each body part generated, we create for each pair a bipartite graph. We have two outputs from our neural network: the heatmaps for keypoints candidates and affinity fields. We use a technique called Non-maximum suppression to identify the optimal position of each keypoint and reduce the space of search considering the heatmaps [1], [15]. With all possible candidates (points) obtained, finding the real connections between them is necessary. To find the right relationship, we use the affinity fields that were generated by the network. Affinity fields are nothing more than vector fields that we use to weight the connections between part candidates. An integral line is computed along the segment described by each pair of candidates given its weights [1], [15]. After following a matrix interpretation, the results are analyzed following an assignment problem. We use a variation of the Hungarian algorithm called the Jonker-Volgenant algorithm [28] to create a connection between two parts. This process is performed in each frame.

In the next subsections, we first briefly review the structure of the Attention block, the factorized convolutions, and then the detailed descriptions of our proposed modules and methods are presented.

A. Attention block

The original SE-Net proposed a "Squeeze-and-Excitation" block to adaptively highlight the channel-wise feature maps by modeling weights applied to each channel. It was proved that SE blocks bring significant improvements in performance for existing state-of-the-art CNNs at slight additional computational cost.

In the excitation step, we fully capture channel-wise dependencies. A simple gating mechanism with a sigmoid activation (S) is applied. Consider an input with ch channels. This mechanism is composed by two fully connected layers (FCL) where the first has $\frac{ch}{r}$ neurons and a LeakyReLU activation, and the second, ch neurons. Finally, the second FCL is followed by the sigmoid activation. The hyper parameter ratio r allows us to vary the capacity and computational cost of the SE blocks in the network and can be evaluated empirically. In the sequence, to generate the output, an element-wise multiplication between the result of the gating mechanism and data input is performed.

In a formal way, let $\mathcal{X} \in R^{H \times W \times ch}$ be the tensor corresponding to the input of attention block and $\mathcal{X}^{'} \in R^{H \times W \times ch}$ its output. In the squeeze operation, a global average pooling is performed on the input to generate the weights z, where each element of z is computed according to the Equation 4:

$$z_c = \frac{1}{H \times W} \sum_{i=0}^{H} \sum_{j=0}^{W} X_c(i,j), \tag{4}$$

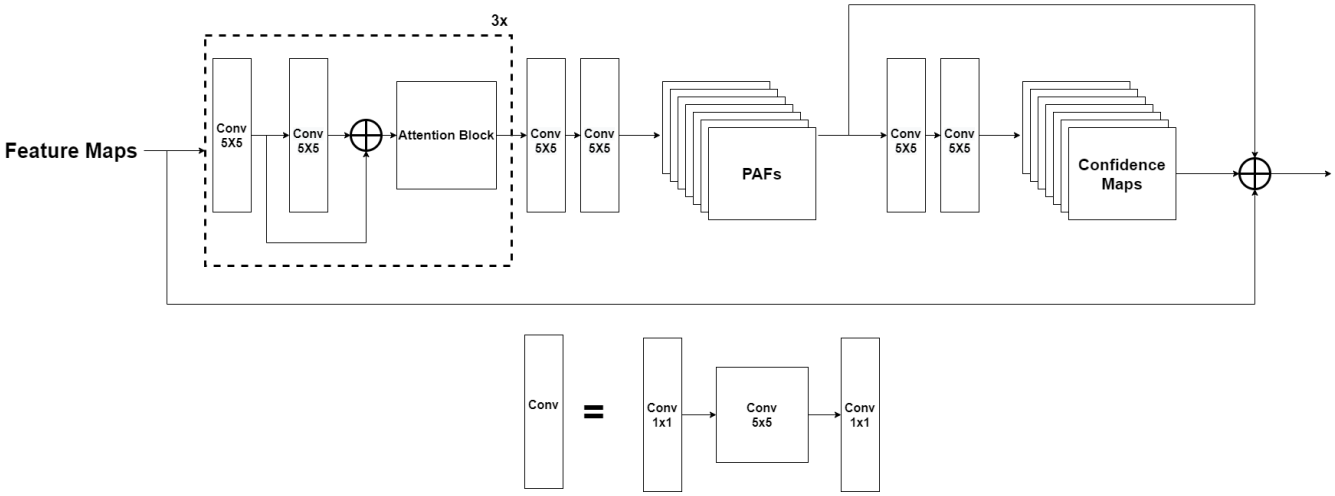

Fig. 1. In our CNN model, we have 3 blocks containing two convolutional layers and an attention block

where X_c corresponds to each channel of $\mathcal{X}$. After that, it is applied the excitation step, on which the information aggregated in the squeeze operation, is used in a operation which aims to fully capture channel-wise dependencies. A simple gatting mechanism with a sigmoid activation is applied as according to this Equation 5:

$$\alpha = sigmoid(W_2(\beta(W_1(z)))). \tag{5}$$

Here, $W_1 \in R^{\frac{ch}{r} \times ch}$ and $W_2 \in R^{ch \times \frac{ch}{r}}$ represent two fully connected layers and β a LeakyRelu activation. The hyperparameter ratio r allows us to vary the capacity and computational cost of the SE blocks in the network and can be evaluated empirically.

In the sequence, the weights α are re-scaled on the input $\mathcal{X}$, generating the output. To do that, a Hadamard product is computed, i.e., an element-wise multiplication between α and $\mathcal{X}$ as described in Equation 6:

$$\mathcal{X}^{'} = \alpha \circ \mathcal{X} \tag{6}$$

Similarly to Su et al. [29], we use spatial attention mechanism to adaptively highlight the task-related regions in the feature maps in addition to the channel-wise. Different from paying attention to the entire image, which can lead to focus on irrelevant features, spatial attention can enhance the overall results [29]. With the spatial input $\mathcal{X} \in R^{H \times W \times ch}$ and its output $\mathcal{X}^{'}$ with same dimensions, the spatial attention is generated by a pointwise convolution followed by a sigmoid activation. The spatial attention is represented by Equation 7:

$$\beta = sigmoid(W\mathcal{X}), \tag{7}$$

where W represents the pointwise convolution. Finally, the output is also re-scaled to achieve $\mathcal{X}^{'}$ according to Equation 8 [29]:

$$x^{'}_{i,j} = \beta_{i,j} * x_{i,j}, \tag{8}$$

which is an element-wise multiplication between the spatial elements of β and $\mathcal{X}$. In our architecture, each attention block is composed of spatial attention modules followed by channelwise SE-Net blocks, as we can see in figure 2.

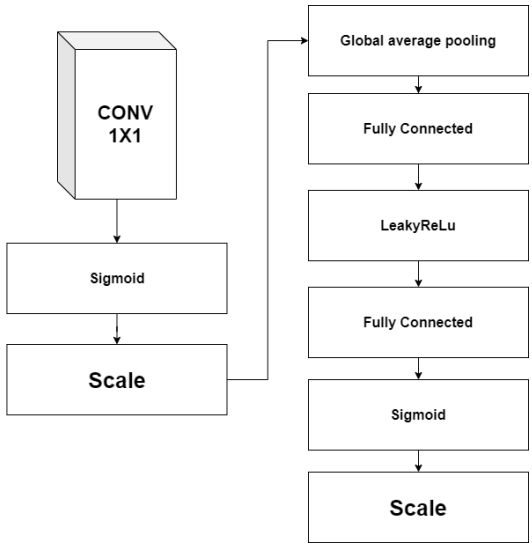

Fig. 2. The attention block used in our architecture. Here we first process spatial attention operations before the channel-wise.

B. Factorized Convolutions

Similarly to TensorPose [1], we apply tensor decompositions to neural networks aiming to factorize their kernels, creating approximations and improving the processing time for inference. We use this strategy as an exploratory way to model a factorized architecture, formed by pointwise convolutions combined with a regular convolution with reduced size. We aim to create a low rank approximation, where we model our architecture following an one-shot whole network compression scheme [18]. It consists of two steps: rank selection and tensor decomposition. We analyze the unfolding mode-3 and mode-4

of each layer's kernel tensor with global analytic variational Bayesian matrix factorization (VBMF) [18]. Our experiments show that an approximation of $1/3$ or $1/4$ of mode-3 and mode-4 of a tensor already presented effective results. We consider just the mode-3 and mode-4 due the fact that the mode-1 and mode-2 represent just the spatial dimension of kernel and they are quite small. The VBMF tries to infer a optimal low rank selection and with these values, after, we apply the Tucker decomposition on each kernel.

Let us consider a regular convolution which maps an input tensor into another with different size by successive operations as one can see in Equation 9:

$$conv(x, y, z) = \sum_{i} \sum_{j} \sum_{k} \mathcal{T}_{(i,j,k,z)} \mathcal{W}_{(x-i,y-j,k)}, \quad (9)$$

where $\mathcal{T}$ is a kernel of size $IJKZ$ and $\mathcal{W}$ an input tensor with size XYK, as we refer typically as an image, for example, with X and Y the image dimensions and K the number of channels. The mode-4 kernel tensor $\mathcal{T}$ can be seen as a kernel in a convolution layer. All operations for its decomposition can be written in the form described in Equation 10 [18]:

$$\mathcal{T}_{x,y,z,k} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \sum_{r_4=1}^{R_4} C_{r_1,r_2,r_3,r_4} U^1_{x,r_1} U^2_{y,r_2} U^3_{z,r_3} U^4_{k,r_4}, \quad (10)$$

where C is a core tensor of size $(R_1 \times R_2 \times R_3 \times R_4)$ and the U_* matrices are factor matrices of sizes $X \times R_1$,$Y \times R_2$,$Z \times R_3$, and $K \times R_4$, respectively.

We rewrite the Equation 10 to consider just the mode-3 and mode-4:

$$\mathcal{T}' = \sum_{r=1}^{R_3} \sum_{r=1}^{R_4} C_{i,j,r_3,r_4} U^k_{r_3}(k) U^z_{r_4}(z), \quad (11)$$

where U_{k,r_3} and U_{z,r_4} are the factor matrices of sizes $K \times R_3$ and $Z \times R_4$, respectively, and C is a core tensor of size (I, J, R_3, R_4). As a final result, we have 2 pointwise convolutions represented by the matrices U and 1 regular convolution with reduced space represented by the core tensor. This leads us to the convolutional block used in our architecture. Even using the VBMF, it was empirically verified that an approximation of $1/3$ or $1/4$ of original output size of the convolution operations already presented effective results considering the performance of our network.

C. Proposed Architecture and Experimental details

1) Architecture: As said before, our architecture is based on the concepts of convolutional pose machines and in this section we present the modifications in the architecture for feature extraction and refinements stages. For feature extraction, we remove the 12 blocks of convolution of a VGG-19 used in the original and replace by a modified architecture of a Mobilenet v2. This architecture has the advantages of being a state-of-the art mobile constrained network with a huge improvement in performance while maintaining the accuracy. Also we set a number of 6 stages for intermediary refinement in the iterative prediction.

2) Training: We trained our model over 120 epochs using Adam optimizer with learning rate varying from $8e^{-5}$ to $7.38e^{-6}$. Also we use as activation function LeakyRelu instead of Relu. Our cost function considers a gaussian kernel distance between the annotations and the inferred data for each body part and the affinty fields, which can be written as in equation 14.

$$f^i_{key} = \sum_{k=1}^{nkeys} \sum_{p} 1 - e^{-w(p)\left\|key^i_k - key^*_k\right\|_2}, \quad (12)$$

$$f^i_{paf} = \sum_{k=1}^{paf} \sum_{p} 1 - e^{-w(p)\left\|paf^i_k - paf^*_k\right\|_2}, \quad (13)$$

$$loss = \sum_{i=0}^{6} f^i_{key} + f^i_{paf} \quad (14)$$

The kernel distances in our functions were similar to the ones proposed by Cao et al. [2]. We weight the loss functions spatially to address a practical issue that some datasets do not completely label all people. Here, key and paf represents the inferred maps for each keypoints and part affinty field respectively and the w is a binary mask were $w(p) = 0$ when the annotation is missing at pixel p.

3) Human Pose estimation: In training and validation stages, we used 110000 and 5000 images respectively, with a maximum of 200 iterations per epoch. In COCO dataset [11], approximately 150000 people and 1.7 million labeled keypoints are available. Considering the inference, we used the COCO validation dataset to compare our modified model and the OpenPose. We use the official COCO evaluation metric called Object Keypoint Similarity (OKS), which measures of how close the predicted Keypoint is to the ground truth annotation. These measures metrics like Precision and Recall at 50% and 75% OKS (AP-50, AP-75, AR-50, AR-75) consider how close a prediction is to annotated data considering the Gaussian standard deviation.

IV. Results

In terms of runtime performance comparisons, we began with the test of the CNN processing. Considering just one image with 23 people, we compared our approach with OpenPose. The tests were performed in a GPU Nvidia RTX 2060 with 6GB of memory, where we vary the scale of the image tested by a factor of 0.5 and repeat each test 1000 times. Our network achieves a performance of almost 20 frames per second when a network resolution of 656×368. We also test a non factorized version of our network. In general, for the same resolution, it has achieved a performance of almost 12 frames per second. As we can see in Table I, in average, our approach has a better performance when compared with the OpenPose, considering our factorized version.

We also perform tests in CPU. As we can see in Table II, in average, our approach has a better performance, considering

Image Resolution	OpenPose	Ours
328 x 193	∼55,08 ms	∼ 45 ms
656 x 368	∼120.224 ms	∼ 51.26 ms
984 x 579	∼174,75 ms	∼ 80.72 ms
1312 x 772	∼320,18 ms	∼ 112 ms

TABLE I
CNN PROCESSING TIME FOR OPENPOSE AND OUR MODEL. WE VARY THE SCALE OF THE INPUT TESTED BY A FACTOR OF 0.5 CONSIDERING 4 IMAGE SCALES. WE DO THIS ONLY FOR THE NETWORK INPUT, THE FINAL RESULT IS SCALED IN THE ORIGINAL IMAGE SIZE.

frames per second, while the original OpenPose is unpractical to be used.

Device	OpenPose	Ours
AMD Ryzen 7 1700 3.5GHZ	0.3 FPS	13 FPS
Mac Pro Intel Core I7 2.7GHZ	0.1 FPS	6 FPS

TABLE II
CNN PROCESSING TIME FOR OPENPOSE AND OUR MODEL IN CPU.

Table III shows our results in contrast to OpenPose [2], [15] and AlphaPose [30]. In terms of precision and Recall, Alphapose has the best results; however, it has a different approach that uses a top-down strategy. It also is impractical to run this model in modest hardware systems. In a High-end GPU, the AlphaPose runs over only 20 FPS. Since our model is highly based on OpenPose and a bottom-up strategy, performance tests with AlphaPose are out of scope. Figure 3 shows a visual comparison between the OpenPose and our technique. We consider the original architecture of OpenPose proposed by Cao et al. [15]. Here, higher OKS means higher overlap between predicted Keypoints and the ground truth. As we can see, our model has a lower accuracy than OpenPose, which is mainly caused by our proposed simplification on the convolutional layers. However, this simplification promotes the advantage of having 9.3 times less operations [1] in these layers than the original OpenPose [1]. We have focused on real-time application and our model's use in low power devices, so this accuracy loss does not represent significant problems. In Figure 3, we can see that OpenPose presents a confusion between semantically similar parts of different persons in this frame, while our method does not find this pattern.

Model	AP 0.50	AP 0.75	AR 0.50	AR 0.75
AlphaPose	0.84	0.715	0.895	0.775
OpenPose	0.782	0.594	0.807	0.650
Ours	0.743	0.501	0.702	0.580

TABLE III
PRECISION AND RECALL FOR ALPHAPOSE, OPENPOSE AND OUR MODEL.

V. CONCLUSIONS

In this work, we proposed a novel deep neural network with a lightweight architecture, attention blocks, and tensor decomposition for pose estimation with improved processing time. We show in our experiments that we have significant improvements in performance, a requirement for real-time applications. We provided an efficient optimization in the CNN model when considering its use in modest GPU hardware and CPU. This is a result of our factorized convolutions involving the Tensor decomposition theory. In parallel to handcrafted approaches for factorized models, this can be considered a technique with great potential. Due to the factorized convolutions, our accuracy of the keypoints detection shows to be slightly smaller. However, our primary objective was the performance gain when considering FPS, maintaining a similar accuracy, which has been achieved. We explored the weakness of convolutional neural networks using attention mechanisms as an auxiliary method to focus on global information for our applications besides the local neighborhood. The redundancy in the parameters of Convolutional Pose Machines following tensor decompositions and attention mechanisms was significantly reduced. For the future, we plan to explore different attention module architectures and increase the accuracy. We also want to extend our work for 3D pose estimation.

REFERENCES

[1] L. J. S. Silva, D. L. S. da Silva, A. B. Raposo, L. Velho, and H. C. V. Lopes, "Tensorpose: Real-time pose estimation for interactive applications," *Computers & Graphics*, vol. 85, pp. 1 – 14, 2019.

[2] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields," in *arXiv preprint arXiv:1812.08008*, 2018.

[3] J. Kossaifi, A. Bulat, G. Tzimiropoulos, and M. Pantic, "T-net: Parametrizing fully convolutional nets with a single high-order tensor," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7822–7831.

[4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[6] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," *arXiv preprint arXiv:1904.09925*, 2019.

[7] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[8] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, "Gather-excite: Exploiting feature context in convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 9401–9411.

[9] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[11] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[12] 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.

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

[14] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4724–4732.

[15] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.

Fig. 3. Comparison between the OpenPose model and ours. The first figure (left) is generated by the OpenPose and the second by our model (right). As we can see, our model is slightly less accurate than the original work in particular cases, considering the exact position of a keypoint. Although, OpenPose presents a confusion between semantically similar parts of different persons in this frame. Also, our model do not consider all the feet keypoints.

[16] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. 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)*, 2018, pp. 269–286.

[17] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[18] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin, "Compression of deep convolutional neural networks for fast and low power mobile applications," *arXiv preprint arXiv:1511.06530*, 2015.

[19] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[20] S. Smith and G. Karypis, "Accelerating the tucker decomposition with compressed sparse tensors," in *European Conference on Parallel Processing*. Springer, 2017, pp. 653–668.

[21] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[22] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[23] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[24] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 2, pp. 179–192, 2010.

[25] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[27] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," *arXiv preprint arXiv:1906.05909*, 2019.

[28] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, no. 4, pp. 325–340, 1987.

[29] K. Su, D. Yu, Z. Xu, X. Geng, and C. Wang, "Multi-person pose estimation with enhanced channel-wise and spatial information," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5674–5682.

[30] H.-S. Fang, S. Xie, Y.-W. Tai, and C. Lu, "Rmpe: Regional multi-person pose estimation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2334–2343.

Fig. 4. Results containing viewpoint variation and occlusion, which are common characteristics in images. Images from COCO dataset.