Semantic graph attention networks and tensor decompositions for computer vision and computer graphics

Luiz Schirmer[1] and Hélio Lopes
Pontifical Catholic University of Rio de Janeiro (PUC-Rio)

Luiz Velho
National Institute of Pure and Applied Mathematics (IMPA)

Abstract—**This thesis proposes new architectures for deep neural networks with attention enhancement and multilinear algebra methods to increase their performance. We also explore graph convolutions and their particularities. We focus here on the problems related to real-time human pose estimation. We explore different architectures to reduce computational complexity, and, as a result, we propose two novel neural network models for 2D and 3D pose estimation. We also introduce a new architecture for Graph attention networks called Semantic Graph Attention.**

I. Introduction

Pose estimation is a challenging problem in computer vision with many real applications in areas including augmented reality, virtual reality, computer animation, and 3D scene reconstruction [1]–[5]. Usually, the problem to be addressed involves estimating the 2D human pose, i.e., the anatomical key points or body "parts" of persons in images or videos [6]. The use of architectures based on conventional convolutional neural networks can achieve high accuracy in this context; however, mistakes caused by occlusion and motion blur are not uncommon. Those models are computationally very intensive for real-time applications [5], [7].

Also, 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 considering its parameters [8]. Furthermore, models with a more significant number of parameters have increased storage and are computationally intensive. We focus on improving the efficiency of CNNs using tensor decompositions. We 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.

Another weakness of convolutional neural networks is that convolution operations consider only local neighborhoods, thus missing global information [9]. We propose a new 2D Convolutional Pose Machine following tensor decomposition models, and we introduce a new architecture with attention mechanisms [10], not only improving performance but also reducing redundancy for pose estimation tasks.

We also explore methods to infer the 3D human pose. Several projects use motion capture to produce computer-animated movies, games, medical applications, or sports analyses. However, in most of these applications, its use depends typically on multiple sensors and commercial systems. This is not only a costly technology but also depends on specialized hardware. Moreover, it is far from being accessible to most producers. To overcome these problems, several papers propose solutions using neural networks [11]–[14]. Considering this problem, to achieve state-of-the-art performance, we propose the use of Graph Convolutional Neural Networks, and we create a novel gating mechanism applied to Semantic Graph Convolutions(*SGCs*) [15] named *Semantic Graph Attention (SGAT)*. We enhance the analysis of global correlation, which is crucial for understanding human actions [16]. Our new layer can learn both channel-wise weights for edges, combine them with kernel matrices, and features inter-dependencies over channels. This work also proposes a novel neural network architecture used for 3D pose estimation from 2D joints. The main feature of this approach is our attention layer combined with Semantic Graph Convolutions. Given a 2D human pose as input, we predict the locations of its corresponding 3D joints in a 3D coordinate space. The proposed method achieves state-of-the-art performance for predicting the 3D human pose from their 2D skeleton (4.9% better than the previous works). Furthermore, it has 58% fewer parameters than the original SGC model. We evaluated our approaches in the following three datasets: Human 3.6M [17], COCO [18] and MPI-INF-3DHP [19].

In summary, the main contributions of this thesis are:

- A novel deep neural network with streamlined architecture and tensor decomposition for 2D pose estimation with improved processing time;
- A novel attention layer for semantic graph convolutions based on a simple but effective gating mechanism;
- A novel lightweight architecture for 3D human pose estimation based SGCs with performance enhanced by our attention layer.

1 - This work is related to the D.Sc. thesis entitled Semantic graph attention networks and tensor decompositions for computer vision and computer graphics, presented on 08/02/2021 at the Pontifical Catholic University of Rio de Janeiro.

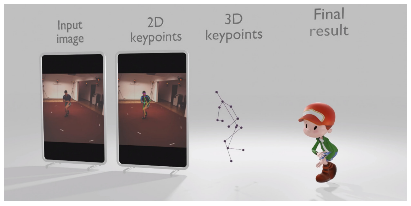

Fig. 1: An example of our 3D human pose algorithm with a single RGB camera. The 2D data is captured with our 2D pose machine and after processed by the 3D neural network. It can be used in applications such as computer animation and game controlling.

II. Pose Estimation and Motion Capture

Wei et al. [20] introduced the *Convolutional Pose Machines* (*CPMs*) that combines the advantages of convolutional neural networks with pose machines. *CPMs* consist of a sequence of convolutional layers that repeatedly produce 2D confidence maps for the location of each body part. Cao et al. [6] proposed an efficient method for 2D multi-person pose estimation with high accuracy on multiple public datasets, extending the original *Convolutional Pose Machines*. We use the concepts from Cao et al. to create our 2D pose estimation model.

Following the success of 2D pose estimation models, several papers propose an end-to-end model to predict the 3D human pose from given in the wild images. Mehta et al. [13] present the first real-time method to capture the 3D pose in a stable, temporally-consistent manner using a single RGB camera, named *VNect*. Their formulation uses a regression for 2D to 3D projection in real-time and creates a kinematic skeleton fitting method for coherent kinematic analysis. *XNect* [14] is an evolution of this work that also predicts the 3D pose of Humans and even infer the bones rotations. They present a real-time approach for multi-person 3D motion capture at over 30 fps. They both present complex frameworks to generate 3D poses and motion capture information. Our model focuses on a lightweight approach to generate 3D poses from 2D data, which is similar to the stage 2 from *XNect*, but surpass its results considering the error metric of this stage.

In a different approach, Martinez et al. [11] propose a simple feed-forward neural network that receives the 2D joint locations and predicts 3D positions. They “lift” the ground-truth 2D joint locations to 3D space. However, this also has limitations such as it does not maintain the bone proportion for all bodies. Zhao et al. [12] has expanded this work by using *Semantic Graph Convolutions*. Their architecture can be also extended to use attention modules and to reduce the projection error. Our approach reduces the reprojection error and network complexity using the attention layer, a way to model inter-dependencies in Semantic Graph Convolutions.In contrast with these previous works, we aim to present a lightweight framework for computer animation using human pose estimation. For this purpose, our models does not need any specialized hardware or even high-end GPU configurations.

III. Methods

A. 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 [21]. Hand-crafted methods can be considered analogous to tensor decomposition, for example, the MobileNet [21], [22] and Xception [23] 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. [24] 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 [25]. 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 [25], [26].

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 [25], [27]–[29].

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 [25]. A rank-one tensor is a mode-N tensor where it can be seen as the outer product of N vectors [25], [26]. In other words, each element of $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \dots I_N}$ is the product of the corresponding element-wise operation defined by Equation 1.

$$t_{i_1 i_2 \dots i_n} = v_{i_1}^{(1)} v_{i_2}^{(2)} \dots v_{i_N}^{(N)} \tag{1}$$

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

$$\mathcal{X} = \sum_{i_n} t_{i_1 \dots . i_N} m_{j_n i_n} \tag{2}$$

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 [24]. 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 [30].

B. Factorized Convolutions

We apply tensor decompositions to neural networks [31] aiming to factorize their kernels, create approximations, and improve inference processing time. We propose a new architecture based on the concepts of Convolutional Pose Machines. For feature extraction, we remove the 12 blocks of convolution of a VGG-19 used in the original and replace them with a modified architecture of a Mobilenet v2 [22]. This approach improves performance while maintaining accuracy. Also, we set a number of 6 stages for intermediary refinement in the iterative prediction. The implementation details where described in previous papers [31], [32].

To improve performance, at first, we train our neural network normally and after decompose its kernels. 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 a one-shot whole network compression scheme [24]. It consists of two steps: rank selection and tensor decomposition. We analyze the unfolding mode-3, and mode-4 of each layer's kernel with global analytic variational Bayesian matrix factorization (VBMF) [24]. 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 mode-3 and mode-4 because mode-1 and mode-2 represent just the spatial dimension of the kernel and are pretty small. The VBMF tries to infer an optimal low-rank selection, and with these values, 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 4:

$$conv(x,y,z) = \sum_i \sum_j \sum_k \mathcal{T}_{(i,j,k,z)} \mathcal{W}_{(x-i,y-j,k)}, \quad (4)$$

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 5 [24]:

$$\mathcal{T}_{x,y,z,k} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \sum_{r_4=1}^{R_4} C_{r_1,r_2,r_3,r_4} U^1_{x,r_1} U^2_{y,r_2} U^3_{z,r_3} U^4_{k,r_4}, \quad (5)$$

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 5 to consider just the mode-3 and mode-4:

$$\mathcal{T}' = \sum_{r=1}^{R_3} \sum_{r=1}^{R_4} C_{i,j,r_3,r_4} U^k_{r_3}(k) U^z_{r_4}(z), \quad (6)$$

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. After the decomposition, we fine-tune our network by 50 epochs.

C. Convolutional Graph Networks (GCNs)

In this section we presents the main mechanisms of our approach to create neural networks for 3D Human Pose Estimation. We choose to model the problem of inferring a 3D skeleton as a graph, and we develop a Graph Convolution Network to solve this task. Following principles of regular *Convolution Neural Networks*, a *Graph Convolution Network* can be considered a way to deal with arbitrary graph structures [33]–[35]. This is highly related to our approach to analyze human pose as a structured graph.

Convolutional Graph Networks (GCNs) share the filter parameters in the graph. The *GCNs* training stage consists in learning structures capable of processing graph information from the node matrix $X \in R^{N \times D}$ (N nodes containing D features) and the adjacency matrix $A \in R^{\|N\| \times \|N\|}$ [34], [35].

Each layer is a non-linear function as follows:

$$H^{(l+1)} = f(H^l, A), \quad (7)$$

with H being the output of each layer and $H^0 = X$. We can rewrite this function following:

$$f(H^l, A) = \sigma(AH^lW^l), \quad (8)$$

where σ represents the *Relu* activation function and W the weight matrix of the network layer. There are some limitations to this approach because the multiplication by the matrix A would only consider features from the neighborhood, but not from the node itself. This problem is addressed by adding the identity matrix to A ($A' = A + I$).

Furthermore, A' should be an unitary matrix to avoid scaling the features vector. We reach it by normalizing A' rows using the *Normalized Laplacian Matrix* $D^{-1/2}AD^{-1/2}$, where D^{-1} is the inverse of the diagonal matrix with the degree of the graph nodes. Substituting in the equation 8, we have:

$$f(H^l, A) = \sigma(D^{'-1/2} A^{'} D^{'-1/2} H^l W^l). \quad (9)$$

There are two clear disadvantages to make the graph convolution considering a regression to work on nodes with arbitrary topologies. The first one is the kernel matrix W is shared by all the edges. As a result, the relationships of neighboring nodes, i.e. internal structure, are not well explored. This is also a limiting factor because the receptive field is fixed with ones [12], the second disadvantage.

A *CNN* with a convolution kernel of size $k \times k$ learns k^2 different transformation matrices. The transformation matrices decode features within the kernel spatial dimension. This formulation can be approximated by learning a vector of weights $\vec{a}_i$ for each position of a pixel in an image or a graph node, and then combining them with a shared transformation matrix W [12].

We can transform an image to a graph by considering the pixels as nodes, and two neighbor pixels being connected by an edge (8-connect neighborhood). So, a kernel size k affects all pixels distant less than $d = \frac{k-1}{2}$. We can extend this approach for *GCNs* by considering that a convolution in a graph using a kernel of size d affects all nodes in a neighborhood of size d [12].

GCNs cannot handle convolution problems directly since it shares the same weight matrix for all edges. Furthermore, the filters just operate in a one step neighborhood. As a solution, Zhao et al. [12] propose to add the weight matrix M to the graph convolution process as follows:

$$f(H^l, A) = \sigma(\phi(A' \odot M)H^l W^l), \tag{10}$$

where the matrix M is a parameter to be learned on the network, σ is a *ReLU* activation and ϕ is a *softmax* function that normalizes the entries of each node, $\odot$ is an element-wise multiplication (*Hadamard*) that returns m_{ij} if $a_{ij} = 1$ or negative values with large exponents after the *softmax*. In this approach, A works like a mask that forces this to the i-th node in the graph. Also as proposed by Zhao et al. [12], this formulation can be extended to consider multiple channels as in traditional convolutions. In our experiments, we use *PreLU* instead of *ReLU* activation because it shows a slightly performance improvement.

D. Attention block for SGC

In this section we present our contribution for graph neural networks: we propose a method to enhance the accuracy of Semantic Graph Convolutions called Semantic Graph Attention. The intuition here is to allow the neural network to perform feature recalibration, i.e., to emphasize more relevant features and suppress less meaningful information. In other words, to give weights to the features over channels of SCGs, in a similar way to the SE-NET [10]. This also can be related to *Global Context Networks* [36], when applying attention to graph networks. We aim to solve the issues of precision and computational complexity, considering both space storage and time complexity, from previous related works.

Considering the computation of features for each node, the idea of adding weights via element-wise multiplication is natural. We intend to identify inter-dependencies between node features. For this purpose, we propose the following gating mechanism for each channel after a regular *SGC*, where we learn a matrix of weights, presented in Equations 11 and 12:

$$g = A' \odot \phi(M_1)W_1^l H^l, \tag{11}$$

where g is composed by a *softmax* ϕ function over the entries of Matrix M_1. This hidden layer performs a dimensionality reduction that reduces drastically the input space by a factor r, where the kernel size of $W_1^l \in R^{\frac{C}{r} \times C}$. Here, C represents the number of features and r is defined empirically.The intuition behind this layer is similar to Principal Components Analysis. We perform a dimensionality reduction to a space that better represents the data given a new basis. Consider that our Graph Neural Network has input data contained in a 2D space. In the first layer, the neural network project the data into frequency space. The first block of our attention module evaluates features in frequency space and forces this neural network stage to consider the most relevant ones as in an orthogonal transformation. In a gating mechanism, the next block will use the data representation in this new space, expanding the data to the original size and given weights to the features.

Equation 12 represents the expansion back to the original input size:

$$s(g) = \alpha(A' \odot \phi(M_2)\sigma_1(g)W_2^l), \tag{12}$$

where kernels $W_2^l \in R^{C \times \frac{C}{r}}$, σ represent a *PReLU* function, α represents a *sigmoid* activation. In our experiments, we use a value $r = 16$, similarly to Hu et. al. [10]. With the output of function $s(g)$ for each channel, we perform an element-wise multiplication operation to give weights to the input data as in Equation 13:

$$H^{l+1} = H^l \circ s(g). \tag{13}$$

At the final process, the channels are also concatenated. Such a gating operation allows us to consider more relevant features after each convolution operation, and thus refine our regression process for the following pose estimation case. As we will see in our experiments, this formulation enhanced our neural network's overall performance and drastically reduced its complexity.

IV. 3D Pose Estimation framework

We propose a 3D human pose estimation framework. The method takes as input a 2D human skeleton detected by our 2D Pose Machine. However, we can use any way to calculate these 2D joints from a single RGB image.The framework starts by calculating the 2D keypoints for each person in a given image using a 2D neural network. Our 3D model only needs the 2D keypoints as input. It makes our architecture flexible and not dependent on a specific 2D pose model to generate keypoints. Afterward, our neural network exploits the power of the semantic graph convolution with a gating mechanism. Our model has an input layer with an *SGC* followed by batch normalization and a *PReLU* activation. The building blocks of our network's internal layers are composed of two *SGC* layers, also followed by batch normalization and *PReLU* activation. The output of the second *SGC* layer is used as the input for our gating mechanism. This is repeated twice, and the blocks also use residual connections. We consider 128 channels for 16 graph nodes in the internal layers, where each node represents a human keypoint. The output layer comprises an *SGC* layer with the 16 nodes and the 3D positions as output data. Our 3D network model was trained over 100 epochs, using the *Adam optimizer* with a learning rate of $1e-3$, rate decay of

0.5, and batches of size 64. We also use the *Xavier* normal function to initialize the weights of each layer. Furthermore, we use a function based on the Mean Squared Error as our loss function. More details about the implementation can be found in the full thesis text.

V. Results

In terms of runtime performance comparisons for our 2D pose machine, 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 varied the scale of the image tested by a factor of 0.5 and repeated each test 1000 times. Our network achieves a performance of almost 20 frames per second with a network resolution of 656×368. We also test a non-factorized version of our network. In general, it has achieved a performance of almost 12 frames per second for the exact resolution. As we can see in Table I, on average, our approach has a better performance when compared with the OpenPose, considering our factorized version.

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.

We also perform tests in the CPU. On average, our approach has better performance, considering frames per second, while the original OpenPose is unpractical to be used, as we can see in Table II. We evaluate the precision and recall of our 2D model in COCO dataset [18], where we achieve an accuracy of 0.743 and recall 0.702.

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.

We also compare our 3D pose estimation network with state-of-the-art approaches. We first evaluate different design choices on a separate validation set for Human3.6, and then, we use the best choice to compare to SGC [12] and Martinez et al. [11]. We compare our model with the state-of-art approaches for 2D joints to 3D pose regression, following two configurations: with and without the attention layer. Table III reports the result.

Our technique outperforms the state-of-the-art *SGC*, for 3D pose regression, [12] by 4.9%. Also, our model with attention layers surpasses the model only with regular *SGCs* (without attention) by almost 10%. It is noteworthy that our approach has much fewer parameters, meaning that using the attention module drastically reduces the network's computational complexity and improves the overall performance. We have approximately 58% fewer parameters than the baseline *SGC* [12] and 95% fewer parameters than the model from Martinez et al. [11].

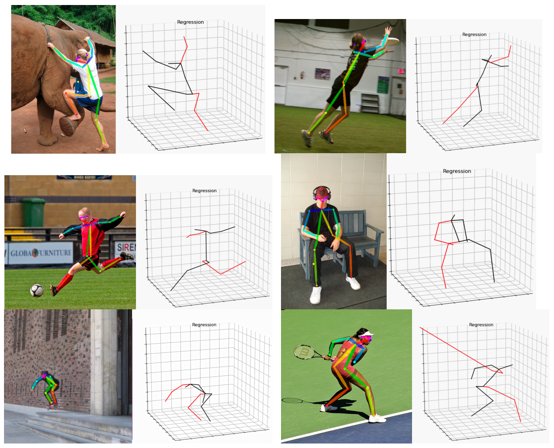

Fig. 2: Visual results of our method on in-the-wild images from COCO dataset [18] . In most cases, our technique can effectively predict 3d joints in different situations. Small errors can be seen considering the image scale and camera projection. In the last row, in an example with self-occlusion, our model cannot predict data from incomplete data.

We evaluated our 3D unprojection model following the dataset Human 3.6M. Table IV shows the result using 2D ground-truth of Human3.6M, our CPM, and Stacked HourGlass predictions for testing. We consider our model trained and tested with ground-truth data as an upper bound of our method since it uses only 2D ground truth (GT) as the input.

Consider that most other methods have sophisticated frameworks [37]–[39] or learning strategies [39], [40]. It was expected that they have better performance, including ground truth, than our approach. However, this is not true. Our model surpasses most of the previous works, considering the Mean Per Joint Position Error (*MPJPE*), proving the potential of the attention layer. The tests consider each action of the motion capture dataset. Table IV shows the error in millimeters for each step following the MPJPE. In Table V, we compare the 3D pose output on the MPI-INF-3DHP dataset [19] using the 3D Percentage of Correct Keypoints (3DPCK) (higher is better) and MPJPE. We prove the robustness of our method. Again, note that most of these methods are built over sophisticated frameworks and can predict multi-person poses and shapes. As we can see when considering our 2D CPM detection, our performance is reduced but still competitive.

Model	# of Parameters	MPJPE (mm)
SGC [12]	0.43 M	43.8
Martinez et al. [11]	4.29 M	45.5
Ours without attention (2 blocks and 128 ch)	0.16 M	46.71
Ours with attention (2 blocks and 128 ch)	0.18 M	**41.75**

TABLE III: 3D pose regression errors and the parameter numbers of our networks with different settings on Human3.6M. For each technique, we use the 2D ground truth data for the training and evaluation.

Protocol	Direct	Discuss	Eating	Greet	Phone	Photo	Pose	Purch.	Sitting	SittingD	Smoke	Wait	WalkD	Walking	WalkT	Average
Martinez et al. ICCV'17 [11]	51.8	56.2	58.1	59	69.5	78.4	55.2	58.1	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9
Yang et al. CVPR'18 [37]	51.5	58.9	50.4	57.0	62.1	65.4	49.8	52.7	69.2	85.2	57.4	58.4	43.6	60.1	47.7	58.6
Mehta et al. SIGGRAPH'17 [13]	62.6	78.1	63.4	72.5	88.3	93.8	63.1	74.8	106.6	138.7	78.8	73.9	82.0	55.8	59.6	80.5
Hossain & Little ECCV'18 [38]	48.4	50.7	57.2	55.2	63.1	72.6	53.0	51.7	66.1	80.9	59.0	57.3	62.4	46.6	49.6	58.3
Zhao et al CVPR'19 [12]	**37.8**	49.4	37.6	40.9	45.1	**41.4**	**40.1**	48.3	50.1	**42.2**	53.5	44.3	40.5	47.3	39.0	43.8
Pavllo et al. CVPR'19 [39]	45.1	47.4	42.0	46.0	49.1	56.7	44.5	44.4	57.2	66.1	47.5	44.8	49.2	32.6	34.0	47.1
Dabra et al ECCV'18 [40]	44.8	50.4	44.7	49.0	52.9	43.5	45.5	63.1	87.3	51.7	61.4	48.5	**37.6**	52.2	41.9	52.1
Our Model (CPM)	52.3	62.8	60.4	62.14	87.73	79.76	58.33	60.44	85.42	88.64	69.82	64.69	66.67	52.92	55.19	69.5
Our Model (GT)	38.07	**44.83**	**36.68**	**38.14**	**41.74**	49.73	40.82	**38.02**	50.43	59.19	**41.91**	**40.78**	41.30	**30.78**	**33.70**	**41.75**

TABLE IV: Results under Protocol of Mean Per Joint Position Error on Human3.6M (no rigid alignment in post-processing). We show the results of our model (trained and tested with ground truth data(GT) and 2D predictions from a Convolutional Pose Machine (CPM). Note that, on average, our model surpasses the previous state-of-the-art approach. The results of all approaches are obtained from the original papers.

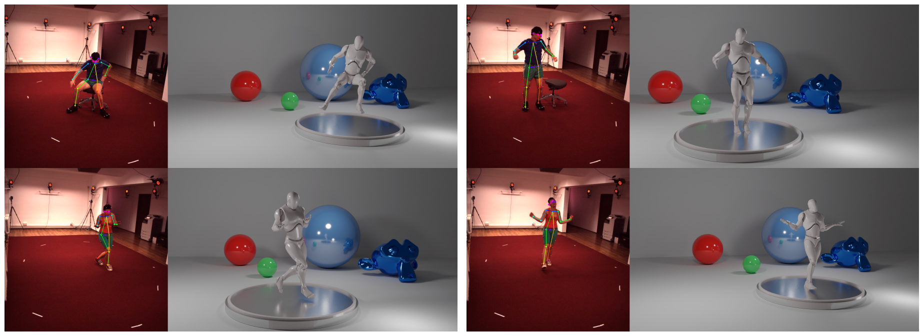

Fig. 3: Visual results of our method on Human3.6M [17]. As we can see, our method is robust but still has minor issues considering joint rotations.

Model	MPJPE (mm)	3D PCK
Vnect [13]	124.7	76.7
M3DHP [19]	117.6	75.7
Mehta [41]	122.2	75.2
Xnect (stage 2) [14]	98.4	82.8
Xnect (stage 3) [14]	115.0	77.8
Kanazawa [42]	124.2	72.9
Kundu [43]	103.8	82.1
VIBE [44]	96.6	89.3
Ours (CPM)	105.17	81.27
Ours (GT H3.6)	80.28	91.0
Ours (GT)	**76.39**	**92.0**

TABLE V: Comparison on the single person MPI-INF-3DHP dataset. Top part are methods designed and trained for single-person capture.The Xnect is multi-person method, however we evaluate only single person predictions. We tested models trained over the 2D ground-truth from Human3.6M, 2D predictions from a CPM, and ground truth data of MPI-INF-3DHP.

We also tested our model trained on 2D ground truth data for Human 3.6 on MPI-INF-3DHP test data. Moreover, considering runtime performance, our 3D network took, on average, 10 seconds to evaluate 1062 poses. The tests were performed in a GPU Nvidia RTX 2060 with 6GB of memory, where we repeated each test 1000 times. In terms of the number of parameters, our network has 0.18M, while the model proposed by Zhao et al. [12] has 0.43M.

A. Qualitative results

Figure 2 illustrates some results generated using images from COCO dataset [18]. Our model can accurately predict 3D poses from these images indicating that it effectively encodes relationships among body joints and can generalize the results to different situations. The input of the method is the 2D joints generated using our 2D Pose Machine. One of our limitations is when we are using data predicted by a *CPM*, if the 2D detector output fails to detect all body keypoints, it is impossible for our model to recover the missing information. Also, it is not uncommon to see images with occluded or incomplete poses for in the wild examples. Figure 3 shows the results of our technique applied on Human3.6M. In another approach, the input is generated by our pose estimation framework, and from the output, we created a *BVH* model to generate 3D animations.

VI. Publications

As results of this thesis, we published in several top tier conferences and journals including Eurographics [45], Computers & Graphics [31] and SIBGRAPI [32].Also, the presented framework were used in several projects that includes motion capture and digital humans [31], [45].

VII. Conclusion

We propose novel architectures for 2D and 3D Pose estimation. Also we propose a novel attention layer for graph convolutions. With this approach, we build a lightweight 3D human pose estimation framework to project 2D keypoints from the output of our 2D pose network in a 3D space. The use of tensor decomposition reduces the computational complexity and improve real time performance of our 2D pose estimation model. Also, for the 3D, the combination of *SGCs* with attention layers achieve state-of-the-art performance with 58% fewer parameters.

References

[1] R. Ranjan, V. M. Patel, and R. Chellappa, "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *IEEE TPAMI*, vol. 41, no. 1, pp. 121–135, 2019.

[2] V. A. Sindagi and V. M. Patel, "A survey of recent advances in cnn-based single image crowd counting and density estimation," *Pattern Recognition Letters*, vol. 107, pp. 3–16, 2018.

[3] L. Ge, "Real-time 3d hand pose estimation from depth images," Ph.D. dissertation, 2018.

[4] S. Schwarcz and T. Pollard, "3d human pose estimation from deep multi-view 2d pose," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 2326–2331.

[5] M. Lin, L. Lin, X. Liang, K. Wang, and H. Cheng, "Recurrent 3d pose sequence machines," in *Proceedings of the IEEE CVPR*, 2017, pp. 810–819.

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

[7] Y. Luo, J. Ren, Z. Wang, W. Sun, J. Pan, J. Liu, J. Pang, and L. Lin, "Lstm pose machines," in *Proceedings of the IEEE CVPR*, 2018, pp. 5207–5215.

[8] 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 CVPR*, 2019, pp. 7822–7831.

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

[10] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE CVPR*, 2018, pp. 7132–7141.

[11] J. Martinez, R. Hossain, J. Romero, and J. J. Little, "A simple yet effective baseline for 3d human pose estimation," in *Proceedings of the IEEE ICCV*, 2017, pp. 2640–2649.

[12] L. Zhao, X. Peng, Y. Tian, M. Kapadia, and D. N. Metaxas, "Semantic graph convolutional networks for 3d human pose regression," in *Proceedings of the IEEE CVPR*, 2019, pp. 3425–3435.

[13] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt, "Vnect: Real-time 3d human pose estimation with a single rgb camera," *ACM Transactions on Graphics*, 2017.

[14] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, M. Elgharib, P. Fua, H.-P. Seidel, H. Rhodin, G. Pons-Moll, and C. Theobalt, "Xnect: Real-time multi-person 3d motion capture with a single rgb camera," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 82–1, 2020.

[15] B. Zhao, X. Wu, J. Feng, Q. Peng, and S. Yan, "Diversified visual attention networks for fine-grained object classification," *IEEE Transactions on Multimedia*, vol. 19, no. 6, pp. 1245–1256, 2017.

[16] Q. Huang, F. Zhou, J. He, Y. Zhao, and R. Qin, "Spatial–temporal graph attention networks for skeleton-based action recognition," *Journal of Electronic Imaging*, vol. 29, no. 5, p. 053003, 2020.

[17] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE TPAMI*, vol. 36, no. 7, pp. 1325–1339, 2013.

[18] 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.

[19] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt, "Monocular 3d human pose estimation in the wild using improved cnn supervision," in *2017 Proceedings of 3DV*. IEEE, 2017, pp. 506–516.

[20] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proceedings of the IEEE CVPR*, 2016, pp. 4724–4732.

[21] 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.

[22] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE CVPR*, 2018, pp. 4510–4520.

[23] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE CVPR*, 2017, pp. 1251–1258.

[24] 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.

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

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

[27] 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.

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

[29] 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.

[30] 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.

[31] 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*, 2019.

[32] L. Schirmer, D. Lúcio, A. Raposo, L. Velho, and H. Lopes, "A lightweight 2d pose machine with attention enhancement," in *2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2020, pp. 324–331.

[33] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[34] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, 2016.

[35] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[36] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[37] W. Yang, W. Ouyang, X. Wang, J. Ren, H. Li, and X. Wang, "3d human pose estimation in the wild by adversarial learning," in *Proceedings of the IEEE CVPR*, 2018, pp. 5255–5264.

[38] M. Rayat Imtiaz Hossain and J. J. Little, "Exploiting temporal information for 3d human pose estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 68–84.

[39] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli, "3d human pose estimation in video with temporal convolutions and semi-supervised training," in *Proceedings of the IEEE CVPR*, 2019, pp. 7753–7762.

[40] R. Dabral, A. Mundhada, U. Kusupati, S. Afaque, A. Sharma, and A. Jain, "Learning 3d human pose from structure and motion," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 668–683.

[41] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, S. Sridhar, G. Pons-Moll, and C. Theobalt, "Single-shot multi-person 3d pose estimation from monocular rgb," in *2018, Proceedings of 3DV*. IEEE, 2018, pp. 120–130.

[42] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-end recovery of human shape and pose," in *Proceedings of the IEEE CVPR*, 2018, pp. 7122–7131.

[43] J. N. Kundu, S. Seth, V. Jampani, M. Rakesh, R. V. Babu, and A. Chakraborty, "Self-supervised 3d human pose estimation via part guided novel image synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6152–6162.

[44] M. Kocabas, N. Athanasiou, and M. J. Black, "Vibe: Video inference for human body pose and shape estimation," in *Proceedings of the IEEE/CVF CVPR*, 2020, pp. 5253–5263.

[45] L. J. S. Silva, D. L. S. da Silva, L. Velho, and H. Lopes, "An end-to-end framework for 3d capture and human digitization with a single rgb camera." in *Eurographics*, 2020, pp. 1–2.