POTTER: Pooling Attention Transformer for Efficient Human Mesh Recovery

Ce Zheng[1]*, Xianpeng Liu[2], Guo-Jun Qi[3,4], Chen Chen[1]
[1]Center for Research in Computer Vision, University of Central Florida
[2] North Carolina State University
[3] OPPO Seattle Research Center, USA [4] Westlake University
cezheng@knights.ucf.edu; xliu59@ncsu.edu; guojunq@gmail.com; chen.chen@crcv.ucf.edu

Abstract

Transformer architectures have achieved SOTA performance on the human mesh recovery (HMR) from monocular images. However, the performance gain has come at the cost of substantial memory and computational overhead. A lightweight and efficient model to reconstruct accurate human mesh is needed for real-world applications. In this paper, we propose a pure transformer architecture named POoling aTtention TransformER (POTTER) for the HMR task from single images. Observing that the conventional attention module is memory and computationally expensive, we propose an efficient pooling attention module, which significantly reduces the memory and computational cost without sacrificing performance. Furthermore, we design a new transformer architecture by integrating a High-Resolution (HR) stream for the HMR task. The high-resolution local and global features from the HR stream can be utilized for recovering more accurate human mesh. Our POTTER outperforms the SOTA method METRO by only requiring 7% of total parameters and 14% of the Multiply-Accumulate Operations on the Human3.6M (PA-MPJPE metric) and 3DPW (all three metrics) datasets. The project webpage is https://zczcwh.github.io/potter_page/.

1. Introduction

With the blooming of deep learning techniques in the computer vision community, rapid progress has been made in understanding humans from monocular images such as human pose estimation (HPE). No longer satisfied with detecting 2D or 3D human joints from monocular images, human mesh recovery (HMR) which can estimate 3D human pose and shape of the entire human body has drawn increasing attention. Various real-world applications such as gaming, human-computer interaction, and virtual reality (VR) can be facilitated by HMR with rich human body information. However, HMR from single images is extremely challenging due to complex human body articulation, occlusion, and depth ambiguity.

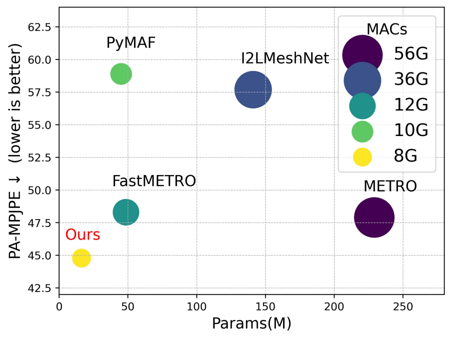


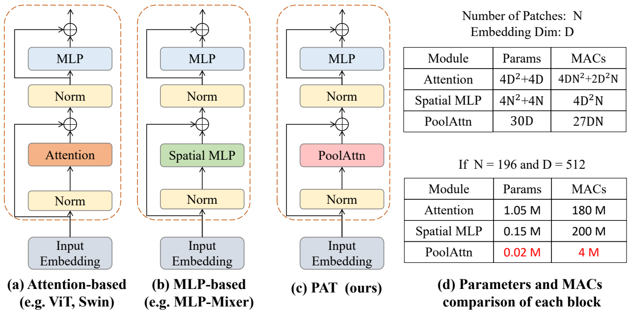
Figure 1. HMR performance comparison with Params and MACs on 3DPW dataset. We outperform SOTA methods METRO [17] and FastMETRO [3] with much fewer Params and MACs. PA-MPJPE is the Procrustes Alignment Mean Per Joint Position Error.

Recently, motivated by the evolution of the transformer architecture in natural language processing, Vision Transformer (ViT) [4] successfully introduced transformer architecture to the field of computer vision. The attention mechanism in transformer architecture demonstrates a strong ability to model global dependencies in comparison to the Convolutional Neural Network (CNN) architecture. With this trend, the transformer-based models have sparked a variety of computer vision tasks, including object detection [23, 26], semantic segmentation [2, 38], and video understanding [22, 31] with promising results. For HMR, the SOTA methods [3, 17] all utilize the transformer architecture to exploit non-local relations among different human body parts for achieving impressive performance.

However, one significant limitation of these SOTA HMR methods is model efficiency. Although transformer-based methods [17, 18] lead to great improvement in terms of accuracy, the performance gain comes at the cost of a substantial computational and memory overhead. The large CNN backbones are needed for [17, 18] to extract features first.

*Work conducted during an internship at OPPO Seattle Research Center, USA.

Then, computational and memory expensive transformer architectures are applied to process the extracted features for the mesh reconstruction. Mainly pursuing higher accuracy is not an optimal solution for deploying HMR models in real-world applications such as human-computer interaction, animated avatars, and VR gaming (for instance, SOTA method METRO [17] requires 229M Params and 56.6G MACs as shown in Fig. 1). Therefore, it is important to also consider the memory footprint and computational complexity when evaluating an HMR model.

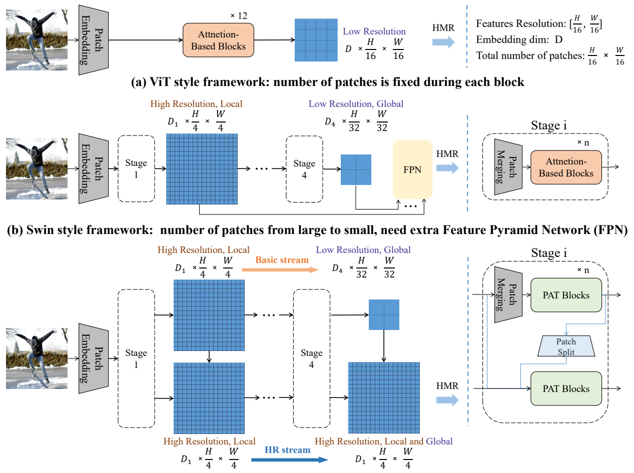


Module	Params	MACs
Attention	$4D^2+4D$	$4DN^2+2D^2N$
Spatial MLP	$4N^2+4N$	$4D^2N$
PoolAttn	30D	27DN

Module	Params	MACs
Attention	1.05 M	180 M
Spatial MLP	0.15 M	200 M
PoolAttn	0.02 M	4 M

Figure 2. Transformer blocks of different models. We suppose the number of patches (N) and the embedding dimension (D) for each block are the same when comparing the Params and MACs.

To bridge this gap, we aim to design a lightweight end-to-end transformer-based network for efficient HMR. Observing that the transformer blocks (attention-based approaches in Fig. 2 (a) and MLP-based approaches in Fig. 2 (b)) are usually computational and memory consuming, we propose a Pooling Attention Transformer (PAT) block as shown in Fig. 2 (c) to achieve model efficiency. After patch embedding, the image input becomes $X = [D, \frac{H}{p}, \frac{W}{p}]$, where D is the embedding dimension and the number of patches is $N = \frac{H}{p} \times \frac{W}{p}$ when patch size is $p \times p$. The input for transformer block is often written as $X_{in} = [N, D]$. To reduce the memory and computational costs, we design a Pooling Attention (PoolAttn) module in our PAT block. The PoolAttn consists of patch-wise pooling attention and embed-wise pooling attention. For the patch-wise pooling attention block, we preserve the patches' spatial structure based on the input $X_{in} = [D, \frac{H}{p}, \frac{W}{p}]$, then apply patch-wise pooling attention to capture the correlation of all the patches. For the embed-wise pooling attention block, we maintain the 2D spatial structure of each patch (without flattening to 1D embedded features). The input is reshaped to $X_{in} = [N, D_h, D_w]$, where $D_h \times D_w = D$ is the embedding dimension. The embed-wise pooling attention is applied to model the dependencies of the embedding dimensions in each patch. A detailed explanation is provided in Section 3.2. The Params and MACs comparison between the PoolAttn and conventional attention module or MLP-based module is shown in Fig. 2 (d). Thus, PAT can reduce the Params and MACs significantly while maintaining high performance, which can be utilized for efficient HMR.


Patch Embedding
× 12
Attention-Based Blocks
Low Resolution
$D \times \frac{H}{16} \times \frac{W}{16}$
HMR
Features Resolution: $[\frac{H}{16}, \frac{W}{16}]$
Embedding dim: D
Total number of patches: $\frac{H}{16} \times \frac{W}{16}$
(a) ViT style framework: number of patches is fixed during each block
High Resolution, Local
$D_1 \times \frac{H}{4} \times \frac{W}{4}$
Low Resolution, Global
$D_4 \times \frac{H}{32} \times \frac{W}{32}$
Stage 1
Stage 4
FPN
Stage i
× n
Patch Merging
Attention-Based Blocks
(b) Swin style framework: number of patches from large to small, need extra Feature Pyramid Network (FPN)
Basic stream
PAT Blocks
Patch Split
High Resolution, Local and Global
HR stream

(c) POTTER: maintains high-resolution while capturing both local and global correlations for HMR

Figure 3. The illustration in terms of patches during each stage in transformer architectures.

Equipped with PAT as our transformer block, the next step for building an efficient and powerful transformer-based HMR model is to design an overall architecture. The naive approach is to apply a Vision Transformer [4] (ViT) architecture as shown in Fig. 3 (a). The image is first split into patches. After patch embedding, a sequence of patches is treated as tokens for transformer blocks. But in ViT, patches are always within a fixed scale in transformer blocks, producing low-resolution features. For the HMR task, high-resolution features are needed because human body parts can vary substantially in scale. Moreover, ViT architecture focuses on capturing the global correlation, but the local relations can not be well modeled. Recently, Swin [21] introduced a hierarchical transformer-based architecture as shown in Fig. 3 (b). It has the flexibility to model the patches at various scales, the global correlation can be enhanced during hierarchical blocks. However, it also produces low-resolution features after the final stage. To obtain high-resolution features, additional CNN networks such as Feature Pyramid Network [19] (FPN) are required to aggregate hierarchical feature maps for HMR. Thus, we propose our end-to-end architecture as shown in Fig. 3 (c), the hierarchical patch representation ensures the self-attention can be modeled globally through transformer blocks with patch merge. To overcome the issue that high-resolution representation becomes low-resolution after patch merge, we propose a High-Resolution (HR) stream that can maintain high-resolution representation through patch split by leveraging the local and global features from the basic stream. Finally, the high-resolution local and global features are used for reconstructing accurate human mesh. The entire framework is also lightweight and efficient by applying our PAT block as the transformer block.

Our contributions are summarized as follows:

- We propose a Pooling Transformer Block (PAT) which is composed of the Pooling Attention (PoolAttn) module to reduce the memory and computational burden without sacrificing performance.
- We design a new transformer architecture for HMR by integrating a High-Resolution (HR) stream. Considering the patch's merging and split properties in transformer, the HR stream returns high-resolution local and global features for reconstructing accurate human mesh.
- Extensive experiments demonstrate the effectiveness and efficiency of our method – POTTER. In the HMR task, POTTER surpasses the transformer-based SOTA method METRO [17] on Human3.6M (PA-MPJPE metric) and 3DPW (all three metrics) datasets with only 7 % of Params and 14 % MACs.

2. Related Work

Since the HMR is one of the fundamental tasks in computer vision with a long history, here we focus on the more recent and relevant approaches. Readers can explore more detailed information about HMR in a recent and comprehensive HMR survey [29].

HMR from a single image. HMR has attracted increasing attention in recent years. Most of the HMR methods [8, 9, 39, 41, 43] utilize a parametric human model such as SMPL [24] to reconstruct human mesh by estimating the pose and shape parameters of the parametric model. Kolotouros et al. [14] present a graph convolution neural network to learn the vertex-vertex relations. They regress the 3D mesh vertices directly instead of the SMPL model parameters. SPIN [13] combines the regression and optimization process in a loop. The regressed output served as better initialization for optimization (SMPLify). Similarly, PyMAF [43] propose a pyramidal mesh alignment feedback where the mesh-aligned evidence is exploited to correct parametric errors. ProHMR [15] is a probabilistic model that outputs a conditional probability distribution using conditional normalizing flow. Dwivedi et al. [5] propose a differentiable semantic rendering loss to exploit richer image information about clothed people.

Transformers in HMR. Transformers are first proposed by [34] in the field of NLP. Inspired by the success of transformer's attention mechanism when dealing with token input, many researchers apply the transformer in various vision tasks such as object detection [1, 48], image classification [4, 21], segmentation [47], human pose estimation [44, 46], etc. METRO [17] is the first transformer-based method for HMR. After extracting the image features by CNN backbone, a transformer encoder is proposed to model vertex-vertex and vertex-joint interaction. Although METRO outperforms the previous SOTA methods by a large margin (more than 10 MPJPE on Human3.6M and 3DPW datasets), METRO requires substantial memory and computational costs to achieve this impressive performance. As an extended version of METRO, MeshGraphormer [18] further combines the graph convolutional network (GCN) with a transformer to model local and global interactions among mesh vertices and joints. It still incurs substantial memory and computational overhead. Zeng et al. [42] propose a Token Clustering Transformer (TCFormer) to merge tokens from different locations by progressive clustering procedure. However, the performance of TCFormer can not beat METRO and MeshGraphormer.

Efficient models for HMR. As mentioned above, [17, 18] pursue higher accuracy by sacrificing computational and memory efficiency. For real-world applications, model efficiency is also a key metric when evaluating HMR models, while less studied before. Although FastMETRO [3] reduces the computational and memory costs for the transformer part, it still relies on the heavy CNN backbone to achieve impressive performance. Another attempt is to reconstruct human mesh from a 2D human pose, which is proposed by GTRS [45]. A lightweight transformer model employing a lightweight 2D pose detector can reduce computational and memory costs significantly. However, the performance of GTRS is not comparable to the SOTA methods since it only uses the 2D pose as input, some information such as human shape is missing.

3. Methodology

3.1. Overall Architecture

We propose an end-to-end transformer network named POTTER for the HMR task as shown in Fig. 4. Following the general hierarchical transformer architecture (such as Swin [21]), there are four hierarchical stages of transformer blocks. After the patch embedding, the input image $X_{img} \in \mathbb{R}^{3 \times H \times W}$ is embedded to the input features of "stage 1" $X_{in1} \in \mathbb{R}^{D_1 \times \frac{H}{4} \times \frac{W}{4}}$, where D_1 is the embedding dimension of "stage 1", H and W are the height and width of the input image, respectively. The total number of patches is $\frac{H}{4} \times \frac{W}{4} = \frac{HW}{16}$ and the resolution of the features is $[\frac{H}{4}, \frac{W}{4}]$. After the transformer blocks modeling the attention, the output X_{out1} keeps the same size as the input $X_{in1} \in \mathbb{R}^{D_1 \times \frac{H}{4} \times \frac{W}{4}}$.

In the basic stream, we follow the Swin [21] style hierarchical architecture, a patch merging block is applied between two adjacent stages to build hierarchical feature maps, which reduces the number of patches between two adjacent stages. Thus, the output of "stage 2" becomes $X^{basic}_{out2} \in \mathbb{R}^{D_2 \times \frac{H}{8} \times \frac{W}{8}}$, the total number of patches is reduced to $\frac{H}{8} \times \frac{W}{8} = \frac{HW}{64}$, and the resolution of the features is decreased to $[\frac{H}{8}, \frac{W}{8}]$. This procedure is the same for "stage 3" and "stage 4", where the output is $X^{basic}_{out3} \in \mathbb{R}^{D_3 \times \frac{H}{16} \times \frac{W}{16}}$ and $X^{basic}_{out4} \in \mathbb{R}^{D_4 \times \frac{H}{32} \times \frac{W}{32}}$, respectively.

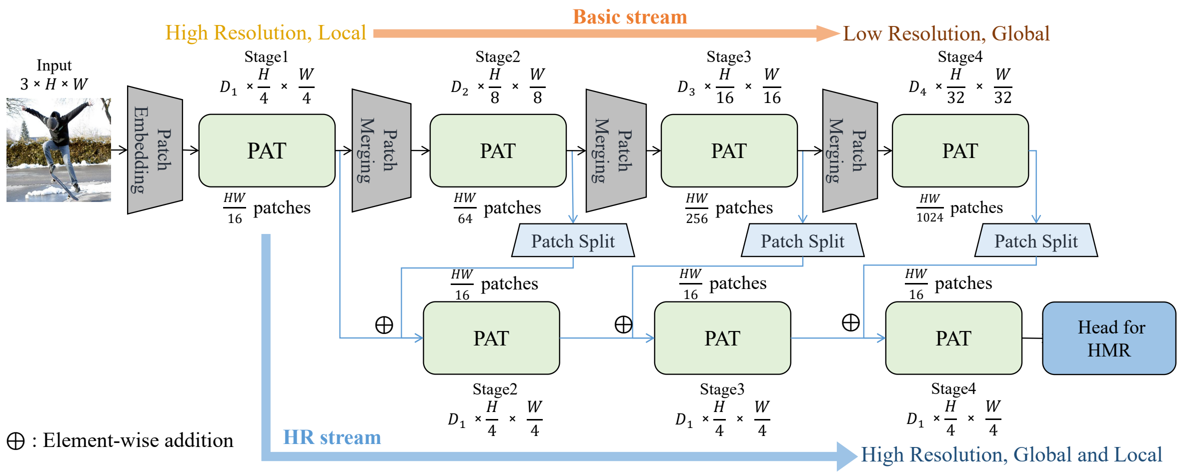

Basic stream
High Resolution, Local
Low Resolution, Global
Input
3 × H × W
Patch Embedding
Stage1
$D_1 \times \frac{H}{4} \times \frac{W}{4}$
PAT
$\frac{HW}{16}$ patches
Patch Merging
Stage2
$D_2 \times \frac{H}{8} \times \frac{W}{8}$
$\frac{HW}{64}$ patches
Stage3
$D_3 \times \frac{H}{16} \times \frac{W}{16}$
$\frac{HW}{256}$ patches
Stage4
$D_4 \times \frac{H}{32} \times \frac{W}{32}$
$\frac{HW}{1024}$ patches
Patch Split
Head for HMR
⊕ : Element-wise addition
HR stream
High Resolution, Global and Local

Figure 4. The overall architecture of our POTTER. PAT is our proposed Pooling Attention Transformer block. The basic stream of POTTER adopts hierarchical architecture with 4 stages [21], where the number of patches is gradually reduced for capturing more global information with low-resolution features ($\frac{H}{32} \times \frac{W}{32}$). Our proposed HR stream maintains the high-resolution ($\frac{H}{4} \times \frac{W}{4}$) feature representation at each stage. The global features from the basic stream are fused with the local features by patch split blocks in the HR stream. Thus, the high-resolution local and global features are utilized for the HMR task.

In the High-Resolution (HR) stream, a patch split block is applied between the basic stream and the HR steam, which splits the merged patches to maintain a high-resolution feature representation. Thus, the output for "stage 3" and "stage 4" are $X_{out3}^{HR} \in \mathbb{R}^{D_1 \times \frac{H}{4} \times \frac{W}{4}}$ and $X_{out4}^{HR} \in \mathbb{R}^{D_1 \times \frac{H}{4} \times \frac{W}{4}}$, respectively. The total number of patches during the HR stream is kept as $\frac{H}{4} \times \frac{W}{4} = \frac{HW}{16}$ and the resolution of the features is maintained as $[\frac{H}{4}, \frac{W}{4}]$.

3.2. The Design of Pooling Attention

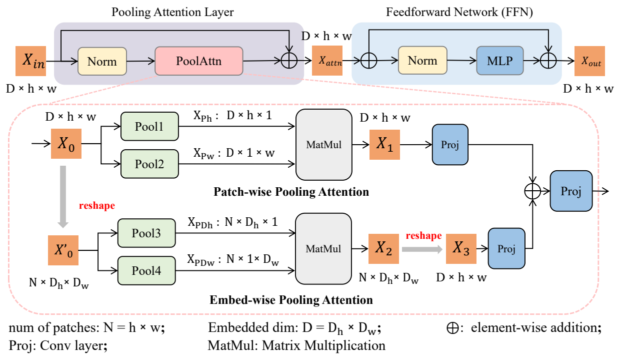


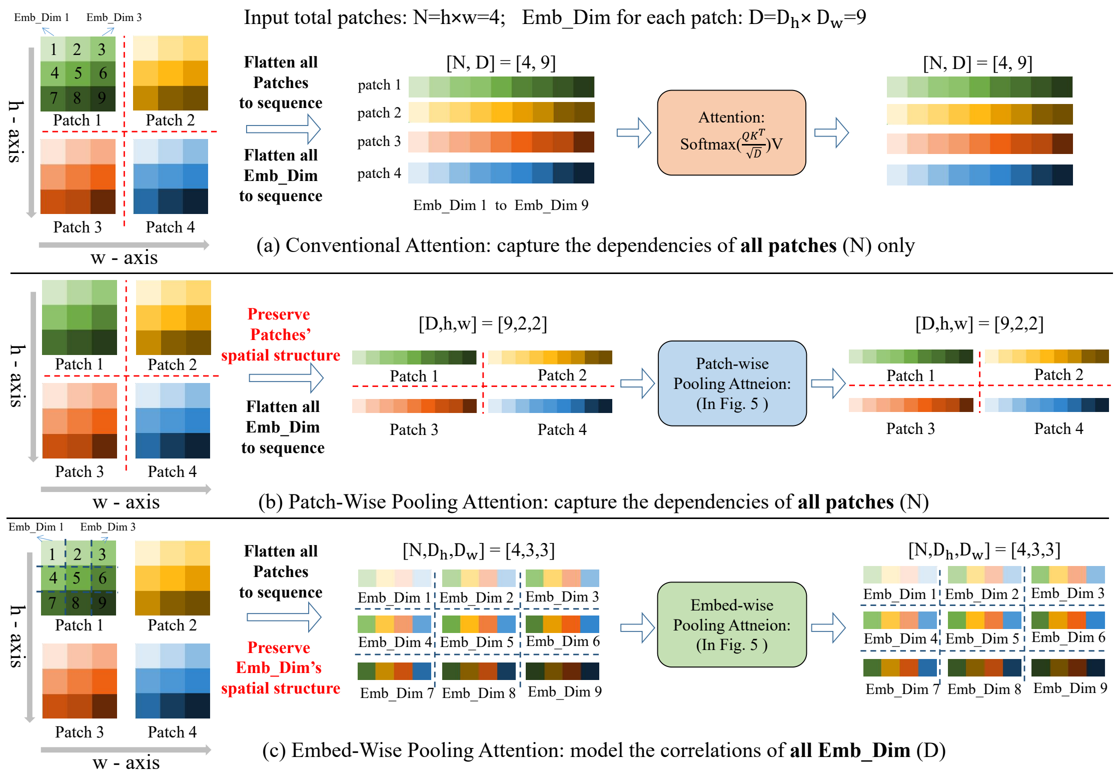
Figure 5. The Pooling Attention Transformer (PAT) block

Following the conventional transformer models, our Pooling Attention Transformer (PAT) block has a similar structure as shown in Fig. 2. Among various transformer-based architecture, PoolFormer block [40] is one of the closely related works to our PoolAtten block. In PoolFormer [40], a simple spatial pooling layer is used to replace the attention for patch mixing. Specifically, given the input $X_{in} \in \mathbb{R}^{D \times h \times w}$ where D is the embedding dimension and $h \times w$ is the number of patches, an average 2D pooling layer ($pool\_size = 3 \times 3, stride = 1, padding = 1$) is applied as patch mixer. The output $X_{out} \in \mathbb{R}^{D \times h \times w}$ keeps the same size as the input. Surprisingly, the performance of PoolFormer surpasses many complicated transformer-based models with much less computational and memory complexity.

Different from PoolFormer, we apply the pooling attention module to model the patch-wise attention and embed-wise attention while reducing computational and memory costs. The detailed structure of our PAT block is shown in Fig. 5. The Pooling Attention (**PoolAttn**) consists of patch-wise pooling attention and embedding dimension-wise pooling attention (embed-wise for short). Given the input $X_{in} \in \mathbb{R}^{D \times h \times w}$, where D is the embedding dimension and $h \times w$ is the number of patches, we first apply a Layer Normalization (LN) operation to normalize the input X_{in} as X_0, then, a PoolAttn module is used for calculating the attention based on the squeezed 2D features.

A graphical illustration of the procedures of the patch-wise pooling attention and the embed-wise pooling attention is presented in Fig. 6. We suppose the number of patches is $h \times w = 2 \times 2$ and the embedding dimension for each patch is $D = 3 \times 3 = 9$ in this simple illustration. *Unlike the conventional attention block that all the patches are reordered as a sequence and the embedded features of each patch are flattened, we preserve the patch's spatial structure before applying the patch-wise pooling attention. Similarly, for each patch, we maintain the 2D spatial embedded features representation (without flattening to 1D features) before applying embed-wise pooling attention.*

For the **patch-wise pooling attention**, we squeeze the X_0 along the $h-axis$ and $w-axis$ by two average pooling layers ($Pool_1$ and $Pool_2$), returning the $X_{Ph} \in \mathbb{R}^{D \times h \times 1}$ and the $X_{Pw} \in \mathbb{R}^{D \times 1 \times w}$. The matrix multiplication (Mat-



Input total patches: N=h×w=4; Emb_Dim for each patch: D=D_h× D_w=9
Emb_Dim 1
Emb_Dim 3
h - axis
w - axis
Patch 1
Patch 2
Patch 3
Patch 4
Flatten all Patches to sequence
Flatten all Emb_Dim to sequence
[N, D] = [4, 9]
patch 1
patch 2
patch 3
patch 4
Emb_Dim 1 to Emb_Dim 9
Attention: Softmax(QK^T/√D)V
(a) Conventional Attention: capture the dependencies of all patches (N) only
Preserve Patches' spatial structure
[D,h,w] = [9,2,2]
Patch-wise Pooling Attneion: (In Fig. 5)
(b) Patch-Wise Pooling Attention: capture the dependencies of all patches (N)
Preserve Emb_Dim's spatial structure
[N,D_h,D_w] = [4,3,3]
Emb_Dim 1
Emb_Dim 2
Emb_Dim 3
Emb_Dim 4
Emb_Dim 5
Emb_Dim 6
Emb_Dim 7
Emb_Dim 8
Emb_Dim 9
Embed-wise Pooling Attneion: (In Fig. 5)
(c) Embed-Wise Pooling Attention: model the correlations of all Emb_Dim (D)

Figure 6. The illustration of the patch-wise pooling attention and the embed-wise pooling attention. The conventional attention blocks not only reorder all the patches as a sequence, but also flatten the embedded features of each patch to 1D features. For patch-wise attention, we preserve the patch's spatial structure, then apply pooling attention to capture the correlations between all the patches. For the embed-wise attention, we maintain the spatial structure of 2D embedded features for each patch, then apply pooling attention to model the correlations between the embedding dimensions.

Mul) results of these 2D features X_{Ph} and X_{Pw} is the patch attention, named $X_1 \in \mathbb{R}^{D\times h\times w}$.

$$X_{Ph} = Pool_1(X_0), \quad X_{Ph} \in \mathbb{R}^{D\times h\times 1} \tag{1}$$

$$X_{Pw} = Pool_2(X_0), \quad X_{Pw} \in \mathbb{R}^{D\times 1\times w} \tag{2}$$

$$X_1 = MatMul(X_{ph}, X_{pw}) \tag{3}$$

Previous transformer-based methods only model the attention given the number of patches. We further exploit the correlation given the embedding dimensions. For the **embed-wise pooling attention**, we reshape the $X_0 \in \mathbb{R}^{D\times h\times w}$ as the $X_0' \in \mathbb{R}^{N\times D_h\times D_w}$, where $N = h \times w$ and $D = D_h \times D_w$. Similarly, we squeeze the X_0' along the $D_h - axis$ and $D_w - axis$ by two average pooling layers ($Pool_3$ and $Pool_4$). The squeezed 2D features $X_{PDh} \in \mathbb{R}^{N\times D_h\times 1}$ and the $X_{PDw} \in \mathbb{R}^{N\times 1\times D_w}$ are used for compute the embedding attention $X_2 \in \mathbb{R}^{N\times D_h\times D_w}$ by the matrix multiplication.

$$X_{PDh} = Pool_3(X_0'), \quad X_{PDh} \in \mathbb{R}^{N\times D_h\times 1} \tag{4}$$

$$X_{PDw} = Pool_4(X_0'), \quad X_{PDw} \in \mathbb{R}^{N\times 1\times D_w} \tag{5}$$

$$X_2 = MatMul(X_{PDh}, X_{PDw}) \tag{6}$$

Next, the embedding attention X_2 is reshaped back to $X_3 \in \mathbb{R}^{D\times h\times w}$. A projection layer (implemented by a CONV layer, same as in [40]) projects the sum of patch attention X_1 and the embedding attention X_3 as the PoolAttn's output.

$$X_3' = Proj_3(Proj_1(X_1) + Proj_2(X_3)) \tag{7}$$

Thus, the patch-wise pooling attention preserves the patch's spatial locations when capturing the correlations between all the patches. In the meantime, the embed-wise pooling attention maintains the spatial embedded features representation for each patch when modeling the correspondences of embedded features. Compared with the simple pooling module in PoolFormer [40], the PoolAttn module boosts the performance by a large margin without increasing memory and computational cost which is verified in Section 4.3.

With the PoolAttn module, one PAT block returns the output X_{out} given the block's input X_{in}, and can be expressed as:

$$X_{attn} = PoolAttn(LN(X_{in})) + X_{in} \tag{8}$$

$$X_{out} = MLP(LN(X_{attn})) + X_{attn} \tag{9}$$

Our PoolAttn operation significantly reduces the computational and memory costs of the PAT. The detailed parameters and MACs comparison of one PoolAttn module is shown in the Supplementary C. For instance, the input is with the shape of $[512, 16, 16]$ where the embedding dimension is 512 and the number of patches is $16 \times 16 = 196$. One attention block such as in ViT or Swin requires 1.1M params and 180M MACs, while our PoolAttn only requires 0.02M params (2%) and 4M MACs (2%) without downgrading the performance.

3.3. High-Resolution Stream

Different from the general hierarchical transformer architecture, the output of "stage 1" X_{out1} is also used by the HR streams as shown in Fig .4. For the HR stream, we aim to maintain the high-resolution features which are critical for the HMR task. Thus, we do not need a patch merging layer to merge the patches, which downgrades the high-resolution representation to the low-resolution representation. Instead, we aggregate the information from the basic stream to the HR stream. The patch split layer can convert the merged patches X_{out2}^{basic} back to the high-resolution patches.

During the basic stream (which is the architecture of Swin [21] equipped with our PAT block), the high-resolution and local features gradually become the low-resolution and global features. Thus, it is served as the general vision backbone (usually for the classification task). To reconstruct human mesh, global features from the basic stream are aggregated with the local features in the HR stream, returning high-resolution local and global features. The input of the transformer block of each stage during the HR stream can be expressed as:

$$X_{out\ i+1}^{HR} = PatchSplit(X_{out\ i}^{basic}) + X_{out\ i}^{HR} \tag{10}$$

Thus, the output for the i stage is $X_{out\ i}^{HR} \in \mathbb{R}^{D_1 \times \frac{H}{4} \times \frac{W}{4}}$, where the total number of patches is always $\frac{H}{4} \times \frac{W}{4} = \frac{HW}{16}$. Finally, the high-resolution features containing both local and global information are utilized for HMR.

4. Experiments

First, we evaluate the effectiveness of our PAT block for the vision transformer backbone, which means we train POTTER without the HR stream for the classification task on ImageNet. The pretrained weights on ImageNet can be used as a good initialization for downstream tasks such as HMR. Next, we train the entire POTTER (with HR stream, as shown in Fig. 4) for reconstructing human mesh.

4.1. Image Classification

The framework of POTTER without HR stream for the image classification (named POTTER_cls) is shown in Fig. 7 where PAT is utilized as the transformer block.

Dataset and Implementation Details: ImageNet-1k is one of the most commonly used datasets for computer vision tasks, which consists of 1.3M training images and 50K validation images within 1k classes.

We follow the same training scheme as [33, 40]. Our models are trained for 300 epochs with peak learning rate $lr = 2e^{-3}$ and batch size of 1024. AdamW [11] optimizer with cosine learning rate schedule is adopted. We use the same Layer Normalization as implemented in [33]. More details are provided in the Supplementary D.

Table 1. Performance of different types of models on ImageNet-1k classification task. All these models are only trained on the ImageNet1k training set. The top-1 accuracy on the validation set is reported in this table. More results comparisons are provided in the Supplementary D.

	Image Size	Params (M)	MACs (G)	Top-1 Acc ↑
ViT-L/16 [4]	224	307	190.7	76.5
RSB-ResNet-18 [37]	224	12	1.8	70.6
RSB-ResNet-34 [37]	224	22	3.7	75.5
PVT-Tiny [36]	224	13	1.9	75.1
MLP-Mixer-B/16 [30]	224	59	12.7	76.4
ResMLP-S12 [32]	224	15	3.0	76.6
PoolFormer-S12 [40]	224	12	1.8	77.2
POTTER_cls	224	12	1.8	**79.0**

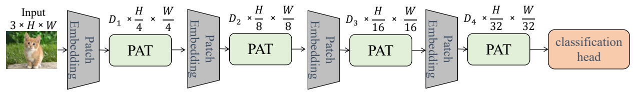

Input
$3 \times H \times W$
Patch Embedding
$D_1 \times \frac{H}{4} \times \frac{W}{4}$
PAT
Patch Embedding
$D_2 \times \frac{H}{8} \times \frac{W}{8}$
PAT
Patch Embedding
$D_3 \times \frac{H}{16} \times \frac{W}{16}$
PAT
Patch Embedding
$D_4 \times \frac{H}{32} \times \frac{W}{32}$
PAT
classification head

Figure 7. The framework for image classification.

Results: The performance of POTTER_cls on ImageNet is reported in Table 1. As a small network focusing on efficiency, POTTER_cls achieves superior performance with only 12M parameters and 1.8G MACs compared with the commonly-used CNN and transformer-based models. The CNN model RSB-ResNet-34 [37] (ResNet [6] trained with improved training procedure for 300 epochs) only achieves 75.5 % of the top-1 accuracy. The transformer-based methods with small params and MACs such as PVT-tiny [36], MLP-Mixer-S12 [30], and PoolFormer-S12 [40] obtain the top-1 accuracy around 75.1 % - 77.2 %. Our POTTER_cls outperforms them by achieving 79.0 % of top-1 accuracy with fewer Params and MACs. PoolFormer [40] demonstrates that using an extremely simple pooling layer without attention design as a patch mixer can still achieve highly competitive performance. We claim that an efficient pooling-based attention mechanism can further boost performance. With our PoolAttn design, POTTER_cls outperforms PoolFormer-S12 by 1.8% without increasing the memory and computational costs. *Thus, POTTER has the potential to enhance other tasks, including HMR.*

4.2. HMR

After pretaining on the ImageNet with POTTER_cls, we load the pretrained weight and train the entire network POTTER with the architecture illustrated in Fig. 4 for the HMR

Table 2. 3D Pose and Mesh performance comparison with SOTA methods on Human3.6M [7] and 3DPW [35] datasets. The “*” indicates that HybrIK [16] uses ResNet34 as the backbone and with predicted camera parameters. The comparison with [41] is provided in the Supplementary D.

					Human3.6M		3DPW		
	Model	Year	Params(M)	MACs(G)	MPJPE ↓	PA-MPJPE ↓	MPJPE ↓	PA-MPJPE ↓	MPVE ↓
CNN-based	HMR [9]	CVPR 2018	-	-	88.0	56.8	130.0	76.7	-
	GraphCMR [14]	CVPR 2019	-	-	-	50.1	-	70.2	-
	SPIN [13]	ICCV 2019	-	-	62.5	41.1	96.9	59.2	116.4
	VIBE [12]	CVPR 2020	-	-	65.6	41.4	82.9	51.9	99.1
	I2LMeshNet [27]	ECCV 2020	140.5	36.6	55.7	41.1	93.2	57.7	-
	HybrIK* [16]	CVPR2021	27.6	12.7	57.3	36.2	75.3	45.2	87.9
	ProHMR [15]	ICCV 2021	-	-	-	41.2	-	59.8	-
	PyMAF [43]	ICCV 2021	45.2	10.6	57.7	40.5	92.8	58.9	110.1
	DSR [5]	ICCV 2021	-	-	60.9	40.3	85.7	51.7	99.5
	OCHMR [10]	CVPR 2022	-	-	-	-	89.7	58.3	107.1
Transformer -based	METRO [17]	CVPR 2021	229.2	56.6	54.0	36.7	77.1	47.9	88.2
	GTRS [45]	ACM MM 2022	71.5	3.8	64.3	45.4	88.5	58.9	106.2
	TCFormer [42]	CVPR 2022	-	-	62.9	42.8	80.6	49.3	-
	FastMETRO-S [3]	ECCV 2022	32.7	8.9	57.7	39.4	79.6	49.3	91.9
	FastMETRO-L [3]	ECCV 2022	48.5	11.8	**53.9**	37.3	77.9	48.3	90.6
	POTTER		16.3	7.8	56.5	**35.1**	**75.0**	**44.8**	**87.4**

task. An HMR head HybrIK [16] is applied for generating the final human mesh.

Dataset and Implementation Details: Human3.6M [7] is one of the commonly used datasets with the indoor setting which consists of 3.6M video frames performed by 11 actors. Following [13, 17, 27, 42], there are 5 subjects (S1, S5, S6, S7, S8) used for training and 2 subjects (S9, S11) used for testing. 3DPW [35] a widely used outdoor dataset that contains 60 video sequences with 51k frames. Unlike Human3.6M which only provides the annotation of key joints, accurate 3D mesh annotation is available for the 3DPW. Thus, we evaluate the Mean Per Joint Position Error (MPJPE) [29] and MPJPE after Procrustes Alignment (PA-MPJPE) [29] on Human3.6M. For 3DPW, we report the MPJPE, PA-MPJPE, and e Mean Per Vertex Error (MPVE). Following previous work [3, 16–18], Human3.6M, MPI-INF-3DHP [25], COCO [20], and 3DPW are used for mixed training. We train our POTTER with the architecture illustrated in Fig. 4. The Adam [11] optimizer is utilized for training where the learning rate is 1×10^{-3} with a batch size of 32. All experiments are conducted on four NVIDIA RTX5000 GPUs with Pytorch [28] implementation. The 3D joint loss and vertex loss are applied during the training. More implementation details are provided in the Supplementary D.

Results: Table 2 compares POTTER with previous SOTA methods for the HMR task on Human3.6M and 3DPW datasets. As a pure transformer architecture, POTTER outperforms the previous transformer-based SOTA method METRO [17] (a hybrid CNN+transformer architecture) by showing significant computational and memory reduction. To be more specific, POTTER only requires 16.3 M Params and 7.8 G MACs (**7% of Params and 14% MACs compared with METRO**) to achieve the new SOTA performance. Although FastMETRO [3] reduces the computational and memory costs of METRO, it is still much more expensive than our POTTER with worse performance. *Without bells and whistles, POTTER demonstrates its exceptional performance both in terms of mesh recovery accuracy and model efficiency.*

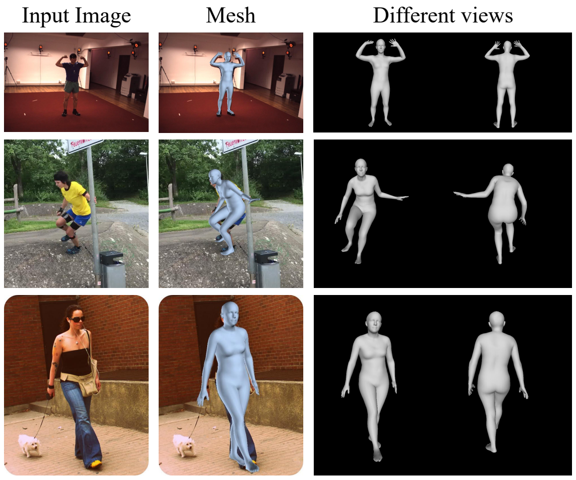


Figure 8. Mesh visualizations of POTTER. Images are taken from Human3.6M, 3DPW, and COCO datasets.

Mesh visualization: We show qualitative results of POTTER on Human3.6M, 3DPW, and COCO dataset in Fig. 8. POTTER can estimate reliable human poses and meshes given various input images. More qualitative results are presented in the Supplementary B.

4.3. Ablation Study

Effectiveness of Pooling Attention Design: First, we verify the effectiveness of the PoolAttn module proposed in Section 3.2. We report the top-1 accuracy using the different block combinations on the ImageNet classification task. “Pooling” denotes that only a pooling layer is used without any pooling attention design (which is the exact architecture of the PoolFormer [40]). The top-1 accuracy is 77.2 % with 11.9 M Params and 1.79 G MACs. When only applying patch-wise pooling attention, the performance is improved by 1.5 % with almost the same Params and MACs. Similarly, when only applying embed-wise pooling attention, the performance is also improved by 1.3 % with a slight increase in the Params and MACs. For our PoolAttn module which integrates two types of pooling attention, the performance is further boosted by 1.8 %, which demonstrates the effectiveness of the PoolAttn in the backbone.

Table 3. Ablation study of different modules in the transformer block on ImageNet classification.

Module	Params(M)	MACs(G)	Top-1 Acc ↑
Pooling	11.9	1.79	77.2
Patch-Wise Pooling Attention	12.0	1.82	78.7
Embed-Wise Pooling Attention	12.2	1.83	78.5
PoolAttn (Patch-Wise and Embed-Wise)	12.4	1.84	79.0

Table 4. Ablation study of different modules in the transformer block on 3DPW dataset for HMR.

			3DPW		
Module	Params(M)	MACs(G)	MPJPE ↓	PA-MPJPE ↓	MPVE ↓
Pooling	15.9	7.7	77.3	47.4	89.9
PoolAttn	16.3	7.8	75.0	44.8	87.4

Next, we evaluate the effectiveness of the PoolAttn on the HMR task. As shown in Table 4, replacing the pooling block (PoolFormer's architecture) by our PoolAttn, the performance of all metrics (MPJPE, PA-MPJPE, and MPVE) on 3DPW dataset can be improved. The increase in memory and computational cost can be neglected. Thus, PoolAttn design is critical for efficient HMR with high accuracy.

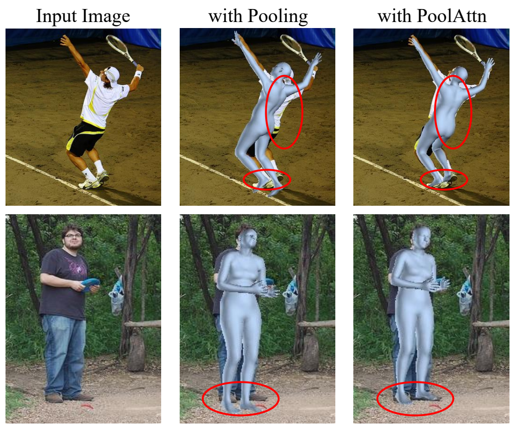


Figure 9. The visual comparison between applying the Pooling layer and PoolAttn layer. The red circles highlight locations where PoolAttn is more accurate than Pooling.

We also show the visualization of using the Pooling layer compared with using PoolAttn layer in Fig. 9. The areas highlighted by red circles indicate that PoolAttn outputs more accurate meshes than Pooling.

Effectiveness of HR stream: We investigate the use of the HR stream in Table 5. If we use the Pooling block in transformer (PoolFormer's architecture [40]), the results can be improved by a large margin (3.8 of MPJPE, 2.1 of PA-MPJPE, and 3.6 of MPVE) when adding the HR stream. If we apply our PoolAttn module in transformer (our PAT design), the HR stream can also boost the performance notably (2.8 of MPJPE, 0.8 of PA-MPJPE, and 2.4 of MPVE).

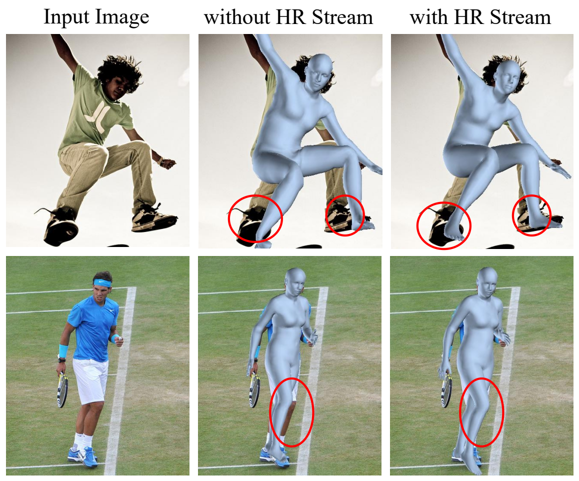


Figure 10. The visual comparison between with HR stream and without HR stream. The red circles highlight locations where POTTER with HR stream is more accurate.

We further compare the mesh visualization of POTTER (with HR stream) and without HR stream as shown in Fig. 10. The areas highlighted by red circles indicate that the HR stream can improve the quality of the reconstructed mesh.

Table 5. Ablation study of the HR stream in the transformer architecture on 3DPW dataset for HMR.

		3DPW		
Block	Architecture	MPJPE ↓	PA-MPJPE ↓	MPVE ↓
Pooling	Without HR-stream	81.1	49.5	93.5
	With HR-stream	77.3	47.4	89.9
PoolAttn	Without HR-stream	77.8	45.6	89.8
	With HR-stream	75.0	44.8	87.4

5. Conclusion

In this paper, we present POTTER, a pure transformer-based architecture for efficient HMR. A Pooling Attention Transformer block is proposed to reduce the memory and computational cost without sacrificing performance. Moreover, an HR stream in the transformer architecture is proposed to return high-resolution local and global features for the HMR task. Extensive experiments show that POTTER achieves SOTA performance on challenging HMR datasets while significantly reducing the computational cost.

As an image-based method, one limitation is that POTTER can not fully exploit the temporal information given video sequences input. We will extend POTTER to a video-based version that can output smooth 3D motion with better temporal consistency.

Acknowledgement: This work is supported by gift funding from OPPO Research.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 3

[2] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in Neural Information Processing Systems*, 34:17864–17875, 2021. 1

[3] Junhyeong Cho, Kim Youwang, and Tae-Hyun Oh. Cross-attention of disentangled modalities for 3d human mesh recovery with transformers. In *European Conference on Computer Vision (ECCV)*, 2022. 1, 3, 7

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 1, 2, 3, 6

[5] Sai Kumar Dwivedi, Nikos Athanasiou, Muhammed Kocabas, and Michael J Black. Learning to regress bodies from images using differentiable semantic rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11250–11259, 2021. 3, 7

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[7] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014. 7

[8] Wen Jiang, Nikos Kolotouros, Georgios Pavlakos, Xiaowei Zhou, and Kostas Daniilidis. Coherent reconstruction of multiple humans from a single image. In *CVPR*, 2020. 3

[9] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 3, 7

[10] Rawal Khirodkar, Shashank Tripathi, and Kris Kitani. Occluded human mesh recovery. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 7

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6, 7

[12] Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. Vibe: Video inference for human body pose and shape estimation. In *CVPR*, 2020. 7

[13] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2252–2261, 2019. 3, 7

[14] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 3, 7

[15] Nikos Kolotouros, Georgios Pavlakos, Dinesh Jayaraman, and Kostas Daniilidis. Probabilistic modeling for human mesh recovery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11605–11614, 2021. 3, 7

[16] Jiefeng Li, Chao Xu, Zhicun Chen, Siyuan Bian, Lixin Yang, and Cewu Lu. Hybrik: A hybrid analytical-neural inverse kinematics solution for 3d human pose and shape estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3383–3393, 2021. 7

[17] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1954–1963, 2021. 1, 2, 3, 7

[18] Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphormer. In *ICCV*, 2021. 1, 3, 7

[19] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 2

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 7

[21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *International Conference on Computer Vision (ICCV)*, 2021. 2, 3, 4, 6

[22] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. *arXiv preprint arXiv:2106.13230*, 2021. 1

[23] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2949–2958, 2021. 1

[24] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM TOG*, 2015. 3

[25] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *3D Vision (3DV), 2017 Fifth International Conference on*. IEEE, 2017. 7

[26] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2906–2917, 2021. 1

[27] Gyeongsik Moon and Kyoung Mu Lee. I2l-meshnet: Image-to-lixel prediction network for accurate 3d human pose and mesh estimation from a single rgb image. In *ECCV*, 2020. 7

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 7

[29] Yating Tian, Hongwen Zhang, Yebin Liu, and Limin Wang. Recovering 3d human mesh from monocular images: A survey. *arXiv preprint arXiv:2203.01923*, 2022. 3, 7

[30] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34:24261–24272, 2021. 6

[31] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems*, 2022. 1

[32] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 6

[33] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 6

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3

[35] Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *European Conference on Computer Vision (ECCV)*, sep 2018. 7

[36] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 568–578, 2021. 6

[37] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021. 6

[38] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021. 1

[39] Xiangyu Xu, Hao Chen, Francesc Moreno-Noguer, László A Jeni, and Fernando De la Torre. 3d human shape and pose from a single low-resolution image with self-supervised learning. In *ECCV*, 2020. 3

[40] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10819–10829, 2022. 4, 5, 6, 7, 8

[41] Mihai Zanfir, Andrei Zanfir, Eduard Gabriel Bazavan, William T Freeman, Rahul Sukthankar, and Cristian Sminchisescu. Thundr: Transformer-based 3d human reconstruction with markers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12971–12980, 2021. 3, 7

[42] Wang Zeng, Sheng Jin, Wentao Liu, Chen Qian, Ping Luo, Wanli Ouyang, and Xiaogang Wang. Not all tokens are equal: Human-centric visual analysis via token clustering transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11101–11111, 2022. 3, 7

[43] Hongwen Zhang, Yating Tian, Xinchi Zhou, Wanli Ouyang, Yebin Liu, Limin Wang, and Zhenan Sun. Pymaf: 3d human pose and shape regression with pyramidal mesh alignment feedback loop. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 3, 7

[44] Weixi Zhao, Yunjie Tian, Qixiang Ye, Jianbin Jiao, and Weiqiang Wang. Graformer: Graph convolution transformer for 3d pose estimation. *arXiv preprint arXiv:2109.08364*, 2021. 3

[45] Ce Zheng, Matias Mendieta, Pu Wang, Aidong Lu, and Chen Chen. A lightweight graph transformer network for human mesh reconstruction from 2d human pose. In *ACM Multimedia*, 2022. 3, 7

[46] Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3d human pose estimation with spatial and temporal transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11656–11665, October 2021. 3

[47] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6881–6890, 2021. 3

[48] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3