VITA: ViT Acceleration for Efficient 3D Human Mesh Recovery via Hardware-Algorithm Co-Design

Shilin Tian[1], Chase Szafranski[1], Ce Zheng[1], Fan Yao[1], Ahmed Louri[2], Chen Chen[1], Hao Zheng[1]
[1]University of Central Florida, [2]The George Washington University
{shilin.tian,chase.szafranski,ce.zheng,fan.yao}@ucf.edu,louri@gwu.edu,chen.chen@crcv.ucf.edu,hao.zheng@ucf.edu

ABSTRACT

Vision Transformers (ViTs) have emerged as a promising solution to enable efficient 3D Human Mesh Recovery (HMR) in augmented and virtual reality (AR/VR) applications. Despite many advancements in algorithm design, it remains a challenge to efficiently accelerate ViT-based HMR due to high computational complexity, substantial memory footprint, and compromised data locality. In this paper, we propose VITA, a hardware and algorithm co-design framework for ViT-based HMR with improved performance and energy efficiency. Specifically, on the algorithm side, we propose an average pooling model to replace conventional multi-head attention, which is further optimized with improved data locality. On the hardware side, we propose an accelerator architecture that can efficiently support various dataflows and computations demanded by pooling, normalization, and convolution operations. We evaluate the proposed VITA, and the evaluation result shows that the proposed VITA design can achieve 5.05× and 69.12× speedups on average over the state-of-the-art GPUs and CPUs on HMR tasks.

ACM Reference Format:
Shilin Tian[1], Chase Szafranski[1], Ce Zheng[1], Fan Yao[1], Ahmed Louri[2], Chen Chen[1], Hao Zheng[1]. 2024. VITA: ViT Acceleration for Efficient 3D Human Mesh Recovery via Hardware-Algorithm Co-Design. In *61st ACM/IEEE Design Automation Conference (DAC '24), June 23–27, 2024, San Francisco, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649329.3656518

1 INTRODUCTION

Human Mesh Recovery (HMR) has been widely deployed in many critical computer vision applications, including gaming, artificial/virtual reality (AR/VR), and human-computer interaction [1, 2]. HMR can efficiently reconstruct 3D human shapes and poses from monocular images, driven by recent advancements in deep learning. However, conventional deep neural networks still fall short in HMR performance due to the complex articulation of the human body, varying levels of occlusion, and the ambiguity of perception [3, 4].

With the recent surge of Vision Transformer (ViT) [5], attention-based models have been explored to enhance the performance of HMR tasks. However, the computation complexity and large model size of ViT-based HMR hinder its deployment in real-world applications, particularly in embedded systems like headsets. To address this challenge, prior works [3, 6] have been proposed to reduce ViT model size and the number of Multiply-Accumulate (MAC) operations. For instance, POTTER [3] achieves a 92.8% and 86.2% reduction in model size and MAC operations, respectively, compared to METRO [6] on GPUs. However, this substantial reduction in model size and MAC operations translates to only a 45% reduction in inference time, as shown in Fig. 1. More importantly, achieving the desired inference speed — greater than 30 frames per second (fps) — for state-of-the-art ViT-based HMR on high-end GPUs remains a challenge, let alone on AR/VR headsets.

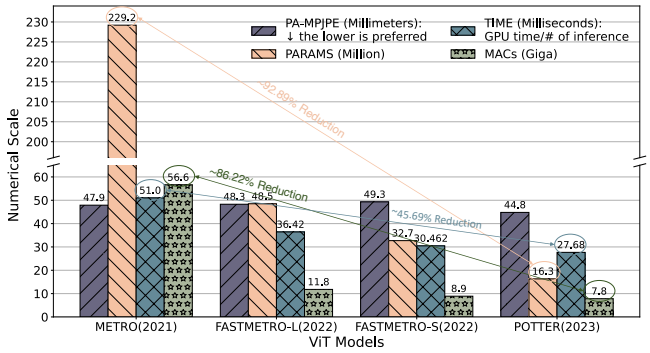

Figure 1: HMR performance comparisons on 3DPW dataset. The scaling of PA-MPJPE [7] (Procrustes Analysis-Mean-Per-Joint-Position-Error, number of parameters, number of MAC operations, and GPU inference time for ViT-based HMR.

Even though specialized accelerators [8, 9] have been proposed for ViT models, they have limited applicability to support HMR tasks efficiently. For example, ViTCod [8] reordered the attention maps to find either denser or sparser fixed patterns, improving computation efficiency. Vitality [9] replaced vanilla softmax attention with a linear attention and introduced a sparse approximated attention, thereby reducing computation requirements. Unfortunately, prior accelerators are customized for multi-head attention-based transformer models. Neither previous algorithms nor architectures have direct applicability to HMR tasks.

To this end, we aim to explore an efficient algorithm and hardware co-design to push the performance envelope of ViT-based HMR, enabling real-time inference capability in resource-constrained devices. Through our application profiling, we observed that current ViT-based HMR models suffer from poor data locality even though MAC operations and model size have been theoretically reduced. Specifically, POTTER [3] involves redundant and irregular memory access due to the reshaping operations of input features. To address the mentioned issues, we propose VITA, a hardware and algorithm co-design to reduce model size and MAC operations, and optimize data locality and DRAM access. Our contributions are summarized as follows:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
DAC '24, June 23–27, 2024, San Francisco, CA, USA
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0601-1/24/06…$15.00
https://doi.org/10.1145/3649329.3656518

Figure 2: The overview of the proposed vision transformer architecture.

To the best of our knowledge, VITA is the first ViT accelerator designed for Human Mesh Recovery (HMR) that incorporates various hardware constraints into the algorithm design.

On the algorithm side, we propose an average pooling block (APB) to prune and refine the conventional multi-head attention model. The proposed APB is designed to efficiently capture cross-channel spatial correlations without compromising the model accuracy. Moreover, the proposed APB not only supports regular and efficient memory access, but also preserves comparable model size and MAC count as compared to counterparts.

On the hardware side, VITA presents a unified PE architecture capable of handling a variety of ViT operations including pooling, normalization, and convolution. The proposed PE array can efficiently optimize spatial and temporal data locality of various pooling operations, thereby reducing DRAM access significantly.

We evaluate the proposed VITA, and the evaluation result shows that the proposed VITA design can achieve 5.95× and 69.12× speedups on average over the state-of-the-art GPUs and CPUs on HMR tasks.

2 PROPOSED VITA ALGORITHM

2.1 Overall Architecture of the Proposed ViT for HMR

In this work, we introduce our customized transformer architecture, specifically tailored for HMR tasks, as depicted in Fig 2. This design aims to achieve two critical goals. Firstly, it maintains a delicate balance between capturing global, fine-grained information at low resolutions and processing high-resolution patches for local feature extraction within the image. Secondly, the architecture is optimized for hardware, enhancing efficiency in computation complexity, memory requirement, and data locality.

Inspired by POTTER[3], our approach incorporates a dual-stream structure. To capture more global information at lower resolutions, as illustrated in Fig. 2, we utilize a total of 4 average pooling blocks. The employed blocks in our design hierarchically reduce the patch size, following a similar rationale to that of the Swin transformer block design[10]. Consequently, the upper stream progressively reduces the number of patches, while the bottom stream consistently maintains high-resolution feature representation. The upper stream's global features are then fused with the local features from the bottom stream via patch split blocks.

For example, in the upper stream, the output of "stage 2" is denoted as $f_2 \in \{\frac{H}{8} \cdot \frac{W}{8}\}$, effectively reducing the total patch count to $\frac{H}{8} \cdot \frac{W}{8}$ and lowering the feature resolution to $\frac{H}{8} \cdot \frac{W}{8}$. This reduction process is consistently applied to subsequent stages before fusion with the bottom stream.

Figure 3: The overview of average pooling block (APB).

Figure 4: The illustration of the proposed APB block: (a) Patch-wise pooling computation and (b) Position-wise average pooling computation.

Similarly, the bottom HMR stream path comprises multiple stages, each connected to the basic stream through a patch split block. The upper stream's merged patches are added to this path, preserving the high-resolution feature representation. For instance, the output for the third and fourth stages retains the same feature resolutions as $f_1 \in \{\frac{H}{4} \cdot \frac{W}{4}\}$.

To optimize the design for a hardware-friendly execution, we will introduce our proposed average pooling block (APB) to address the irregular memory access problem during the pooling operation in previous designs.

2.2 Proposed Average Pooling Block

Employing pooling operations exclusively for patch mixing, this method surpasses numerous complex transformer-based models, providing enhanced computational and memory efficiency. Although pooling operations are known for their ability to reduce memory footprints and the number of MAC operations, as discussed in [3], they necessitate access to varying regions of input features. This requirement, without strategic optimizations, can result in irregular memory access patterns and poor data locality. Traditional designs often overlook the detrimental effects of such irregularities. To address this challenge, we introduce an innovative average pooling block.

Figure 5: Overall VITA accelerator architecture.

This new pooling operation not only re nes data access patterns but also e ectively models the patch mixing functionality inherent in Attention layers [5], enhancing accuracy. As shown in Fig. 3, our VITA model employs two separate pathways in the APB to extract features from the input images. The APB consists of patch-wise and position-wise average pooling paths. Patch-wise pooling focuses on the overall structure by analyzing inter-patch correlations, while position-wise average pooling delves into detailed features within each patch. This ne-grained control over the patches ensures a comprehensive understanding of the input with lower computational and memory requirements.

Speci cally, in the*Patch-wise Pooling* , This stream captures inter-patch correlations while preserving their spatial structure [11]. As depicted in Fig. 4 (a), this process involves applying a pooling module that computes attention over compressed 2D features along height and width dimensions. This approach e ectively encapsulates the spatial relationships between patches. It is achieved by reducing the number of rows and columns of input tensors through pooling operations applied over the rows and columns associated with the respective patches.

On the other hand, the*Position-wise Average Pooling* is designed to explore dependencies within the embedding dimensions of each patch, retaining their 2D spatial structure as illustrated in Fig. 4 (b). Unlike baseline ViT adaptations [6, 12] that atten these features, we design this pathway that keeps the 2D structure, enabling a nuanced analysis of the intra-patch feature correlations for a deeper understanding of each patch's characteristics.

Therefore, the patch-wise pooling attention conserves the spatial positioning of each patch while capturing inter-patch relationships. Simultaneously, the position-wise average pooling attention preserves each patch's spatial embedding for intricate modeling of feature correspondences. The outputs from both pooling then pass through a depth-wise convolution layer [13], forming $"_1$ and $"_2$ before progressing to the Feedforward network.

3 PROPOSED VITA ACCELERATOR DESIGN

3.1 Overall Architecture

The proposed VITA accelerator architecture can support a variety of operations in the proposed ViT algorithm, including normalization, pooling, and convolution. The overall accelerator architecture is shown in Fig. 6. Speci cally, the proposed VITA architecture consists of a nonlinear function unit, a global bu er (GLB) for output, weight, and input, and a processing element (PE) array. As various pooling operations involve di erent memory access patterns, the input GLB adopts a multi-banked bu er design to enhance endpoint bandwidth for input features. The GLB is connected to the nonlinear function unit and the PE array via a crossbar network.

Figure 6: MAC con gurations for distinct operations in VITA.

3.1.1 Proposed PE Architecture. The proposed PE array can be used to simultaneously support distinct operations and their data ows. Speci cally, as shown in Fig. 6, we use a 4×4 PE array to illustrate the concept, in which PEs are connected in a torus topology. Each PE consists of 16 multiply-accumulation (MAC) units for parallel computations and local bu ers for input features, weights, and partial sums. We further present a novel MAC unit to support various computations tailored for our proposed transformer model. The proposed MAC unit can support square operation, division by 2 using a shifter, accumulation, and multiplication. This enables our MAC units to e ciently handle a variety of operations in one uni ed PE, which could improve PE utilization and data locality.

Walkthrough Examples. As illustrated in Fig. 6, the MAC datapath can be con gured to support four representative operations - patch-wise pooling, position-wise pooling, matrix multiplication, and depth-wise convolution. For example, to support patch-wise pooling operations, the feature and partial sum bu ers are connected to the adder as shown in Fig. 6 (a), and the summation of one patch of data will be further divided by its patch size. The MAC architecture can be further con gured for position-wise pooling operations as shown in Fig. 6 (b). Furthermore, it supports another two operation models for general matrix multiplication (Fig. 6 (c)) and convolution (Fig. 6 (d)) operations. The matrix multiplication is used to reconstruct the matrix dimensions after the pooling layers, as such both inputs are stored in the partial sum bu ers, whereas convolution operation needs both weight and input matrices.

3.1.2 Inverse Square Root Computation. We implement the inverse square root computation with a Lookup Table methodology, optimized for both e ciency and precision. This approach is based on a pre-populated LUT [14], encompassing a wide spectrum of precomputed inverse square root values to cover the anticipated range of input data. Once the variance value is received, the proposed unit will identify the nearest candidate and retrieve its approximated variance value from LUT. If a higher precision is needed, the values in LUT could be further re ned o ine.

Figure 7: (a) An example of input features, (b) data ow for patch-wise pooling, and (c) data ow for position-wise average pooling with an aggregation example for elements at position 0.

3.1.3 GeLU Approximation. Many state-of-the-art transformer models, such as SWIN transformer [10], employ the Gaussian Error Linear Unit (GELU) activation function [15] for their fully connected layers and MLPs instead of the traditional ReLU function. GELU, often considered a smoother variant of ReLU, o ers enhanced non-linearity that contributes to faster and more e ective convergence. The original formulation of GELU is presented as equation 1 [15], with an approximation provided in equation 2 [15]. However, implementing GELU activation using these equations imposes considerable hardware overhead.

$$GELU(x) = x \cdot P(X \le x) = 0.5x\left(1 + \mathrm{erf}\left(\frac{x}{\sqrt{2}}\right)\right) \quad (1)$$

$$GELU(x) \approx 0.5x\left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right)\right) \quad (2)$$

In the proposed accelerator, this challenge is addressed within a dedicated nonlinear function unit. This unit utilizes a local bu er to store precomputed GELU values based on equation 2. Upon receiving feature elements, the unit sequentially searches for the nearest value in the LUT [14], retrieves the approximated GELU value, and then returns the processed data to the global bu er. This approach e ciently approximates GELU activation while mitigating the computational and memory overhead typically associated with direct GELU computation.

3.2 Proposed Data ow and Mapping

Data ow exploitation [16] is a critical step that can determine the o -chip memory access. The proposed VITA accelerator architecture can e ciently support a variety of data ows, thus improving data locality and reducing DRAM access. We will use the following examples to illustrate the proposed data ow for various normalization, pooling, and convolution operations.

3.2.1 The proposed Dataflow for APB. Pooling layers are generally structured in a regular pattern, also re ected in their memory access. The major problem is that conventional pooling attention includes multiple pooling operations across di erent matrix dimensions. Consequently, di erent memory access patterns of pooling operations could lead to poor data locality. To this end, we propose a customized data ow for the proposed APB to unify the memory access patterns of multiple pooling operations with optimized data locality and reuse.

For example, as shown in Fig. 7 (b), all the elements of each channel ( ) of input features (,) are loaded into each PE. This is because patch-wise pooling is performed individually on each channel. As such, each PE can perform patch-wise pooling without any inter-PE communication.

On the other hand, position-wise pooling operation needs to collect the elements across channels as shown in Fig. 7 (c). This requires an accumulation operation among all the PEs, which inevitably increases the bandwidth requirement per link. To address this issue, we leverage the 2D-ring allreduce algorithm [17] to reduce link bandwidth requirement.

3.2.2 The proposed dataflow for normalization. As mentioned, normalization is a critical step in the proposed ViT model. The normalization algorithm used in this accelerator is a layer normalization [18]. As such, each element in the layer is normalized with the mean and variance computed across all elements, according to Equation 3 [18].

$$\sim = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} W + V \quad (3)$$

where W and V are learnable parameters for input shape and element-wise a ne information, and ϵ is a value added to the denominator for numerical stability.

The primary issue with layer normalization lies in the calculation of the mean, necessitating the aggregation of all elements within the layer. Much like the data ow for APB, we utilize a torus topology to facilitate the reduction operations needed to compute the mean value. For instance, each PE calculates the mean of elements housed in its local bu er. This computed mean is then propagated to the subsequent PE via vertical rings. Following vertical accumulation, a horizontal-wise aggregation is executed to derive the nal mean value. Subsequently, this mean is transmitted to the inverse square root unit, and the output result is broadcast to all PEs.

3.2.3 The proposed dataflow for Convolutional Neural Network. The selection of data ow has been extensively studied in prior works, as it can signi cantly a ect DRAM access. Typically, data ow for CNNs is subject to many factors, including matrix size, tiling factors, and others. In this work, we select output stationary and weight stationary data ows for the basic and HMR streams.

Output stationary data ow for convolutional layers at the basic stream: In the basic stream, the input matrix is partitioned and distributed to all the PEs across dimension. As such, the weight

Table 1: Design Parameters of VITA Accelerator

Accelerator Parameters	Configuration	PE Parameters	Configuration
# of PEs	8×8	# of MACs	64
Feature GLB	262 KB	Feature LocalBuf.	16,384B
Weight GLB	62 KB	Weight LocalBuf.	264 B
Output GLB	262 KB	Psum LocalBuf.	8196B
Clock Freq.	330 MHz		
Bandwidth Configuration for Simulation			
DRAM: 64 GB/s		GLB to PE Array: 80 GB/s	

matrix is also partitioned on D dimension which will be distributed to each PE. Given such a spatial parallelism strategy, no partial sum accumulation is needed between PEs. As such, we select the output stationary to support temporal partial sum accumulation within the PE.

Weight stationary dataflow for convolutional layers at the HMR stream: In the HMR stream, the input matrix is spatially partitioned and sent all the PEs, in which attributes D, H, and W are spatially parallelized. As mentioned earlier, a depth-wise convolution will be performed to recover the full dimensions of the feature matrix. As such, the weight matrix ($D \times 3 \times 3$) will be duplicated in every PE. The partial sums produced by each PE need to be accumulated spatially. Consequently, we select the weight stationary dataflow to improve the temporal reuse of the weight matrix.

4 EXPERIMENT RESULTS

4.1 Experiment Setting

Dataset and Training configurations: We evaluate the proposed VITA model as compared to state-of-the-art Vision Transformer (ViT)-based models. For image classification, we compare VITA against multiple models, including RSB-Resnet-18/34 [19], ResMLP-S12 [20], and PoolFormer_S12 [3]. To evaluate the performance of human mesh recovery, VITA compares itself with POTTER_HMR [3], METRO [6], and FASTMETRO [12]. The datasets include ImageNet-1K [21] for image classification and 3DPW[22] for HMR. Following the training regimen outlined in [23], we pre-trained the VITA model with the basic stream on the ImageNet-1K dataset for 310 epochs and fine tune the entire VITA model on Human3.6M afterwards. During the pre-training process, we set the peak learning rate to $lr = 2 \times 10^{-3}$ and used a total batch size of 1024.

Baseline architectures: To assess the performance of the VITA accelerator, particularly for image classification and Human Mesh Recovery, we include several high-performance computing platforms to evaluate the performance the proposed VITA accelerator architecture, which includes a server-class CPU Xeon(R) Gold 6240 CPU, and two advanced GPUs: the NVIDIA V100 and the NVIDIA H100 GPU. Given that existing ViT accelerators do not support ViT-based HMR, we are unable to perform a fair comparison.

Accelerator Configurations: The design of VITA accelerator is implemented with Verilog RTL for synthesis, and we developed a customized simulator to fairly reflect the hardware configurations as shown in table 1. In order to ensure a fair and realistic comparison with actual hardware implementations and to obtain relatively accurate performance metrics, all parameters were configured to align closely with feasible values for generic ASIC designs. For instance, the clock frequency employed in our design is set at 330 MHz. The GLB size is selected based on the storage needs of input features for all stages in both basic and HMR streams as illustrated in Fig. 2. Considering an image size of $3 \times 224 \times 224$, the input feature size for four stages are $64 \times 64 \times 64$, $128 \times 32 \times 32$, $320 \times 16 \times 16$ and $512 \times 8 \times 8$, respectively. As such, we find that a small GLB size of 262KB is sufficient to accommodate the storage demands. However, a larger local buffer is anticipated due to the fact that patch-wise pooling computations are executed locally within PE. Regarding on-chip bandwidth, our configuration tries to saturate bandwidth of typical DDR4 memory in FPGA with more than 4 channels.

Table 2: Performance comparison on image classification task. Models are train on training set and the reported top-1 accuracy is obtained from the validation set.

Models			
Name	Params(M)	MACs(G)	Top-1 Acc(%)
ViT-L/16 [5]	307	190.7	76.5
RSB-ResNet-18 [19]	12	1.8	70.6
RSB-ResNet-34 [19]	22	3.7	75.5
POTTER_S12 [3]	12	1.8	77.2
VITA	12	1.8	**78.08**

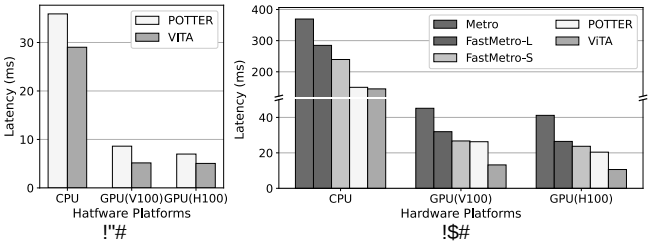

Figure 8: The performance comparisons between VITA over multiple platforms for HMR task.

4.2 Evaluation

4.2.1 Evaluation of the Proposed ViT algorithm. The performance of our VITA model for the image classification task is reported in Table 2. We compare it with both transformer-based models (ViT-L/16 [5], POTTER_S12 [3]) and CNN-based models(RSB-ResNet-18 [19] and RSB-ResNet-34 [19]). RSB-ResNet-34 is trained with 300 epochs, whereas other models are trained within 100 epochs. In general, VITA achieves an increase of 4.58% in Top-1 accuracy compared to RSB-ResNet-34, while reducing the model size and MAC operations by 45.45% and 51.35% respectively. When compared with the transformer-based model, POTTER_S12, VITA slightly increases the top-1 accuracy with similar model size and MAC operations. Regarding inference performance improvement, Fig. 8 (a) illustrates that VITA can achieve a speedup of 1.23×, 1.67×, and 1.38× compared to POTTER on CPU, Nvidia V100, and Nvidia H100 platforms, respectively. We use 3DPW dataset to evaluate VITA performance on HMR tasks. VITA can achieve 77.6, 46.7, 90.2 in Mean Per Joint Position Error (MPJPE) [24], Procrustes Alignment (PA-MPJPE) [7], and MPVE [25], which significantly outperforms METRO and FASTMETRO. The MPJPE, PA-MPJPE, and MPVE of VITA are slightly higher than that of POTTER (75.0, 44.8, and 87.4).