Efficient Vision Transformer for Human Pose Estimation via Patch Selection

Kaleab A. Kinfu
kinfu@seas.upenn.edu
René Vidal
vidalr@seas.upenn.edu

Center for Innovation in Data Engineering and Science (IDEAS), University of Pennsylvania
Philadelphia, PA, USA

Abstract

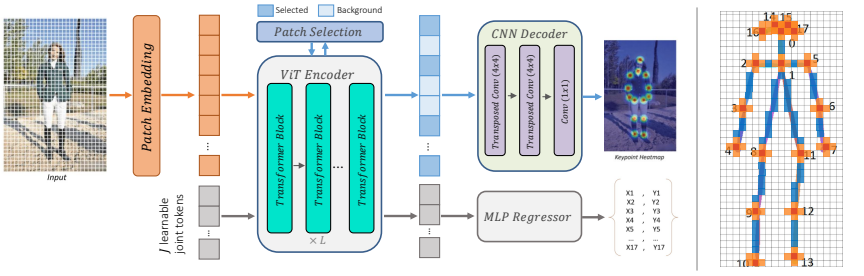
While Convolutional Neural Networks (CNNs) have been widely successful in 2D human pose estimation, Vision Transformers (ViTs) have emerged as a promising alternative to CNNs, boosting state-of-the-art performance. However, the quadratic computational complexity of ViTs has limited their applicability for processing high-resolution images. In this paper, we propose three methods for reducing ViT's computational complexity, which are based on selecting and processing a small number of most informative patches while disregarding others. The first two methods leverage a lightweight pose estimation network to guide the patch selection process, while the third method utilizes a set of learnable joint tokens to ensure that the selected patches contain the most important information about body joints. Experiments across six benchmarks show that our proposed methods achieve a significant reduction in computational complexity, ranging from 30% to 44%, with only a minimal drop in accuracy between 0% and 3.5%.

1 Introduction

In recent years, Human Pose Estimation has emerged as an important problem in computer vision, with numerous applications in fields such as surveillance [13], motion analysis [31], virtual and augmented reality [17, 23]. Classical pose estimation algorithms relied on handcrafted features [9, 12, 25], but recent advances in deep learning have led to significant improvements based on learned features [33, 41]. For instance, Convolutional Neural Networks (CNNs) have proven to be successful by exploiting spatial correlations among pixels.

The recent emergence of Vision Transformers (ViTs) has challenged the dominance of CNNs. Unlike CNNs, ViTs rely on self-attention mechanisms to model the long-range dependencies between patches, which has been shown to be highly effective [7]. Nevertheless, the computational complexity of ViTs presents a significant challenge for processing high-resolution images. In particular, the computational cost of ViTs scales quadratically with the number of input tokens, making them intractable for practical use.

To address this issue, several recent works have proposed various methods for reducing the number of tokens that need to be processed by ViTs, thereby lowering their computational cost. Token Learner [28] is one approach that aims to merge and reduce the input tokens into



(a) (b)

Figure 1: **Overall architecture of EViTPose –** (a) The input image is first passed through a patch embedding layer to obtain patches of size 16×16. These patches, along with J learnable joint tokens, are processed by a ViT with L transformer blocks. Patches are selected either before processing with ViT for neighboring and skeleton techniques or progressively across blocks for joint tokens based patch selection. The output of the last ViT block is then used by a CNN-based decoder to estimate the heatmap of J joints, while a simple MLP joint regressor estimates joints directly from joint tokens. (b) Skeletal visualization of selected patches via the proposed patch selection methods. The red, orange, and blue patches correspond to the body joints, neighboring patches, and skeleton patches, respectively.

a small set of important learned tokens. Token Pooling [19] clusters the tokens and down-samples them, whereas DynamicViT [26] introduces a token scoring network to identify and remove redundant tokens. Although these techniques successfully reduce the computational complexity of ViTs in classification tasks, the additional pooling and scoring network can introduce extra computational overhead. Besides, the extension of these approaches to dense prediction tasks, such as human pose estimation, remains an open question.

In this work, we propose to reduce the computational complexity of transformer based human pose estimation networks by selecting and processing a small subset of patches that are most likely to contain body joints or limbs. The rationale behind this choice is that results from [40] suggest that pixels neighboring a joint or a limb are more informative about joint locations. We first propose two patch selection methods that utilize fast yet imprecise out-of-the-shelf pose estimation networks to guide the selection process. We then propose a patch selection method that uses learnable joint tokens to progressively select the most informative patches. As a result, our approach significantly enhances computational efficiency, albeit with a minor trade-off in accuracy. To validate the effectiveness of our proposed methods, we conducted experiments on six 2D human pose estimation benchmarks. The experiments show that our approach significantly reduces computational complexity (30% to 44% drop in GFLOPs), while only having a minimal drop in accuracy (between 0% and 3.5%).

2 Related Work

2.1 Human Pose Estimation

Human Pose Estimation is an essential task in computer vision that involves identifying and estimating the location of human body joints from 2D images or videos. In recent years, deep learning methods have been successful in 2D human pose estimation, with most meth-

ods employing CNNs to learn a mapping between the input image and the corresponding 2D pose. There are two main deep learning pipelines used in single person pose estimation: regression-based and heatmap-based approaches. Regression-based methods directly map the input image to 2D joint positions [33], while heatmap-based methods predict approximate joint locations using 2D Gaussian heatmaps centered at the body joint [36]. The heatmap-based approach is effective and commonly used, and will be used for inference in this work although both will be utilized for training.

Multi-person pose estimation is a difficult task that requires determining the number of people and their positions, as well as grouping key points. Two main approaches are top-down and bottom-up. Top-down approaches [21, 32, 38, 40, 42] use person detectors to extract boxes from input images and then apply single-person pose estimators to produce multi-person poses. In contrast, bottom-up techniques [3, 4, 22, 30] identify all body joints in a single image and group them for each person. This work will follow the top-down approach, which has been shown to be effective.

2.2 Vision Transformer

The Vision Transformer (ViT) architecture proposed by Dosovitskiy *et al.* [7] has demonstrated remarkable performance on image classification tasks. Consequently, several transformer based architectures have been proposed for human pose estimation. Transpose [40] is a transformer network that estimates 2D pose using a CNN-based backbone. TokenPose [15] is another transformer based on explicit token representation for each body joint. HRFormer [42] is a transformer that adopts HRNet design along with convolution and local-window self-attention. ViTPose [39] is a ViT-based approach that uses a shared encoder training on multiple datasets to improve performance. These architectures demonstrate the effectiveness of transformer-based models in human pose estimation.

However, vision transformers have quadratic computational complexity. To improve the efficiency of ViT, researchers have proposed methods such as sparsifying the attention matrix [5, 27], token pooling [19], and estimating the significance of tokens [26]. Hierarchical Visual Transformer [24] removes redundant tokens via token pooling, while TokenLearner [28] introduces a learnable tokenization module. Adaptive Token Sampler [8] adaptively down-samples input tokens, which assigns significance scores to every token based on the attention weights of the class token in ViT. Similarly, EViT [16] determines tokens' importance scores via attention weights. However, it is still an open question whether these approaches can be extended to dense prediction tasks, such as human pose estimation. In our third patch selection method, we will follow a similar approach to [8, 16] and extend it to the human pose estimation task.

3 ViT based Human Pose Estimation with Patch Selection

In this section, we describe our approach to human pose estimation, which includes revisiting the standard ViT-based method, incorporating learnable joint tokens and patch selection techniques. We present the overall architecture of our method in Figure 1a. Given an input image $X \in R^{H\times W\times 3}$, the task is to find a mapping from X to $Y \in \mathbb{R}^{J\times 2}$, where J is the number of body joints, for each person in the image. We first embed X into patches of size 16×16, resulting in a set of patch tokens $\mathbf{P} \in \mathbb{R}^{N\times C}$, where $N = \frac{H}{16} \times \frac{W}{16}$ and C represents the channel dimension. We then extend the patch tokens with J learnable joint tokens, $\mathbf{J} \in \mathbb{R}^{J\times C}$, that

explicitly embed each one of the joints that are later used to regress the joint 2D positions in the image, resulting in a concatenated set of input tokens $\mathbf{I} \in \mathbb{R}^{(N+J)\times C}$. The patch and joint tokens are then fed to a standard ViT encoder with L transformer blocks, each of which consists of a multi-head self-attention (MSA) layer and a feed-forward network (FFN).

Here, we revisit the standard self-attention mechanism. Later, we will show how this mechanism is modified in the case of the joint-token-based patch selection method. In the self-attention mechanism, the output tokens $\mathbf{O}$ and the attention matrix $\mathcal{A}$ are computed as

$$\mathbf{O} = \mathcal{A}\mathcal{V} \quad \text{and} \quad \mathcal{A} = \text{Softmax}(\frac{\mathcal{Q}\mathcal{K}^T}{\sqrt{C}}), \tag{1}$$

where $\mathcal{Q} \in \mathbb{R}^{(N+J)\times C}$, $\mathcal{K} \in \mathbb{R}^{(N+J)\times C}$ and $\mathcal{V} \in \mathbb{R}^{(N+J)\times C}$ are, respectively, the queries, keys, and values, which are computed from the input tokens $\mathbf{I}$ as in the standard ViT [7].

Given the final feature map of the patches produced by ViT, denoted as $F_{out} \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times C}$, we use a classical decoder with two deconvolution blocks, each with a deconvolution layer, batch normalization, and ReLU activation to estimate the heatmap of J joints. Similarly, a LayerNorm layer followed by a fully connected MLP layer is used to directly regress the J joint coordinates from the joint tokens. In this way, the joint tokens are enforced to learn the important joint-level information to be able to successfully regress the joint 2D positions.

3.1 Improving Efficiency via Patch Selection

Although ViT can model long-range dependencies and is able to generate a global representation of the overall image, the computational complexity increases quadratically with respect to the number of tokens. However, not all patches in an image contribute equally to the human pose estimation task. Recent research [40] indicates that the long-range dependencies between predicted joints are mostly restricted to the body part regions. Therefore, computing MSA between every patch in the image while only a few patches are relevant to the body parts is unnecessary. To this end, we propose three methods to select a small number of relevant patches while discarding irrelevant and background patches without retraining the vision transformer. By selecting only the relevant patches, we can significantly reduce the computational complexity as shown in [8, 20, 26] for the classification task.

The first two approaches utilize an off-the-shelf lightweight pose estimator to guide the patch selection. Our first approach is based on a breadth-first neighboring search algorithm that selects body joint patches and its neighbors given estimated pose predictions, as outlined in Section 3.2. In our second approach, we extend the first approach to select patches formed by a skeleton of the joints. Here, the objective is to select body patches where the lines formed by body joint pairs cross. To accomplish this, we utilize Bresenham's algorithm to select the relevant patches, as outlined in Section 3.3. It is important to note, however, that by selecting a few patches of the image and processing them with the ViT encoder, we only get the features of the patches that were chosen. However, we must create a feature map for all patches for further processing. As a result, since the goal of the task is to produce a Gaussian heatmap centered at the body joint and zero elsewhere, we fill the non-body-part patches with zeros. Whilst the aforementioned methods can remove irrelevant patches before they are processed with ViT and thus enhance its efficiency, their reliance on the accuracy of off-the-shelf pose estimators is a limitation. As a result, we present an alternative approach for

Algorithm 1 (Select body joint patches and neighbors given joint prediction)

Require: joint prediction $\mathcal{B} \in \mathbb{R}^{J\times 2}$, patch size P, neighboring search function $\mathcal{N}$ and n number of neighboring patches.

function SELECTJOINTPATCHES
  $BP = \{\}$ ▷ set of body part patches
  $c = \frac{W}{P}$ ▷ number of columns
  for $j \leftarrow 1$ to J **do**
    $x^j = \lfloor \frac{\mathcal{B}_x^j}{P} \rfloor$ ▷ get column
    $y^j = \lfloor \frac{\mathcal{B}_y^j}{P} \rfloor$ ▷ get row
    for $(k,l) \in \mathcal{N}(x^j, y^j, n)$ **do**
      $x^n = (x^j + k)$
      $y^n = (y^j + l)$
      $p = y^n \times c + x^n$
      $BP \leftarrow p$ ▷ add patch to set
    end for
  end for
end function

Algorithm 2 (Select body part patches formed by a skeleton)

Require: joint prediction $\mathcal{B} \in \mathbb{R}^{J\times 2}$, body joint pairs $\mathcal{P}$, patch size P

function SELECTSKELETONPATCHES
  $BP = \{\}$
  for $j \leftarrow 1$ to J **do**
    $x_0^j = \lfloor \frac{\mathcal{B}_x^j}{P} \rfloor, y_0^j = \lfloor \frac{\mathcal{B}_y^j}{P} \rfloor$
    for $l \in \mathcal{P}(j)$ **do**
      $x_1^j = \lfloor \frac{\mathcal{B}_x^l}{P} \rfloor, y_1^j = \lfloor \frac{\mathcal{B}_y^l}{P} \rfloor$
      $\Delta_x = x_1 - x_0, \Delta_y = y_1 - y_0$
      $\varepsilon = 2\Delta_y - \Delta_x$
      $y = y_0$
      for $x \in [x_0, x_1]$ **do**
        $BP \leftarrow (x,y)$
        if $\varepsilon \geq 0$ **then**
          $y = y + 1$
          $\varepsilon = \varepsilon - 2\Delta_x$
        end if
        $\varepsilon = \varepsilon + 2\Delta_y$
      end for
    end for
  end for
end function

automatically selecting body part patches via learnable joint tokens that enable the selection of relevant patches using their corresponding attention maps, as outlined in Section 3.4.

3.2 Neighboring Patch Selection Method

In our first approach, we assume we are provided with an estimate of the joint locations $\mathcal{B} \in \mathbb{R}^{J\times 2}$ via a lightweight pose estimation network, where J denotes the number of joints. To select the body joint patch and its n neighboring patches, we employ a Breadth-First Search (BFS) algorithm, as presented in Algorithm 1. Specifically, for every joint located at 2D patch location $(x,y) \in \mathcal{B}$, we identify the nearest four neighboring patches located at $(x,y+1)$, $(x,y-1)$, $(x-1,y)$, and $(x+1,y)$, and store them in a queue. We continue searching for neighboring patches until we have selected n patches, ensuring that we do not revisit any previously visited patches. To provide a visual representation of this approach, we depict in Figure 1b a skeletonized human figure where the red patches represent the joint patches and the set of orange patches correspond to the selected neighboring patches.

3.3 Skeleton Patch Selection Method

A limitation of the aforementioned neighboring selection method is that it only covers the body joint patches and its neighbors. Our second approach extends this method to encompass all patches between joints. This can lead to improved performance with a minimal increment

in the computational complexity. To achieve this, we adopt a different strategy by identifying the patches where the line formed by the body joint pair (segments) crosses, based on Bresenham's algorithm [2]. Originally proposed as a canonical line-drawing algorithm for pixellated grids, our extension of Bresenham determines the patches that need to be selected in the line between (x_0, y_0) and (x_1, y_1), corresponding to the start and end 2D locations of patches containing body joints. As we move across the $x-$ or $y-$ axis in unit intervals, we select the x or y value between the current and next value that is closer to the line formed by the body joint pairs. To make this decision, we require a parameter ε. Our objective is to track the slope error from the last increment, and if the error exceeds a certain threshold, we increment our coordinate values and subtract from the error to re-adjust it to represent the distance from the top of the new patch, as presented in Algorithm 2. As depicted in Figure 1b, the set of blue patches represents the patches selected by this approach.

3.4 Joint-Token-based Patch Selection Method

The limitation of the first two patch selection methods is that they rely on the performance of the lightweight off-the-shelf pose estimator. This limitation can become especially problematic when dealing with complex scenes, as the accuracy of the pose estimator is often compromised by occlusion, motion, or variations in camera perspective. As a consequence, this might result in the selection of irrelevant patches and removing important patches, leading to suboptimal performance. Therefore, a more robust approach is required that can adapt to these challenging scenes without the need for an off-the-shelf pose estimator.

Selecting most informative patches via learnable joint tokens. We propose to overcome this limitation by selecting patches using the learnable joint tokens, which serve as a powerful feature representation for distinguishing the relevant body part patches. Specifically, we aim to determine the importance of each patch in relation to the joint tokens, thereby enabling us to select the most informative body part patches. To achieve this, we harness the attention matrix similar to [8, 10, 16], as the values in $\mathcal{A}$ serve as the weight of contribution of input tokens to output tokens. For example, $\mathcal{A}_{N+1,1:N}$ denotes the contribution weights of the N patch tokens to the $(N+1)^{\text{th}}$ output token, *i.e.* first joint token. Thus, we can calculate the average contribution weight of a patch token l to the J joint tokens, as follows:

$$\mathcal{W}_l = \frac{1}{J}\sum_{j=1}^{J} \mathcal{A}_{N+j,l}. \tag{2}$$

Following [8], we take the norm of $\mathcal{V}_l$ into account for calculating the importance score. Thus, the importance score of the patch token l is:

$$\mathcal{I}_l = \frac{\mathcal{W}_l \times \|\mathcal{V}_l\|}{\sum_{k=1}^{N} \mathcal{W}_k \times \|\mathcal{V}_k\|}, \tag{3}$$

where $l, k \in \{1, \ldots, N\}$. Once the importance scores of each patch token have been computed, we select L patch tokens with the highest scores for further processing, where $L \ll N$.

Pruning attention matrix. Our subsequent step involves pruning the attention matrix $\mathcal{A} \in \mathbb{R}^{(N+J)\times(N+J)}$ by selecting the rows that correspond to the chosen L patch tokens and J joint tokens, designated as $\mathcal{A}^s \in \mathbb{R}^{(L+J)\times(L+J)}$. We then compute the output tokens $\mathbf{O}^s \in \mathcal{R}^{(L+J)\times C}$, given by:

$$\mathbf{O}^s = \mathcal{A}^s \mathcal{V}^s. \tag{4}$$

These output tokens are then passed as input for the next blocks.

Refining background patches via joint tokens. Although only $\mathbf{O}^s$ will be processed in the next blocks of ViT, the non-selected patch tokens will still be used during the heatmap decoding. Therefore, it is important to have a refined representation of the non-selected patch tokens before they are excluded from further processing in the next blocks. Thus, we propose an efficient method that updates these tokens using the joint tokens. This approach is motivated by the fact that joint tokens learn global information and therefore can be used to update the patch tokens in a computationally efficient manner without the need to compute contributions from all tokens, which can be computationally expensive. We start by selecting the rows of the attention matrix $\mathcal{A}$ that correspond to the non-selected patch tokens and the columns that correspond to the joint tokens, resulting in a sub-matrix $\mathcal{A}^o \in \mathbb{R}^{(N-L)\times J}$. We then update the non-selected patch tokens using the joint tokens as follows:

$$\mathbf{O}^o = \mathcal{A}^o \mathcal{V}^j, \tag{5}$$

where $\mathcal{V}^j$ corresponds to the values of the joint tokens.

4 Experiments

4.1 Implementation details

In our experiments, we employ the common top-down setting for human pose estimation. We follow most of the default training and evaluation settings of the mmpose [6] framework but use the AdamW optimizer with a learning rate of $5e-4$ and UDP as a post-processing method. We use ViT-B and ViT-L as backbones and refer to the corresponding models as EViTPose-B and EViTPose-L. The backbones are pre-trained with MAE [11] weights and trained with multiple datasets as in [39].

4.2 Dataset details

The proposed methods are evaluated on six 2D pose estimation benchmarks: namely MPII [1], COCO [18], AI Challenger [37], CrowdPose [14], JRDB-Pose [35], and OCHuman [43]. The datasets are challenging and diverse, with varying numbers of images, person instances, and annotated joints. The first five datasets are used to train and test the proposed method, and OCHuman is only used to test the models in dealing with occluded scenes.

4.3 Evaluation metrics

On the MPII benchmark, we adopt the standard PCKh metric as our performance evaluation metric. PCKh [1] is an accuracy metric that measures if the predicted joint and the true joint are within a certain distance threshold (50% of the head segment length). On the remaining benchmarks, we adopt standard average precision (AP) as our main performance evaluation metric. AP is calculated using Object Keypoint Similarity (OKS), which measures how close the predicted joint location is to the ground-truth joint. Additionally, we will evaluate the computational complexity of the models by measuring the total number of Floating Point Operations (FLOPs) that each model needs to perform.

Table 1: Performance of the proposed patch selection methods on three benchmarks, namely COCO val set, MPII test set, and OCHuman test set. The best results among the base and large model variants are highlighted in bold.

Model	Patch Selection	Params	FLOPs	COCO	MPII	OCH
Lite-HRNet [41]	None	1M	0.2G	64.8	86.1	51.9
SimpleBaseline [38]	None	69M	15.7G	72.0	89.0	58.2
HRNet-W48 [32]	None	64M	14.6G	75.1	90.1	60.4
TransPose-H/A6 [40]	None	18M	21.8G	75.8	92.3	-
TokenPose-L/D24 [15]	None	28M	11.0G	75.8	-	-
HRFormer-B [42]	None	43M	12.2G	75.6	-	49.7
ViTPose-B [39]	None	86M	18.0G	77.1	**93.3**	87.3
ViTPose-L [39]	None	307M	59.8G	78.7	**94.0**	90.9
ViTPose-H [39]	None	632M	122.8G	79.5	94.1	90.9
EViTPose-B (Ours)	None	90M	19.8G	**77.6**	92.4	**93.0**
EViTPose-B (Ours)	Neighbors	90M	11.1G	74.1	91.8	89.5
EViTPose-B (Ours)	Skeleton	90M	13.3G	75.0	92.1	90.1
EViTPose-B (Ours)	Joint Tokens	90M	13.7G	*76.5*	92.5	*92.3*
EViTPose-L (Ours)	None	309M	66.7G	**78.7**	92.8	**94.3**
EViTPose-L (Ours)	Neighbors	309M	35.6G	75.7	92.1	90.0
EViTPose-L (Ours)	Skeleton	309M	38.3G	76.3	92.4	92.6
EViTPose-L (Ours)	Joint Tokens	309M	45.5G	*77.3*	*92.7*	*93.6*

4.4 Results

The performance of our proposed methods and other convolutional and transformer-based methods on three datasets, namely the COCO val set, the MPII test set, and the OCHuman test set, are presented in Table 1. We used LiteHRNet [41], a lightweight and less accurate pose estimation network, to guide the first two patch selection methods. Additional results of the joint-token-based patch selection method on AI Challenger val set, CrowdPose test set, and JRDB-Pose val set is presented in Table 2. The remaining patch selection methods were not evaluated on these benchmarks as we were not able to find lightweight pose estimators trained on them. Our experiments show that EViTPose without patch selection outperforms all, but is computationally expensive. However, our proposed patch selection methods proves to be beneficial in this regard, as it significantly reduces computational costs while maintaining high accuracy. For instance, we achieve a reduction of 30% to 44% in GFLOPs with a slight drop in accuracy ranging from 1.1% to 3.5% for COCO, 0% to 0.6% for MPII, and 0.7% to 3.5% for OCHuman. We can also control the drop in accuracy by changing the number of patches to be selected. The trade-off between performance and computational complexity for the neighboring and joint-token-based patch selection methods is depicted in Figure 2. The neighboring and skeleton patch selection remove irrelevant patches before they are processed by ViT, while the joint-token-based selection method learns to remove them on the fly. Thus, the first two approaches prioritize efficiency over accuracy by removing patches early on. For example, they are effective for addressing the low end range in Figure 2, where the joint-token-based selection method performs poorly. Some qualitative results of our method on sample images from the benchmarks are illustrated in Figure 4.

Table 2: Performance of the proposed joint-token-based patch selection method on the other three benchmarks, namely AI Challenger val set, CrowdPose test set, and JRDB-Pose val set.

Model	Patch Selection	Params	FLOPs	AIC	CPose	JRDP
SimpleBaseline [38]	None	69M	15.7G	29.9	60.8	-
HRNet-W48 [32]	None	64M	14.6G	33.5	-	42.4
HRFormer-B [42]	None	43M	12.2G	34.4	72.4	-
EViTPose-B (Ours)	None	90M	19.8G	**36.6**	**76.3**	**73.9**
EViTPose-B (Ours)	Joint Tokens	90M	13.7G	*35.0*	*74.5*	*72.8*
EViTPose-L (Ours)	None	309M	66.7G	**38.3**	**77.9**	**74.9**
EViTPose-L (Ours)	Joint Tokens	309M	45.5G	*37.1*	*76.7*	*74.2*

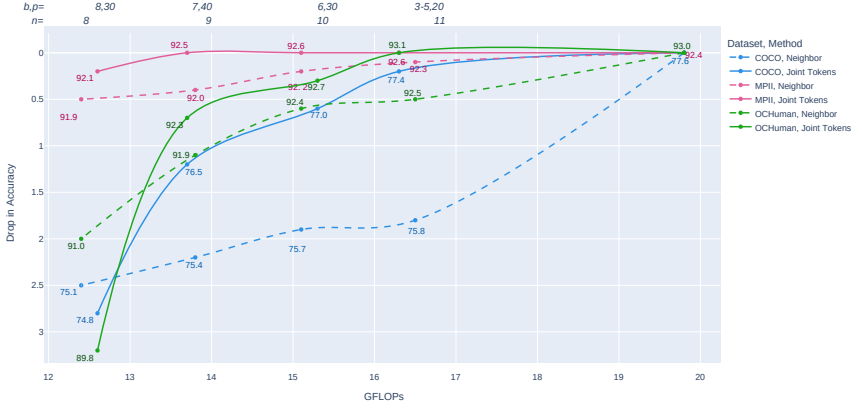


Figure 2: **Trade-off between accuracy and GFLOPs on three benchmarks: COCO, MPII, and OCHuman –** The performance of EViTPose-B with two patch selection methods: Neighbors (dashed line) and Joint-Token-based (solid line). n denotes to the number of neighbors selected and b, p refers to p number of patches that are removed at block b in the Joint Tokens method.

Ablation. We conducted a run-time comparison (measured in frames per second, FPS) among EViTPose, ViTPose and TokenPose, presented in Table 3. The results show that our Joint-Token-based Patch Selection method (EViTPose-B/JT) achieves an 88% reduction in GFLOPs and 10× increase in FPS with respect to ViTPose-H, with a minimal drop in accuracy of upto 2.9%. Furthermore, we evaluate the effect of the lightweight pose network in the performance of EViTPose with the first two patch selection methods as shown in Table 3. EViTPose maintains consistent performance across three different lightweight pose estimation models, namely LiteHRNet [41], MobileNetv2 [29, 34], and ShuffleNet [34, 44].

5 Conclusion

In this work, we have proposed EViTPose, a Vision Transformer-based human pose estimation network with patch selection methods that greatly reduces the computational com-

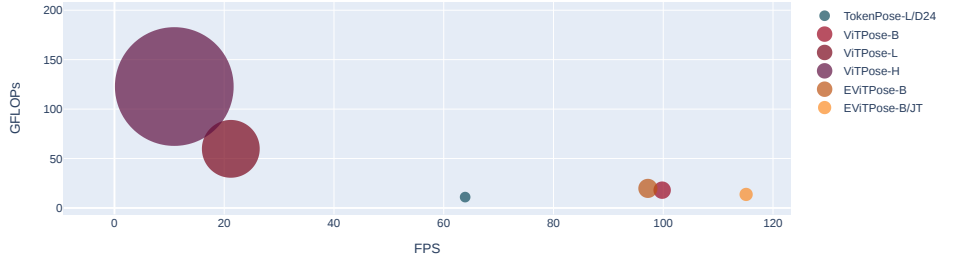


Figure 3: **Runtime (FPS) vs GFLOPs comparison –** The Joint-Token-based Patch Selection method (EViTPose-B/JT) achieves an 88% reduction in GFLOPs and a 10× (955%) increase in FPS compared to ViTPose-H, with a minimal accuracy drop of up to 2.9%.

Table 3: Effect of lightweight pose network in performance of EViTPose on the MPII dataset.

Neighbors			Skeleton		
LiteHRNet	MobileNetv2	ShuffleNet	LiteHRNet	MobileNetv2	ShuffleNet
92.1	91.9	92.0	91.8	91.8	91.8

plexity of ViTs while maintaining high accuracy. The proposed methods involve selecting and processing a small subset of patches that contain the most important information about body joints. Two of the methods utilize a fast yet imprecise out-of-the-shelf pose estimation network to guide the patch selection process, while the third method uses learnable joint tokens to progressively select the most informative patches. The experimental results on six widely-used 2D pose estimation benchmarks demonstrate that our methods significantly improve speed and reduce computational complexity, with reductions ranging from 30% to 44% in GFLOPs, with only a slight drop in accuracy, between 0% and 3.5%.

Figure 4: Qualitative results on sample images from the evaluation benchmarks.

Acknowledgments. The authors thank Carolina Pacheco and Yutao Tang for their valuable input and feedback throughout the development of this work. This research is based upon

work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via [2022-21102100005]. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[2] Jack Bresenham. Algorithm for computer control of a digital plotter. *Seminal graphics*, 1965.

[3] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:172–186, 2021.

[4] Bowen Cheng, Bin Xiao, Jingdong Wang, Humphrey Shi, Thomas S. Huang, and Lei Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5385–5394, 2020.

[5] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *ArXiv*, abs/1904.10509, 2019.

[6] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. https://github.com/open-mmlab/mmpose, 2020.

[7] 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. *ArXiv*, abs/2010.11929, 2021.

[8] Mohsen Fayyaz, Soroush Abbasi Koohpayegani, Farnoush Rezaei Jafari, Sunando Sengupta, Hamid Reza Vaezi Joze, Eric Sommerlade, Hamed Pirsiavash, and Juergen Gall. Adaptive token sampling for efficient vision transformers. In *European Conference on Computer Vision*, 2022.

[9] Georgia Gkioxari, Pablo Arbeláez, Lubomir D. Bourdev, and Jitendra Malik. Articulated pose estimation using discriminative armlet classifiers. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3342–3349, 2013.

[10] Saurabh Goyal, Anamitra R. Choudhury, Venkatesan T. Chakaravarthy, Saurabh ManishRaje, Yogish Sabharwal, and Ashish Verma. Power-bert: Accelerating bert inference for classification tasks. *ArXiv*, abs/2001.08950, 2020.

[11] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Doll'ar, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15979–15988, 2022.

[12] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *British Machine Vision Conference*, 2010.

[13] Alberto Lamas, Siham Tabik, Antonio Cano Montes, Francisco Pérez-Hernández, Jorge García-Torres Fernández, Roberto Olmos, and Francisco Herrera. Human pose estimation for mitigating false negatives in weapon detection in video-surveillance. *Neurocomputing*, 489:488–503, 2022.

[14] Jiefeng Li, Can Wang, Hao Zhu, Yihuan Mao, Haoshu Fang, and Cewu Lu. Crowdpose: Efficient crowded scenes pose estimation and a new benchmark. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10855–10864, 2018.

[15] Yanjie Li, Shoukui Zhang, Zhicheng Wang, Sen Yang, Wankou Yang, Shutao Xia, and Erjin Zhou. Tokenpose: Learning keypoint tokens for human pose estimation. In *IEEE/CVF International Conference on Computer Vision*, pages 11293–11302, 2021.

[16] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. *ArXiv*, abs/2202.07800, 2022.

[17] H. Lin and Ting-Wen Chen. Augmented reality with human body interaction based on monocular 3d pose estimation. In *ACIVS*, 2010.

[18] Tsung-Yi Lin, Michael Maire, Serge J. 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*, 2014.

[19] Dmitrii Marin, Jen-Hao Rick Chang, Anurag Ranjan, Anish K. Prabhu, Mohammad Rastegari, and Oncel Tuzel. Token pooling in vision transformers. *ArXiv*, abs/2110.03860, 2021.

[20] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12299–12308, 2021.

[21] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, 2016.

[22] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. *ArXiv*, abs/1611.05424, 2017.

[23] Stepán Obdrzálek, Gregorij Kurillo, Jay J. Han, Richard Ted Abresch, and Ruzena Bajcsy. Real-time human pose detection and tracking for tele-rehabilitation in virtual reality. *Studies in health technology and informatics*, 173:320–4, 2012.

[24] Zizheng Pan, Bohan Zhuang, Jing Liu, Haoyu He, and Jianfei Cai. Scalable vision transformers with hierarchical pooling. *IEEE/CVF International Conference on Computer Vision*, pages 367–376, 2021.

[25] Deva Ramanan. Learning to parse images of articulated bodies. In *Neural Information Processing Systems*, 2006.

[26] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *Neural Information Processing Systems*, 2021.

[27] Aurko Roy, Mohammad Taghi Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.

[28] Michael S. Ryoo, A. J. Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: What can 8 learned tokens do for images and videos? *ArXiv*, abs/2106.11297, 2021.

[29] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[30] Dahu Shi, Xing Wei, Liangqi Li, Ye Ren, and Wenming Tan. End-to-end multi-person pose estimation with transformers. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11059–11068, 2022.

[31] Jan Stenum, Cristina Rossi, and Ryan T. Roemmich. Two-dimensional video-based analysis of human gait using pose estimation. *PLoS Computational Biology*, 17, 2020.

[32] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5686–5696, 2019.

[33] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660, 2014.

[34] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

[35] Edward Vendrow, Duy-Tho Le, and Hamid Rezatofighi. Jrdb-pose: A large-scale dataset for multi-person pose estimation and tracking. *ArXiv*, abs/2210.11940, 2022.

[36] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[37] Jiahong Wu, He Zheng, Bo Zhao, Yixin Li, Baoming Yan, Rui Liang, Wenjia Wang, Shipei Zhou, Guosen Lin, Yanwei Fu, Yizhou Wang, and Yonggang Wang. AI Challenger: A large-scale dataset for going deeper in image understanding. *ArXiv*, abs/1711.06475, 2017.

[38] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *European Conference on Computer Vision*, 2018.

[39] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation. *ArXiv*, abs/2204.12484, 2022.

[40] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Keypoint localization via transformer. *IEEE/CVF International Conference on Computer Vision*, pages 11782–11792, 2021.

[41] Changqian Yu, Bin Xiao, Changxin Gao, Lu Yuan, Lei Zhang, Nong Sang, and Jingdong Wang. Lite-hrnet: A lightweight high-resolution network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10435–10445, 2021.

[42] Yuhui Yuan, Rao Fu, Lang Huang, Weihong Lin, Chao Zhang, Xilin Chen, and Jingdong Wang. Hrformer: High-resolution transformer for dense prediction. *ArXiv*, abs/2110.09408, 2021.

[43] Song-Hai Zhang, Ruilong Li, Xin Dong, Paul L. Rosin, Zixi Cai, Xi Han, Dingcheng Yang, Haozhi Huang, and Shimin Hu. Pose2seg: Detection free human instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 889–898, 2019.

[44] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.