TransPose: Towards Explainable Human Pose Estimation by Transformer

Sen Yang* Zhibin Quan Mu Nie Wankou Yang

Abstract

Deep Convolutional Neural Networks (CNNs) have made remarkable progress on human pose estimation task. However, there is no explicit understanding of how the locations of body keypoints are predicted by CNN, and it is also unknown what spatial dependency relationships between structural variables are learned in the model. To explore these questions, we construct an explainable model named TransPose based on Transformer architecture and low-level convolutional blocks. Given an image, the attention layers built in Transformer can capture long-range spatial relationships between keypoints and explain what dependencies the predicted keypoints locations highly rely on. We analyze the rationality of using attention as the explanation to reveal the spatial dependencies in this task. The revealed dependencies are image-specific and variable for different keypoint types, layer depths, or trained models. The experiments show that TransPose can accurately predict the positions of keypoints. It achieves state-of-the-art performance on COCO dataset, while being more interpretable, lightweight, and efficient than mainstream fully convolutional architectures.

1 Introduction

Deep Convolutional Neural Networks have dominated the field of human pose estimation, with DeepPose [42] being the early classic method. Afterwards, fully convolutional networks such as [45, 25, 45, 27, 48, 6, 29, 46, 38] have become the mainstream by predicting keypoints heatmaps, which *implicitly* learn spatial relationships between body parts. Most prior works take deep CNN as a powerful *black box predictor* and focus on improving the network structure to boost the performances, but what exactly happens inside the model or how these convolutional architectures capture the dependencies between body parts remains unexplained. From the scientific and practical perspectives, such interpretability of the model is imperative. It can provide insights into why the models behave the way they do, increase the safety and transparency of the model, help model users for decision-making, and help researchers understand how the structural variables, *e.g.*, body parts in this task, are jointly related to each other to reach the final prediction.

*All authors are with Southeast University, Nanjing, China. E-mail: yangsenius@seu.edu.cn

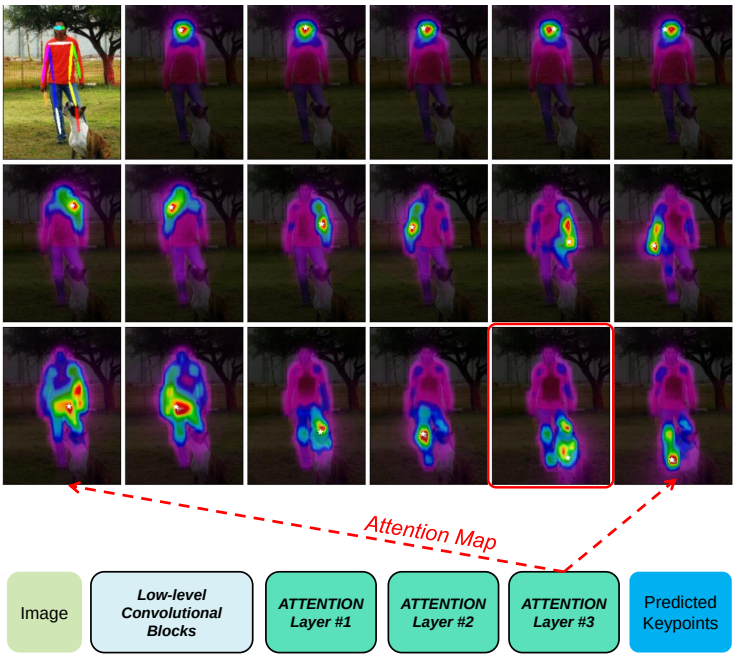


Figure 1: A schematic diagram of TransPose. **Below:** The inference pipeline of the model. **Above:** Dependencies areas for all predicted keypoints by inspecting the attention map from the predicted locations, which are annotated by white pentagrams. In this example, the person's left-ankle is occluded by a dog, but the model can still accurately infer its location. ***Why?*** the attention map (red box) may give a meaningful explanation: the predicted location of the left ankle relies on the image clues provided by the left knee and some joints on the other side of the symmetry.

We empirically summarize some reasons that might hinder the explainability of human pose estimation models.

- *Deepness*. Since the used convolutional neural networks are usually very deep such as [45, 27, 46, 38], one cannot easily figure out what role each layer plays, especially the high-level layers.
- *Relationships and features are coupled*. The global spatial relationships between body parts are not only implicitly encoded in the parameters of convolution kernels, but also expressed in the activations of neurons. Only looking at the intermediate feature activations like [22, 52] cannot reveal the spatial relationship. It is also challenging to decouple and understand these coupled relationships solely from numerical values.
- *Limited memory and expressiveness for large numbers of images*. The explanations we expect should be variable, image-specific, and fine-grained.

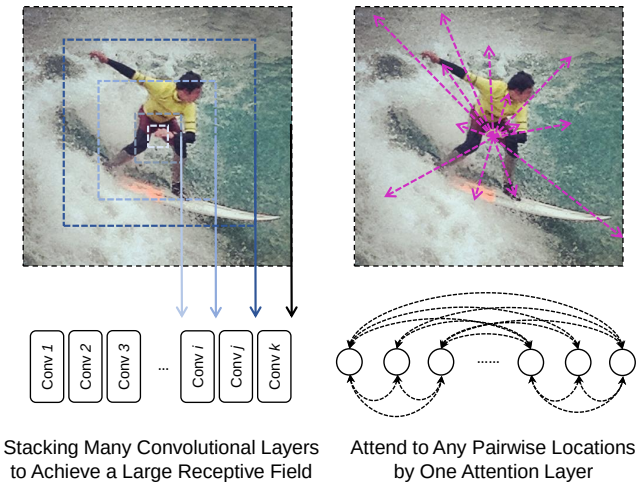


Figure 2: CNN vs. Attention. **Left:** The receptive filed enlarges in the deeper convolutional layer. **Right:** One self-attention layer can capture the pairwise relationship between any pair of locations.

When CNN does inference, however, the *static* convolution kernels parameters are limited in the ability to represent variables due to their limited working memory [15, 16, 18], thereby making it difficult for CNN to explicitly express the variability of the image-dependent spatial dependencies as the postures of the human body vary in a large joint configuration space.

- *Lack of tools.* Although there are already visualization techniques like Maximizing Activations [11], DeconvNet [49], Saliency Map [37, 13], Feature Visualization [28], or Class Activation Mapping [54], etc., most of them aim to find out class-specific input patterns or saliency maps rather than to explain the fine-grained dependencies between related and structural predicted variables. By far, how to construct or develop explainable pose estimation models is still an open question.

In this work, we aim to build a human pose estimation model that can explain its predictions and directly reveal the image-dependent spatial relationships between keypoints, as illustrated in Fig. 1. We argue that convolution has advantages in extracting low-level features, but deeply stacking convolutional layers at high-level to enlarge the receptive field is not efficient to capture global dependencies. And such deepness increases the difficulty in interpreting CNN predictions. The Transformer architecture [44] commonly used in NLP tasks, has a natural advantage over convolutional architectures in terms of drawing pairwise or higher-order interactions. As shown in Fig. 2, one attention layer can capture interactions between any pairwise locations, and the attention map acts as an immediate memory to store these dependencies. Thus

Transformer would be eligible for human pose estimation since this task requires the model to be able to capture long-range dependencies.

Based on these considerations, we design a novel model called *TransPose*, using convolutions to extract features at low-level and Transformer encoder layers to capture global dependencies at high-level. More importantly, the attention maps store the pairwise dependencies measured by the similarities of query-keys transformed from feature vectors at pairs of locations, the model can explain which regions the predicted keypoints locations highly rely on. Given a specific image, we can explicitly understand what image clues significantly contribute to the predictions (maximum activation position) and which parts are affected by the other ones, by unfolding and visualizing the attention map in each attention layer.

TransPose model is simple and lightweight. It can precisely predict keypoints positions based on heatmaps and obtain excellent performances. It is on par with state-of-the-art – SimpleBaseline [46], HRNet [38], and DarkPose [50] baseline via fewer model parameters and faster speeds. In summary, our contributions are as follow:

- To the best of our knowledge, we are the first to use Transformer architecture to capture the spatial relationships between structural human body parts.
- TransPose can explain what spatial dependencies the predicted keypoints rely on. Qualitative analysis reveals the fine-grained spatial dependencies, which are image-specific and variable for different keypoint types, layer depths, and trained models.
- TransPose achieves 75.8 AP and 75.0 AP on COCO validation set and test-dev set, with 73% fewer parameters and 1.4× faster than HRNet-W48.
- We find that position embedding is important for accurately predicting the 2D positions of keypoints and helps model generalize better on unseen input resolutions.

2 Related Works

2.1 Explainability

Explainability means a better understanding for human of how the model makes predictions. Many works define the goal for explanation is often to determine what inputs are the most relevant to the prediction, as surveyed by [35]. Activation Maximizing [11, 22] performs gradient descent in the input space to find out what input patterns can maximize (explain) a given unit. [37, 12] further consider generating the image-specific class saliency maps. [49] uses DeConvNet to generate feature activities to show what has been learned in each convolutional layer. [22] visualizes the feature maps and finds that some maximally activated neurons work like body parts patterns/templates

detector. [52] uses channel-wise attention mechanism to find out the channels of feature activations associated with the occlusion patterns for pedestrian detection. There are also works like Network Dissection [2], Feature Visualization [28], Excitation Backprop [51], LRP [1], CAM [54], and Grad-CAM [36], which aim to explain the prediction of CNN classifier or visualize the saliency area significantly affecting the class. Different from most prior works, we aim to explain the fine-grained dependencies between body joints variables in the structural skeleton. And our model can directly exploit the ingredient of itself to explain predictions without the help of external tools. It is worth noting that there are some works to explain how the neural network predicts the positions and stores the position information by designing proxy tasks such as CoordConv [24] and Zero Padding [19]. Our work is carried out on real-world task, explores the importance of position embedding for predicting the locations and its generalization on unseen input scales.

2.2 Human Pose Estimation

Deep CNNs have achieved great success in human pose estimation. The inductive biases of vanilla convolution kernel [21, 20] are locality and translation equivariance. It proves to be efficient to extract low-level image feature. For human pose estimation, capturing global dependencies is crucial [34, 41, 45, 29], but the locality nature of convolution makes it impossible to capture long-range interactions. To address this issue, an effective but brute solution is to enlarge the receptive field, *e.g.* by downsampling feature map size, increasing the model depth or expanding the kernel size, etc. Further, more sophisticated design strategies are proposed such as multi-scale feature fusion [27, 32, 48, 6, 38, 9, 7], stacking modules and multiple stages to refine the prediction [45, 46, 27], or high-resolution representation [38], etc; meanwhile, many successful architectures have emerged such as CPM [45], Stacked Hourglass Network [27], FPN [48], CPN [6], SimpleBaseline [46], HRNet [38], RSN [4], even automated architectures [47, 14, 26, 8, 53]. But as the model architecture design becomes more complex, it is more challenging but imperative than ever to seek the transparency and interpretability of the human pose estimation model. Our model, in contrast, can capture the spatial relationship in an efficient yet interpretable way.

2.3 Transformer

Transformer architecture was proposed by Vaswani *et al.* [44] for neural machine translation (NMT) task [40]. Recently, Transformer or attention-augmented layers have merged as new choices for vision tasks such as image classification [31, 33, 3, 10, 43] and object detection [5]. Unlike DETR [5], ViT [10], and DeiT [43] applying Transformer to predict the object instances set or image categories, we use Transformer to predict the heatmaps represented with 2D spatial distributions of keypoints, and use attention maps as the explanations for this 2D-structural prediction task.

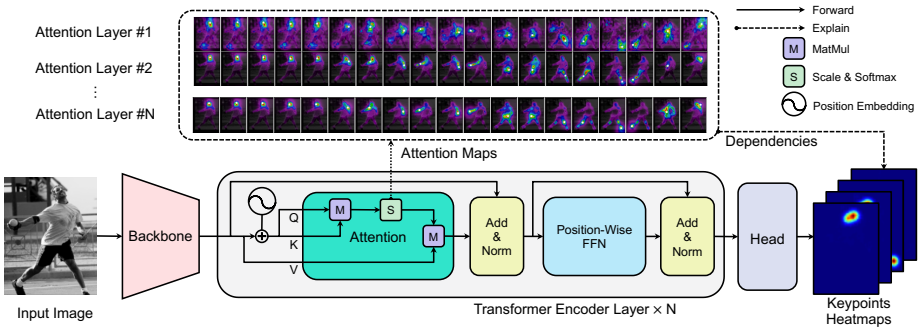


Figure 3: The TransPose model architecture. Firstly, the feature maps are extracted by a CNN backbone and flattened into a sequence. Next, the Transformer encode layers iteratively capture dependencies from the current sequence by query-key-value attention. Then, a simple head is used to predict the keypoints heatmaps. The model can explain what regions or joints significantly contribute to the predicted locations by the attention maps in Transformer encoder layers, and the spatial dependencies can be further revealed.

3 Method

Our goal is to build a model that can capture the spatial dependencies between human body parts and explain its predictions. In this section, we describe the architecture of the proposed model, and then we recall the attention mechanism and further analyze how it explains what the predicted keypoints depend on when given a specific image.

3.1 Architecture

As illustrated in Fig. 3, TransPose model consists of three components: a CNN backbone to extract low-level image feature; a Transformer Encoder to capture long-range spatial interactions between feature vectors across the locations; a head to predict the keypoints heatmaps.

Backbone. Many common CNNs can be taken as the backbone. For better comparisons, we choose two typical CNN architectures: ResNet [17] and HRNet [38]. To keep the simplicity of the model, we only retain the initial part of the original ImageNet pretrained CNN as the initial layers to extract low-level feature from input image. We name them ResNet-S and HRNet-S (including HRNet-S-W32 and HRNet-S-W48). The numbers of parameters of candidate CNNs are only 1.4M, 7.3M and 16.7M, which are 5.5%, 25.6% and 24.3% of the original ResNet-50 (25.6M), HRNet-W32 (28.5M), and HRNet-W48 (68.6M).

Transformer. We follow the standard Transformer architecture [44] as closely as possible. And only the encoder is employed, as we believe that pure heatmaps prediction task is simply an encoding task, which compresses the original im-

age information into a compact positions representation of human body keypoints. Given an input image $I \in \mathbb{R}^{3\times H_I \times W_I}$, we assume that the CNN backbone outputs a 2D spatial structure image feature $\mathbf{X}_f \in \mathbb{R}^{d\times H\times W}$ whose feature dimension has been transformed to d by a 1×1 convolution. Then, the image feature map is flattened into a sequence $\mathbf{X} \in \mathbb{R}^{L\times d}$, *i.e.*, L d-dimensional feature vectors where $L = H \times W$. It enters Transformer Encoder and goes through N attention layers and feed-forward networks (FFN). We provide details in Section 3.4.

Head. A simple and lightweight head is attached to Transformer Encoder output $\mathbf{E} \in \mathbb{R}^{L\times d}$ to predict K types of keypoints heatmaps $P \in \mathbb{R}^{K\times H^*\times W^*}$ where $H^*, W^* = H_I/4, W_I/4$ by default. We firstly reshape $\mathbf{E}$ back to $\mathbb{R}^{C\times H\times W}$ shape. If H, W equal H^*, W^*, only a 1×1 convolution reduces the channel dimension of $\mathbf{E}$ from d to K; if H, W equal $H^*/2, W^*/2$, we use a bilinear interpolation or a 4×4 deconvolution to do $2\times$ upsampling before 1×1 convolution[1].

3.2 Resolution Settings.

The computational complexity of per self-attention layer is $O\left((HW)^2 \cdot d\right)$. Considering the trade-off between the memory footprint for attention layers and the loss in detailed information, we restrict the attention layers to operate at a resolution with $r\times$ downsampling rate w.r.t. the original input, *i.e.*, $H, W = H_I/r, W_I/r$; we adopt $r = 8$ and $r = 4$ setting for ResNet-S and HRNet-S backbone. In the common human pose estimation architectures [45, 27, 46, 38], $32\times$ downsampling is usually adopted as a standard setting to obtain a very low resolution map containing global information. By contrast, our model can directly capture long-range interactions at a higher resolution, while preserving the fine-grained local feature information.

3.3 Position Embedding

Without the position information embedded in the input sequence, the Transformer Encoder is a permutation-equivariant architecture:

$$\text{Encoder}\left(\rho\left(\mathbf{X}\right)\right) = \rho\left(\text{Encoder}\left(\mathbf{X}\right)\right), \tag{1}$$

where ρ is any permutation for the pixel locations or the order of sequence. To make the order of sequence or the spatial structure of the image feature map matter, the original Transformer adds sine and cosine positional encodings to the input embeddings.

2D Sine position embedding. Likewise, we follow the sine positional encodings [44] but further hypothesize that the position information is independent at x (horizontal) and y (vertical) direction of an image. Thus we adopt both x-direction and y-direction position embedding for image feature, like [31, 5]. It is injected into the input sequence before self-attention layer.

[1] In fact, a 1×1 convolution is completely equivalent to a position-wise linear layer that reduces the dimension of vector in each position of the output sequence.

Model Name	Backbone	Resolution for Attention	Upsampling	#layers	heads	d	h	Params
TransPose-R*	ResNet-S*	1/8	Bilinear Interpolation	3	8	256	512	5.0M
TransPose-R-A3	ResNet-S	1/8	Deconvolution	3	8	256	1024	5.2M
TransPose-R-A4	ResNet-S	1/8	Deconvolution	4	8	256	1024	6.0M
TransPose-H-S	HRNet-S-W32	1/4	None	4	1	64	128	8.0M
TransPose-H-A4	HRNet-S-W48	1/4	None	4	1	96	192	17.3M
TransPose-H-A6	HRNet-S-W48	1/4	None	6	1	96	192	17.5M

Table 1: Architecture configurations for different TransPose models. More details about the backbones are described in Appendix.

Method	Input Size	AP	AR	Params	FLOPs	FPS
SimpleBaseline-Res50 [46]	256×192	70.4	76.3	34.0M	8.9G	114
SimpleBaseline-Res50 + DarkPose	256×192	72.0	77.6	34.0M	8.9G	114
TransPose-R*	256×192	71.5	76.9	5.0M (↓85%)	5.4G	137 (↑20%)
TransPose-R-A3	256×192	71.7	77.1	5.2M (↓85%)	8.0G	141 (↑23%)
TransPose-R-A4	256×192	**72.6**	**78.0**	6.0M (↓82%)	8.9G	138 (↑21%)
HRNet-W32 [38]	256×192	74.4	73.7	28.5M	7.2G	28
HRNet-W32 + DarkPose [50]	256×192	75.6	76.7	28.5M	7.2G	28
HRNet-W48 [38]	256×192	75.1	80.4	63.6M	14.6G	27
TransPose-H-S	256×192	74.2	78.0	8.0M (↓72%)	10.2G	45 (↑61%)
TransPose-H-A4	256×192	75.3	80.3	17.3M (↓73%)	17.5G	41 (↑52%)
TransPose-H-A6	256×192	**75.8**	**80.8**	17.5M (↓73%)	21.8G	38 (↑41%)

Table 2: Results on COCO validation set, provided with the same detected human boxes. TransPose-R, TransPose-H-S, and TransPose-H-A* achieve competitive results to SimpleBaseline, HRNet, and HRNet-DarkPose, with fewer parameters and faster speeds.

We use 2D sine position embedding by default for all TransPose models. See Appendix for how to construct 2D sine position embedding.

Learnable position embedding and w/o position embedding. In experiments, we also use a learnable position embedding LPE $\in \mathbb{R}^{L\times d}$ to explore whether the model can implicitly learn the position information. In addition, we also conduct the experiment without adding any position information to explore the importance of the position information for predicting 2D-structure heatmaps.

3.4 Explaining by Attention

Attention mechanism. The core mechanism of Transformer [44] is multi-head self-attention. It first projects an input sequence $\mathbf{X} \in \mathbb{R}^{L\times d}$ into queries $\mathbf{Q} \in \mathbb{R}^{L\times d}$, keys $\mathbf{K} \in \mathbb{R}^{L\times d}$ and values $\mathbf{V} \in \mathbb{R}^{L\times d}$ by three matrices $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{d\times d}$. Note that in this work the position embedding will be added into the

input sequences except for computing the values $\mathbf{V}$. Then, the attention scores matrix[2] $\mathbf{A} \in \mathbb{R}^{N \times N}$ is computed by:

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right). \tag{2}$$

Each query $\boldsymbol{q}_i \in \mathbb{R}^d$ belonging to the token (feature vector) $\boldsymbol{x}_i = \mathbf{X}_i^\top$ computes similarities with all the keys to achieve a weight vector $\mathbf{w}_i = \mathbf{A}_{i,:} \in \mathbb{R}^{1 \times L}$, which determines how much dependency is needed from each token in the previous sequence for a new token. Then an incremental update (residual connection) for $\boldsymbol{x}_i$ is achieved by linearly combining each value in Value matrix $\mathbf{Q}$ with the corresponding weight in $\mathbf{w}_i$. By doing this, the attention maps can be seen as *dynamic weights* that store the similarities dependent on the current input context or feature activations, and weight the distributions in the forward propagation. This mechanism plays a crucial role in capturing and explaining how much contribution the final predictions aggregate from the context token at each location of the sequence. Note that *the pairwise and global interactions mostly occur at the attention layers*[3]. The subsequent layers in feed-forward network (FFN) and head only serve as *position-wise* transformation. They are unable to capture global interactions but approximately linearly transform the contributions from all positions by shared weights.

Specifically, the last attention layer in Transformer Encoder, whose attention scores are seen as the dynamic (*image-dependent*) weights, has the most direct effect on the predictions. Although the weights in FFN or head cannot be ignored, they are static (*image-independent*) during inference and shared across all locations. We make *gradient analysis* in Appendix G to analyze the rationality of using the attention scores of the last attention layer as the explanations for this task.

Explaining the predicted locations by attention maps. Similar with the ideas in [11, 37], the interpretability behind TransPose lies in: the regions which can maximize a given prediction (location) can explain what this prediction is looking for or depend on.

In this task, the learning target is to expect the output value h_{i^*} in the heatmap to be maximally activated where i^* represents the groundtruth location of a keypoint:

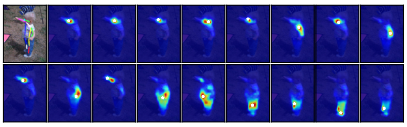
$$\theta^* = \arg\max_\theta h_{i^*}(\theta, I). \tag{3}$$

Assuming the model has been optimized with parameters θ^* and it predicts the location of a particular keypoint as i (maximally activated in heatmaps), why the model predicts such prediction can be explained by the fact that those locations $\mathbf{J}$, whose element j has higher attention score ($\geq \delta$) with i, are the dependencies that significantly contribute to the prediction. These locations can be found by:

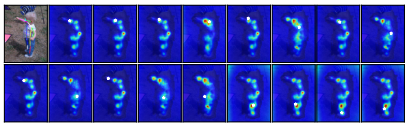
$$\mathbf{J} = \left\{j | \mathbf{A}_{i,j}\left(\theta^*, I\right) \geq \delta\right\}, \tag{4}$$

[2]Here we consider single-head self attention. For multi-head self-attention, the attention matrix is the average of attention maps in all heads.

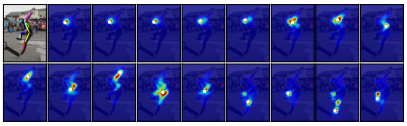
[3]We assume that the used convolutions are responsible to extract feature in a limited local patch.



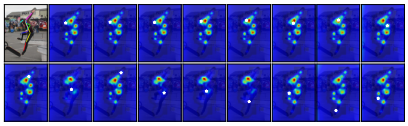
(a) **TP-R:** predictions and dependencies of each keypoints for **input A**.



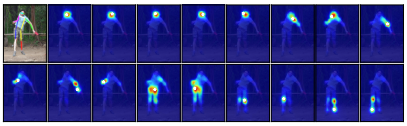
(b) **TP-H:** predictions and dependencies of each keypoint for **input A**.



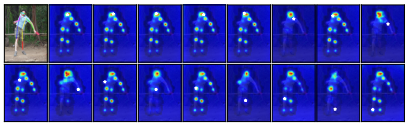
(c) **TP-R:** predictions and dependencies of each keypoint for **input B**.



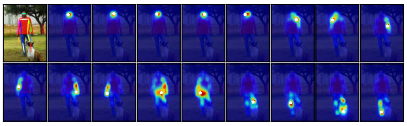
(d) **TP-H:** predictions and dependencies of each keypoint for **input B**.



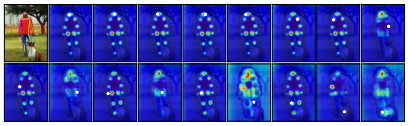
(e) **TP-R:** predictions and dependencies of each keypoint for **input C**.



(f) **TP-H:** predictions and dependencies of each keypoints for **input C**.



(g) **TP-R:** predictions and dependencies of each keypoint for **input D**.



(h) **TP-H:** predictions and dependencies of each keypoints for **input D**.

Figure 4: In each sub-figure, the first one is the original input plotted with predicted skeleton. The other maps are the unfolded attention maps in the final layer of TP-R and TP-H inspected by the predicted locations of keypoints, each position of which is annotated by a white pentagram. All the maps are visualized by the unnormalized attention maps.

where $\mathbf{A} \in \mathbb{R}^{L \times L}$ is the attention map of the last attention layer and also a function w.r.t θ^* and I. Given an image I and a query location i, $\mathbf{A}_{i,:}$ can reveal what dependencies a predicted location i highly relies on, we named it *dependency area*; $\mathbf{A}_{:,j}$ can reveal what positions a specified location j mostly affects, we name it *affected area*.

Different from Activation Maximizing methods [11, 22, 49] or Saliency Map Visualization [37], we do not need external tools or extra training costs to learn some explainable patterns, but only need to inspect the ingredients of the trained models to reveal what spatial relationships the TransPose has learned. In addition, unlike the visualizations on the fixed patterns that the convolutional kernels prefer to look for, the dynamic characteristic of attention map can show changeable dependencies w.r.t input images, layer depths, types of keypoints or trained models. We make further analysis in the Section 4.2.

4 Experiments

Dataset. We evaluate our models on COCO dataset [23], which contains more than 200k images in the wild and 250k person instances. We only use COCO train2017 for training, which consists of 57k images and 150k person instances. Val2017 set contains 5k images and test-dev2017 consists of 20k images. Object keypoint similarity (OKS) is the evaluation metric for keypoints locating accuracy [4].

Technical details. We follow the top-down human pose estimation paradigm. The training samples are the cropped images with single person. We resize all input images into 256×192 resolution. We use the same training strategies, data augmentation and person detected results as [38]. We also adopt the coordinate decoding strategy proposed by [50] to reduce the quantisation error when decoding from downscaled heatmaps. The feed forward layers are trained with 0.1 dropout and ReLU activate function. Next, we name the models based on ResNet-S and HRNet-S *TransPose-R* and *TransPose-H*, abbreviated as ***TP-R*** and ***TP-H***. The architecture details are reported in the Tab. 1. We use Adam optimizer for all models. Training epochs are 230 for TP-R and 240 for TP-H. The cosine annealing learning rate decay is used. The learning rates for TP-R-A4 and TP-H-A6 models decay from 0.0001 to 0.00001, we recommend to use such a schedule for all models. Considering the compatibility with backbone and the memory consumption, we adjust the hyperparameters of Transformer encoder to make the model capacity not very large.

Comparisons with state-of-the-art methods. We compare TransPose based on ResNet-S and HRNet-S with SimpleBaseline, HRNet and the stronger models proposed by [50]. Specially, we trained the SimpleBaseline-ResNet50-DarkPose on our machines according to the official code. The others results showed in Tab. 2 are come from the papers. We test all models on a single NVIDIA 2080Ti GPU with the same experimental conditions to compute the average FPS. As shown in Tab. 2, TransPose-R-A4 and TransPose-H-A6 have obviously overperformed Simple-Res50-DarkPose (+0.6AP), HRNet-W32-Dark (+0.2AP), and HRNet-W48 (+0.7AP) baseline models, with significantly fewer model parameters and faster speeds. Tab. 4 shows the results on COCO test set.

Position Embedding	Params	FLOPs	AP
✗	4.999M	7.975G	70.4
Learnable	5.195M	7.976G	70.9
2D Sine (Fixed)	5.195M	7.976G	71.7

Table 3: Results for different position embedding schemes for TransPose models. The input size is 256×192.

[4] http://cocodataset.org/

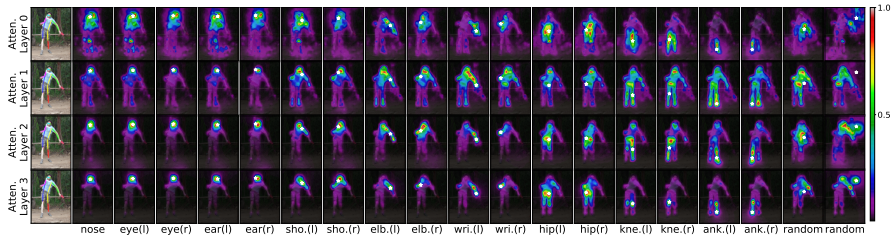


Figure 5: **Dependency areas** for the particular positions in the different layers of **TP-R** for **input E**.

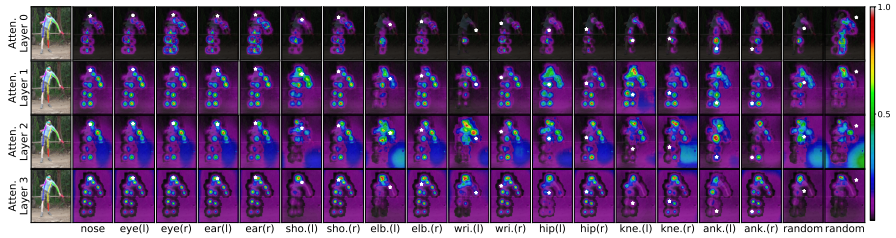
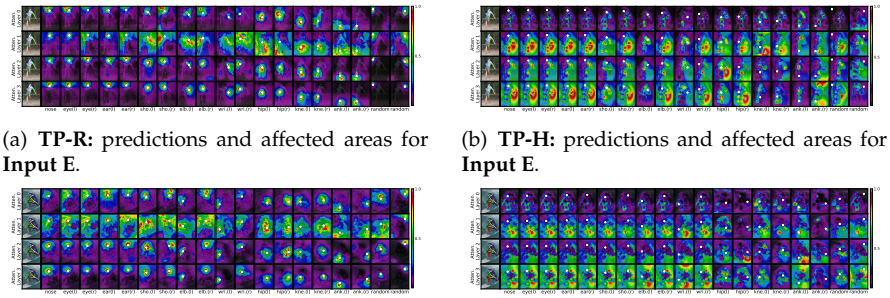


Figure 6: **Dependency areas** for the particular positions in the different layers of **TP-H** for **input E**.

4.1 Ablations

The importance of position embedding. Without position embedding, the 2D spatial structure information loses in Transformer. To explore the importance of position embedding, we conduct experiments on TransPose-R-A3 models with three position embedding strategies: 2D sine position embedding, learnable position embedding, and no position embedding. As expected, the models with position embedding have better performances, particularly for 2D sine position embedding, as shown in Tab. 3. But note that TransPose without position embedding can also perform well with only losing 1.3 AP. The reason perhaps is that the sparse position encoding in the target heatmaps makes the 2D spatial structure less important. We also observe what has been learned in the learnable position embedding, by computing the cosine similarities between the vectors in any paired locations of the learnable position embedding. The results (see Appendix) indicate that each vector in the learnable embedding has similar values with its neighbours in the 2D grid, which means the coarse 2D position information is learned and exploited in the model.

Position embedding helps generalize better on unseen input resolutions. The top-down paradigm scales all the cropped images containing a single person to a fixed size. But for some cases even with a fixed input size or the bottom-up multi-person pose estimation paradigm, the scale of the human body in the input might be various, the robustness of handling with different scales is also important. To test the generalization of models, we design an extreme experiment: we test SimpleBaseline-ResNet50-DarkPose and TransPose-R-A3 models on unseen 128×96, 384×288, 512×388 input resolutions, all of



(a) **TP-R:** predictions and affected areas for **Input E**.

(b) **TP-H:** predictions and affected areas for **Input E**.

(c) **TP-R:** predictions and affected areas for **Input F**.

(d) **TP-H:** predictions and affected areas for **Input F**.

Figure 7: **Affected areas** for the particular positions in the different depths of attention layer.

Method	Input size	Params	FLOPs	AP	$AP_{0.5}$	$AP_{0.75}$	AP_M	AP_L	AR
G-RMI [30]	353×257	42.6M	57.0G	64.9	85.5	71.3	62.3	70.0	69.7
Integral [39]	256×256	45.0M	11.0G	67.8	88.2	74.8	63.9	74.0	-
CPN [6]	384×288	-	-	72.1	91.4	80.0	68.7	77.2	78.5
RMPE [12]	320×256	28.1M	26.7G	72.3	89.2	79.1	68.0	78.6	-
SimpleBaseline [46]	384×288	68.6M	35.6G	73.7	91.9	81.1	70.3	80.0	79.0
HRNet-W32 [38]	384×288	28.5M	16.0G	74.9	92.5	82.8	71.3	80.9	80.1
HRNet-W48 [38]	384×288	63.6M	32.9G	75.5	92.5	83.3	71.9	81.5	80.5
DarkPose [50]	384×288	63.6M	32.9G	76.2	92.5	83.6	72.5	82.4	81.1
TransPose-H-S	256×192	5.2M	10.3G	73.4	91.6	81.1	70.1	79.3	78.6
TransPose-H-A4	256×192	17.3M	17.5G	74.7	91.9	82.2	71.4	80.7	79.9
TransPose-H-A6	256×192	17.5M	21.8G	75.0	92.2	82.3	71.3	81.1	80.1

Table 4: Comparisons with state-of-the-art on COCO test-dev set.

which only have been trained with 256×192 input images. Interestingly, the results in Fig. 8 demonstrate that SimpleBaseline and TransPose-R without position embedding have obvious performance collapses on these unseen input resolutions, particularly on 128×96, but TransPose-R with learnable or 2D Sine position embedding have significantly better generalization, especially for the Sine position embedding with minimal performance degradation. We conjecture that 1) it is hard for the models with a fixed receptive field to adapt the changes in the feature scale; 2) building associations with *relative position information* encoded in Sine position embedding [44] may help the model generalize on different sizes of inputs or feature maps.

4.2 Explainability Analysis

In this section, we show the dependencies exhibited by the attention maps are dynamic across different trained models, types of predicted keypoints, depths of attention layers, and input images. We choose cropped images by GT boxes

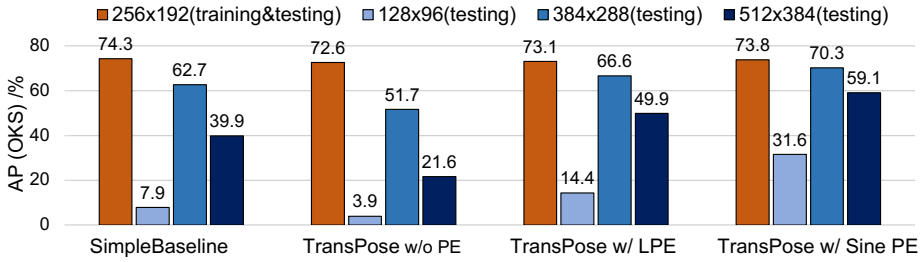


Figure 8: Generalization performances on unseen input resolutions. SimpleBaseline and TransPose without Position Embedding (PE) generalize worse than TranPose with PE obviously.

from COCO val dataset and use the trained models TP-R (TP-R-A4 model) and TP-H (TransPose-H-S model) with 75.1AP and 76.1 AP performances as exemplars, to make qualitative analysis by controlling variables.

The ranges of dependencies for predictions are longer in the model with higher performance. The attention map in the final attention layer directly reflects the dependencies for predicting the keypoints locations. For example, comparing Fig. 4(a) with Fig. 4(b), although TP-R and TP-H predict exactly the same locations of keypoints in the input A, TP-H obviously exploits the image cues from the longer-distance joints to predict most keypoints. In contrast, TP-R prefers to attend to the short-range (local) image cues around the target keypoint. This characteristic can be further confirmed by the affected areas illustrated in Fig. 7, in which the keypoints have larger affected areas in TP-H. The larger receptive field and information fusion of HRNet-S might account for this.

Dependencies and influences vary for different types of keypoints. Overall, the upper limb and the keypoints belonging to it, especially the head, have a greater impact on predicting positions of keypoints in the lower limb. Such influences are more obvious in TP-H, mainly due to its stronger long-distance capturing ability than TP-R model. As shown in Fig. 4(b), Fig. 4(d), Fig. 4(h), Fig. 4(f), and Fig. 6, we can further observe that a *well-performed model might gather more significant clues from more other parts to predict the target keypoint*. We thus suppose that learning strong associations between accurate keypoints locations can reinforce the model to localize any one of them more accurately. This can explain why the model still can predict the location of an occluded keypoint accurately, and the occluded keypoint with ambiguity feature would have less impact on the other predictions, such as the left-ankle joint shown in Fig. 4(g) and Fig. 4(h). Moreover, the joints with a high degree of freedom, *e.g.*, elbows, wrists, knees, and ankles, also depend on the nearby limbs and joints on the symmetrical side.

Attentions gradually focus on the more fine-grained dependencies with the depth of layer increasing. Observing all attention layers, as shown in the 1,2,3-th rows of Fig. 6, we surprisingly find that *even without the intermediate*

GT locations supervision, TP-H model can still attend to the accurate locations of joints in the early layers. For both models, with the depth of layer increasing, the predictions gradually depend on the more fine-grained body part image clues or keypoints positions, as shown in Fig. 5 and Fig. 6.

The dependencies slightly change for different input image context. Different from the static dependencies encoded in the weights and bias of CNN after training, the attention maps are dynamic to inputs. By choosing a model alone and testing different inputs for it, *e.g.*, input A, B, C or D for TP-R in Fig. 10, we can observe that despite the statistical commonalities on the dependency relationships for the predicted keypoints, the fine-grained dependencies would slightly change according to the image context. With the existence of occlusion or invisibility in a given image such as input D (Fig. 4(h)), the model can still localize the position of the partially obscured keypoint by looking for more significant image clues and reduces reliance on the invisible keypoint to predict the other ones. Such a dynamic characteristic can depict the unique dependencies existing in an unique image when inferring.

5 Conclusion

In this work, we explored using Transformer encoder with low-level convolutional blocks to construct an explainable human pose estimation model. We analyzed and showed that the attention mechanism is very promising to capture and explain the image-dependent spatial relationships in this structure prediction task. With lightweight architectures, TransPose models match the state-of-the-art on COCO Keypoint Detection task that has been dominated by deep fully convolutional architectures, and there seems to have further space to improve the upper limit of model performance by expanding the size of TransPose. Furthermore, we validate the importance of position embedding, and our qualitative analysis reveals the image-specific and variable spatial dependencies for different layer depths, keypoints types, or trained models. We believe that these ideas could help researchers or users have a deeper understanding of how neural networks make structural predictions for human pose estimation task, and further improve the model design.

References

[1] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 5, 22

[2] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *CVPR*, pages 3319–3327, 2017. 5

[3] Irwan Bello, Barret Zoph, Quoc Le, Ashish Vaswani, and Jonathon Shlens. Attention augmented convolutional networks. In *ICCV*, pages 3286–3295, 2019. 5

[4] Yuanhao Cai, Zhicheng Wang, Zhengxiong Luo, Binyi Yin, Angang Du, Haoqian Wang, Xinyu Zhou, Erjin Zhou, Xiangyu Zhang, and Jian Sun. Learning delicate local representations for multi-person pose estimation. In *ECCV*, 2020. 5

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, Cham, 2020. 5, 7

[6] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *CVPR*, pages 7103–7112, 2018. 1, 5, 13

[7] Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S. Huang, and Lei Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation. In *CVPR*, June 2020. 5

[8] Hsin-Pai Cheng, Feng Liang, Meng Li, Bowen Cheng, Feng Yan, Hai Li, Vikas Chandra, and Yiran Chen. Scalenas: One-shot learning of scale-aware representations for visual recognition. *arXiv preprint arXiv:2011.14584*, 2020. 5

[9] Xiao Chu, Wei Yang, Wanli Ouyang, Cheng Ma, Alan L. Yuille, and Xiaogang Wang. Multi-context attention for human pose estimation. In *CVPR*, pages 5669–5678, 2017. 5

[10] 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 preprint arXiv:2010.11929*, 2020. 5

[11] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *Technical Report, University of Montreal*, 1341(3):1, 2009. 3, 4, 9, 10

[12] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *ICCV*, pages 2334–2343, 2017. 4, 13

[13] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*, pages 3449–3457, 2017. 3

[14] Xinyu Gong, Wuyang Chen, Yifan Jiang, Ye Yuan, Xianming Liu, Qian Zhang, Yuan Li, and Zhangyang Wang. Autopose: Searching multi-scale branch aggregation for pose estimation. *arXiv preprint arXiv:2008.07018*, 2020. 5

[15] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014. 3

[16] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016. 3

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6, 21

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[19] Amirul Islam, Sen Jia, and Neil D. B. Bruce. How much position information do convolutional neural networks encode. In *ICLR*, 2020. 5

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012. 5

[21] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[22] Sijin Li, Zhi-Qiang Liu, and Antoni B. Chan. Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network. *IJCV*, 113(1):19–36, 2015. 2, 4, 10

[23] 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 *ECCV*, pages 740–755. Springer, 2014. 11

[24] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *NeurIPS*, pages 9605–9616, 2018. 5

[25] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 1

[26] William McNally, Kanav Vats, Alexander Wong, and John McPhee. Evopose2d: Pushing the boundaries of 2d human pose estimation using neuroevolution. *arXiv preprint arXiv:2011.08446*, 2020. 5

[27] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499. Springer, 2016. 1, 2, 5, 7

[28] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization. 3, 5

[29] George Papandreou, Tyler Zhu, Liang-Chieh Chen, Spyros Gidaris, Jonathan Tompson, and Kevin Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *ECCV*, 2018. 1, 5

[30] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *CVPR*, pages 4903–4911, 2017. 13

[31] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018. 5, 7

[32] Tomas Pfister, James Charles, and Andrew Zisserman. Flowing convnets for human pose estimation in videos. In *ICCV*, pages 1913–1921, 2015. 5

[33] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NIPS*, pages 68–80, 2019. 5

[34] Varun Ramakrishna, Daniel Munoz, Martial Hebert, James Andrew Bagnell, and Yaser Sheikh. Pose machines: Articulated pose estimation via inference machines. volume 8690, pages 33–47, 2014. 5

[35] Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature, 2019. 4

[36] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017. 5, 22

[37] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR (Workshop Poster)*, 2013. 3, 4, 9, 10, 22

[38] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, June 2019. 1, 2, 4, 5, 6, 7, 8, 11, 13, 20, 21

[39] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. Integral human pose regression. In *ECCV*, pages 529–545, 2018. 13

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NIPS*, volume 27, pages 3104–3112, 2014. 5

[41] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *NeurIPS*, pages 1799–1807, 2014. 5

[42] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, pages 1653–1660, 2014. 1

[43] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020. 5

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 3, 5, 6, 7, 8, 13, 20

[45] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *CVPR*, pages 4724–4732, 2016. 1, 2, 5, 7

[46] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *ECCV*, pages 466–481, 2018. 1, 2, 4, 5, 7, 8, 13

[47] Sen Yang, Wankou Yang, and Zhen Cui. Pose neural fabrics search. *arXiv preprint arXiv:1909.07068*, 2019. 5

[48] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation. In *ICCV*, pages 1281–1290, 2017. 1, 5

[49] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014. 3, 4, 10

[50] Feng Zhang, Xiatian Zhu, Hanbin Dai, Mao Ye, and Ce Zhu. Distribution-aware coordinate representation for human pose estimation. In *CVPR*, pages 7093–7102, 2020. 4, 8, 11, 13

[51] Jianming Zhang, Zhe L. Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. In *ECCV*, pages 543–559, 2016. 5

[52] Shanshan Zhang, Jian Yang, and Bernt Schiele. Occluded pedestrian detection through guided attention in cnns. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6995–7003, 2018. 2, 5

[53] Wenqiang Zhang, Jiemin Fang, Xinggang Wang, and Wenyu Liu. Efficientpose: Efficient human pose estimation with neural architecture search. *arXiv preprint arXiv:2012.07086*, 2020. 5

[54] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929, 2016. 3, 5

A What position information has been learned in the TransPose model with learnable position embedding?

We show what position information has been learned in the TransPose (TransPose-R) with learnable position embedding. It has been discussed in the Section 4.2

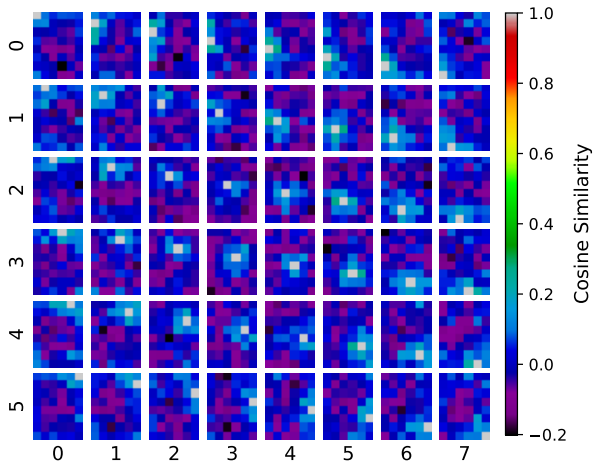


Figure 9: The cosine similarities between the learned position embedding vectors, which have been reshaped into 2D grid and interpolated with 0.25 scale factor for a better illustration (the original shape is $(24, 32)$). Each map in x-row and y-col of the figure represents the cosine similarities between the embedding vector in position (x, y) and the embedding vectors at other locations.

of the paper. As shown in Fig. 9, we visualize the similarities by calculating the cosine similarity between vectors at any pair of locations of the learnable position embedding and reshaping it into a 2D grid-like map. We find that the embedding in each location of learnable position embedding has a unique vector value in the d-dim vector space, but it has relatively higher cosine similarity values with the neighbour locations in 2D-grid and lower values with those far away from it. The results indicate the coarse 2D position information has been implicitly learned in the learnable position embedding. We suppose that the learning sources of the position information might be the 2D-structure groundtruth heatmaps and the similar features existing in the 1D-structure sequences. The model learns to build associations between position embedding and input sequences, as a result it can predict the target heatmaps with 2D Gaussian peaking at groundtruth keypoints locations.

B 2D Sine Position Embedding

We follow the sine positional encodings but further hypothesize that the position information is independent at x (horizontal) and y (vertical) direction of an image. Concretely, we keep the original 2D-structure respectively with $d/2$

channels for x, y-direction:

$$\begin{aligned} PE_{(2i,p_y,:)} &= \sin\left(2\pi * p_y/(H * 10000^{2i/\frac{d}{2}})\right), \\ PE_{(2i+1,p_y,:)} &= \cos\left(2\pi * p_y/(H * 10000^{2i/\frac{d}{2}})\right), \\ PE_{(2i,:,p_x)} &= \sin\left(2\pi * p_x/(W * 10000^{2i/\frac{d}{2}})\right), \\ PE_{(2i+1,:,p_x)} &= \cos\left(2\pi * p_x/(W * 10000^{2i/\frac{d}{2}})\right), \end{aligned} \tag{5}$$

where $i = 0, 1, ..., d/2 - 1$, p_x or p_y is the position index along x or y-direction. Then they are stacked and flattened into a shape $\mathbb{R}^{L\times d}$.

C Transformer Encoder Layer

The Transformer Encoder layer [44] we used can be formulated as:

$$\begin{aligned} \mathbf{Z} &= \text{LayerNorm}\left(\text{MultiheadSelfAttention}\left(\mathbf{X}\right) + \mathbf{X}\right), \\ \mathbf{X}^* &= \text{LayerNorm}\left(\text{FFN}\left(\mathbf{Z}\right) + \mathbf{Z}\right), \end{aligned} \tag{6}$$

where $\mathbf{X}$ is the original input sequence that has not yet been added with position embedding. The position embedding will be added to $\mathbf{X}$ for computing querys and keys. $\mathbf{X}^*$ is the output sequence of the current Transformer Encoder layer, as the input sequence of next encoder layer. The formulations of Multihead Self-Attention and FFN are defined in [44].

D Detailed Results on COCO test-dev2017 Set

In Tab. 7, we show the results on COCO test-dev2017 set on AP (Average Precision) and AR (Average Recall) metrics, which are reported by the COCO official server.

E Architecture Details

We report the architecture details of ResNet-S and HRNet-S-W32(48) in Tab. 5 and Tab. 6. The ResNet-S* only differs from ResNet-S in that ResNet-S* has 10 Bottleneck-c128 blocks. More details about HRNet-W32 and HRNet-W48 are described in [38].

F Ablation Study on the Size of Transformer Encoder

We analyze the effect of the size of encoder layers on the model performances, as shown in Tab. 8. For TransPose-R models, with the number of layers increas-

Backbone	ResNet-S
Stem	Conv-k7-s2-c64, BN, ReLU Pooling-k3-s2
Blocks	3×Bottleneck-c64 Conv-k3-s2-c128, BN 4×Bottleneck-c128 Conv-k1-s1-c256

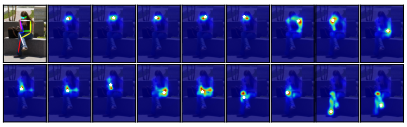
Table 5: The detailed configurations for ResNet-S. Conv-k7-s2-c64 means a convolutional layer with 7×7 kernel size, 2 stride, and 64 output channels, followed by a BN and ReLU; the same below. The Bottleneck-c64 includes Conv-k1-s1-c64-BN-ReLU, Conv-k3-s1-c64-BN-ReLU, and Conv-k1-s1-c256-BN. Bottleneck-c128 includes Conv-k1-s1-c128-BN-ReLU, Conv-k3-s1-c128-BN-ReLU, and Conv-k1-s1-c512-BN. See details in [17].

Backbone	HRNet-S-W32(48)
Stem	Conv-k3-s2-c64, BN, ReLU Conv-k3-s2-c64, BN, ReLU 4×Bottleneck-c64
Blocks	transition1∼stage2 transition2∼stage3 Conv-k1-s1-c64(92)

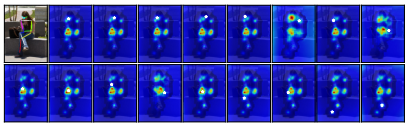
Table 6: The detailed configurations for HRNet-S-W32(48). More detailed information about the transition layer and stage blocks are described in the HRNet paper [38].

Method	Input size	Params	FLOPs	AP	$AP^{0.5}$	$AP^{0.75}$	AP^{M}	AP^{L}	AR	$AR^{0.5}$	$AR^{0.75}$	AR^{M}	AR^{L}
TransPose-H-S	256× 192	5.2M	10.3G	73.4	91.6	81.1	70.1	79.3	78.6	95.0	85.6	74.5	84.3
TransPose-H-A6	256×192	17.5M	21.8G	75.0	92.2	82.3	71.3	81.1	80.1	95.4	86.7	75.9	85.9

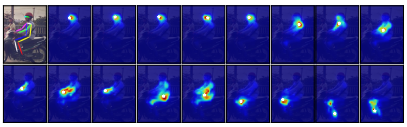
Table 7: Comparisons with state-of-the-art on COCO test-dev set.



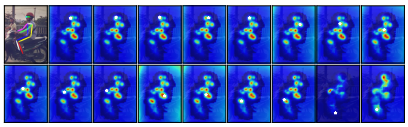
(a) **TP-R:** predictions and dependencies of each keypoint for **input 1**.



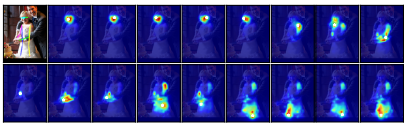
(b) **TP-H:** predictions and dependencies of each keypoint for **input 1**.



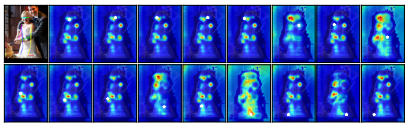
(c) **TP-R:** predictions and dependencies of each keypoint for **input 2**.



(d) **TP-H:** predictions and dependencies of each keypoints for **input 2**.



(e) **TP-R:** predictions and dependencies of each keypoint for **input 3**.



(f) **TP-H:** predictions and dependencies of each keypoints for **input 3**.

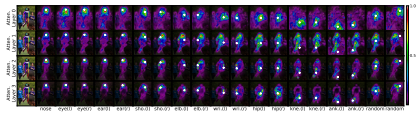
Figure 10: In each sub-figure, the first one is the original input plotted with predicted skeleton. The other maps are the unfolded attention maps in the last layer of TP-R and TP-H inspected by the predicted locations of keypoints, each position of which is annotated by a white pentagram. All the maps are visualized by the unnormalized attention maps.

ing to 6, the performance improvement gradually tends to saturate or degenerate. But we have not observed such a phenomenon on TransPose-H models (base on HRNet-S-W48). We partially attribute this to the limited model capacity of TransPose-R based on the very lightweight backbone ResNet-S. And there seems to have a room to improve the performance of TransPose-H by expanding the model sizes, *e.g.*, increasing the number of attention layers, the dimensions d, or hidden units in FFN.

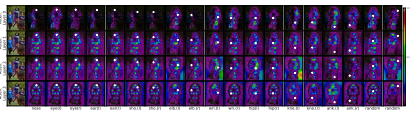
G Gradient Analysis

As revealed by [37, 1, 36], the gradient information indicates how much importance a token $\boldsymbol{x}_j \in \mathbb{R}^d$ (feature vector) at location j affects the final prediction in the heatmap. That assumption is based on that tiny change in the input token with the most important feature value causes a large change in what the output of the model would be.

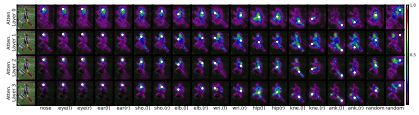
Assuming that $\boldsymbol{h}_i \in \mathbb{R}^K$ is the scores for all K types of keypoints at location i; $\boldsymbol{z}_i \in \mathbb{R}^d$ is the intermediate feature outputted by the last attention



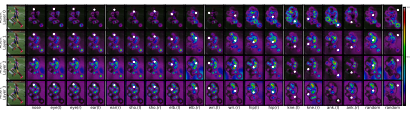
(a) **TP-R:** predictions and dependency areas of each keypoint in different attention layers.



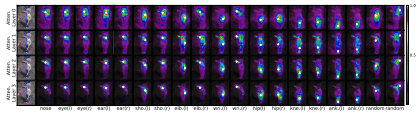
(b) **TP-H:** predictions and dependency areas of each keypoint in different attention layers.



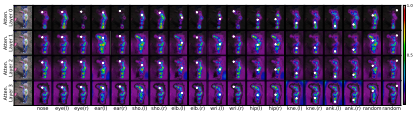
(c) **TP-R:** predictions and dependency areas of each keypoint in different attention layers.



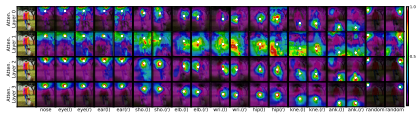
(d) **TP-H:** predictions and dependency areas of each keypoint in different attention layers.



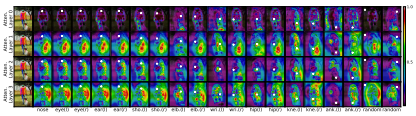
(e) **TP-R:** predictions and dependency areas of each keypoint in different attention layers.



(f) **TP-H:** predictions and dependency areas of each keypoint in different attention layers.



(g) **TP-R:** predictions and affected areas of each keypoint in different attention layers.



(h) **TP-H:** predictions and affected areas of each keypoint in different attention layers.

Figure 11: **Dependency areas** and **Affected areas** for different input images.

Model	#Layers	d	h	Params	FLOPs	FPS	AP	AR
TransPose-R	2	256	1024	4.4M	7.0G	174	69.6	75.0
	3	256	1024	5.2M	8.0G	141	71.7	77.1
	4	256	1024	6.0M	8.9G	138	**72.6**	**78.0**
	5	256	1024	6.8M	9.9G	126	72.2	77.6
	6	256	1024	7.6M	10.8G	109	72.2	77.5
TransPose-H	4	64	128	17.0M	14.6G	-	75.1	80.1
	4	192	384	18.5M	27.0G	-	75.4	80.5
	4	96	192	17.3M	17.5G	41	75.3	80.3
	5	96	192	17.4M	19.7G	40	75.6	80.6
	6	96	192	17.5M	21.8G	38	**75.8**	**80.8**

Table 8: Ablation study on the hyperparameters of Transformer Encoder. #Layers, d and h are the number of encoder layers, the dimensions d, and the number of hidden units of FFN.

layer before being fed into FFN. There is only a ReLU excluding the linear and convolutions (head) layers after the last attention layer. ReLU (rectified linear unit) activation function in FFN can be empirically regarded as the negative contribution filter, which only retains positive contributions. Next we choose numerator layout for computing the derivative of a vector with respect to a vector. We thus assume the mapping from $\boldsymbol{z}_i$ to $\boldsymbol{h}_i$ can be approximated as a linear function f with learned weights $\mathbf{W}_f \in \mathbb{R}^{K\times d}$ and bias $\mathbf{b} \in \mathbb{R}^K$ by computing the first-order Taylor expansion at a given point $\boldsymbol{z}_i^0$, *i.e.*, $\boldsymbol{h}_i \approx \mathbf{W}_f \boldsymbol{z}_i + \mathbf{b}$, $\mathbf{W}_f = \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{z}_i}\Big|_{\boldsymbol{z}_i^0}$. Then we compute the derivative of $\boldsymbol{h}_i$ w.r.t the token $\boldsymbol{x}_j$ at location j of the input sequence of the last attention layer:

$$
\begin{aligned}
\frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{x}_j} &= \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{z}_i}\frac{\partial \boldsymbol{z}_i}{\partial \boldsymbol{x}_j} \\
&= \frac{\partial f(\boldsymbol{z}_i)}{\partial \boldsymbol{z}_i}(\mathbf{1} + \frac{\partial \mathbf{w}_i \mathbf{V}}{\partial \boldsymbol{x}_j}) \\
&\approx \mathbf{W}_f(\mathbf{1} + \frac{\partial \mathbf{w}_{i,0}\boldsymbol{v}_0 + ... + \mathbf{w}_{i,j}\boldsymbol{v}_j + ... + \mathbf{w}_{i,L-1}\boldsymbol{v}_{L-1}}{\partial \boldsymbol{x}_j}) \\
&= \mathbf{W}_f(\mathbf{1} + \frac{\partial \mathbf{w}_{i,j}\boldsymbol{v}_j}{\partial \boldsymbol{x}_j}) \\
&= \mathbf{W}_f(\mathbf{1} + \frac{\partial \mathbf{A}_{i,j}\mathbf{W}_v^\top \boldsymbol{x}_j}{\partial \boldsymbol{x}_j})
\end{aligned}
\tag{7}
$$

where $\boldsymbol{v}_j \in \mathbb{R}^d$ is the value vector transformed from $\boldsymbol{v}_j = \mathbf{W}_v^\top \boldsymbol{x}_j$. $\mathbf{A}_{i,j}$ is a scalar value that is computed by the dot-product between $\boldsymbol{q}_i$ and $\boldsymbol{k}_j$. We assume $\mathcal{G} := \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{x}_j}$ as a function w.r.t. a given attention score $\mathbf{A}_{i,j}$, then:

$$
\begin{aligned}
\mathcal{G}\left(\mathbf{A}_{i,j}\right) &= \mathbf{W}_f(\mathbf{1} + \frac{\partial \mathbf{A}_{i,j}\mathbf{W}_v^\top \boldsymbol{x}_j}{\partial \boldsymbol{x}_j}) \\
&= \mathbf{W}_f\left(\mathbf{1} + \mathbf{A}_{i,j}\mathbf{W}_v^\top\right) \\
&= \mathbf{A}_{i,j}\mathbf{W}_f\mathbf{W}_v^\top + \mathbf{W}_f \\
&= \underbrace{\mathbf{A}_{i,j}}_{\textbf{Image-Dependent: dynamic weights}} \cdot \underbrace{\mathbf{W}_f \cdot \mathbf{W}_v^\top + \mathbf{W}_f}_{\textbf{Learned: static weights}} \\
&= \mathbf{A}_{i,j} \cdot \mathbf{K} + \mathbf{B}
\end{aligned}
\tag{8}
$$

where $\mathbf{K}, \mathbf{B} \in \mathbb{R}^{K\times d}$ are static weights shared across all positions. We can see that how $\boldsymbol{x}_j$ affects $\boldsymbol{h}_i$ could be approximated as a linear combination of dynamic weights – attention score $\mathbf{A}_{i,j}$ and learned static weights, *i.e.*, the subsequent layers almost linearly and equally transform the attention scores from tokens across all the positions. The last attention layer in Transformer Encoder, whose attention scores are seen as the image-dependent weights, has the most direct effect on the predictions. Though the weights in FFN or head cannot

be ignored, they are image-independent during inference and shared across all locations. Note that $\mathbf{Q} = (\mathbf{X} + \mathbf{P})\,\mathbf{W}_q, \mathbf{K} = (\mathbf{X} + \mathbf{P})\,\mathbf{W}_k, \mathbf{V} = \mathbf{X}\mathbf{W}_v$ where $\mathbf{P}$ is the position embedding. Because $\mathbf{A}_{i,j} \propto \mathbf{Q}_i \mathbf{K}_j^\top$, the position embedding values also affect the attention score to some extent. This section is associated with Section 3.4.

H More Attention Maps Visualizations

In this section, we show more visualization results of the attention maps from TP-R (TransPose-R) and TP-H (TransPose-H-S) models. The attention maps of the last attention layers of two models are shown in Fig. 10. The attention maps in different attention layers of two models are shown in Fig. 11.