DETRPose: Real-Time End-to-End Multi-Person Pose Estimation via Modified Transformer Decoder and Novel Denoising Keypoints

Sebastian Janampa[1] and Marios Pattichis[1]

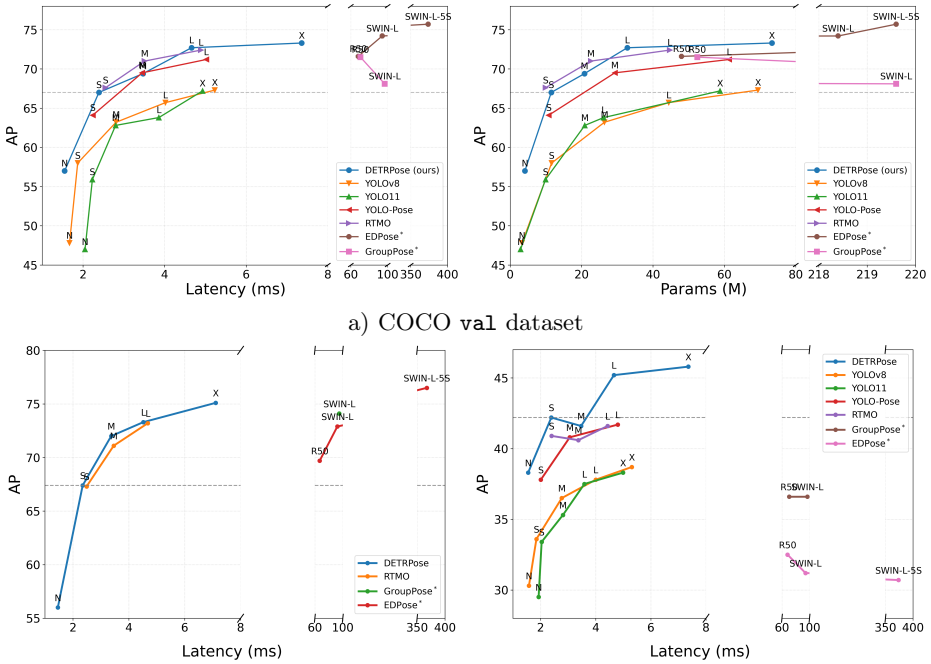
The University of New Mexico

Abstract. Multi-person pose estimation (MPPE), which involves detecting body joint positions (keypoints) for every person in an image, is a fundamental task in computer vision. Despite recent advances, no transformer-based model currently achieves real-time performance. This work addresses the latency challenge by introducing DETRPose, the first family of real-time, end-to-end transformer models for multi-person 2D pose estimation. DETRPose significantly enhances the GroupPose decoder, enabling real-time inference. For training, a novel denoising keypoint technique is proposed to accelerate convergence. The varifocal loss is also extended for keypoints, termed Keypoint Similarity VariFocal loss, to improve query quality. Extensive evaluation demonstrates that DETRPose models achieve accuracy comparable to or exceeding that of leading alternatives while requiring five to ten times fewer training epochs. DETRPose-S matches the accuracy of YOLOv8-Pose-X and YOLO11-Pose-X on the COCO dataset (67.0 vs 67.3 and 67.2 in AP) with 81% fewer parameters (11.5M vs 69.4M and 58.8M) and 52% faster inference speed (2.39ms vs 5.23ms and 4.93ms). On the CrowdPose dataset, DETRPose-X has 13.1× fewer FLOPs (232.3G vs 3048.1G) and only 2% fewer precision (75.1 vs 76.6 in AP) than ED-Pose-SwinL-5S. On the OCHuman dataset, DETRPose-S surpasses all previous models, showing the robustness of DETRPose on out-of-distribution datasets. Code is available at `https://github.com/SebastianJanampa/DETRPose`

1 Introduction

2D multi-person pose estimation (MPPE) is a critical preprocessing step for tasks including activity recognition, 2D-to-3D human pose estimation, and virtual reality. As a result, low latency is essential for real-time deployment. Although numerous studies on transformer models for MPPE focus on accuracy improvements, their high latency limits their applicability in real-time scenarios.

The majority of fast MPPE methods can be categorized as either top-down or single-stage approaches. Top-down models first detect individuals, crop the object regions, and then process the cropped images to estimate keypoints. These models typically achieve higher accuracy than alternative methods, but their inference times increase linearly with the number of people in an image. In contrast, single-stage methods estimate human keypoints and subsequently group



a) COCO `val` dataset

b) CrowdPose `test` dataset

c) OCHuman `test` dataset

Fig. 1: Comparisons with other pose estimation models in mutiple datasets. We measure end-to-end latency using TensorRT FP16 on an NVIDIA RTX A5500 GPU.

them for each person. Although bottom-up methods provide faster inference, they are generally less accurate because the models must simultaneously learn detection and pose estimation.

Recent developments in single-stage models (*e.g.* YOLOv8/11-Pose [2,3] and RTMO [13]) successfully adapted YOLO models for MPPE, providing a good trade-off between speed and precision. However, these methods rely on Non-Maximum Suppression (NMS) during inference, resulting in inconsistent latency.

To overcome the limitations associated with NMS, several studies have extended the DETR framework to pose estimation [21, 28]. DETR-based models achieve consistent inference times and strong precision. However, prior transformer models have prioritized high precision over computational speed. Although real-time DETR object detectors have been developed [17, 23, 32], no real-time adaptation of DETR for pose estimation exists, thereby restricting the broader application of the DETR framework.

This paper introduces DETRPose, the first real-time end-to-end transformer-based pose estimation model. To achieve low, consistent latency while maintaining high accuracy, several training innovations are presented, and the GroupPose decoder (which predicts keypoints from queries) is adapted to reduce processing time. A new denoising keypoint technique that uses Keypoint Similarity (KS)

is proposed for constructing keypoint queries, significantly accelerating convergence. Additionally, the Keypoint Similarity VariFocal (KSVF) loss function and an auxiliary Pose-LQE (Localization Quality Estimation) classification layer are introduced to refine the quality of keypoint predictions and classification accuracy (same purpose as the LQE layers [9] used in D-FINE[1] [17]). Architecturally, the D-FINE strategy is adopted, and projection layers are removed to further reduce decoder latency.

The proposed approach achieves competitive results compared to previous methods while maintaining low and consistent latency. Specifically, DETRPose-L attains 71.2% Average Precision (AP) on the COCO `test-dev2017` dataset with a latency of 4.66 ms, surpassing prior end-to-end methods. Compared to RTMO [13], the current state-of-the-art model for real-time MPPE, DETRPose-L offers fewer parameters, reduced latency, faster convergence, a comparable AP score (only 0.4 points lower), and a higher AR score (+3.2 points). Against popular frameworks like YOLOv8-Pose and YOLO11-Pose, DETRPose-S matches the biggest version of these two YOLO models on the COCO `val` dataset (see Fig. 1a). On the CrowdPose `test` dataset, DETRPose-X exceeds expectations, surpassing all transformer models with the Swin-L backbone by at least 1 point in AP, while using significantly fewer parameters (73.3M vs. 210M). On the OCHuman dataset, DETRPose-S shows DETRPose's superior robustness on out-of-distribution datasets, surpassing RTMO-L (42.2 vs 41.6 in AP) while being 1.85× faster (see Fig. 1c).

The primary contributions of this work are:

- A new denoising technique for pose estimation. The proposed approach employs object Keypoint Similarity (KS) to construct both positive and negative keypoint queries.
- An auxiliary classification layer, the Pose-LQE layer, which enhances classification performance. Its lightweight design enables integration without increasing latency, addressing the challenge that DETRPose predicts keypoints rather than objects.
- A new loss function, the Keypoint Similarity VariFocal (KSVF) loss, that uses the Keypoint Similarity (KS) metric and the predicted score to promote the selection of high-quality object queries.
- A modified GroupPose decoder to support real-time end-to-end MPPE for transformer-based models.

2 Related Works

Single-stage multi-person pose estimation methods can be categorized into two primary groups: (i) non-end-to-end methods, and (ii) end-to-end methods.

[1] The use of LQE is not mentioned in the D-FINE paper, but it is found in D-FINE's decoder code from its official repository.

2.1 Non-End-to-End Methods

Non-end-to-end methods [3, 14, 15, 22, 24, 25] can be further divided into single-stage and two-stage approaches. Single-stage methods [2, 3, 13] are fast because they estimate the keypoint instances in a single forward pass. In contrast, two-stage approaches include both top-down and bottom-up methods. Top-down approaches generally achieve higher accuracy by first performing person detection and then pose estimation. During training, top-down methods detect humans by assigning bounding boxes to each person, then estimate pose within each bounding box. Conversely, bottom-up methods estimate all keypoints in the first stage and subsequently group keypoints into instances in the second stage. A key limitation of non-end-to-end methods is their reliance on hand-crafted processes such as Non-Maximum Suppression (NMS). Furthermore, two-stage processes introduce significant variability in inference time due to the need to process bounding boxes or group keypoints.

2.2 End-to-End Methods

Current end-to-end multi-person pose estimation (MPPE) models utilize the DETR [1] framework to eliminate the need for NMS. PETR [21] is the first DETR-based model for pose estimation and employs two decoders: a pose decoder and a joint decoder. The pose decoder coarsely detects human instances by estimating their poses, while the joint decoder promotes interaction between keypoints of the same person, resulting in more precise estimations. Subsequently, QueryPose [27] and ED-Pose [28] adopt a top-down approach by incorporating a human detector decoder. Notably, QueryPose implements two parallel branches: a bounding box detector and a pose detector. ED-Pose further advances these developments by employing sequential decoder layers, with initial layers dedicated to detection and subsequent layers estimating human keypoints. In contrast, GroupPose [11] utilizes group self-attention to eliminate the need for detection-only decoder layers. Group self-attention comprises two sequential self-attention modules: the first operates among keypoint queries within the same human instance, and the second operates among queries of the same keypoint type across different instances. Although these models demonstrate effectiveness, they do not fully exploit DETR's capabilities for pose estimation. Notably, no prior work has applied a denoising technique to keypoints, despite these being the primary instances being estimated.

2.3 Denoising Techniques

DETR-based methods use query vectors to propose candidate instances for bounding box, keypoint, or mask estimation [1, 7, 28]. Many works have focused on developing strong queries by associating them with specific spatial positions to improve results [12, 26]. However, none of them focus on the Hungarian matcher (the algorithm that enables DETR to perform end-to-end inference), which is unstable, especially during early training.

a) Denoising Box [6, 29] b) Denoising Keypoint (ours)

Fig. 2: Denoising techniques. **Left**. Denoising box [6,29] produced by shifting and scaling the ground-truth box, used for object detection. **Right**. Denoising keypoint consists of shifting the ground-truth keypoint. The ground-truth instances are in black. The positive and negative queries are located within the green and red areas, respectively.

The denoising strategy was initially introduced in DN-DETR [6], and later improved in DINO [29], to stabilize the Hungarian Matcher. The denoising strategy generates both positive and negative queries. It trains the model to accept positive queries with bounding boxes close to the ground truth (based on their classification score) and to reject negative queries whose boxes are far from the ground truth. As shown in Fig. 2a, DETR-based object detectors generate these two types of queries for object detection. For each ground-truth bounding box (black square), its center and dimensions are randomly shifted or scaled to build a positive (green area) or negative (red area) query, using the width and height of the ground-truth.

In practice, ED-Pose [28] applies the DINO denoising strategy to the detection layers. It then removes the denoising queries, keeping only the selected top-k queries for the human-to-keypoints layers. GroupPose [11] does not employ denoising, as it estimates only keypoint locations. Interestingly, previous pose estimation models have not applied denoising to keypoints, despite many DETR models relying on this approach for optimal performance [17, 23, 32].

Unlike a bounding box that has 4 points (corners), keypoints are single points. This difference makes it challenging to shift or translate keypoints. Additionally, not all keypoints belong to the same joint type. To address these challenges, we propose a novel denoising keypoint strategy that uses the keypoint similarity to generate both positive and negative queries (see 2b).

3 Methodology

Figure 3 shows the DETRPose architecture. It consists of a hierarchical backbone, a hybrid encoder, and a transformer decoder. DETRPose estimates all keypoints for every human instance simultaneously by employing group self-attention in each decoder layer. We use the backbone and encoder from D-FINE, initialize them with D-FINE's Object365 weights. We introduce a new training strategy named Denoising Keypoints strategy to accelerate convergence and to enhance overall model performance. We also develop the Pose-LQE head to re-

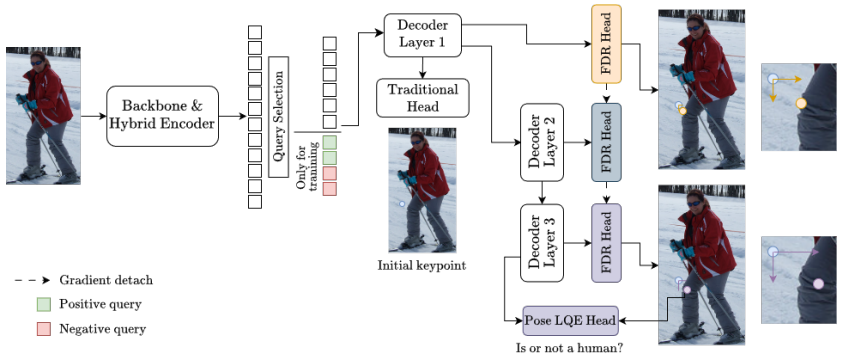


Fig. 3: DETR-POSE architecture. All keypoints are estimated simultaneously. For simplicity, we present the computation for a single person. The arrows in the right-side images (knee) represent the vertical and horizontal offsets generated by the FDR heads.

fine classification scores and the Keypoint Similarity VariFocal loss to improve query selection quality.

3.1 Denoising Keypoints

We next describe our denoising keypoint strategy. We begin by describing the keypoint similarity. Let $\kappa > 0$ denote a fall-off parameter that will allow us to vary our similarity metric for each type of keypoint. The κ values are taken from the COCO and CrowdPose repositories. Let s be the object scale, and $0 \leq s^2 \leq HW$, the human's segmented area. H and W are the image dimensions. Then, using the distance d between the ground-truth and a sampling keypoint, we define the keypoint similarity (KS) as:

$$\text{KS} = \exp(-d^2/(2s^2\kappa^2)). \tag{1}$$

In order to derive bounds for KS, we note that the sampling keypoint $\hat{p}$ can be written as a variation of the ground-truth keypoint p given by:

$$\hat{p} = p + \alpha_{pose}\,\overrightarrow{n} \tag{2}$$

where $\alpha_{\text{pose}} \geq 0$ is a non-fixed scaling factor and $\overrightarrow{n}$ is a random unit vector. In order to estimate α_{pose} for generating positive and negative queries, we note that $d(p, \hat{p}) = \alpha_{pose}$, and substitute it in Eq. (1) to get:

$$\alpha^2_{\text{pose}} = -2\ln(\text{KS})s^2\kappa^2. \tag{3}$$

$$\alpha_{\text{pose}} = s\kappa\sqrt{-2\ln(\text{KS})}. \tag{4}$$

Next, we describe how we generate positive and negative queries. We sample values for KS from a uniform distribution $\mathcal{U}$, then substitute them into (4)

to get α_{pose}. This α_{pose} value sets the magnitude of the shift. We generate the sampling keypoint, $\hat{p}$, by translating the ground-truth keypoint, p, along a random unit vector $\overrightarrow{n}$ as in (2). For positive queries, we sample KS from $\mathcal{U}(0.5, 1)$. For negative queries, we sample KS from $\mathcal{U}(0.1, 0.5)$.

During training, we apply an attention mask that *i)* allows the denoising groups to interact with the prediction group (but not vice-versa), and *ii)* that blocks cross interaction between denoising groups. We used a fixed number of 100 positive and 100 negative queries. For the CrowdPose dataset [8], we define the segmented area as $s^2 = \text{A}_{\text{bbox}} \cdot 0.53$ where A_{bbox} is the area of the bounding box, and the 0.53 is taken from the CrowdPose's evaluation code.

3.2 Query Selection

We use a linear layer processes the encoder's feature maps to get the top-N pixels with the highest probability of containing an instance query. Then, another linear layer generates K keypoints for each selected pixel (giving a total of $K \times N$ keypoints). Then, we redefine the instance query as the mean value of the K keypoints for each human instance. Overall, the decoder receives $N \cdot (K + 1)$ queries in total.

3.3 Decoder

Each decoder layer includes a within-instance self-attention layer, where keypoints from the same human instance interact. Then, an across-instance self-attention layer enables keypoints of the same joint to interact with one another. To separate negative, positive, and prediction instances, we apply a mask attention in the across-instances self-attention layer. Finally, a deformable attention mechanism allows all queries to interact with the encoder's feature maps.

Our decoder introduces two critical architectural enhancements over GroupPose: the integration of D-FINE's pre-pose and Fine-Grained Distribution Refinement (FDR) layers, and our novel Pose Localization Quality Estimation (Pose-LQE) layer, which enhances classification scores.

Pose-LQE Layer The Pose-LQE layer extracts content information from the encoder's highest-spatial feature map at each keypoint. It produces a vector for each keypoint and keeps only the top-k channels. A subsequent linear layer processes these values and combines them with those from traditional classification layers to produce a classification score. This approach creates a more confident classification score by combining information from the output queries, the predicted keypoint locations, and content features. Fig. 4 illustrates this procedure.

3.4 Keypoint Similarity VariFocal Loss

Since DETRPose does not estimate bounding boxes, it can not be trained using an IoU-based loss. We propose the Keypoint Similarity VariFocal (KSVF) loss

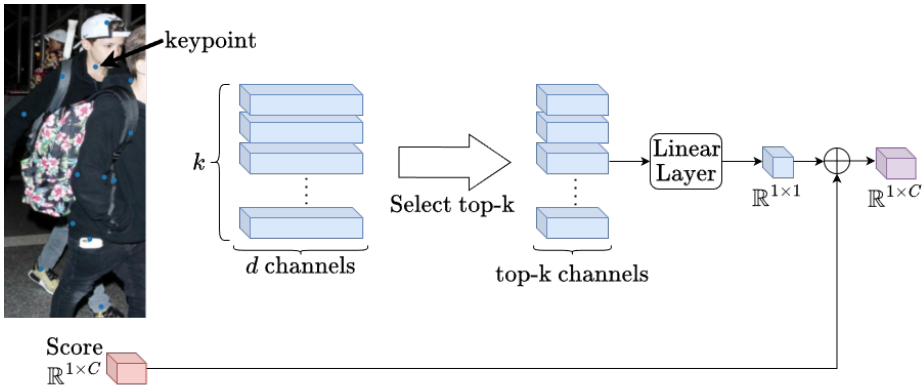


Fig. 4: Pose-LQE layer for improving classification accuracy.

Table 1: Parameter for DETRPose family.

	DETRPose-S	DETRPose-M	DETRPose-L	DETRPose-X
# decoder layers	3	4	6	6
# human instances	60	60	60	60
hidden dim.	256	256	256	384
backbone	HGNetv2-B0	HGNetv2-B2	HGNetv2-B4	HGNetv2-B5
epochs	96	60	48	48
lr	1e-4	1e-4	1e-4	1e-4
backbone lr	1e-4	1e-5	1e-5	5e-5
barch size	16	16	16	16

defined as:

$$\text{KSVF}(q, \tilde{c}) = \begin{cases} -q(q\log(\tilde{c}) + (1-q)\log(1-\tilde{c})), & q > 0 \\ -\alpha\tilde{c}^{\gamma}\log(1-\tilde{c}), & q = 0 \end{cases} \tag{5}$$

where q is the object keypoint similarity (OKS), $\tilde{c}$ denotes the predicted classification score, and α and γ are constants that control the loss function for low-similarity cases. We set $\alpha = 0.25$ and $\gamma = 2.0$ during training.

Our KSVF is an extension of the VariFocal loss [30], which is used in RT-DETR and D-FINE. The main purpose of KSVF is to reduce false-positive predictions by making the network focus on the predictions with high object keypoint similarity. A second benefit is that we can reduce the number of predicted persons because KSVF produces higher quality predictions. Since pose estimation transformers treat each keypoint as a query, reducing the number of predicted humans significantly reduces the decoder's FLOPs.

4 Results

4.1 Datasets

We conduct experiments on two standard benchmarks: COCO [10], CrowdPose [8], and OCHuman [31]. The COCO dataset includes over 150K images and 250K person instances annotated with 17 keypoints. CrowdPose, a more challenging dataset due to its highly crowded and occluded scenes, contains approximately 20K images and 80K person instances. The OCHuman dataset is only for testing and includes 5081 images and 10375 person instances with 17 keypoints.

We train DETRPose on the COCO `train` set and report results on the COCO `val` and `test-dev` sets, as well as the OCHuman `test` set. For CrowdPose, we use the `train-val` set for training and evaluate on the `test` set.

4.2 Model Variants

We develop four DETRPose models of varying sizes: DETRPose-S, DETRPose-M, DETRPose-L, and DETRPose-X. Tab. 1 provides information about the decoder parameters, backbone name, and training hyperparameters. Each DETRPose variant adopts the same backbone and encoder architecture as its corresponding D-FINE counterpart [17].

4.3 Implementation Details

All DETRPose variants are trained using the AdamW optimizer with a batch size of 16 on 4 Tesla V100 GPUs (cloud service). For data augmentation, we adopt the DEIM [23] protocol, which includes horizontal flipping, color jittering, mosaic composition, mix-up, and multi-scale sampling, but we do not use its cosine annealing strategy. Following the transfer learning strategy of RTMO [13] (see the supplementary material for more information), we initialize our backbone and encoder with D-FINE Object365 pretrained weights [17].

We use average precision (AP) to measure and compare model performance. During validation, input images are resized to 640×640. For ED-Pose and GroupPose, we apply letterboxing with a fixed input size of 800×1300 to preserve the image aspect ratio and provide a padding mask as an additional input. We report latency using TensorRT with FP16 precision on a local Nvidia RTX A5500. For a fair and consistent comparison, we use the benchmarking tool developed by Roboflow [19], which reports latency as the median rather than the mean. Since this work was done before YOLO26's release [4], we do not compare to it in the main paper but do so in the supplemental material.

4.4 Comparisons

COCO Dataset Comparisons We provide comprehensive comparisons against several other methods using the COCO `test-dev` dataset in Tab. 2. Among the end-to-end models not using the Swin-L backbone, DETRPose-L yields the best

Table 2: Performance comparison on the COCO `test-dev` dataset. The model with † is trained without transfer learning. Times with * are computed using an input size of 800 × 1300.

Method	Backbone	Epochs	#Params	Time (ms)	AP	AP_{50}	AP_{75}	AP_M	AP_L	AR
Non-end-to-end methods										
KAPAO-S [16]	CSPNet	500	12.6M	-	63.8	88.4	70.4	58.6	71.7	71.2
KAPAO-M [16]	CSPNet	500	35.8M	-	68.8	90.5	76.5	64.3	76.0	76.3
KAPAO-L [16]	CSPNet	500	77.0M	-	70.3	*91.2*	77.8	66.3	76.8	77.7
YOLO-Pose-S [14]	CSPDarknet	300	10.8M	2.24	63.2	87.8	69.5	57.6	72.6	67.6
YOLO-Pose-M [14]	CSPDarknet	300	29.3M	3.45	68.6	90.7	75.8	63.4	77.1	72.8
YOLO-Pose-L [14]	CSPDarknet	300	61.3M	5.02	70.2	91.1	77.8	65.3	78.2	74.3
RTMO-S [13]	CSPDarknet	600	9.9M	2.55	66.9	88.8	73.6	61.1	75.7	70.9
RTMO-M [13]	CSPDarknet	600	22.6M	3.51	70.1	90.6	77.1	65.1	78.1	74.2
RTMO-L [13]	CSPDarknet	600	44.8M	4.89	*71.6*	91.1	*79.0*	*66.8*	79.1	75.6
End-to-end methods with Swin-L backbone										
PETR [21]	Swin-L	100	213.8M	-	70.5	91.5	78.7	65.2	78.0	-
QueryPose [27]	Swin-L	79	-	-	73.3	91.3	79.5	68.5	81.2	-
ED-Pose [28]	Swin-L	60	218.4M	94.11*	72.7	92.3	80.9	67.6	80.0	-
GroupPose* [11]	Swin-L	60	219.6M	96.77*	74.8	91.6	82.1	69.4	83.0	-
End-to-end methods										
PETR* [21]	ResNet-50	100	43.7M	-	67.6	89.8	75.3	61.6	76.0	-
QueryPose* [27]	ResNet-50	40	-	-	68.7	88.6	74.4	63.8	76.5	-
ED-Pose* [28]	ResNet-50	60	48.0M	68.10*	69.8	90.2	77.2	64.3	77.4	-
GroupPose* [11]	ResNet-50	60	52.4M	70.44*	70.2	90.5	77.8	64.7	78.0	-
DETRPose-L† (ours)	ResNet-50	48	42.7M	4.98	68.7	89.3	75.2	62.6	78.3	75.8
DETRPose-S (ours)	HGNetv2-B0	96	11.5M	2.39	66.5	87.4	72.5	59.8	76.8	73.0
DETRPose-M (ours)	HGNetv2-B2	60	20.8M	3.47	68.5	88.6	74.6	62.1	78.3	74.6
DETRPose-L (ours)	HGNetv2-B4	48	32.8M	4.66	71.2	*91.2*	78.1	65.8	*79.9*	*78.1*
DETRPose-X (ours)	HGNetv2-B5	48	73.3M	7.36	**72.2**	**91.4**	**79.3**	**67.0**	**80.6**	**78.8**

results while being 14.6× faster. Compared with the second-best model, GroupPose [11], DETRPose-L has 37% fewer parameters (32.8M vs 52.4M). Compared to the best non-end-to-end models, RTMO-L [13], DETRPose-L has 27% fewer parameters (32.8M vs 44.8M), 5% lower latency (4.89 vs 4.66), 0.5% lower AP score (71.2 vs 71.6), and 4% higher AR score. DETRPose-X , our biggest model, achieves competitive results against end-to-end models with Swin-L backbones (> 200M parameters). It is important to note that PETR, QueryPose, ED-Pose, and GroupPose resize the image so that its shortest side is 800 and preserve the original image aspect ratio, which explains why these models achieve better results than DETRPose-L with ResNet50.

We report a comparison on the COCO `val` dataset on Tab. 3. DETRPose-X surpasses all previous methods by at least 0.9 and 3.1 points in AP and AR, respectively. Fig. 1a shows that DETRPose-S has a similar AP to YOLOv8/11-L models while having 32% lower latency and 40% fewer parameters. DETRPose-M matches YOLOv8/11-X-Pose in the AP score (69.4 vs 69.2 and 69.5), and has 64% fewer parameters, and 21% lower latency. GroupPose-Swin-L-5S's performance lags behind its official PyTorch results because this GroupPose variant was trained with 1 image per GPU across 16 GPUs.

Table 3: Performance comparison on the COCO `val` dataset. Best and second results are in **bold** and *italic*, respectively.

Method	# Params	Time (ms)	FLOPs (G)	AP	AP_{50}	AP_{75}	AP_M	AP_L	AR
YOLO-Pose-S [14]	10.8M	2.24	18	64.1	87.1	70.2	57.9	74.1	68.2
YOLO-Pose-M [14]	29.3M	3.45	48	69.5	89.9	76.2	63.7	78.9	73.3
YOLO-Pose-L [14]	61.3M	5.02	98	71.2	90.1	78.2	65.6	80.1	74.9
RTMO-S [3]	9.9M	2.55	12	67.6	87.8	73.6	61.4	77.3	71.5
RTMO-M [3]	22.6M	3.51	32	71.0	89.5	77.8	65.3	79.8	74.8
RTMO-L [3]	44.7M	4.89	68	72.4	89.9	78.8	67.1	81.0	76.2
YOLOv8-S [2]	11.6M	1.87	30	58.0	82.4	64.7	49.8	70.0	66.7
YOLOv8-M [2]	26.4M	2.81	81	63.2	85.6	70.8	56.6	73.1	71.8
YOLOv8-L [2]	44.4M	4.02	169	65.7	86.6	73.0	59.4	75.1	74.2
YOLOv8-X [2]	69.4M	5.23	263	67.3	87.5	75.0	61.5	76.1	75.7
YOLO11-S [3]	9.9M	2.23	23	55.9	81.2	62.1	47.8	67.6	66.1
YOLO11-M [3]	20.9M	2.80	72	62.8	85.4	70.6	55.9	71.4	72.2
YOLO11-L [3]	26.2M	3.86	91	63.8	86.2	71.4	58.2	72.5	73.3
YOLO11-X [3]	58.8M	4.93	203	67.2	87.5	74.8	62.3	75.5	76.3
End-to-end methods with Swin-L backbone									
ED-Pose-SWIN-L* [28]	218.4M	94.11	1954	74.2	91.8	81.4	68.3	82.8	83.0
ED-Pose-SWIN-L-5S* [28]	218.7M	373.72	3061	75.7	92.7	83.1	70.0	83.8	84.2
GroupPose-SWIN-L* [11]	219.6M	96.77	1977	68.1	89.8	74.7	60.3	79.2	78.2
End-to-end methods									
ED-Pose-R50* [28]	48.0M	68.10	591	71.6	90.1	78.0	66.0	80.0	81.2
GroupPose-R50* [11]	52.4M	70.44	614	71.5	89.5	78.8	66.2	79.8	81.5
DETRPose-S (ours)	11.5M	2.39	33	67.0	88.1	72.9	60.4	77.3	73.5
DETRPose-M (ours)	20.8M	3.47	67	69.4	89.8	75.5	63.1	79.1	75.5
DETRPose-L (ours)	32.8M	4.66	107	*72.7*	**91.0**	*79.2*	*66.7*	*82.2*	*78.7*
DETRPose-X (ours)	73.3M	7.36	240	**73.3**	*90.5*	**79.4**	**67.5**	**82.7**	**79.4**

CrowdPose Dataset Comparisons We provide comprehensive comparisons on the CrowdPose dataset in Tab. 4 and Fig. 1b. The results on CrowdPose are more significant, as it is a more challenging dataset. DETRPose-M surpasses the ED-Pose-R50 while having 50% fewer parameters, and 16× lower latency. Compared with non-end-to-end models, DETRPose models outperform their RTMO counterparts (S, M, and L) and converge faster. For example, DETRPose-S and -M require 156 and 72 epochs, which is 4.5× and 9.7× more than for RTMO counterparts.

DETRPose-X sets a new state-of-the-art record for real-time MPPE models, achieving a score of 75.1 on the AP metric. DETRPose-X surpasses most transformer models equipped with Swin-L while having 3× less parameters. Only ED-Pose-SWIN-L-5S beats us for 1.5 points; however, its full model was pretrained on the Object365 dataset [20], and uses 5 feature maps for prediction. Clearly, DETRPose-X provides strong evidence that our proposed transformer-based method holds great promise for pose estimation.

Table 4: Performance comparison on the CrowdPose `test` dataset. Models with * use input size of 800 × 1300. Best and second results are in **bold** and *italic*.

Method	Epochs	# Params	Time (ms)	FLOPs (G)	AP	AP_{50}	AP_{75}	AP_E	AP_M	AP_H
Non-end-to-end methods										
RTMO-S [13]	700	9.8 M	2.49	12	67.3	88.2	72.9	73.7	68.2	59.1
RTMO-M [13]	700	22.4M	3.46	32	71.1	87.7	77.1	77.4	71.9	63.4
RTMO-L [13]	700	44.5M	4.68	68	73.2	90.7	79.3	79.2	*74.1*	65.3
End-to-end methods with Swin-L backbone										
ED-Pose-SWIN-L* [28]	80	218.3M	92.22	1940	72.9	91.1	79.8	80.5	73.8	63.8
ED-Pose-SWIN-L-5S* [28]	80	218.6M	367.45	3048	76.5	92.6	83.3	83.0	77.3	68.3
GroupPose -SWIN-L* [11]	80	219.7M	94.89	1969	74.1	91.3	80.4	80.8	74.7	66.4
End-to-end methods										
ED-Pose-R50* [28]	80	48.0M	66.91	578	69.7	89.0	75.8	77.7	70.6	60.9
DETRPose-S (ours)	156	11.5M	2.35	31	67.4	88.6	72.9	74.7	68.1	59.3
DETRPose-M (ours)	72	20.7M	3.37	65	72.0	91.0	77.8	78.6	72.6	64.5
DETRPose-L (ours)	48	32.7M	4.52	104	*73.3*	*91.6*	*79.4*	*79.5*	74.0	*66.1*
DETRPose-X (ours)	48	73.3M	7.11	232	**75.1**	**92.1**	**81.3**	**81.3**	**75.7**	**68.1**

OCHuman Dataset Comparisons We provide comparisons on the OCHuman dataset in Tab. 5 to evaluate models' robustness. For fair comparisons, all the models are trained on the COCO `train` set. DETRPose-S surpasses all methods by at least 0.6 while being the 4th fastest model in Tab. 5. The Big models, ED-Pose and GroupPose, yield the lowest results, indicating they overfit the COCO dataset. Fig. 1c validates this statement since ED-Pose-R50 performs better than it bigger variations. In terms of model scaling, the difference in AP between RTMO-S and RTMO-L is 0.7, while it is 3.0 for DETRPose, indicating that DETRPose scales better than other models on out-of-distribution datasets.

4.5 Ablation study

Component Analysis We provide a step-by-step analysis for each proposed component in DETRPose in Tab. 6. For each step, we fully train DETRPose-L on the COCO `train` set and report results on the COCO `val` set. To initiate our study, we create RT-GroupPose by adding D-FINE's backbone, encoder, and FDR heads to the original GroupPose (first row in Tab. 6), achieving an AP of 69.9. Reducing the number of instance queries from 100 to 60 (Model 2) lowers AP and GFLOPs. Adding the proposed components individually (Models 3–5) improves performance, with the denoising strategy giving the largest gain.

We explore the synergy between different components (Models 6–8). The KSVF loss works well with either the denoising strategy or Pose-LQE, achieving 70.9 and 70.5, respectively. However, pairing the denoising strategy with Pose-LQE performs worse than using either component alone. Training DETRPose-L with all three proposed components achieves an AP of 71.5 (Model 9).

Table 5: Performance comparison on the OCHuman dataset. Models with * use input size of 800×1300. Best and second results are in **bold** and *italic*.

Method	# Params	Time (ms)	AP	AP_{50}	AP_{75}	AP_L
YOLO-Pose-S [14]	10.8M	2.01	37.8	52.6	42.9	73.4
YOLO-Pose-M [14]	29.3M	3.06	40.8	53.2	46.1	78.2
YOLO-Pose-L [14]	61.3M	4.80	41.7	53.2	47.0	79.4
RTMO-S [3]	9.9M	2.40	40.9	53.5	45.8	77.3
RTMO-M [3]	22.6M	3.37	40.6	51.9	45.4	79.7
RTMO-L [3]	44.7M	4.43	41.6	51.9	46.1	81.0
YOLOv8-S [2]	11.6M	1.85	33.6	49.3	37.4	68.3
YOLOv8-M [2]	26.4M	2.77	36.5	50.3	40.8	72.5
YOLOv8-L [2]	44.4M	4.00	37.8	51.1	42.4	74.4
YOLOv8-X [2]	69.4M	5.31	38.7	51.1	43.3	76.3
YOLO11-S [3]	9.9M	2.04	33.4	49.6	37.6	67.4
YOLO11-M [3]	20.9M	2.81	35.3	49.7	39.9	71.5
YOLO11-L [3]	26.2M	3.59	37.5	51.4	42.1	73.0
YOLO11-X [3]	58.8M	4.99	38.3	51.1	43.2	75.0
End-to-end methods with Swin-L backbone						
ED-Pose-SWIN-L* [28]	218.4M	94.11	31.2	38.8	34.7	89.4
ED-Pose-SWIN-L-5S* [28]	218.4M	373.72	30.7	37.9	34.0	90.4
GroupPose-SWIN-L* [11]	219.6M	96.77	36.6	45.7	41.1	88.6
End-to-end methods						
ED-Pose-R50* [28]	48.0M	68.10	32.5	40.8	36.2	87.8
GroupPose-R50* [11]	52.4M	70.44	36.6	45.2	40.5	88.7
DETRPose-S (ours)	11.5M	2.39	42.2	54.3	47.5	85.2
DETRPose-M (ours)	20.8M	3.47	41.6	53.0	46.9	86.2
DETRPose-L (ours)	32.8M	4.66	*45.2*	*55.4*	*50.4*	**89.0**
DETRPose-X (ours)	73.3M	7.36	**45.8**	**55.7**	**51.1**	*87.8*

Pretraining Weights. DETRPose-L achieves a score of 71.5 without pretraining initialization. Initializing the model with DFINE's COCO-pretrained weights lowers performance, while Object365 weights yield the best result at 72.7. These experiments show that initializing DETRPose with DFINE weights is unnecessary to surpass YOLOv8 and YOLO11.

Denoising Keypoint Analysis To understand how the per-keypoint fall-off parameter κ and the segmented area s^2 affect the denoising keypoint strategy, we run multiple experiments varying the available information and report our findings in Tab. 7. We start by assigning a constant $\kappa = 1/(\#\text{ keypoints})$ for each type of keypoint, and define the object area $s^2 = \mathrm{A}_{\mathrm{bbox}} \cdot 0.53$. Using the denoising strategy without prior knowledge of the dataset performs better than training with KSFVF and Pose-LQE (Model 8). Providing the official COCO κ values

Table 6: Ablation study using DETRPose-L on the COCO`val` set.

Model number	# Detections	KSVF	Pose-LQE	DK	Pretraining	mAP	# Params	FLOPs (G)	Latency (ms)
Component Analysis									
Model 1	100				None	69.9	32.8M	121.6	5.24
Model 2	60				None	68.9	32.7M	107.1	4.58
Model 3	60	✓			None	69.9	32.7M	107.1	4.58
Model 4	60		✓		None	69.5	32.8M	107.1	4.66
Model 5	60			✓	None	70.0	32.7M	107.1	4.58
Model 6	60		✓	✓	None	69.6	32.8M	107.1	4.66
Model 7	60	✓		✓	None	70.9	32.7M	107.1	4.58
Model 8	60	✓	✓		None	70.5	32.8M	107.1	4.66
Model 9	60	✓	✓	✓	None	71.5	32.8M	107.1	4.66
Pretraining Analysis									
Model 9	60	✓	✓	✓	None	71.5	32.8M	107.1	4.66
Model 10	60	✓	✓	✓	Coco	71.1	32.8M	107.1	4.66
Model 11	60	✓	✓	✓	Obj2Coco	72.4	32.8M	107.1	4.66
Model 12	60	✓	✓	✓	Obj365	72.7	32.8M	107.1	4.66

Table 7: Ablation study of the denoising keypoints on the COCO-`val` dataset. κ and s^2 represent the per-keypoint fall-off parameter and the segmented area, respectevily.

ID	κ	s^2	AP
ID13	constant	box	71.1
ID14	coco	box	71.2
ID15	constant	mask	71.3
ID16	coco	mask	71.5

adds 0.1, even without masks to extract the area value. Keeping κ constant but computing the area from masks adds another 0.1. Using both COCO κ values and masks (Model 16) gives the best result, with a final score of 71.5. These experiments show that DETRPose can be fine-tuned on custom 2D MPPE datasets that lack mask information (see [5,18] for how to address this problem).

Flexibility of DETRPose We modify the number of layers and the number of queries in DETRPose-L and report results in Tab. 8. We use the DETRPose-L trained on the COCO dataset, and the results in this section do not require any additional training. First, we remove deeper layers from the decoder. We observe that using 4 layers results in a 1.2% drop in AP score, while latency reduces by 13.5% compared to the original architecture (6 decoder layers). To our surprise, when we use 3 layers in the decoder, the AP significantly drops.

We observe that reducing the number of queries has a small effect on performance. One reason is that the average and maximum numbers of people in an image are significantly lower than 40. We also evaluate DETR-Pose-L with 25 queries and 4 and 5 decoder layers (last two rows of Tab. 8), achieving better results than YOLOv8/11-X while being faster than YOLOv8/11-L.

Table 8: Study of the flexibility of DETRPose-L on the COCO-`val` dataset.

# Queries	# Layers	GFLOPs	# Params	AP	Latency (ms)
Varying the # of decoder layers					
60	6	107	32.8M	72.7	4.66
60	5	104	31.4M	72.6	4.25
60	4	100	30.0M	71.8	4.03
60	3	96	28.6M	70.6	3.69
Varying the # of queries					
50	6	104	32.8M	72.6	4.48
40	6	100	32.8M	72.4	4.29
30	6	96	32.8M	71.9	4.05
25	6	95	32.8M	71.5	3.96
Varying the # of decoder layers and queries					
25	5	93	31.4M	71.4	3.73
25	4	91	30.0M	70.7	3.53

5 Conclusion

In this paper, we have presented a new real-time end-to-end pose estimator. DETRPose achieves new state-of-the-art results and faster convergence. Results on multiple datasets demonstrate that DETRPose's excellent performance, proving that DETR-based models can be effectively adapted for pose estimation, opening a promising future for DETR models in the MPPE task. A limitation of this work is that DETRPose has higher latency than YOLO model, despite having similar FLOPs. The main reason for this is the group attention, which requires multiple transpose operations, breaking the contiguity of tensors.

Acknowledgements

This work was supported in part by the National Science Foundation under Grant 1949230, Grant1842220, and Grant 1613637. We would like to thank Lambda.ai and UNM Center for Advanced Research Computing for providing the high performance computing resources used in this work.

References

1. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Proceedings of the European Conference on Computer Vision. pp. 213–229 (2020)
2. Jocher, G., Chaurasia, A., Qiu, J.: Ultralytics yolov8 (2023), `https://github.com/ultralytics/ultralytics`
3. Jocher, G., Qiu, J.: Ultralytics yolo11 (2024), `https://github.com/ultralytics/ultralytics`
4. Jocher, G., Qiu, J.: Ultralytics yolo26 (2026), `https://github.com/ultralytics/ultralytics`
5. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything. arXiv:2304.02643 (2023)
6. Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L.: Dn-detr: Accelerate detr training by introducing query denoising. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13619–13627 (2022)
7. Li, F., Zhang, H., xu, H., Liu, S., Zhang, L., Ni, L.M., Shum, H.Y.: Mask dino: Towards a unified transformer-based framework for object detection and segmentation (2022)
8. Li, J., Wang, C., Zhu, H., Mao, Y., Fang, H.S., Lu, C.: Crowdpose: Efficient crowded scenes pose estimation and a new benchmark. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10863–10872 (2019)
9. Li, X., Wang, W., Hu, X., Li, J., Tang, J., Yang, J.: Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. arXiv preprint arXiv:2011.12885 (2020)
10. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Proceedings of the European Conference on Computer Vision. pp. 740–755 (2014)
11. Liu, H., Chen, Q., Tan, Z., Liu, J., Wang, J., Su, X., Li, X., Yao, K., Han, J., Ding, E., Zhao, Y., Wang, J.: Group pose: A simple baseline for end-to-end multi-person pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2023)
12. Liu, S., Li, F., Zhang, H., Yang, X., Qi, X., Su, H., Zhu, J., Zhang, L.: Dab-detr: Dynamic anchor boxes are better queries for detr. arXiv preprint arXiv:2201.12329 (2022)
13. Lu, P., Jiang, T., Li, Y., Li, X., Chen, K., Yang, W.: RTMO: Towards high-performance one-stage real-time multi-person pose estimation (2023)
14. Maji, D., Nagori, S., Mathew, M., Poddar, D.: Yolo-pose: Enhancing yolo for multi person pose estimation using object keypoint similarity loss. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2637–2646 (2022)
15. Mao, W., Tian, Z., Wang, X., Shen, C.: Fcpose: Fully convolutional multi-person pose estimation with dynamic instance-aware convolutions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9034–9043 (2021)
16. McNally, W., Vats, K., Wong, A., McPhee, J.: Rethinking keypoint representations: Modeling keypoints and poses as objects for multi-person human pose estimation. arXiv preprint arXiv:2111.08557 (2021)

17. Peng, Y., Li, H., Wu, P., Zhang, Y., Sun, X., Wu, F.: D-fine: Redefine regression task in detrs as fine-grained distribution refinement (2024)
18. Ravi, N., Gabeur, V., Hu, Y.T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Alwala, K.V., Carion, N., Wu, C.Y., Girshick, R., Dollár, P., Feichtenhofer, C.: Sam 2: Segment anything in images and videos. arXiv preprint arXiv:2408.00714 (2024), https://arxiv.org/abs/2408.00714
19. Robinson, I., Robicheaux, P., Popov, M.: Rf-detr. https://github.com/roboflow/rf-detr (2025), sOTA Real-Time Object Detection Model
20. Shao, S., Li, Z., Zhang, T., Peng, C., Yu, G., Li, J., Zhang, X., Sun, J.: Objects365: A large-scale, high-quality dataset for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8425–8434 (2019)
21. Shi, D., Wei, X., Li, L., Ren, Y., Tan, W.: End-to-end multi-person pose estimation with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11069–11078 (2022)
22. Shi, D., Wei, X., Yu, X., Tan, W., Ren, Y., Pu, S.: Inspose: instance-aware networks for single-stage multi-person pose estimation. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 3079–3087 (2021)
23. Shihua Huang, Zhichao Lu, X.C.Y.Y.X.Z., Shen, X.: Deim: Detr with improved matching for fast convergence (2025)
24. Tian, Z., Chen, H., Shen, C.: Directpose: Direct end-to-end multi-person pose estimation. arXiv preprint arXiv:1911.07451 (2019)
25. Wang, D., Zhang, S.: Contextual instance decoupling for robust multi-person pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11060–11068 (June 2022)
26. Wang, Y., Zhang, X., Yang, T., Sun, J.: Anchor detr: Query design for transformer-based detector. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 2567–2575 (2022)
27. Xiao, Y., Su, K., Wang, X., Yu, D., Jin, L., He, M., Yuan, Z.: Querypose: Sparse multi-person pose regression via spatial-aware part-level query. In: Advances in Neural Information Processing Systems. pp. 1–14 (2022)
28. Yang, J., Zeng, A., Liu, S., Li, F., Zhang, R., Zhang, L.: Explicit box detection unifies end-to-end multi-person pose estimation. In: International Conference on Learning Representations. pp. 1–17 (2023)
29. Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L., Shum, H.: Dino: Detr with improved denoising anchor boxes for end-to-end object detection. In: International Conference on Learning Representations. pp. 1–18 (2022)
30. Zhang, H., Wang, Y., Dayoub, F., Sünderhauf, N.: Varifocalnet: An iou-aware dense object detector. In: CVPR (2021)
31. Zhang, S.H., Li, R., Dong, X., Rosin, P., Cai, Z., Han, X., Yang, D., Huang, H., Hu, S.M.: Pose2seg: Detection free human instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
32. Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., Chen, J.: Detrs beat yolos on real-time object detection (2023)