Supplementary Material for Point-Query Quadtree for Crowd Counting, Localization, and More

Chengxin Liu[1] Hao Lu[1] Zhiguo Cao[1*] Tongliang Liu[2]
[1] Key Laboratory of Image Processing and Intelligent Control, Ministry of Education
School of Artificial Intelligence and Automation
Huazhong University of Science and Technology, China
[2]The University of Sydney, Australia
{cx_liu, hlu, zgcao}@hust.edu.cn

Table 1: Performance of sparse point queries with different point-query stride K.

Stride K	16	10	8	4
SH PartA MAE	62.19	55.03	**53.59**	54.16

A. Analysis on the Point-Query Quadtree

In this section, we first introduce how we select the initial point-query stride K. Then, we analyze the depth of the quadtree. In addition, we also give a more detailed discussion on the quadtree loss in Eq. (4).

A.1. Selecting the Initial Point-Query Stride K

We consider two criteria when selecting the initial point-query stride K: i) It should achieve moderate performance; ii) Its value should fall into an appropriate range, *i.e.*, neither too large nor too small. Therefore, we first investigate the performance of sparse point queries with different stride K. In case of misunderstanding, **we remark that here we do not use the quadtree, but only span sparse point queries across an image.**

As shown in Table 1, one can observe that: i) With the stride reduced, the MAE first decreases and then rises. The performance tends to be stable with K is around 8, which suggests that the model is relatively insensitive to K within a certain interval; ii) A large point-query stride, *e.g.*, $K = 16$, yields inferior results. This is reasonable because too few points lead to systematic underestimation. For example, when dealing with congested regions, the number of point queries is not sufficient to cover all persons. As a result, the MAE surges in such scenarios, thus affecting the overall MAE. *Note that this result has nothing to do with robustness, but the natural pitfall of using a large point-query stride*; iii) A small point-query stride, *e.g.*, $K = 4$, does not bring further improvement. This could attribute to the ambiguity during bipartite matching. To be specific, a ground-truth point may correspond to several similar point queries on sparse regions, which impedes the model to discriminate valid points. In addition, a small point-query stride also results in a large computational cost.

To achieve a balance between performance and computational cost, we set the initial point-query stride $K = 8$.

Table 2: Statistics of the maximum count of existing datasets inside a 256×256 patch. The statistics are computed on training images.

Dataset	SH PartA	UCF-QNRF	JHU-Crowd	NWPU
Maximum Count	973	1350	1204	974

A.2. The Depth of the Quadtree

We analyze the depth of the point-query quadtree from two perspectives, including the statistics perspective and the distance perspective. The former determines the depth of the quadtree by analyzing the statistics of crowd, while the latter inspects the distance distribution between ground-truth points and point queries. Both of them suggest that splitting once is generally sufficient for crowd prediction.

Statistics Perspective. Recall that we crop 256×256 patches from the input images for training. An intuitive way to determine the depth of the quadtree is to obtain the statistics of crowd inside a 256×256 patch. In another word, we should ensure that the number of point queries is larger than the maximum number of persons in a patch.

To acquire such statistics, we use a sliding window to obtain all possible 256×256 patches inside an image, and

*corresponding author

Figure 1: **Visualization of the matched querying points**. Red and yellow points are ground-truth points and matched querying points, respectively. Best viewed with zoom in.

Table 3: Statistics of $\mathcal{D}$ on ShanghaiTech PartA and UCF-QNRF datasets, where $\mathcal{D}' = \{d_i | d_i > K, i \in \{1, \ldots, N\}$.

ID	Dataset	Stride K	Quantile of $\mathcal{D}$			$\frac{\lvert\mathcal{D}'\rvert}{\lvert\mathcal{D}\rvert}$
			50%	90%	100%	
B1	SH PartA	8	3.3	5.2	84.5	4.5%
B2		4	1.6	2.3	5.9	0.022%
B3	UCF-QNRF	8	3.4	5.9	346.9	7.0 %
B4		4	1.6	2.3	16.1	0.074%

compute the maximum count among all patches. The statistics of existing datasets are listed in Table 2. UCF-QNRF dataset has the highest crowd density, with a maximum count of 1350. For comparison, using a stride of 8 could produce at most 1024 points. After one-time splitting, the maximum number of point queries could reach 4096, which is significantly larger than 1350. Therefore, splitting once is generally sufficient for crowd prediction.

Distance Perspective. Here we analyze the distance between ground-truth points and point queries. For each image, we first span the querying points across it with stride K, and then compute the bipartite matching between these querying points and point annotations. This results in a set of matched points pairs $\mathcal{S} = \{(\mathbf{p}_i, \mathbf{q}_i) | i \in \{1, \ldots, N\}\}$, where $\mathbf{p}_i$ is the i^{th} point annotation, $\mathbf{q}_i$ is the matched querying point, and N is the total number of point annotations. Based on $\mathcal{S}$, we further obtain a set of matched distances $\mathcal{D} = \{d_i | i \in \{1, \ldots, N\}\}$, where d_i is the Euclidean distance between $\mathbf{p}_i$ and $\mathbf{q}_i$. Table 3 shows the statistics of $\mathcal{D}$ on two crowd counting datasets.

Take B1 for example, when the stride of querying points equals 8 ($K = 8$), we observe that 90% of the matched distance in $\mathcal{D}$ is less than 5.2 while the maximum matched distance is 84.5. In addition, 4.5% of the matched distance in $\mathcal{D}$ is larger than 8. After splitting (B2), *i.e.*, stride K is reduced from 8 to 4, only 0.022% of the matched distance in $\mathcal{D}$ is larger than 4 while the maximum matched distance is 5.9. Similarly, for UCF-QNRF, only 0.074% of

Figure 2: **Example images that contain dense regions**. Yellow boxes highlight the dense regions.

the matched distance is larger than 4 after splitting. This suggests that splitting once is generally sufficient to deal with crowd estimation. Fig. 1 shows some examples of the matched querying points. One can observe:

- For sparse regions, $K=8$ is sufficient, *i.e.*, the quadtree is unnecessary to split;
- For congested regions, some matched querying points are far from ground-truth points when $K=8$. After splitting ($K=4$), the matched querying points are sufficiently close to ground-truth points.

To summarize, the above analysis both conclude that splitting once is generally sufficient to deal with crowd prediction when the initial point-query stride is set to 8.

A.3. Discussion on the Quadtree Loss in Eq. (4)

For convenience, we post Eq. (4) here:

$$\ell_{split} = \mathbb{1}(\texttt{dense})(1-\max(M_s)) + \min(M_s)\,, \quad (1)$$

where M_s is the split map, and $\mathbb{1}(\texttt{dense})$ equals 1 if the input image has dense regions, otherwise 0. We consider an image has dense regions if its crowd density is high. The crowd density $C_{\texttt{den}}$ is defined by the average nearest distance between ground-truth points. Given a set of ground-truth points $\mathcal{Y}=\{\mathbf{y}_i\}_{i=1}^{M}$, $C_{\texttt{den}}$ is computed as:

$$C_{\texttt{den}} = \frac{1}{M}\sum_{i} \min_{j\in\{1,\ldots,M\},j\neq i} d(\mathbf{y}_i,\mathbf{y}_j)\,, \quad (2)$$

where $d(\cdot,\cdot)$ denotes Euclidean distance. An image is considered to have dense regions if $C_{\texttt{den}}$ is smaller than $2K$, where K is the point-query stride. Fig. 2 shows some example images that contain dense regions.

One may consider that whether the definition of dense regions has significant impact on the quadtree splitter. Interestingly, we found that the definition is not that important. Eq. (4) works even if we do not define dense regions. To be specific, we can simply eliminate the indicator function of dense regions and rewrite Eq. (4) as follows:

$$\ell_{split} = (1-\max(M_s)) + \min(M_s). \quad (3)$$

Although Eq. (3) works in an unsupervised manner, the quadtree splitter can still output reasonable split maps and

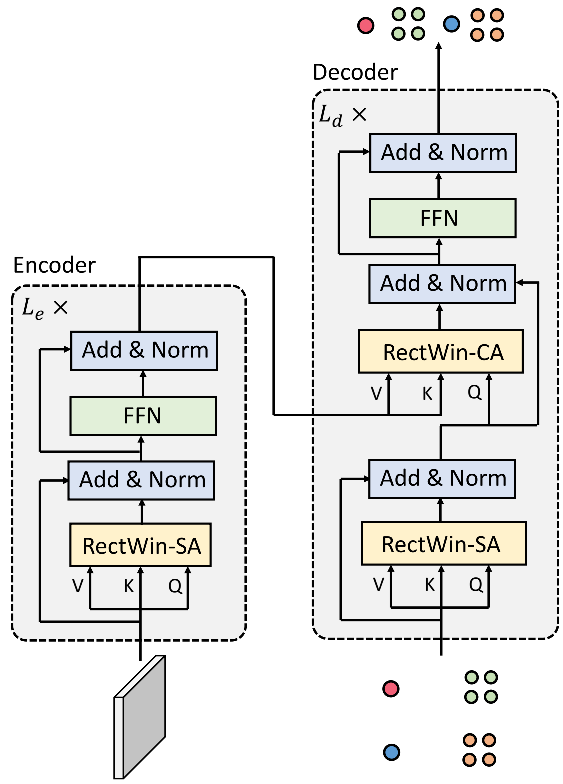

Figure 3: **Detailed architecture of transformer encoder and decoder.**

the performance of Eq. (3) is similar to Eq. (4). This phenomenon can attribute to the weak supervision of Eq. (3), as it only samples one element in M_s when computing the loss of dense and sparse regions. Such weak supervision enables the model to discriminate dense regions without external guidance. Given that Eq. (3) can be computed within a batch instead of one sample, the first term is often valid because the training images often contain dense regions. Therefore, the quadtree splitter can still be correctly trained.

B. Detailed Architecture of Transformer

Fig. 3 shows the detailed architecture of transformer encoder and decoder.

Encoder. The encoder attention is computed as:

$$\begin{aligned}\hat{\mathbf{x}}^l &= \texttt{LN}(\texttt{RectWin-SA}(\mathbf{x}^{l-1}) + \mathbf{x}^{l-1})\,,\\ \mathbf{x}^l &= \texttt{LN}(\texttt{FFN}(\hat{\mathbf{x}}^l) + \hat{\mathbf{x}}^l)\,,\end{aligned} \quad (4)$$

where $\mathbf{x}^{l-1}$ and $\mathbf{x}^l$ are the output features of the encoder layer $l-1$ and the layer l, respectively. Note that $\mathbf{x}^0$ is initialized with CNN feature $\mathcal{F}$. `RectWin-SA`, `FFN`, and `LN` denote rectangle window self-attention, feed-forward network, and layer normalization, respectively. In particular, feed-forward network consists of two MLP layers with

ReLU activation. The input, hidden, and output dimension of feed-forward network is 256, 512, and 256. Regarding rectangle window self-attention, the input dimension and number of head are set to 256 and 8. In addition, the number of layer L_e is set to 4. For the first two layer, we use a rectangle window with a size of $s_e \times r_e s_e$, where $s_e = 16$ and $r_e = 2$. For the last two layers, we adopt a smaller rectangle window with a size of $\frac{1}{2}s_e \times \frac{1}{2}r_e s_e$.

Decoder. The decoder attention is computed as:

$$\begin{aligned}
\hat{\mathbf{z}}^l &= \mathtt{LN}(\mathtt{RectWin\text{-}SA}(\mathbf{z}^{l-1}) + \mathbf{z}^{l-1})\,, \\
\hat{\mathbf{z}}^l &= \mathtt{LN}(\mathtt{RectWin\text{-}CA}(\hat{\mathbf{z}}^l, \mathbf{x}^N) + \hat{\mathbf{z}}^l)\,, \\
\mathbf{z}^l &= \mathtt{LN}(\mathtt{FFN}(\hat{\mathbf{z}}^l) + \hat{\mathbf{z}}^l)\,,
\end{aligned} \tag{5}$$

where $\mathbf{x}^N$ is the final output of the transformer encoder, $\mathbf{z}^{l-1}$ and $\mathbf{z}^l$ are the output features of the decoder layer $l-1$ and the layer l, respectively. Note that $\mathbf{z}^0$ is initialized with the representation of point queries. `RectWin-SA` and `RectWin-CA` denote the rectangle window self-attention and the rectangle window cross-attention, respectively. The number of layer L_d is set to 2. The configurations of feed-forward network and attention are the same as transformer encoder.

Recall that we adopt a point-query quadtree with a depth of 2. For sparse point queries, the rectangle window is with a size of $\frac{1}{2}s_e \times \frac{1}{2}r_e s_e$. For dense point queries, we use a smaller window with a size of $\frac{1}{4}s_e \times \frac{1}{4}r_e s_e$.

Bipartite Matching. The output of PET is a set of candidate crowd $\mathcal{Q} = \{\mathbf{q}_i\}_{i=1}^N$. We optimize the network based on the bipartite matching between these predictions and ground truths. Let $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1}^M$ denote the set of ground-truth points, we define the cost matrix between $\mathcal{Q}$ and $\mathcal{Y}$ as follows:

$$\mathcal{C}_{\text{match}}(\mathcal{Q}, \mathcal{Y}) = (-c_i + \alpha \|\mathbf{q}_i - \mathbf{y}_j\|_2)_{i \in \{1,\dots,N\}, j \in \{1,\dots,M\}}, \tag{6}$$

where c_i is the classification probability, $\|\cdot\|$ denotes ℓ_2 distance, α is a balancing factor, and $N > M$. Eq. (6) jointly considers the classification and localization information, aiming to achieve optimal matching. The above matching process will output a matched index σ (Line 505 in the paper). Note that α is set to 0.05 during training.

C. More Quantitative Results

Results on the UCF_CC_50 Dataset. To further demonstrate the effectiveness of our approach on dense scenes, we conduct experiments on the UCF_CC_50 dataset. We follow previous work to perform a 5-fold evaluation. As shown in Table 4, our PET significantly outperforms state-of-the-art methods, achieving an MAE of 159.96 and MSE of 223.79. This supports the adaption of PET on dense scenes.

Table 4: Crowd counting results on the UCF_CC_50 dataset.

Method	Venue	MAE	MSE
CSRNet [4]	CVPR'18	266.1	397.5
CAN [5]	CVPR'19	212.2	243.7
BL+ [6]	ICCV'19	229.3	308.2
S-DCNet [9]	ICCV'19	204.2	301.3
DM-Count [8]	NeurIPS'20	211.0	291.5
ASNet [3]	CVPR'20	174.84	<u>251.63</u>
P2PNet [7]	ICCV'21	<u>172.72</u>	256.18
GauNet+CSRNet [2]	CVPR'22	215.4	296.4
PET - Ours	-	**159.96**	**223.79**

Table 5: Results of crowd distribution generalization. The model is trained on ShanghaiTech PartB and tested on ShanghaiTech PartA.

Method	D2CNet [1]	P2PNet [7]	PET (Ours)
MAE / MSE	164.5 / 286.4	144.3 / 251.5	**132.4 / 245.7**

Table 6: Impact of different encoder windows. r_e stands for aspect ratio.

Encoder Window	$r_e = 2$	$r_e = 3$	$r_e = 4$
SH PartA MAE	49.34	49.09	49.15

Crowd distribution generalization. To justify the generalization capability of our approach, we train the model on ShanghaiTech PartB and test it on ShanghaiTech PartA. The idea is to investigate whether the model can transfer from low crowd density data to high crowd density data. Table 5 reports the results. We observe that PET exhibits good generalization capability, outperforming existing localization-based methods by a considerable margin.

Effect of Encoder Window. Here we investigate the effect of different encoder windows by adjusting aspect ratios r_e. The results are shown in Table 6. One can observe that PET is insensitive to the configuration of encoder window. Although using a larger aspect ratio (*e.g.*, $r_e = 3$) could slightly improve the performance, it will increase computational cost. Therefore, we simply set r_e as 2.

D. Qualitative Results of Model Predictions

Here we show some qualitative results of model predictions on ShanghaiTech PartA (Fig. 4) and UCF-QNRF datasets (Fig. 5). The left column shows the ground-truth points, while the right column shows the model predictions, in conjunction with the split map.

Figure 4: **Qualitative results of model predictions on the ShanghaiTech PartA dataset**. Red regions denote congested regions that require the quadtree to split.

Figure 5: **Qualitative results of model predictions on the UCF-QNRF dataset**. Red regions denote congested regions that require the quadtree to split.

E. Qualitative Results of Attention Maps

Fig. 6 and Fig. 7 show more qualitative results of attention maps. We can observe that a higher attention value occurs in similar crowd.

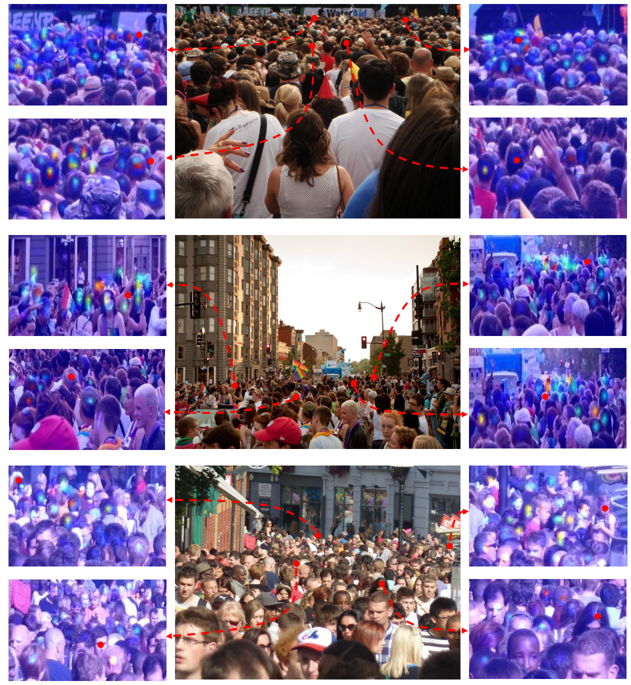

Figure 6: **Qualitative results of encoder attention maps**. Red points in the original images denote reference points.

Figure 7: **Qualitative results of decoder attention maps**. Red points in the original images denote point queries.

References

[1] Jian Cheng, Haipeng Xiong, Zhiguo Cao, and Hao Lu. Decoupled two-stage crowd counting and beyond. *IEEE TIP*, 30:2862–2875, 2021. 4

[2] Zhi-Qi Cheng, Qi Dai, Hong Li, Jingkuan Song, Xiao Wu, and Alexander G. Hauptmann. Rethinking spatial invariance of convolutional networks for object counting. In *CVPR*, pages 19638–19648, 2022. 4

[3] Xiaoheng Jiang, Li Zhang, Mingliang Xu, Tianzhu Zhang, Pei Lv, Bing Zhou, Xin Yang, and Yanwei Pang. Attention scaling for crowd counting. In *CVPR*, pages 4705–4714, 2020. 4

[4] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *CVPR*, pages 1091–1100, 2018. 4

[5] Weizhe Liu, Mathieu Salzmann, and Pascal Fua. Context-aware crowd counting. In *CVPR*, pages 5099–5108, 2019. 4

[6] Zhiheng Ma, Xing Wei, Xiaopeng Hong, and Yihong Gong. Bayesian loss for crowd count estimation with point supervision. In *ICCV*, pages 6141–6150, 2019. 4

[7] Qingyu Song, Changan Wang, Zhengkai Jiang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yang Wu. Rethinking counting and localization in crowds: A purely point-based framework. In *ICCV*, pages 3365–3374, 2021. 4

[8] Boyu Wang, Huidong Liu, Dimitris Samaras, and Minh Hoai Nguyen. Distribution matching for crowd counting. In *NeurIPS*, 2020. 4

[9] Haipeng Xiong, Hao Lu, Chengxin Liu, Liang Liu, Zhiguo Cao, and Chunhua Shen. From open set to closed set: Counting objects by spatial divide-and-conquer. In *ICCV*, pages 8361–8370, 2019. 4