Few-shot Object Counting with Similarity-Aware Feature Enhancement
– *Supplementary Material* –

Zhiyuan You[1], Kai Yang[2], Wenhan Luo[3], Xin Lu[2], Lei Cui[4], Xinyi Le[1*]
[1]Shanghai Jiao Tong University, [2]SenseTime, [3]Sun Yat-sen University, [4]Tsinghua University
zhiyuanyou@foxmail.com, {yangkai, luxin}@sensetime.com, whluo.china@gmail.com
cuil19@mails.tsinghua.edu.cn, lexinyi@sjtu.edu.cn, *Corresponding Author

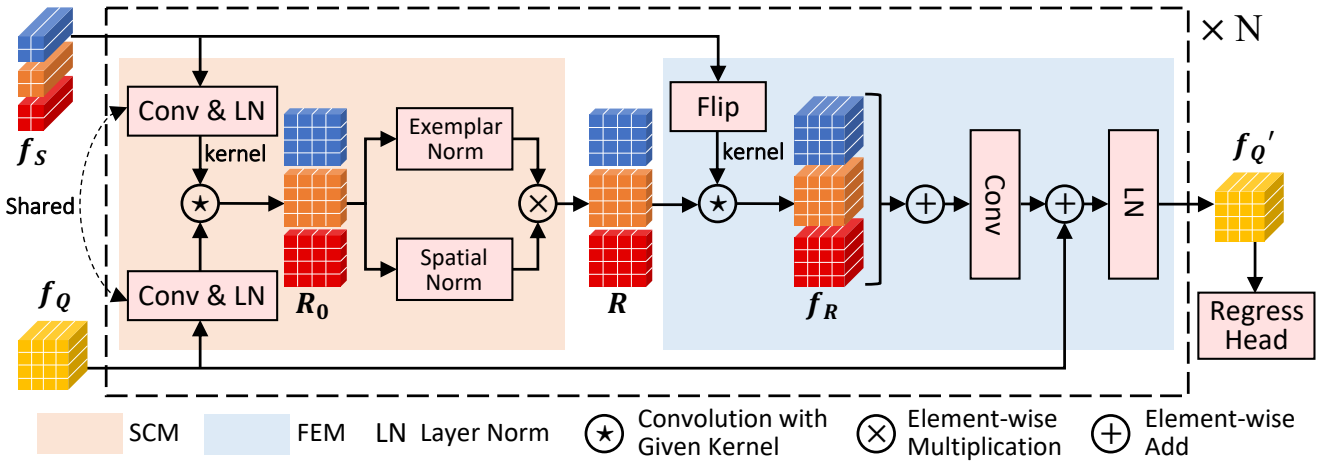


Figure 1. **Illustration of the similarity-aware feature enhancement block** under the 3-shot case.

1. Network Architecture and Training Configurations

This part describes the detailed architecture of our SAFECount block and other assistant modules, followed by the training configurations. To make the supplementary material self-contained, we recall the proposed similarity-aware feature enhancement block in Fig. 1.

Feature Extractor. We select ResNet-18 [6] pre-trained on ImageNet [4] as the feature extractor.[1] Given a query image, $Q \in \mathbb{R}^{3\times512\times512}$, we resize the outputs of the first three residual stages of ResNet-18 to the same size, 128×128, and concatenate them along the channel dimension. Afterward, a 1×1 convolutional layer is applied to reduce the channel dimension to 256, resulting in the query feature, $\boldsymbol{f_Q} \in \mathbb{R}^{256\times128\times128}$. The size of ROI pooling [17] is set as 3×3, so the support feature, $\boldsymbol{f_S}$, has the shape of $K \times 256 \times 3 \times 3$ in the K-shot case. The backbone is frozen during training, while the 1×1 convolutional layer is not.

[1]We borrow the checkpoint here.

Similarity Comparison Module (SCM). Our SCM is implemented with three steps: *learnable feature projection*, *feature comparison*, and *score normalization*. The *feature comparison* is implemented by convoluting the query feature, $\boldsymbol{f_Q}$, with the support feature, $\boldsymbol{f_S}$, as kernels, deriving a score map, $\boldsymbol{R_0}$. This process is illustrated intuitively in Fig. 2a. Other components in SCM have been detailed in the paper. The SCM finally outputs a similarity map, $\boldsymbol{R} \in \mathbb{R}^{K\times1\times128\times128}$.

Feature Enhancement Module (FEM). The FEM is composed of two steps: *weighted feature aggregation* and *learnable feature fusion*. The *weighted feature aggregation* treats the values in the similarity map, $\boldsymbol{R}$, as weighting coefficients to integrate $\boldsymbol{f_S}$, producing the similarity-weighted feature, $\boldsymbol{f_R}$. This process is realized by the convolution, as shown in Fig. 2b. Besides, before serving as the convolutional kernels, $\boldsymbol{f_S}$ is flipped both horizontally and vertically. As illustrated in Fig. 3, the flipping helps $\boldsymbol{f_R}$ inherit the spatial structure from $\boldsymbol{f_S}$. In Fig. 3, $\boldsymbol{R}$ is a unit impulse function, meaning that only one position

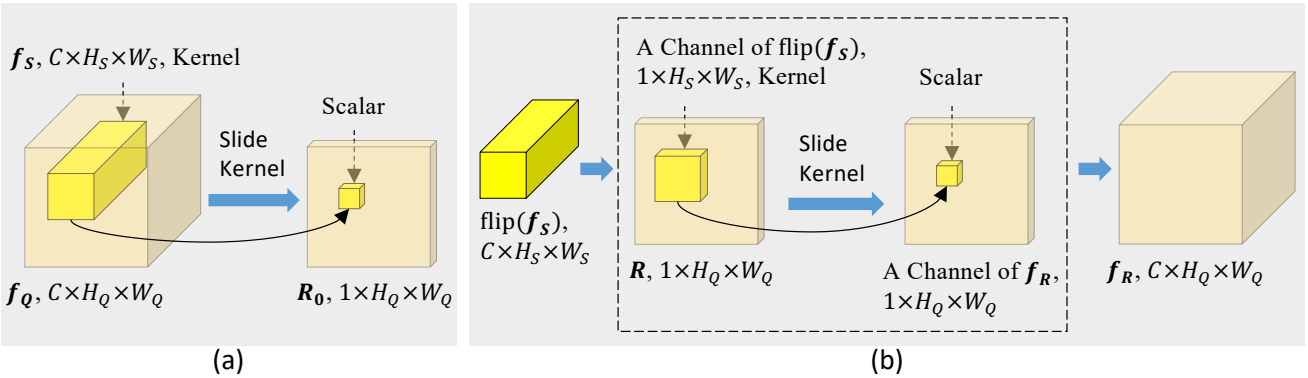


Figure 2. **(a) Illustration of the *feature comparison*** in SCM under the 1-shot case, where the feature projection is omitted. **(b) Illustration of the *weighted feature aggregation*** in FEM under the 1-shot case.

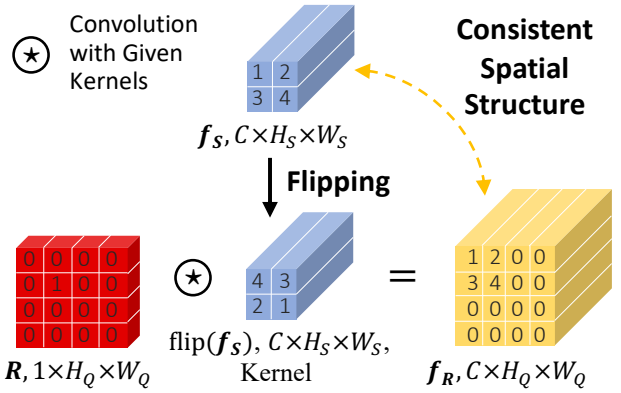


Figure 3. **Illustration of kernel flipping in FEM**, which helps f_R inherit the spatial structure from f_S. The convolution is implemented with the same padding strategy.

Table 1. **Network architectures** of (a) the *learnable feature fusion* in FEM, where the skip connection and the layer normalization are omitted, and (b) the regress head.

(a) Architecture of the learnable feature fusion

layer	kernel	in	out	activation
Conv	3×3	256	1024	Leaky ReLU
Conv	3×3	1024	256	-

(b) Architecture of the regress head

layer	kernel	in	out	activation	followed by
Conv	5×5	256	128	Leaky ReLU	$2\times$Upsample
Conv	3×3	128	64	Leaky ReLU	$2\times$Upsample
Conv	1×1	64	32	Leaky ReLU	-
Conv	1×1	32	1	ReLU	-

has the maximum similarity with f_S, while other positions have no similarity with f_S at all. Therefore, f_R should have a sub-part exactly the same with f_S in the position corresponding to the maximum similarity, while the others should be zero vectors. The *weighted feature aggregation* constructs f_R following the above insights via flipping and convolution. The *learnable feature fusion* is completed by a 2-layer convolutional network, skip connection, and layer normalization. The architecture of the convolutional network is shown in Tab. 1a. Other components in FEM have been detailed in the paper. Eventually, the FEM produces the enhanced feature, $f'_Q \in \mathbb{R}^{256\times 128\times 128}$.

Regress Head. The regress head regresses the density map, $D \in \mathbb{R}^{512\times 512}$, from the enhanced feature, f'_Q. The regression head is composed of a sequence of convolutional layers, followed by the Leaky ReLU activation and bi-linear upsampling, as shown in Tab. 1b.

Multi-block Architecture. The enhanced feature derived by one block, f'_Q, could serve as the input to the next block by taking the place of the query feature, f_Q, forming a multi-block architecture. As for another input of the next block, the support feature, f_S, there are two choices. If the support image is cropped from the query image, f_S is updated by ROI pooling on newly obtained f'_Q. If not, f_S does not change in different blocks.

Training Configuration on FSC-147 [15]. The sizes of the query image, the query feature, and the support feature are selected as 512×512, 128×128, and 3×3, respectively. The SAFECount block number is set as 4. The model is trained with Adam optimizer [9] for 200 epochs with batch size 8. The hyper-parameter ϵ in Adam optimizer is set as 4e-11, much smaller than the default 1e-8, considering the small norm of the losses and the gradients. The learning rate is set as 2e-5 initially, and it is dropped by 0.25 after every 80 epochs. Data augmentation methods including random horizontal flipping, color jittering, and random gamma transformation are adopted.

2. Ablation Studies

This part conducts comprehensive ablation studies on the components of our approach.

Table 2. **Ablation studies** regarding (a) loss weight term α in Eq. (1), (b) size of ROI pooling.

(a) α in Eq. (1)

α	Val Set		Test Set	
	MAE	RMSE	MAE	RMSE
1e-3	15.15	52.02	15.42	95.77
1e-4	**14.18**	53.65	**13.55**	89.69
1e-5	15.11	56.05	14.63	93.41
0	15.28	**47.20**	14.32	**85.54**

(b) Size of ROI Pooling

Size of ROI Pooling	Val Set		Test Set	
	MAE	RMSE	MAE	RMSE
1×1	15.83	54.65	16.13	95.52
3×3	**15.28**	**47.20**	**14.32**	**85.54**
5×5	15.57	53.79	15.18	89.32

Loss Function. The loss function described in the paper is MSE loss. Actually, we also implement experiments with another SSIM term as follows,

$$\mathcal{L} = \texttt{MSE}(\boldsymbol{D}, \boldsymbol{D_{GT}}) - \alpha \texttt{SSIM}(\boldsymbol{D}, \boldsymbol{D_{GT}}), \qquad (1)$$

where $\texttt{SSIM}(\cdot)$ is the structural similarity function [21], which measures the local pattern consistence between the predicted density map and the ground-truth, α is the weight term. The results with different α are shown in Tab. 2a. Adding the SSIM term promotes the performance of MAE but with the sacrifice of RMSE. Compared with MAE, RMSE relies more heavily on the prediction of the samples with extremely large count. Therefore, we speculate that the SSIM term is beneficial to some samples, but may harm the samples with extremely large count. We finally decide not to add the SSIM term, because the performance drop of RMSE is too large.

Size of ROI Pooling. To study the influence of the ROI pooling size, we conduct experiments with different ROI pooling sizes. The results are shown in Tab. 2b. The performance is the worst with the ROI pooling size as 1×1, *i.e.* pooling to a support vector, since pooling to a support vector fully omits the spatial information of the support image. Adding the ROI pooling size to 3×3 brings stable improvement. However, further increasing the ROI pooling size to 5×5 decreases the performance slightly. This may be because too large ROI pooling size would slightly hinder the accurate localization of target objects. Accordingly, we select the ROI pooling size as 3×3 for FSC-147.

3. More Results

This part presents more experimental results, including the quantitative evaluation on various class-specific counting datasets [2, 3, 7, 24], as well as some visual samples.

3.1. Class-specific Object Counting

Our method is designed to be a general class-agnostic FSC approach. Nonetheless, we still evaluate our method on class-specific counting tasks to further testify its superiority.

Class-specific Counting Datasets. We select five class-specific counting datasets including two car counting datasets: CARPK [7] and PUCPR+ [7] and three crowd counting datasets: ShanghaiTech (PartA and PartB) [24], UCSD [2], and Mall [3]. The details of these datasets are given in Tab. 3.

Table 3. Class-specific counting datasets.

Type	Dataset	#Images	#Objects
Car	CARPK [7]	1448	89,777
	PUCPR+ [7]	125	16,916
Crowd	PartA [24]	482	241,677
	PartB [24]	716	88,488
	UCSD [2]	2000	49,885
	Mall [3]	2000	62,325

Training Configuration on Class-specific Counting. The size of the support feature is set as 1×1. The block number is set as 2. Data augmentation methods including random flip, color jitter, random rotation, and random grayscale are used to prevent over-fitting and improve the generalization ability. Other setups are the same as **FSC-147**.

Car Counting. Car counting tasks are conducted on CARPK [7] and PUCPR+ [7]. 5 support images are randomly sampled from the training set and *fixed for both training and test*. Our method is compared with 4 categories of baselines: object detectors, single-class car counting methods, multi-class counting methods, and FSC methods. Note that multi-class counting methods could only count classes in training set, while FSC methods can count unseen classes. The quantitative results are shown in Tab. 4a. Our approach surpasses all multi-class counting methods and FSC methods with a large margin, and achieves comparable performance with single-class car counting methods.

Crowd Counting. Crowd counting tasks are implemented on UCSD [2], Mall [3], and ShanghaiTech [24]. We randomly sample 5 support images from the training set and *fixed them for both training and test*. 3 kinds of competitors are included: single-class crowd counting methods, multi-class counting methods, and FSC methods. The results of MAE are reported in Tab. 4b. For UCSD and Mall where the crowd is relatively sparse, our approach surpasses all counterpart methods stably. For ShanghaiTech, our approach outperforms all multi-class counting methods and FSC methods with a large margin, and achieves competitive performance on par with specific crowd counting methods. It is emphasized that, our method is not tailored to the specific crowd counting task, while the compared methods are.

Table 4. **Counting performance on class-specific datasets**, including CARPK [7], PUCPR+ [7], UCSD [2], Mall [3], and ShanghaiTech (Part A & Part B) [24].

(a) Car Counting

	Method	CARPK		PUCPR+	
		MAE	RMSE	MAE	RMSE
1	YOLO [16]	48.89	57.55	156.00	200.42
	F-RCNN [17]	47.45	57.39	111.40	149.35
	S-RPN [7]	24.32	37.62	39.88	47.67
	RetinaNet [12]	16.62	22.30	24.58	33.12
2	LPN [7]	23.80	36.79	22.76	34.46
	HLCNN [8]	**2.12**	**3.02**	**2.52**	**3.40**
4	One Look [14]	59.46	66.84	21.88	36.73
	IEP Count [19]	51.83	-	15.17	-
	PDEM [5]	6.77	8.52	7.16	12.00
5	GMN [13]	7.48	9.90	-	-
	FamNet [15]	18.19	33.66	14.68†	19.38†
	Ours	**5.33**	**7.04**	**2.42**	**3.55**

(b) Crowd Counting (MAE)

	Method	UCSD	Mall	PartA	PartB
3	Crowd CNN [23]	1.60	-	181.8	32.0
	MCNN [24]	**1.07**	-	110.2	26.4
	Switch-CNN [1]	1.62		90.4	21.6
	CP-CNN [18]	-	-	73.6	20.1
	CRSNet [11]	1.16	-	68.2	10.6
	RPNet [22]	-	-	**61.2**	8.1
	GLF [20]	-	-	61.3	**7.3**
4	LC-FCN8 [10]	1.51	2.42	-	13.14
	LC-PSPNet [10]	1.01	2.00	-	21.61
5	GMN [13]	-	-	95.8	-
	FamNet [15]	2.70†	2.64†	159.11†	24.90†
	Ours	**0.98**	**1.69**	**73.70**	**9.98**

[1] Detectors provided by the benchmark [7].
[2] Single-class car counting methods.
[3] Single-class crowd counting methods.
[4] Multi-class counting methods (classes for training and test must be the same).
[5] Few-shot counting methods.
† trained and evaluated by ourselves with the official code.

3.2. More Qualitative Results

Qualitative Results on FSC-147 [15]. The qualitative results of FSC-147 are shown in Fig. 4, Fig. 5, and Fig. 6a. For each class, the images from top to down are the query image and the predicted density map. The objects circled by the red rectangles are the support images. The texts below the density map describe the counting results. Our SAFECount could successfully count objects of all categories with various densities and scales, demonstrating strong generalization ability and robustness. Specifically, for both objects with extremely high density (*e.g.*, Legos in Fig. 5) and objects with quite sparse density (*e.g.*, Prawn Crackers in Fig. 5), both small objects (*e.g.*, Birds in Fig. 4) and large objects (*e.g.*, Horses in Fig. 4), both round objects (*e.g.*, Apples in Fig. 4) and square objects (*e.g.*, Stamps in Fig. 5), both vertical strip objects (*e.g.*, Skis in Fig. 5) and horizontal strip objects (*e.g.*, Shirts in Fig. 5), our approach could precisely count objects of interest with high localization accuracy.

Qualitative Results on Class-specific Object Counting. Our method is evaluated on two car counting datasets and three crowd counting datasets. For each dataset, five support images are randomly sampled from the training set and fixed for both training and test, as shown in Fig. 6b. The qualitative results on CARPK [7], PUCPR+ [7], UCSD [2], Mall [3], and ShanghaiTech [24] are shown in Fig. 6c-h. *(1) Car Counting:* Our approach could localize and count cars with different angles and scales successfully. Especially, in the cases that some cars are in the deep shadows (*e.g.*, the $7^{th}, 11^{th}$ examples in Fig. 6c, the 11^{st} example in Fig. 6d) or partly hidden under the trees (*e.g.*, the $3^{rd}, 10^{th}$ examples in Fig. 6c, the $5^{th}, 12^{nd}$ examples in Fig. 6d), our method still accurately localizes these cars, indicating the superiority of our approach. *(2) Crowd Counting:* In the cases of UCSD and Mall where the crowd density is relatively sparse, our approach could count the number of persons precisely with extremely small error. For ShanghaiTech PartA, if the persons in the crowd are distinguishable (*e.g.*, the $1^{st}, 11^{th}$ examples in Fig. 6g), our model could localize each person precisely. If the persons are too crowded to distinguish (*e.g.*, the $2^{nd}, 5^{th}$ examples in Fig. 6g), our method could predict an accurate density estimate for crowds. For ShanghaiTech PartB where most persons are distinguishable, our approach successfully localizes and counts persons, indicating that our approach is capable of crowd counting with various crowd densities.

Figure 4. **Qualitative results** on unseen classes in FSC-147 (from Ants to Keyboard Keys). There are only 2 images of Flowers in FSC-147.

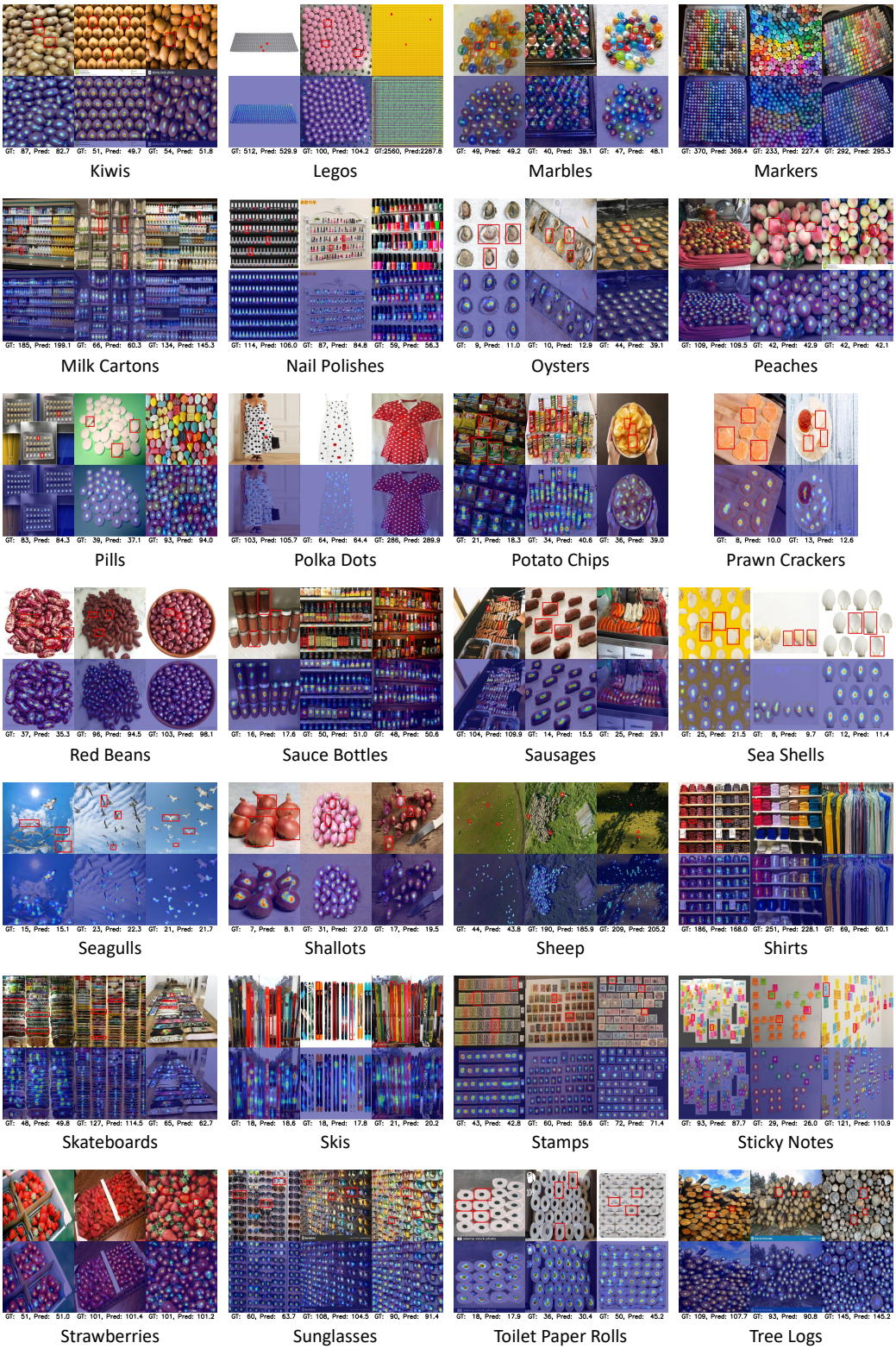
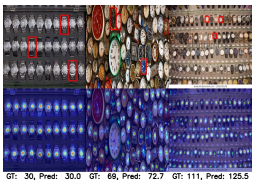
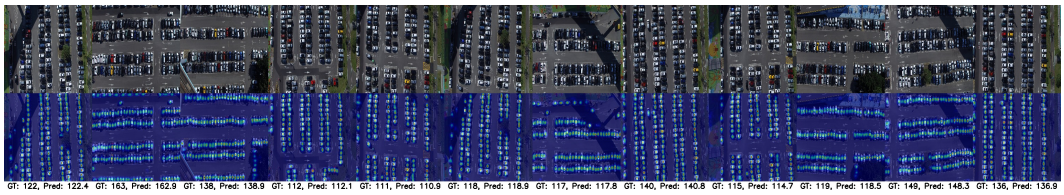


Figure 5. **Qualitative results** on unseen classes in FSC-147 (from Kiwis to Tree Logs). There are only 2 images of Prawn Crackers in FSC-147.



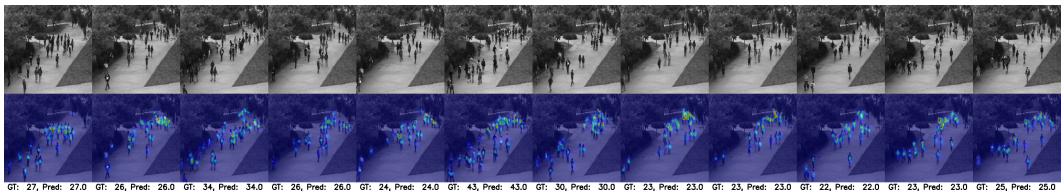
(a) Watches in FSC-147

(b) Support Images of Class-specific Datasets

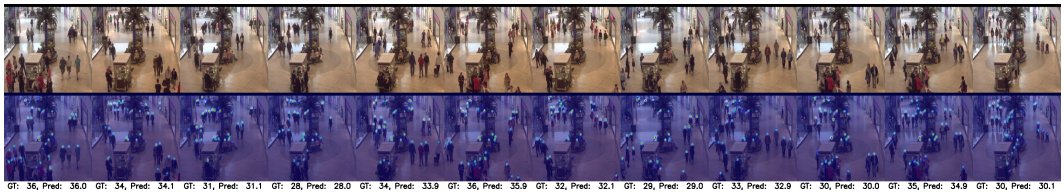


(c) CARPK

(d) PUCPR+



(e) UCSD



(f) Mall

(g) ShanghaiTech PartA

(h) ShanghaiTech PartB

Figure 6. (a) Qualitative results on unseen classes (Watches) in FSC-147. (b) Support images of class-specific datasets. (c-h) Qualitative results on class-specific datasets.

References

[1] Deepak Babu Sam, Shiv Surya, and R Venkatesh Babu. Switching convolutional neural network for crowd counting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5744–5752, 2017. 4

[2] Antoni B Chan, Zhang-Sheng John Liang, and Nuno Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1–7, 2008. 3, 4

[3] Ke Chen, Chen Change Loy, Shaogang Gong, and Tony Xiang. Feature mining for localised crowd counting. In *Brit. Mach. Vis. Conf.*, page 3, 2012. 3, 4

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 248–255, 2009. 1

[5] Eran Goldman, Roei Herzig, Aviv Eisenschtat, Jacob Goldberger, and Tal Hassner. Precise detection in densely packed scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5227–5236, 2019. 4

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 770–778, 2016. 1

[7] Meng-Ru Hsieh, Yen-Liang Lin, and Winston H Hsu. Drone-based object counting by spatially regularized regional proposal network. In *Int. Conf. Comput. Vis.*, pages 4145–4153, 2017. 3, 4

[8] Ersin Kilic and Serkan Ozturk. An accurate car counting in aerial images based on convolutional neural networks. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–10, 2021. 4

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[10] Issam H Laradji, Negar Rostamzadeh, Pedro O Pinheiro, David Vazquez, and Mark Schmidt. Where are the blobs: Counting by localization with point supervision. In *Eur. Conf. Comput. Vis.*, pages 547–562, 2018. 4

[11] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1091–1100, 2018. 4

[12] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Int. Conf. Comput. Vis.*, pages 2980–2988, 2017. 4

[13] Erika Lu, Weidi Xie, and Andrew Zisserman. Class-agnostic counting. In *Asian Conf. Comput. Vis.*, pages 669–684, 2018. 4

[14] T Nathan Mundhenk, Goran Konjevod, Wesam A Sakla, and Kofi Boakye. A large contextual dataset for classification, detection and counting of cars with deep learning. In *Eur. Conf. Comput. Vis.*, pages 785–800, 2016. 4

[15] Viresh Ranjan, Udbhav Sharma, Thu Nguyen, and Minh Hoai. Learning to count everything. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3394–3403, 2021. 2, 4

[16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 779–788, 2016. 4

[17] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Adv. Neural Inform. Process. Syst.*, pages 91–99, 2015. 1, 4

[18] Vishwanath A Sindagi and Vishal M Patel. Generating high-quality crowd density maps using contextual pyramid cnns. In *Int. Conf. Comput. Vis.*, pages 1861–1870, 2017. 4

[19] Tobias Stahl, Silvia L Pintea, and Jan C Van Gemert. Divide and count: Generic object counting by image divisions. *IEEE Trans. Image Process.*, pages 1035–1044, 2019. 4

[20] Jia Wan, Ziquan Liu, and Antoni B Chan. A generalized loss function for crowd counting and localization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1974–1983, 2021. 4

[21] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, pages 600–612, 2004. 3

[22] Yifan Yang, Guorong Li, Zhe Wu, Li Su, Qingming Huang, and Nicu Sebe. Reverse perspective network for perspective-aware object counting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4374–4383, 2020. 4

[23] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 833–841, 2015. 4

[24] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 589–597, 2016. 3, 4