Class-aware Object Counting

Andreas Michel
Fraunhofer IOSB
andreas.michel@iosb.fraunhofer.de

Wolfgang Gross
Fraunhofer IOSB
wolfgang.gross@iosb.fraunhofer.de

Fabian Schenkel
Fraunhofer IOSB
fabian.schenkel@iosb.fraunhofer.de

Wolfgang Middelmann
Fraunhofer IOSB
wolfgang.middelmann@iosb.fraunhofer.de

Abstract

Estimating the correct number of objects in a given natural scene is a common challenge in computer vision. Natural scenes usually contain multiple object categories and varying object densities. Detection-based algorithms are well suited for class-aware object counting and low object counts. However, they underperform with high or varying numbers of objects. To address this challenge, we propose an end-to-end approach to enhance an existing detection-based method with a multi-class density estimation branch. The results of both branches are fed into a successive count-estimation network, which estimates object counts for each category. Although these numbers do not contain any localization information, they can be used as a valuable indicator for verifying the exactness of the object detector results and improving its counting performance. In order to demonstrate the effectiveness, we evaluate our method on common object detection datasets.

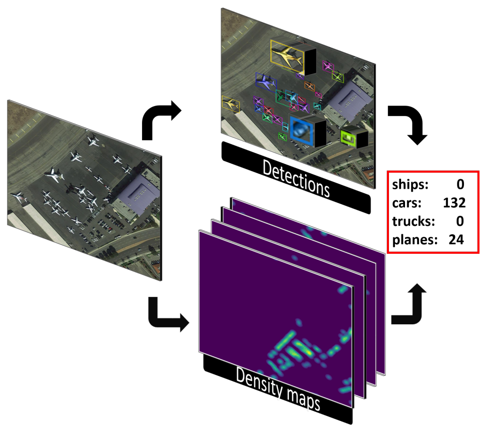

Figure 1: **Class-aware object counting.** The basic idea behind our approach is to detect relevant objects while simultaneously creating multi-class density maps. Results from both branches are fused to create a more precise object count for each class.

1. Introduction

Tasks, which are usually easy for a human, can be hard for a machine [21]. Counting objects and persons is such a task. In real-world scenes, the challenges are diverse, including but not limited to occlusions, the presence of multiple object categories, and varying object densities [10]. Figure 2 shows examples of natural scenes. In particular, for the coast guard, not only the number of ships is relevant but also the vessel type. Furthermore, counting has many valuable applications, such as traffic monitoring, surveillance, public safety, and urban planning [27]. Since manual object counting is a time-consuming task, it is not feasible on a large scale and with time-critical applications. In order to address this issue, crowd counting presents a solution as it can be generalized to object counting. Commonly, crowd counting approaches are based on detection, regression, or density estimation [6].

Object detection-based approaches are usually divided into one- and two-stage detectors. One-stage detectors [24, 17, 14, 1] only utilize a dense prediction head and are usually more time-efficient. In contrast, two-stage detectors [7, 22] are generally more accurate by utilizing an additional sparse prediction head. Recent object detection methods are usually class-aware and deliver localization information of the objects on top of their estimated number. Furthermore, object detection can be fused with segmentation techniques to extract even more information from a scene [8, 11]. Object detection methods excel in low density counts with low to moderate occlusion. However, object detection-based methods underperform in scenes with high and diverse object counts.
In contrast, direct regression methods [2, 26] are better

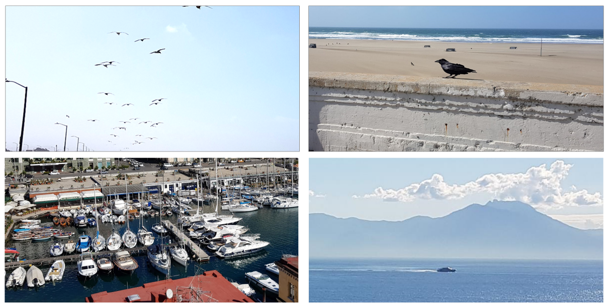

Figure 2: **Overview of real-world scenes.** We study the problem of class-aware counting in natural scenes. Counting in natural scenes is a real-world application with many challenges. Leading counting approaches are usually focused on single-class counting, but realistic scenarios do not only vary strongly in object density and scale but usually include multiple relevant categories.

suited for these conditions, yet any information about the spatial distribution is lost. Therefore, density estimation methods have become popular in recent years [33, 27, 12, 18, 19]. Instead of predicting a global count or localizing objects precisely, a density map of the relevant targets is estimated. The global count can be acquired by summation over the density map. Density estimation methods favor overestimating the number of sparse scenes and, consequently, underachieve in low object counts. Furthermore, nearly all recent density estimation methods are unfit for a current class-aware approach. In a real-world application, this is a big disadvantage in comparison to a detection-based method. Only a few methods drive research towards multi-class density maps [4, 31]. Object counting in the context of a real-world application often requires predicting the number of objects in different categories, *e.g.*, in traffic monitoring. In this scenario, it can be important to count all road users and simultaneously differentiate between cars and trucks. One of the main obstacles in this scenario is the strongly varying object density. Urban regions will usually contain more vehicles compared to rural areas. Detection-based methods will likely be contested on urban scenes, and density estimation-based methods will underperform on rural scenes with a sparse object density. Furthermore, diverse object densities are a common condition in air- and space-borne images, *e.g.*, iSaid Dataset [29]. This can lead to a decrease in the reliability of the counting accuracy. One way to improve the counting accuracy is to combine different counting models. Liu et al. [16] proposed a fusion of detection-based and regression-based methods for crowd counting. Although DecideNet [16] achieves great performance, it is only developed for the single-class problem.

Class-aware object counting in varying object densities is a real-world problem, but common object detection datasets like COCO [15] have a strong bias towards one-digit object counts per image [3]. In contrast, common crowd counting datasets like ShanghaiTech [33] usually have diverse object densities yet include only one class. Yet single class solutions are for many real-world applications not sufficient. As a result, research in the field of class-aware object counting is still needed. Therefore, inspired by DecideNet, we propose a fusion of a modern object detector with a multi-class density estimation network. We integrate our multi-class density estimation branch in the common object detection backbone, the feature pyramid network FPN [13]. Furthermore, we utilize detection and density estimation results to predict accurate object counts for different categories. We see for our approach two use cases: The first use case is class-aware object counting. The second use case is to utilize it as a support structure for object detection. From the deviation between the predicted object count and the detection count, the presence of false positive or false negative detections can be inferred.

Contributions: (1) We propose a novel way to combine class-aware density estimation with object detection. Our approach is well-suited to be integrated into the most common object detection and instance segmentation models and is end-to-end trainable. (2) Our approach includes a count estimation network optimized to predict object counts with different categories in scenes with diverse object densities. The predicted object counts can be used as an indicator to verify the detection results and, therefore, increase its reliability. (3) To display our approach's effectiveness, we evaluate our method on the common object detection datasets.

2. Related Work

R-CNN. Faster R-CNN [7] is one of the most popular two-stage object detectors. Its major components consist of a backbone, a class-agnostic region proposal network (RPN), and a sparse prediction head. The backbone, *e.g.*, a ResNet [9] extracts features from the input data. A pure ResNet has only a bottom-up pathway for semantic features. It allows features only to flow from low- to high-level semantics. By expanding the feature extractor with an FPN [13], semantics can flow in both ways. This is a direct consequence of implementing a top-down pathway and lateral connections in addition to the previous bottom-up pathway. This extended feature extractor increases the performance of the subsequent detection steps, especially its scale awareness. After extracting the features, the feature maps are passed to the RPN. The RPN proposes class-agnostic region proposals, which are filtered by a non-maximum suppression module. Finally, the detection head classifies the filtered region proposals according to their class category and improves their bounding box coordinates. In addition, faster R-CNN can be easily upgraded to Mask R-CNN [8]

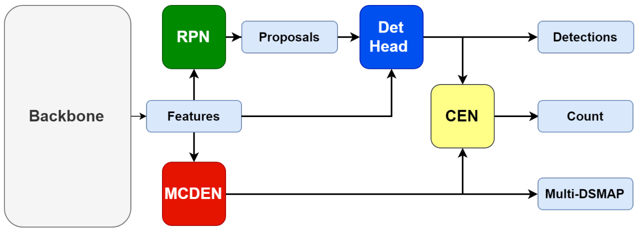

Figure 3: **Network architecture.** A general overview of our proposed method. Features extracted from the backbone are processed in the RPN and in the MCDEN. Subsequently, detection results and density maps are sent to the CEN to refine counting results.

by using a mask head.

Context-aware crowd counting. Liu et al. [18] propose a context-aware crowd counting network (CAN) that combines features extracted from multiple receptive field sizes in order to estimate precise density maps. This is done by utilizing VGG [28] to create feature maps that are average pooled with different sizes to generate scale-aware features. In order to determine the local scale of each image region, the scale-aware features are used to generate contrast features. Eventually, former features are processed into contextual features and are decoded into density maps.

Crowd FPN. In [20], an FPN is extended by a density estimation network. In addition to feature maps, it provides class-agnostic density maps for a given image. Similar to CAN, scale-aware features are generated. However, instead of using average pooling for creating feature maps with multiple receptive field sizes, the different spatial features from the FPN are utilized. Crowd FPN can be integrated into an R-CNN algorithm such as Faster R-CNN or Mask-R-CNN and in combination with an alternating region proposal scheme, increase the average recall of the detection results.
In contrast to Crowd FPN, we expand the class-agnostic approach by a class-aware density estimation network, removing any influence of the density branch on the region proposal process and add a different fusion approach between both branches.

3. Approach

Our proposed approach consists of the multi-class density estimation Network (MCDEN) and the detection pipeline, which are combined to form the input of the count estimation network (CEN).
As described above, natural scenes usually include multiple objects, which vary strongly in object density and consist of multiple categories. Therefore, we propose a combined approach between object detection and class-aware density

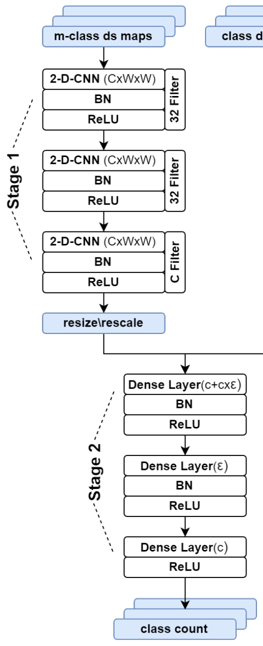

Figure 4: **Count estimation network.** Model architecture of the CEN.

maps to improve the counting performance. Figure 3 shows an overview of the general architecture. First, we utilize an FPN as our backbone to extract feature maps in indifferent receptive sizes and semantic levels. The flow of the feature maps is split into the detection and the density estimation branch. The detection pipeline consists of an RPN, a non-maximum filter, a boxhead, and an optional maskhead. Meanwhile, the density estimation branch consists of the MCDEN, which outputs class-aware density maps. The results of both methods serve as input for the CEN. The CEN aims to predict precise counts for each object category for a given scene.

3.1. Multi-class density estimation network

We consider a set of N training images $\{I_i\}_{1\leq i\leq N}$ with related ground-truth density maps $\{D^{\text{gt}}_{i,j}\}_{1\leq j\leq c}$, for c different object categories.

We aim to learn a non-linear mapping $\mathcal{F}_j$ for each object category parametrized by θ to estimate c different density maps approximating $D^{\text{gt}}_{i,j}$ by minimizing the the L_2 norm of the groundtruth and the predictions.

$$D^{\text{est}}_{i,j}(I_i) = \mathcal{F}_j(I_i, \theta), \tag{1}$$

where θ are the learned parameters. Without loss of generality, we describe the proposed method for a single image I to facilitate the notation.
To start with, the FPN outputs four relevant feature maps f_s, one for each of the scaling factors $s \in \{\frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}\}$.

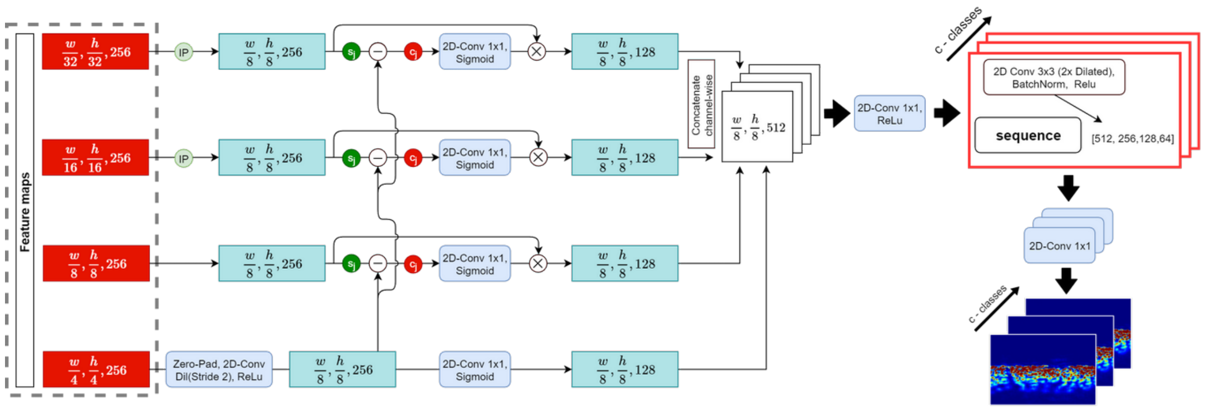
Feature maps
$\frac{w}{32},\frac{h}{32},256$
$\frac{w}{16},\frac{h}{16},256$
$\frac{w}{8},\frac{h}{8},256$
$\frac{w}{4},\frac{h}{4},256$
IP
Zero-Pad, 2D-Conv Dil(Stride 2), ReLu
2D-Conv 1x1, Sigmoid
$\frac{w}{8},\frac{h}{8},128$
Concatenate channel-wise
$\frac{w}{8},\frac{h}{8},512$
2D-Conv 1x1, ReLu
c - classes
2D Conv 3x3 (2x Dilated), BatchNorm, Relu
sequence
[512, 256,128,64]
2D-Conv 1x1

Figure 5: **Multi-class density estimation network.** We aim to predict multiple density maps for each object category. Feature maps, which are generated in an FPN [13], are processed in an encoder network followed by multiple decoder networks. The encoder network exploits contrast features between the different feature maps to increase the scale-awareness of the encoder network.

The first feature map f_{s_1} with the lowest semantic level and the highest resolution is used as the base feature. The downsampling process to our base resolution of $\frac{1}{8}$-th of a given original image I is learned by applying a dilated convolutional layer with a stride of 2. This process leads to a slightly better density estimation performance. The remaining feature maps are upsampled to the base resolution by bilinear interpolation, which is denoted by $\mathcal{I}_{bi}$. Due to the various perspective field sizes of the different feature maps, scale aware features are created according to

$$f_s = \begin{cases} \mathcal{I}_{bi}(\mathcal{F}_{\text{FPN},s_4}(I)) \\ \mathcal{I}_{bi}(\mathcal{F}_{\text{FPN},s_3}(I)) \\ \mathcal{F}_{\text{FPN},s_2}(I) \\ \mathcal{F}_d(\mathcal{F}_{\text{FPN},s_1}(I)) \end{cases}, \tag{2}$$

where $\mathcal{F}_{\text{FPN},s_k}$ for $k = 1, \ldots, 4$ are the feature maps produced by the FPN and $\mathcal{F}_d$ is the learned downsampling operator. In order to increase scale awareness, contrast features δ are calculated

$$\delta_l = s_l - s_1 \quad l \in \{2, 3, 4\}. \tag{3}$$

Subsequently, the individual contrast features are calculated by

$$w_l = \mathcal{F}_{sa}^l(\delta_l, \theta_{sa}^l), \tag{4}$$

where $\mathcal{F}_{sa}^l$ is the output of a 1×1 convolution layer followed by a sigmoid activation function and a scale-aware operator. Finally, we multiply the former learned features elementwise with the scale aware features and concatenate them channel-wise with the base feature map

$$f_I = \left[f_{s_1} | \sum_{l=2}^{4} \omega_l \odot s_l\right]. \tag{5}$$

After encoding the relevant information, in particular scale-information of the original image I, into the feature block we begin with the decoding process. Instead of using only one decoder as in [20], we utilize a specific decoder network $\mathcal{F}_{\text{dc}}$ for every category c

$$D_c^{\text{est}} = \begin{cases} \mathcal{F}_{\text{dc},1}(f_I) \\ \vdots \\ \mathcal{F}_{\text{dc},j}(f_I) \\ \vdots \\ \mathcal{F}_{\text{dc},c}(f_I) \end{cases}. \tag{6}$$

The decoder consists of multiple sequences of dilated convolutional layers followed by batch normalizations and ReLU activation functions. In contrast to Crowd FPN we reduce the number of weights significantly in the decoder block, as each additional category increases the number of required parameters to be learned.

3.2. Detection pipeline

The detection pipeline is derived from Mask R-CNN. The essential parts of the detection pipeline are the RPN, and the boxhead. Furthermore, we add a maskhead, which is an optional step for our approach. The first step is the processing of the feature maps from the backbone in the RPN. The RPN proposes regions in which objects can be found. Then, the NMS filters the output region proposals. Afterwards, the filtered region proposals are sent to the boxhead, where the classification of the objects in the region are performed. The boxhead also refines the spatial properties of the region proposals. In an optional step, a maskhead estimates the segmentation masks of the potential objects.

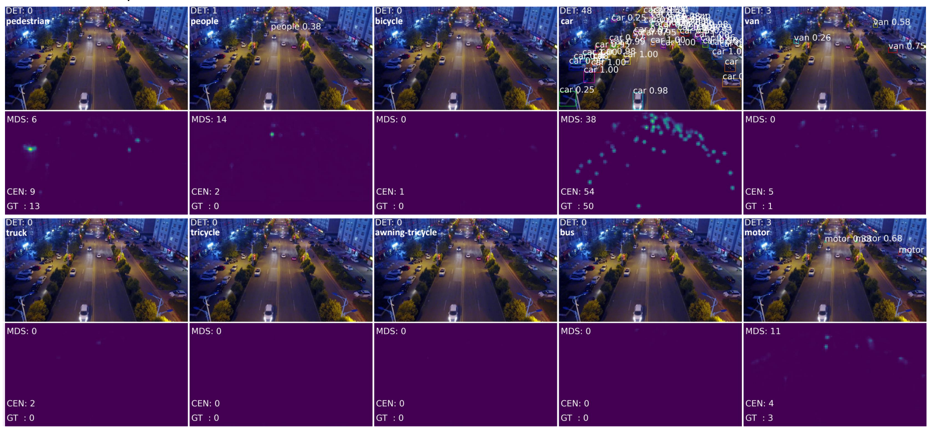
Figure 6: **Sample of the vidsrone dataset(test-dev) [34].** For each target category, a pair of density map and detection overlays are shown. It is easily possible to distinguish between the given density maps and their corresponding density level. The top-left image pair represents the category 'pedestrians' and its right neighbor represents the category 'people'. While the detection (DET) and density branch (MDS) underachieve in differentiating between both types, the count estimation network (CEN) performed well in this case.

3.3. Count estimation network

Figure 4 shows an overview of the CEN. The CEN for a single object category j can be formulated as follows

$$\text{CEN}_j = \mathcal{F}_{\text{stage2}}(B_j^{\text{est}} + \lambda_{res} \cdot \mathcal{I}_{bi}(\mathcal{F}_{\text{stage1}}(D_j^{\text{est}}))). \quad (7)$$

Here, $\mathcal{F}_{\text{stage1}}$ and $\mathcal{F}_{\text{stage2}}$ are the operators describing the corresponding parts from Figure 4. $\mathcal{F}_{\text{stage1}}$ is a sequence of convolutional layers, $\mathcal{F}_{\text{stage2}}$ consists of multiple fully connected layers, and $\mathcal{I}_{bi}$ is a bilinear interpolation function with a fixed output of the size $c \times \epsilon$. The constant ϵ is a hyperparameter, which has to be tuned while training and c is the number of object categories. $\mathcal{B}_{est}$ and $\mathcal{D}_{est}$ are the results from the detection and the density estimation pipeline. $\lambda_{res} \in [0, 1]$ is the rescaling factor, which can be calculated by

$$\lambda_{res,j} = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y}(\mathcal{F}_{\text{stage1}}(D_j^{\text{est}}(x,y))}{\sum_{x=1}^{X}\sum_{y=1}^{Y}\mathcal{I}_{bi}(\mathcal{F}_{\text{stage1}}(D_j^{\text{est}}(x,y)))}, \quad (8)$$

where x, y are the pixel coordinates of the given image with dimension $X \times Y$. The multi-class density maps D_j^{est} and the results B_j^{est} of the detection pipeline are fed forward to the CEN. In the first step of the CEN, the multi-class density maps are processed in a 2D-convolutional layer with a filter size of 32. The convolutional layer is followed by a batch normalization and a ReLu activation function. After a repetition of this sequence of operations, the resulting features are further processed by a 2D-convolutional layer with filter size c. After applying another set of batch normalizations and ReLU activation functions, the results are interpolated to a matrix with the dimensions $c \times \epsilon$. Due to different sizes of the input density maps, resizing the feature map can lead to wrong assumptions about the number of objects. Density maps are trained to approximate a normalized Gaussian distribution in which the sum of all pixels corresponds to the number of objects in a given scene. Applying a simple resize operation to a fixed value can skew the distribution. Therefore, a resizing function has to be followed by a rescaling factor λ_{res}.

Furthermore, for each class j, the quantity of the detected objects from the detection pipeline is calculated and saved in the vector $v \in \mathbb{N}^c$. This vector and the previously predicted number of detections B_j^{est} are concatenated and fed into a sequence of two dense layers with a batch normalization layer, which are activated by ReLu functions. Finally, an additional dense layer outputs the estimated count result for each class.

3.4. Loss functions

We employ a multi-loss function consisting of the detection loss L_{det}, the density loss L_{ds}, and the CEN loss L_{cen}. The overall loss function can be formulated as follows

$$L = L_{\text{det}} + \tfrac{1}{c}\sum\nolimits_{j=1}^{c} L_{\text{ds,j}} + L_{\text{cen}}, \quad (9)$$

where L_{det} is implemented according to [8]. Furthermore, $L_{\text{ds,j}}$ is aggregated over all c object categories and calculated by the L^2-norm with a batch size $N' < N$ of the training images $I' \subset I$

$$L_{\text{ds}} = \frac{1}{2N'}\sum\nolimits_{i=1}^{N'}\left\|D_i^{\text{gt}} - D_i^{\text{est}}\right\|^2 \quad (10)$$

Table 1: Counting results on the Visdrone dataset

Visdrone		[0-1000]	[0-10]	[11-50]	[51-100]	[101-1000]
Faster R-CNN (*standalone*)	**MAE**	4.93	2.6	11.31	17.87	29.21
	RMSE	13.45	3.98	13.56	33.34	64.88
Faster R-CNN (*det-branch*)	**MAE**	4.71	2.4	11.25	17.86	29.43
	RMSE	13.17	3.91	13.82	32.39	64.18
DetectorRS (*standalone*)	**MAE**	4.77	2.35	14.71	18.35	22.86
	RMSE	9.58	3.13	16.85	34.13	51.59
CenterNet (*standalone*))	**MAE**	5.79	3.07	15.69	18.35	22.86
	RMSE	10.92	4.19	18.23	36.65	54.25
OURS-MCDEN	**MAE**	6.53	4.29	14.39	14.81	26.92
	RMSE	11.5	5.74	16	25.78	53.88
OURS-CEN	**MAE**	3.76	2.07	10.49	13.84	23.55
	RMSE	9.56	3.25	12.52	25.07	51.11

. Finally, L_{cen} can be defined as

$$L_{\text{cen}} = \frac{\lambda_{\text{cen}}}{2N'} \sum_{i=1}^{N'} \frac{\left\| D_i^{\text{gt}} - D_i^{\text{est}} \right\|^2}{D_i^{\text{gt}}} \tag{11}$$

. The L_{cen} also utilizes the L^2-norm. In order to reduce the effect of the L_{cen} on the weight update process, we apply an additional constant $\lambda_{\text{cen}} \in [0, 1]$ to L_{cen}. Additionally, we divide the loss term by the number of objects per category to create a relative loss function for balancing the loss between high- and low-density object counts.

4. Experiments and Discussion

In this section, we evaluate the effectiveness of our proposed approach. First, we present details of our implementation. Then, we introduce the evaluation metrics and the benchmark datasets. After that, we discuss the counting results and compare them to state-of-the-art object detection methods. Finally, we conduct an ablation study.

4.1. Implementation details

We use a ResNet50 [9] pretrained on ImageNet [25] as our backbone. For the detection pipeline of our network, we use Faster R-CNN for object detection and Mask R-CNN as a baseline for instance segmentation. We apply the default anchor configuration following [8], but we increase the number of proposals to keep before and after applying the NMS filter to 2000 proposals. Furthermore, we modify the number of possible detections per image to 1000. With this, we want to ensure that our detection pipeline is not limited by hyperparameters in scenes with high object densities. During training, we usually reduce the loss of the density estimation branch by a factor of ten and steadily increase it per epoch. This results in an overall faster training time. In our count estimation network, we set the constant loss reduction factor λ_{cen} to 0.2 and set the interpolation constant to $\epsilon = 128$. During training, we start with a learning rate of $\alpha = 3 \cdot 10^{-3}$ and uniformly reduce α to 10^{-3} steadily over 50 epochs. We choose a batch size of four and utilize the AdamW optimizer [32] with a weight decay of 10^{-4}. For training, ground truth density maps are created by a geometry adaptive Gaussian kernel [33] from the bounding box coordinates.

Table 2: Counting results on the iSaid dataset.

iSaid		Mean	Ship	Truck	Car	Plane
Mask R-CNN (*standalone*)	**MAE**	8.52	3.84	3.82	24.32	1.20
	RMSE	50.68	11.11	9.94	99.15	3.73
Mask R-CNN (*det-branch*)	**MAE**	8.32	3.73	3.87	24.49	1.21
	RMSE	50.58	10.25	9.43	100.13	3.53
OURS-MCDEN	**MAE**	10.97	5.16	3.94	30.95	3.83
	RMSE	37.3	9.48	9.97	73.03	6.65
OURS-CEN	**MAE**	7.85	5.65	4.31	20.02	1.41
	RMSE	31.64	15.01	9.73	60.59	3.7

4.2. Evaluation metrics

Similar to previous works in object counting [4] and crowd counting [18, 33], the mean absolute error (MAE) and the root mean squared error (RMSE) are used as evaluation metrics. MAE_c per category for all N test images in the dataset can be defined as follows:

$$\text{MAE}_j = \tfrac{1}{N} \sum_{i=1}^{N} |Q_{i,j}^{est} - Q_{i,j}^{gt}|, \tag{12}$$

and RMSE_c can be calculated by:

$$\text{RMSE}_j = \sqrt{\tfrac{1}{N} \sum_{i=1}^{N} (Q_{i,j}^{est} - Q_{i,j}^{gt})^2}, \tag{13}$$

where N is the number of test images, $Q_{i,j}^{est}$ is the estimated number of relevant targets in the i-th image for an object category j and $Q_{i,j}^{gt}$ the coresponding ground truth. Furthermore, we define the mMAE as follows:

$$\text{mMAE} = \frac{1}{c} \sum_{j=1}^{c} \text{MAE}_j \tag{14}$$

and the mRMSE as:

$$\text{mRMSE} = \frac{1}{c} \sum_{j=1}^{c} \text{RMSE}_j \tag{15}$$

Q^{est} is calculated for each branch individually. In the detection pipeline, $Q^{est,det}$ is calculated by counting the number of detections. Meanwhile, $Q^{est,ds}$ is derived by integrating over the pixels of the estimated density map in the density branch. Finally, the CEN outputs $Q^{est,cen} = CEN^{est}$ directly.

4.3. Datasets

To our knowledge, there exists no publicly available dataset for multi-class object counting with high and varying object counts. Therefore, we utilized the following common object detection datasets.

Visdrone-Det [34]. This dataset consists of 10,209 images

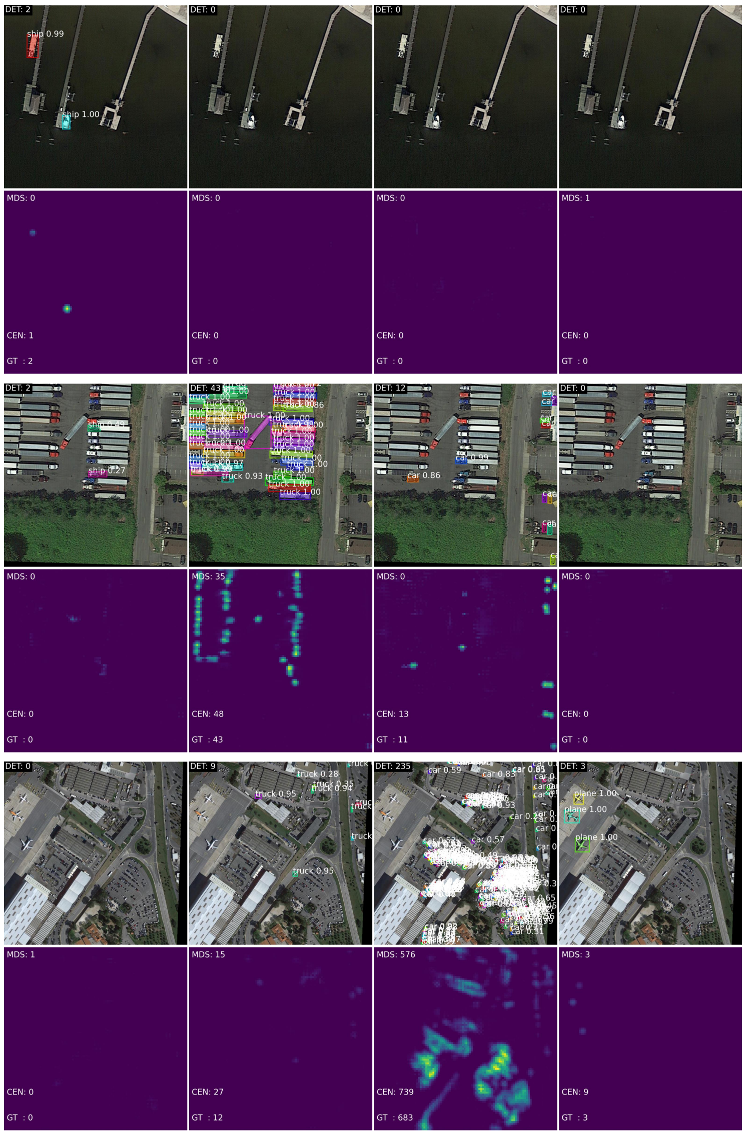

Figure 7: **Visualization results on the iSaid dataset [29]**. From left to right column: ships, trucks, cars, and planes. DET are the numbers of detections, MDS is the sum over the density map and CEN displays the number of the count estimation network, which combines the former results.

with ten categories. Object categories include classes such as 'person', 'car', and 'bicycle' from different viewing angles. In our experiment, we do not reduce the number of classes. We train on the training dataset, which contains only 6,471 images. The validation dataset contains 548 images and we evaluate the training process on the test-dev dataset, consisting of 1,610 images.

iSaid [29]. This dataset is derived from the DOTA [30] dataset and contains object shape information. It includes 2,806 high-resolution images collected from multiple sensors and platforms. It consists of 655,451 object instances of 15 categories. The images are cropped into patches with a resolution of 800×800 for training and evaluation purposes. This paper only uses a subset of the dataset and utilizes object instances of the movable object categories: 'ship', 'car', 'truck', and 'plane'.

4.4. Counting Results

Figure 6 presents a sample of the Visdrone test-dev dataset. The complete results are shown in Table 1, which displays the MAE and RMSE for all categories. In order to verify the effectiveness of our approach, we compare

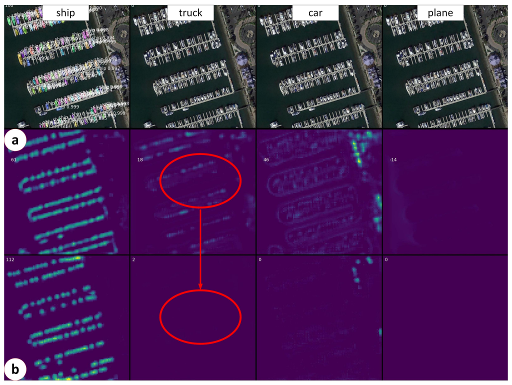

Figure 8: **Comparison between multi-decoder and single-decoder approach.** This Figure illustrates the density maps created from a single-decoder and a multi-decoder approach. Part a shows the single-decoder method with crosstalk between the density maps. In contrast, the density maps from the multi-decoder approach of Part b are almost crosstalk-free.

our method to the state-of-the-art object detectors DetectorRS [23], and CenterNet [5]. Both methods are trained for 50 epochs with hyperparameters similar to the standard settings described in [23] and [5]. Even though they usually predict more precise bounding boxes, their counting results are similar or worse in comparison to Faster R-CNN. The detection branch outperforms the density branch in brackets with low object counts and vice versa for high object densities. This follows the expected performance discussed in the state-of-the-art chapter. Both branches are outperformed in all brackets by our proposed CEN. Furthermore, the Faster R-CNN of our detection pipeline has a similar counting performance as the detection pipeline. Therefore, the multi-branch approach does not influence its detection part negatively in counting performance. Multiple images of the iSaid dataset are visualized in Figure 7 along with the corresponding CEN output. Similar to the Visdrone dataset, the density branch predicts fewer outliers than the detection branch, but it performed worse in the MAE for all categories (Table 2).

4.5. Ablation Study

Finally, we perform an ablation study to confirm our architecture decisions.

Multi-class decoder. Our goal for the MCDEN is to estimate a density map for each category. The first approaches were challenged by crosstalk between each unique density map. In this case, the high-density level in one category influenced the density estimation of other categories. Figure 8 shows the qualitative results of our progress. At first, only a joint decoder for all categories is used. The number of trainable parameters is roughly the same as in the multi-decoder structure. Note that objects from the category ship are the most numerous category in this example. This leads to strong crosstalk between the density map, and the traces of the ship density map can be seen in the other ones Figure 8a). Therefore, theMCDEN systematically overestimated the predicted numbers of the other categories. By adding an additional decoder part for each category and simultaneously reducing the number of trainable parameters in each backbone, the crosstalk decreased dramatically. Figure 8b displays the results after applying these improvements. Simultaneously to the reductions in crosstalk the number of predicted objects in the scene became more precise.

Table 3: Results of multiple CEN variations on the iSaid.

iSaid		[0-10000]	[0-10]	[11-50]	[51-100]	[101-10000]
CEN *(simple resize)*	**MAE**	9.82	3.02	9.8	25.29	66.9
	RMSE	35.29	11.34	22.38	34.71	112.02
CEN *(simple loss)*	**MAE**	8.4	2.56	8.49	18.41	57.28
	RMSE	34.73	10.46	20.133	28.19	109.31
CEN *(base)*	**MAE**	7.85	2.5	6.51	17.86	50.31
	RMSE	31.64	10.26	12.18	26.38	95.57

CEN variants. The proposed CEN stands out for its efficient and versatile design. Table 3 shows the counting results of the CEN on the iSaid dataset over multiple intervals. The first row of Table 3 shows the CEN without the rescaling factor λ_{res}. A performance loss can be seen in comparison to the base implementation of the CEN. This effect becomes more substantial with the increasing variety of the input image resolution. The second row shows the counting results without dividing the L_2-loss by the number of objects. We expected a performance increase in the most common interval and a decrease in the performance of the remaining brackets. But surprisingly, by applying a relative loss term instead of the original L_2-term, the counting performance improved in all intervals.

5. Conclusion and Future Perspectives

In this paper, we propose a combined approach between density estimation and object detection for multi-aware object counting. We have shown that the density branch outperforms the detection branch in high object densities and vice versa in low object counts. Furthermore, we show how to fuse both branches in our proposed count estimation network to improve the prediction accuracy compared to utilizing only a single branch. In order to demonstrate the effectiveness of our approach, we evaluate our method on two common object datasets and compare them with R-CNN algorithms. As discussed above, a proper benchmark for multi-class object counting in strongly varying densities is mandatory to progress research in this field.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[2] Antoni B Chan and Nuno Vasconcelos. Bayesian poisson regression for crowd counting. In *2009 IEEE 12th international conference on computer vision*, pages 545–551. IEEE, 2009.

[3] Prithvijit Chattopadhyay, Ramakrishna Vedantam, Ramprasaath R Selvaraju, Dhruv Batra, and Devi Parikh. Counting everyday objects in everyday scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1135–1144, 2017.

[4] Hisham Cholakkal, Guolei Sun, Fahad Shahbaz Khan, and Ling Shao. Object counting and instance segmentation with image-level supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12397–12405, 2019.

[5] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6569–6578, 2019.

[6] Guangshuai Gao, Junyu Gao, Qingjie Liu, Qi Wang, and Yunhong Wang. Cnn-based density estimation and crowd counting: A survey. *arXiv preprint arXiv:2003.12783*, 2020.

[7] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Mohammad Hossain, Mehrdad Hosseinzadeh, Omit Chanda, and Yang Wang. Crowd counting using scale-aware attention networks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1280–1288. IEEE, 2019.

[11] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9404–9413, 2019.

[12] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1091–1100, 2018.

[13] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[15] 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 *European conference on computer vision*, pages 740–755. Springer, 2014.

[16] Jiang Liu, Chenqiang Gao, Deyu Meng, and Alexander G Hauptmann. Decidenet: Counting varying density crowds through attention guided detection and density estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2018.

[17] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[18] Weizhe Liu, Mathieu Salzmann, and Pascal Fua. Context-aware crowd counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5099–5108, 2019.

[19] Ao Luo, Fan Yang, Xin Li, Dong Nie, Zhicheng Jiao, Shangchen Zhou, and Hong Cheng. Hybrid graph neural networks for crowd counting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11693–11700, 2020.

[20] Andreas Michel, Jonas Mispelhorn, Fabian Schenkel, Wolfgang Gross, and Wolfgang Middelmann. An approach to improve detection in scenes with varying object densities in remote sensing. In Lorenzo Bruzzone, Francesca Bovolo, and Emanuele Santi, editors, *Image and Signal Processing for Remote Sensing XXVI*, volume 11533, pages 107 – 113. International Society for Optics and Photonics, SPIE, 2020.

[21] Marvin Minsky. *Society of mind*. Simon and Schuster, 1988.

[22] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. *arXiv preprint arXiv:2006.02334*, 2020.

[23] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10213–10224, 2021.

[24] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[26] Usman Sajid, Hasan Sajid, Hongcheng Wang, and Guanghui Wang. Zoomcount: A zooming mechanism for crowd counting in static images. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(10):3499–3512, 2020.

[27] Deepak Babu Sam, Shiv Surya, and R Venkatesh Babu. Switching convolutional neural network for crowd counting.

In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4031–4039. IEEE, 2017.

[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[29] Syed Waqas Zamir, Aditya Arora, Akshita Gupta, Salman Khan, Guolei Sun, Fahad Shahbaz Khan, Fan Zhu, Ling Shao, Gui-Song Xia, and Xiang Bai. isaid: A large-scale dataset for instance segmentation in aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 28–37, 2019.

[30] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3974–3983, 2018.

[31] Wei Xu, Dingkang Liang, Yixiao Zheng, and Zhanyu Ma. Dilated-scale-aware attention convnet for multi-class object counting. *arXiv preprint arXiv:2012.08149*, 2020.

[32] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[33] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 589–597, 2016.

[34] Pengfei Zhu, Dawei Du, Longyin Wen, Xiao Bian, Haibin Ling, Qinghua Hu, Tao Peng, Jiayu Zheng, Xinyao Wang, Yue Zhang, et al. Visdrone-vid2019: The vision meets drone object detection in video challenge results. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.