Density-Based Vehicle Counting with Unsupervised Scale Selection

Petr Dobeš, Jakub Špaňhel, Vojtěch Bartl, Roman Juránek, Adam Herout
Graph@FIT, Brno University of Technology
Brno, Czech Republic
{idobes, ispanhel, ibartl, ijuranek, herout}@fit.vutbr.cz

***Abstract*—A significant hurdle within any counting task is the variance in a scale of the objects to be counted. While size changes of some extent can be induced by perspective distortion, more severe scale differences can easily occur, *e.g.* in case of images taken by a drone from different elevations above the ground. The aim of our work is to overcome this issue by leveraging only lightweight dot annotations and a minimum level of training supervision. We propose a modification to the Stacked Hourglass network which enables the model to process multiple input scales and to automatically select the most suitable candidate using a quality score. We alter the training procedure to enable learning of the quality scores while avoiding their direct supervision, and thus without requiring any additional annotation effort. We evaluate our method on three standard datasets: PUCPR+, TRANCOS and CARPK. The obtained results are on par with current state-of-the-art methods while being more robust towards significant variations in input scale.**

***Index Terms*—Object counting, Traffic surveillance, Vehicle counting, Density estimation, Deep learning**

I. Introduction

Counting the number of objects in an image can serve a wide range of practical applications. Specifically, counting vehicle instances is essential for various tasks, including estimation of parking lot occupancy, detection and prevention of traffic congestion, as well as traffic flow monitoring and control. Similarly to other computer vision problems, object counting is currently dominated by deep learning methods, represented by convolutional neural networks. Nevertheless, training of such methods often requires costly annotations. This work therefore aims at using only the minimum necessary level of annotation effort together with partly unsupervised learning procedure in order to design a competitive vehicle counting approach that would be capable of tackling severe changes in the scale of observed objects across different input images.

The area of object count estimation can be regarded as a subset of the detection task [1], [2], *i.e.* a detector or an instance segmentation network can first be applied to extract individual objects from the image, and the count is then obtained directly as the total number of detected instances. However, such an approach induces an unnecessary degree of data annotation complexity when only the object count is desired. In these scenarios, object counting is often approached in a density-based manner. A regression network is trained to predict probability density over the input image, which is subsequently used to infer the final object count [3]–[5].

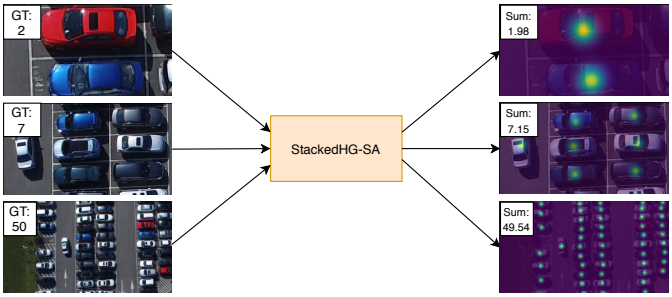

Fig. 1. By modifying Stacked Hourglass network and the corresponding training scheme, we propose a scale-aware model that is able to automatically assess the quality of the generated density maps, and thus to cope with severe changes of input object sizes.

Nevertheless, one of the major hurdles is the varying size of the objects to be counted. In this regard, two main sources of scale change can be identified. Changes in size can be induced either by perspective [6]–[8] or by the overall change of the scale, caused by *e.g.* a difference in drone's elevation above the ground in case of UAV imagery [9], as shown in Fig. 1. While considering the problem of vehicle counting, perspective distortion can create scale variance to some extent; however, shifts of the latter kind can easily result in much more radical scale differences.

In this paper, we follow the density-based methodology. This enables us to utilize simple and cheap-to-obtain training annotations in the form of dots that mark the occurrence of individual objects. This annotation process also follows the natural way of how human annotators label counting data (*i.e.* by successively pointing and clicking at each of the observed objects). The points are then transformed into a probability distribution that is to be learned by a neural network. We further develop on this method by enhancing the training process to enable unsupervised selection of correct input image scale.

We show that a strong-enough baseline model can readily tackle smaller changes in object sizes resulting from perspective distortion. In order to cope with the possibility of more

This work was supported by TACR project "SMARTCarPark", TH03010529. This work was also supported by The Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II); project IT4Innovations excellence in science – LQ1602.

significant changes in the input scale, we extend the training mechanism to enable automatic selection of the appropriate scale for processing.

Without bells and whistles to address further granularity of objects that is caused by perspective distortion, our method achieves results similar to the state-of-the-art approaches that deal with the perspective, and therefore often require providing additional ground truth information during the training process. As a result, our method is able to cope with more extensive changes in the scale of the entire input image, without the necessity of any additional annotation effort.

II. Related Work

The methods utilized for the general task of object count estimation can be categorized into two main groups. The first group tackles counting objects as a detection problem, where individual object instances of the desired class are first identified (using a detector, such as Faster-RCNN [1] or Mask-RCNN [2]), and the overall count is then extracted by simple summation of all detected object occurrences. These methods are therefore commonly referred to as *counting by detection*. Nevertheless, such methods typically require large amounts of costly and precise annotations, and can still lead to worse resultant counting performance [10]. Methods within the second significant branch solve the object counting task through *density estimation*. In density-based approaches, a probabilistic density distribution is first predicted across the entire input image. The final object count is then obtained by integrating over the predicted distribution.

The primary aim of our work is to leverage lightweight annotations and a minimum level of training supervision for the task of vehicle counting, and we thus approach the problem using density estimation. Since our work belongs to the category of density-based methods, in this section, we only review the relevant publications from this group.

The concept of density estimation was first introduced in the pioneering work of Lempitsky and Zisserman [3]. Later learning-based methods, such as HydraCNN [4], then adopted this approach, performing the calculation of the final map using a sliding-window.

While object count can also be regressed directly by providing the final count as the only training supervision, Aich and Stavness [11] show that regulating the intermediate output of the network with density map improves the counting results significantly. The same authors further improve their network [12] by adding global sum pooling operation to allow processing of input images with arbitrary resolution.

Among the notable fully-convolutional models also belongs the Multi-column Network (MCNN) [7] which uses several networks (termed columns) with different kernel sizes to account for variance in the scale of the input objects. In order to further increase the receptive field, Deb and Ventura [13] propose to use dilated filters inside the multi-column architecture and add an aggregation module to consolidate column features.

Li *et al.* [14] argue that multiple columns of the MCNN network all learn nearly identical features, and instead propose to use a deeper architecture based on a pre-trained VGG16 model, followed by a set of dilated convolutional layers to prevent a loss of detail in the final density map. Similarly, Chen *et al.* [15] replace the columns of the MCNN with four parallel dilated convolutions with different rates.

Liu *et al.* [16] divide their counting architecture into two separate networks. The first network generates an attention map which identifies the regions of the image that contain the desired objects. A pixel-wise product of the attention map with the input image is then passed to the second network that forms the final density map prediction. In this way, the network only focuses on relevant image areas, and other potentially noisy and distracting regions are suppressed.

Note that this approach is rather orthogonal to our work, and we intentionally avoid adding such enhancements to our model for the sake of simplicity. Nevertheless, similar attention mechanism could potentially also be incorporated to improve our results further.

Apart from predicting the density map, Shi *et al.* [17] also add two more branches to their network and exploit the point annotations to estimate segmentation focus and global density maps simultaneously. Jiang *et al.* [18] add an auxiliary task of density-level classification and show that learning the multi-task network improves the density map generation. Perspective-aware convolutional neural network (PACNN) [6] learns to regress approximate ground truth perspective maps for two scale levels alongside the density distributions. Perspective maps are then used to assemble the final density prediction.

Kang *et al.* [5] compare various approaches to density map estimation, including methods that are not fully convolutional and operate in a sliding-window fashion. While some of the methods can have better results in terms of precise localization, the best counting performance is delivered by FCNN-skip architecture — a model composed of convolutional downsampling and upsampling parts with skip connections. To some extent, the basic hourglass block of our baseline model is similar to the FCNN-skip structure.

The superiority of the encoder-decoder models is also demonstrated by Thanasutives *et al.* [19], who use a dual-path decoder, where the first branch generates an attention map that is subsequently used in the second branch that produces the density estimates.

Among other notable methods that do not directly use a density map for count prediction but still utilize an only minimum level of annotation labels, are the works of Laradji *et al.* [20] and Liu *et al.* [21] who experiment with training detectors based on dot-level annotations. Furthermore, in Spatial Divide-and-Conquer Network (S-DCNet), Xiong *et al.* [8] reformulate the counting task as a classification problem and use a system of three networks to gradually divide the input image and classify the number of objects in the subdivided parts.

III. METHOD

In this section, we first outline the basic concept of counting based on density estimation. Utilized baseline model is then described. Subsequently, the core changes to the model, as well as the training procedure are presented.

A. Vehicle Counting Based on Density Map Estimation

In our work, we adopt the density-based approach to the problem of vehicle counting. We only use simple and cheap data labels in the form of dots, where each dot represents one vehicle. Mathematically, every dot can be understood as a delta function positioned on the annotated object. Individual delta functions are then convolved with Gaussian kernels to produce probability density function. All resultant density functions are combined to form a density map for the entire image. These maps then serve as a ground truth information to be learned by the neural network.

The important property that every probability density function integrates to one can subsequently be employed to extract total count estimation from a predicted probability map. Note that this property also holds in case of overlaps of Gaussian functions, and the overall result therefore remains unaffected. Moreover, once discrete representation is used, the integration property of probability density function can instead be transformed into summation. Consequently, the final vehicle count estimate can easily be obtained by performing a straightforward sum over the output density map.

It should also be pointed out that when resizing a density map, it is necessary to normalize it by the proportion of the area change, in order to keep the integration property valid. Therefore, whenever a density map is resized in our work, we also perform the corresponding normalization.

B. Hourglass Network for Vehicle Counting

As a baseline model, we choose the Stacked Hourglass Network designed by Newell *et al.* [22]. The network (shown in Fig. 2) is created by stacking blocks with encoder-decoder structure, which are referred to as hourglass modules.

Each of the hourglass modules downsamples and upsamples the features in the spatial dimension. For better gradient flow, the network contains skip-connections and an additional output layer with MSE loss function after each hourglass module. The individual parts of the hourglass module consist of residual blocks, each being composed of $3\times$(BN+ReLu+Conv) layers. Our baseline model for vehicle counting, termed *Stacked-HG*, contains two hourglass stacks. It is a fully convolutional network which, in line with the density-based counting approach, for each image pixel evaluates the likelihood that the pixel contains a vehicle.

Our only modification to the hourglass network is at the level of pre-processing layers. In the original network, the input is processed by residual blocks and downsampled two times (first by strided convolution, and then by max-pooling) before being passed to the stacks of hourglass modules. This results into the output map being $4\times$ smaller than the input. In our model, we remove the max pooling operation and therefore obtain an output that is downsampled only $2\times$. Otherwise, we adopt exactly the same network structure, and we refer the reader to the original paper [22] for further details.

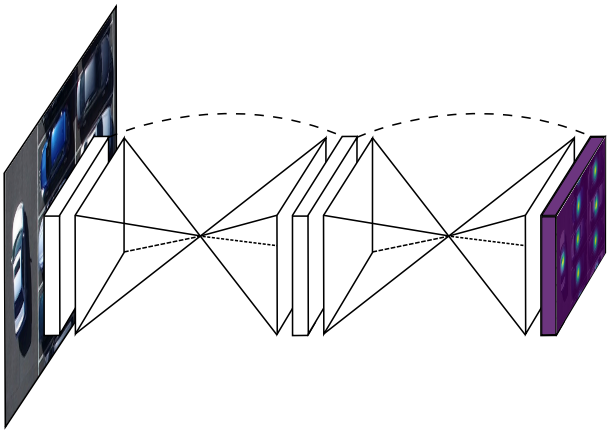

Fig. 2. Stacked Hourglass network [22] adopted as our baseline model. The network takes an images as an input and predicts a map with probability density distribution of the objects to be counted.

C. Hourglass Network with Unsupervised Scale Awareness

We propose an extension to the Stacked Hourglass Network which enables the model to automatically learn to select the best scale of the input and obtain an accurate density map. We implement this selection procedure in a way that does not demand any sort of additional annotations for training supervision. We refer to the proposed architecture as Scale-Aware Stacked Hourglass Network — *StackedHG-SA*.

The input to our scale-aware network is a pyramid of gradually downsampled images. The task of the network is now to identify the most appropriate scale for density map estimation. The core idea is therefore to let the network process all of the provided image scales and produce a quality score together with each of the resultant output maps.

Traditionally, this task could be carried out in a supervised fashion - *i.e.* the quality score would first be encoded by some sort of annotation, and subsequently trained to be regressed by the network. Nevertheless, we aim to avoid increasing the annotation burden, and we thus perform this without providing direct supervision.

To this end, we alter the loss function to include the quality score in a form of a weight that is to be learned by the network along the training process. More precisely, the score produced by the network becomes a multiplicative element for individual components of the loss function. As a result, the network is indirectly forced to learn to maximize the quality score of the most appropriate input scale while weighting down the unsuitable ones, in order to minimize the total loss value.

In the following, we first explain how the quality score is formed by our model. Subsequently, the detailed description of the changes to the loss function are addressed.

Global information about the content of the input image can be expected to be condensed inside the feature vector at the central (bottleneck) part of the hourglass module. These

features should also contain the necessary clues for identifying the scale quality of the particular image. We therefore add a new network branch that takes the central feature block as input and produces a quality coefficient for the corresponding scaled image — this is outlined in Fig. 3. Specifically, we first perform global average pooling [23] over the input features. The global average pooling step is necessary, since the channel dimensions (*i.e.* width and height) of the central feature block generally differ for input images of different scales. The resultant pooled vector is then processed by two projection layers, implemented by 1×1 convolutions, with ReLu activation in between. The second layer outputs only one number that serves as an intermediate quality coefficient. Finally, quality coefficients for all input image scales are gathered and passed through a softmax function to create the quality scores.

During inference, the produced quality score should identify the best density map for extraction of overall vehicle count. However, we avoid supervising the quality score directly, and instead modify the loss function to enable the network to learn the correct scores alongside the training of the density maps.

In particular, the output of the baseline Hourglass network is supervised by standard MSE loss function:

$$L_{MSE} = \frac{1}{N_{px}} \sum_{i=1}^{N_{px}} (m_i - \hat{m}_i)^2, \tag{1}$$

where m_i are the elements of the ground truth density map, $\hat{m}_i$ are the elements of the predicted map, and the N_{px} represents the total number of pixels.

Once the Hourglass network is used to process S scales of an input image, resultant losses can be considered as $L_1, ..., L_S$. These losses could immediately be combined to form the total loss $L = \frac{1}{S} \sum_{i=1}^{S} L_i$. However, we extend the loss function by multiplying each of the losses L_i with the corresponding quality score. Consequently, the scores become weights q_i of the individual loss values, and the loss function is:

$$L = \sum_{i=1}^{S} q_i \cdot L_i. \tag{2}$$

As a result, in order to minimize the total loss value L, this modification forces the network to assign smaller weights to input image scales that do not provide the necessary quality of the corresponding density maps, and vice versa. Note that the quality weights sum to one, since they are formed as an output of a softmax function. This also prevents the degenerate scenario of all weights collapsing to zero.

We also make the entire training procedure compatible with the concept of stacks within the hourglass architecture. As can be seen in Figure 4, each of the stacks produces its own set of quality coefficients. These are then passed through a softmax layer to form quality scores for loss function of the particular stack. Specifically, stack i of scale j produces a quality score $q_{j,i}$ which becomes a factor for the intermediate loss part $L_{j,i}$.

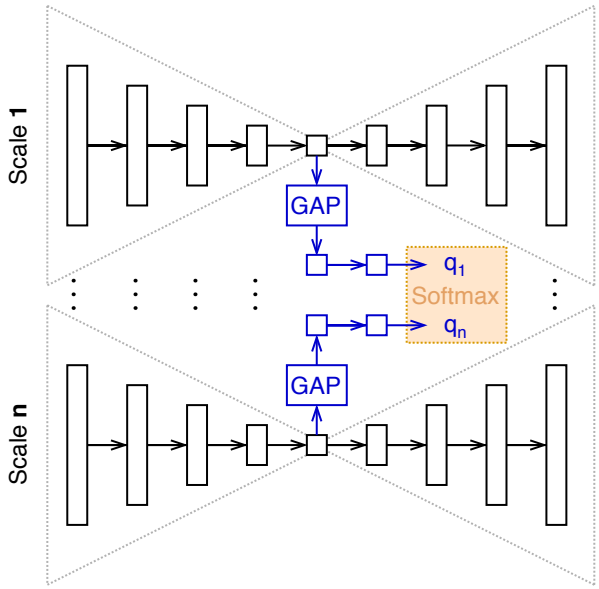

Fig. 3. Detail view of our proposed modification to the hourglass module. Central feature block is processed by global average pooling (GAP), and subsequently by two projection layers, to produce a quality coefficient. All quality coefficients are passed through softmax to obtain the final quality score for each of the input scales. Quality scores are then used within the loss function in order to enable fully automatic scale-awareness.

The final loss is then composed of all individual loss parts from all network stacks:

$$L_{final} = \frac{1}{N_{Stacks}} \sum_{i=1}^{N_{Stacks}} \sum_{j=1}^{N_{Scales}} q_{j,i} \cdot L_{j,i}. \tag{3}$$

While the image is subsampled to several scales that form the inputs to the network during the training phase, it is important to emphasize that the size (controlled by σ) of the Gaussian kernel, which is used to generate the corresponding ground truth output maps, remains the same. In this way, the network is allowed to choose the scale of the input objects that best fit the size of the density functions. By multiplying each of the loss parts with its quality score, the network simultaneously learns to predict the density maps, as well as their qualities. During inference, the same resizing is performed for the input image, and the output is obtained by simply taking the density map prediction with the highest quality score.

One additional note should be made about the normalization of individual loss parts $L_{j,i}$. Commonly, each of the losses can be divided by the number of pixels N_{px} in the map. However, these normalization factors would now differ significantly due to the changes in the size of the map. Consequently, this would distort the magnitudes of the loss function parts and thus hinder the training process. We therefore do not normalize the loss parts by the number of pixels, but by the pixel area under the generated Gaussian distribution functions.

IV. Datasets

In this section, we provide a brief description of available datasets that are used for evaluating the proposed method

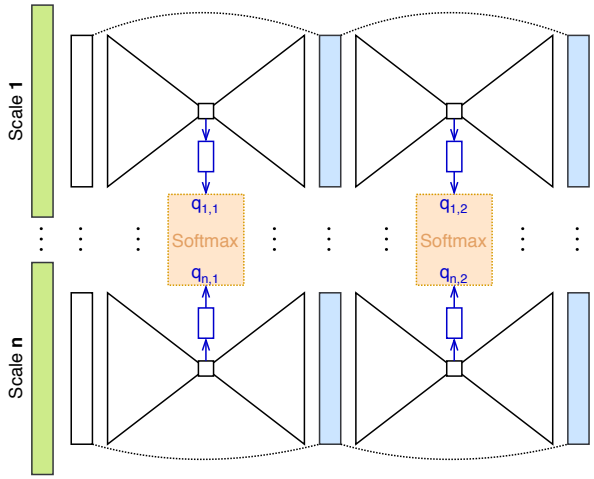

Fig. 4. We make the added quality score branches (as shown in Fig. 3) compatible with the concept of stacks within the Stacked Hourglass network. Specifically, stack i of scale j produces a quality score $q_{j,i}$ which becomes a factor for the intermediate loss part $L_{j,i}$. For more details, see Section III-C.

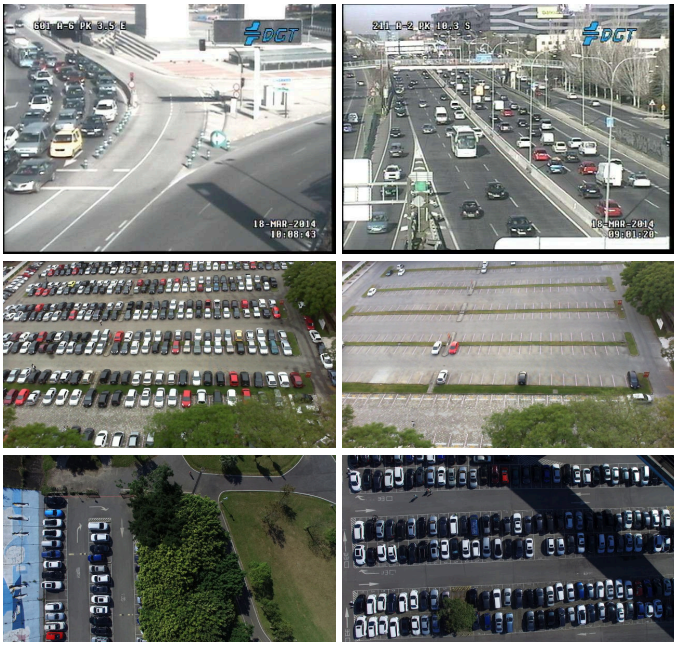

Fig. 5. Samples from datasets used in this work. From top row to bottom — TRANCOS dataset, PUCPR+ dataset and CARPK dataset.

together with a description of evaluation metrics. Note that our method only requires simple dot annotations, and thus, if bounding box annotations are provided by a dataset, we convert them to dots by taking the center point of each bounding box.

A. TRANCOS dataset

Guerrero-Gómez-Olmedo *et al.* [24] published the TRANCOS (TRaffic ANd COngestionS) dataset, which focuses on traffic congestions on highways. The dataset is made by capturing traffic using real surveillance cameras during three weeks. The dataset contains 1,244 images and 46,796 annotated vehicles. These images are split into three sets as follows: Training (403 images), Validation (420 images) and Testing set (421 images). Sample images from TRANCOS dataset are shown in Figure 5 (top row).

The TRANCOS dataset uses dot annotation technique to represent each vehicle's position. Since annotations are only partial, authors provide Region of Interest (ROI), specified by binary mask, to mask the un-annotated parts of the image. Unlike the other available datasets, TRANCOS dataset uses its own metric to measure network's accuracy — the *GAME(L)* metric (see Section IV-D).

B. PUCPR+ dataset

Original Pontifical Catholic University of Parana (PUCPR) dataset was published as a part of PKLot dataset [25]. It was captured from 10th floor of the building by fixed camera which covers parking lot with 331 parking spaces. These captures are taken during various weather conditions such as sunny, overcast and rainy weather. However, the dataset only provides annotation for 100 parking places instead of full parking lot.

Hsieh *et al.* [9] completed annotations (bounding boxes) for every vehicle in all images from the original PUCPR dataset and re-published it as PUCPR+ dataset. Figure 5 (middle row) shows sample images from PUCPR+. The dataset contains 125 images with 16,915 vehicles in total. Only 25 images with 3,920 vehicles are meant for model testing. The PUCPR+ dataset uses *MAE* and *RMSE* metrics which were extended by *GAME(L)* in our case.

C. CARPK dataset

Hsieh *et al.* [9] also proposed a drone-based dataset called CARPK. The CARPK dataset provides large-scale images taken from the point of view of a drone. The drone took images from four different parking lots. There are 1,448 images with approximately 90,000 vehicles in total annotated by bounding boxes. Sample images from CARPK dataset are shown in Figure 5 (bottom row). The dataset is split into a training set (989 images) and testing set (459 images). The resolution of all images is 1280×720 px.

Drone addresses the problem of a bias created by a fixed camera scene, where the point of view is constant. The basic evaluation protocol for CARPK dataset uses the same metrics as PUCPR+ (*MAE* and *RMSE*) and we extended these basic metrics by *GAME(L)* as well.

D. Evaluation Metrics

The primary metric for accuracy evaluation of any counting method is *MAE* (Mean Absolute Error). It calculates the absolute differences between expected and predicted vehicle counts and averages them across all images. Equation 4 shows the general formula to compute *MAE*.

$$\mathrm{MAE} = \frac{1}{N} \cdot \sum_{i=1}^{N} |y - \hat{y}|, \tag{4}$$

where N is the total number of data samples, y is the expected value and $\hat{y}$ is the predicted value.

One of the frequently used metrics is *RMSE* (Root Mean Square Error). Value of this metric is either 0 (a perfect fit) or a positive number. The general formula for *RMSE* calculation is:

$$\text{RMSE} = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} (y - \hat{y})^2}, \tag{5}$$

where N is the total number of data samples, y is the expected value and $\hat{y}$ is the predicted value.

GAME(L) metric, introduced with TRANCOS dataset, measures network performace by partitioning the image into regions. This metric is used for evaluation of object count on local level *GAME(1, 2, ..., L)*, rather than globally on the whole image, where *GAME(0)* becomes equivalent to *MAE*. This metric is defined as:

$$\text{GAME}(L) = \frac{1}{N} \cdot \sum_{n=1}^{N} \left(\sum_{l=1}^{4^L} |e_n^l - gt_n^l| \right), \tag{6}$$

where N is the total number of test images, L is the total levels number (max 4-regions exponent), e_n^l is the estimated count within region l of image n and gt_n^l is the ground truth count within region l of image n. For more details we refer the reader to the original paper [24].

V. Experiments

We run a set of experiments to validate the proposed structure of the vehicle counting network and the corresponding training scheme. We use the above described public datasets, *i.e.* PUCPR+, TRANCOS and CARPK. For each dataset, we train our networks on the training image split and evaluate on the test set. In case a dataset does not provide a designated validation image set, we reserve 10% of the training data to form a validation split, which is used for training control (early stopping, *etc.*).

All networks are trained from scratch, using 256 features and learning rate of $2.5 \cdot 10^{-4}$ with Adam optimizer. We augment the original images using contrast and brightness shifts, Gaussian blur, modifications in HSV space, and applying JPEG compression, as well as random rotations and scaling. We train on image crops of 256×256 pixels and use 6 samples per batch. We experimented with 3 and 5 input scales for our scale-aware network, where 3 scales use downsampling factors of 1, 2, and 4, while 5 scales also include the intermediate factors of 1.5 and 3. During our experiments, we found no significant difference between using these two scale schemes.

A. Experiments on Standard Datasets

We first train and evaluate our baseline network, *StackedHG* (described in III-B). As can be seen from the results on all datasets, when compared to the current state-of-the-art methods, this model already represents a solid baseline. We assume this can be partly attributed to the fact that the hourglass structure contains significantly more parameters when contrasted to other networks that are based, for example, on VGG-16. Most importantly, the encoder-decoder architecture proves to be well suited for the task of per-pixel density estimation.

This high learning capacity of the network therefore enables it, to some extent, to directly capture the variance in the scale of individual vehicles that are induced by the perspective distortion present in the data. Nevertheless, more significant changes in scales of the input result in degraded performance, as also shown later.

Subsequently, we compare the baseline results with the results of the newly proposed architecture. Our *StackedHG-SA* model either outperforms the baseline scores or attains similar values. Moreover, the results are often on par with, or even surpass, those of other state-of-the-art methods, while providing the additional robustness against a broader range of input scale variation.

On TRANCOS dataset, most of the current state-of-the-art methods achieve comparable MAE (*i.e.*, GAME(0)) values of around 3.0 (see Table I). The same can already be observed for our baseline *StackedHG* network. While our scale-aware model (*StackedHG-SA*) does not surpass the baseline in this case, its results are again comparable with other methods.

For PUCPR+ and CARPK datasets, the results are shown in Table II and III, respectively. Apart from the methods that provide their evaluation on these datasets, we also re-implement other proposed architectures, including PACNN [6], which achieves state-of-the-art results on several crowds counting datasets. In the case of PUCPR+, the PACNN model represents the best results available to us.

For the PUCPR+ dataset, our baseline produces similar MAE results as the best PACNN model. While localization precision, manifested by GAME(1-3), degrades slightly more quickly, our scale-aware modification surpasses the baseline model and also produces state-of-the-art results in terms of RMSE and MAE metrics.

The best available results for CARPK dataset are obtained by GSP-224 [12]. Even though our baseline network does not provide results as close to this approach as in case of the other two datasets, our *StackedHG-SA* variant dramatically improves over the baseline scores and achieves values on par with the best known methods.

B. Scale Robustness

In order to demonstrate the contribution towards the robustness against more significant input scale variations, we perform another set of experiments on the CARPK dataset. Instead of evaluating directly (using the original images from the dataset test split), we first resize all of the images ($0.5\times$, $2\times$, $3\times$ and $4\times$) to simulate more extensive changes to the scale of the input objects. The results for all evaluation runs are shown in Table IV. It can be observed that the performance of the baseline model (*StackedHG*) deteriorates rapidly once inputs of varying scales are presented to the network. The

TABLE I
RESULTS FOR TRANCOS DATASET. GAME(0) = MAE.

Method	GAME(0)	GAME(1)	GAME(2)	GAME(3)
CCNN [4]	12.49	16.58	20.02	22.41
Hydra CNN - 3s [4]	10.99	13.75	16.69	19.32
AMDCN [13]	9.77	13.16	15.00	15.87
DA-net [26]	3.08	6.93	09.08	12.85
FCNN-skip [5]	4.61	8.39	11.08	16.10
CSRNet [14]	3.56	5.49	8.57	15.04
LC-ResFCN [20]	3.32	5.2	7.92	12.57
ADCrowdNet [16]	2.39	4.23	6.89	14.82
PSDDN [21]	4.79	5.43	6.68	8.40
E2D [27]	2.88	4.81	7.77	12.47
DADNet [28]	2.79	4.41	6.43	9.27
S-DCNet [8]	2.92	4.29	5.54	7.05
SPN [15]	3.35	4.94	6.47	9.22
CFF [17]	**2.00**	* **3.17**	* **4.45**	* **6.05**
M-SFANet [19]	2.23	3.46	4.86	6.91
DensityCNN-H [18]	3.17	4.78	6.30	8.26
LSC-CNN [29]	4.60	5.40	6.90	8.30
†MCNN [7]	8.78	10.78	11.28	12.21
†PACNN [6]	6.49	8.49	9.22	10.27
StackedHG	2.35	4.10	6.28	9.89
StackedHG-SA	3.43	5.52	7.88	12.44

†— refers to re-implemented and re-trained network.
* — results estimated from graph in the original paper.

TABLE II
RESULTS FOR PUCPR+ DATASET. GAME(0) = MAE.

Method	RMSE	GAME(0)	GAME(1)	GAME(2)	GAME(3)
†CCNN [4]	59.25	51.85	72.13	72.31	75.78
†Hydra CNN-3s [4]	84.25	74.15	83.20	85.21	85.51
†PACNN [6]	3.45	2.86	**4.97**	**6.41**	**7.96**
†PACNN - w/o persp. [6]	8.56	6.14	8.49	9.02	10.13
†MCNN [7]	31.90	26.49	28.82	29.95	30.95
StackedHG	4.28	3.40	5.41	9.91	23.47
StackedHG-SA	3.21	**2.32**	6.77	16.33	32.21

†— refers to re-implemented and re-trained network.

TABLE III
RESULTS FOR CARPK DATASET. GAME(0) = MAE.

Method	RMSE	GAME(0)	GAME(1)	GAME(2)	GAME(3)
GAP-C-224 [12]	9.59	7.65	-	-	-
GAP-PS-224 [12]	21.42	19.20	-	-	-
GSP-224 [12]	**8.09**	**5.46**	-	-	-
GMN model	9.9	7.48			
†CCNN [4]	25.63	30.39	34.32	38.24	40.94
†Hydra CNN-3s [4]	51.65	45.63	49.84	52.23	56.34
†PACNN [6]	20.47	17.54	19.55	22.63	24.71
†PACNN - w/o persp. [6]	11.54	9.03	12.80	14.93	**16.89**
†MCNN [7]	20.56	18.16	23.74	27.32	30.39
StackedHG	13.49	11.35	11.91	13.92	19.91
StackedHG-SA	8.45	6.00	**7.72**	**12.73**	23.67

†— refers to re-implemented and re-trained network.

TABLE IV
RESIZING THE IMAGES IN CARPK TEST SET SIMULATES MORE EXTENSIVE SCALE CHANGES OF THE INPUT OBJECTS. PACNN NETWORK AND OUR BASELINE STACKEDHG BOTH SHOW RAPID DEGRADATION OF PERFORMANCE. WHILE THE RESULTS OF THE SCALE-AWARE STACKEDHG-SA REMAIN STABLE.

Method	Image scale	RMSE	GAME(0)	GAME(1)	GAME(2)	GAME(3)
PACNN [6]	0.5x	78.94	73.64	73.80	73.70	74.37
	1x (baseline)	*20.47*	*17.54*	*19.55*	*22.63*	*24.71*
	2x	39.47	34.78	36.62	39.20	41.71
	3x	66.16	62.05	62.19	63.80	66.24
	4x	101.47	97.47	97.50	97.74	99.09
StackedHG	0.5x	90.53	84.54	84.58	84.88	86.89
	1x (baseline)	*13.49*	*11.35*	*11.91*	*13.92*	*19.91*
	2x	17.62	14.22	16.89	23.79	33.78
	3x	85.13	79.30	79.30	80.96	85.54
	4x	82.83	76.67	77.03	78.54	84.65
StackedHG-SA	0.5x	8.46	6.01	7.72	12.73	23.66
	1x (baseline)	*8.45*	*6.00*	*7.72*	*12.73*	*23.67*
	2x	8.43	5.94	7.67	12.68	23.60
	3x	8.32	5.54	7.39	11.88	21.12
	4x	9.21	6.32	8.47	12.79	21.10

same degradation can be seen in the case of PACNN network. On the other hand, the *StackedHG-SA* architecture, with its scale selection scheme, handles the change of size without any significant loss of accuracy.

C. Generalization to the Unseen Data

Additionally, we train the proposed model on a private dataset available to us[1]. The dataset is composed of images from various parking lots. The primary aim of this experiment is to verify the generalization and practical applicability of our method. A newly mounted camera would normally require manual calibration of image scaling coefficient before the data is processed by a baseline (or similar) neural network. As a result, several trial-and-error attempts would be made by the person setting up the camera in order to find a scaling factor that yields a satisfactory output. Our approach alleviates the burden of manual calibration, as the network is capable of selecting the correct input scale automatically. We successfully verify this by testing the network (trained on the private dataset) on a previously unknown camera observing a previously unseen parking lot. An example is provided in Fig. 6. By removing another step of manual work, our method can directly facilitate large-scale deployment of parking lot security cameras.

VI. CONCLUSION

We adopt a density-based vehicle counting approach and address the possibility of significant changes in the input scale. We extend the Stacked Hourglass network and develop a model that is capable of automatically selecting the best input scale by providing a quality score together with the predicted density map. In order to avoid any additional annotation effort, we modify the training process and enable learning of the quality scores alongside the density maps without the need of direct supervision. Our approach achieves results that are on par with state-of-the-art methods on three standard datasets while offering better robustness against significant variations in input scale. Furthermore, we perform additional experiments on a private dataset available to us and verify the practical applicability of our method.

[1]We are currently negotiating the possibility of publicly releasing the dataset for research purposes. The data may therefore become publicly available in the future.

REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *TPAMI*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[2] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *ICCV*, Oct. 2017, pp. 2980–2988.

[3] V. Lempitsky and A. Zisserman, "Learning To count objects in images," in *NIPS*, Dec. 2010, pp. 1324–1332.

[4] D. Oñoro-Rubio and R. J. López-Sastre, "Towards Perspective-Free Object Counting with Deep Learning," in *ECCV*, 2016, pp. 615–629.

[5] D. Kang, Z. Ma, and A. Chan, "Beyond counting: Comparisons of density maps for crowd analysis tasks - counting, detection, and tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[6] M. Shi, Z. Yang, C. Xu, and Q. Chen, "Revisiting Perspective Information for Efficient Crowd Counting," in *CVPR*, Jun. 2019, pp. 7271–7280.

[7] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-Image Crowd Counting via Multi-Column Convolutional Neural Network," in *CVPR*, Jun. 2016, pp. 589–597.

[8] H. Xiong, H. Lu, C. Liu, L. Liang, Z. Cao, and C. Shen, "From open set to closed set: Counting objects by spatial divide-and-conquer," in *ICCV*, 2019, pp. 8362–8371.

[9] M.-R. Hsieh, Y.-L. Lin, and W. H. Hsu, "Drone-Based Object Counting by Spatially Regularized Regional Proposal Network," in *ICCV*, 2017.

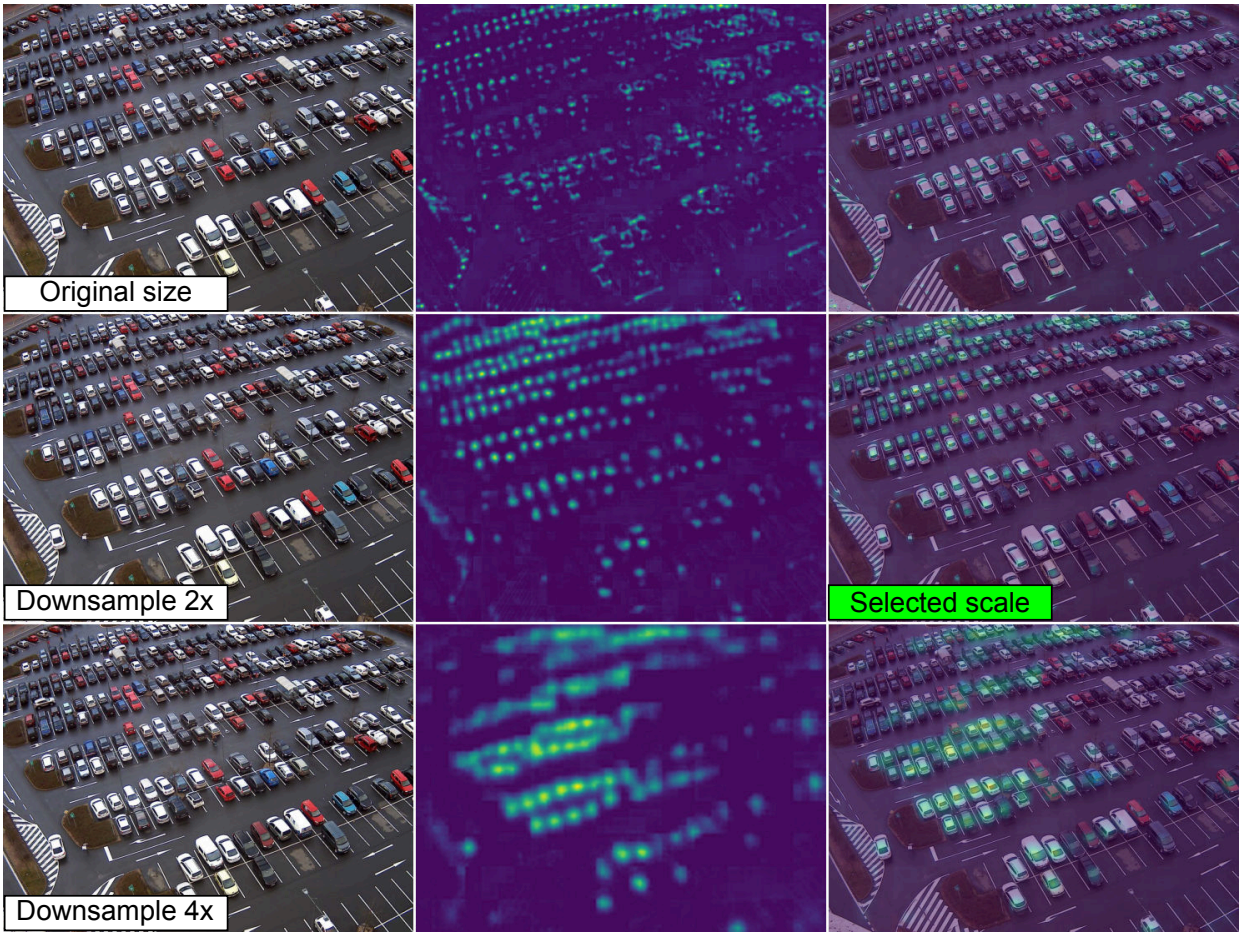

Fig. 6. An example of a result from our proposed scale-aware model trained on a private dataset and evaluated on a previously unknown camera observing a previously unseen parking lot. The network selects the middle input scale which yields the best density map quality. Note that the image also includes a perspective distortion which does not pose any significant problem for the final density map, once the correct scaling factor of the entire image is known. Our model is therefore capable of processing a much wider range of input object sizes, and thus alleviates the necessity of manual scaling factor input in case of newly mounted cameras. We would like to thank the camera owner for providing the data.

[10] P. Chattopadhyay, R. Vedantam, R. R. Selvaraju, D. Batra, and D. Parikh, "Counting Everyday Objects in Everyday Scenes," in *CVPR*, Jul. 2017, pp. 4428–4437.

[11] S. Aich and I. Stavness, "Improving Object Counting with Heatmap Regulation," *CoRR*, May 2018.

[12] S. Aich and I. Stavness, "Global Sum Pooling: A Generalization Trick for Object Counting with Small Datasets of Large Images," *CVPR Deep Vision Workshop*, Sep. 2019.

[13] D. Deb and J. Ventura, "An aggregated multicolumn dilated convolution network for perspective-free counting," in *CVPRW*, 04 2018.

[14] Y. Li, X. Zhang, and D. Chen, "CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes," in *CVPR*, 2018, pp. 1091–1100.

[15] X. Chen, Y. Bin, N. Sang, and C. Gao, "Scale pyramid network for crowd counting," in *WACV*, 2019, pp. 1941–1950.

[16] N. Liu, Y. Long, C. Zou, Q. Niu, L. Pan, and H. Wu, "ADCrowdNet: An attention-injective deformable convolutional network for crowd understanding," in *CVPR*, 06 2019, pp. 3220–3229.

[17] Z. Shi, P. Mettes, and C. G. M. Snoek, "Counting with focus for free," in *ICCV*, 2019.

[18] X. Jiang, L. Zhang, T. Zhang, P. Lv, B. Zhou, Y. Pang, M. Xu, and C. Xu, "Density-aware multi-task learning for crowd counting," *IEEE Transactions on Multimedia*, pp. 1–1, 2020.

[19] P. Thanasutives, K.-i. Fukui, M. Numao, and B. Kijsirikul, "Encoder-decoder based convolutional neural networks with multi-scale-aware modules for crowd counting," *arXiv preprint arXiv:2003.05586*, 2020.

[20] I. H. Laradji, N. Rostamzadeh, P. O. Pinheiro, D. Vazquez, and M. Schmidt, "Where are the blobs: Counting by localization with point supervision," in *ECCV*, 2018, pp. 547–562.

[21] Y. Liu, M. Shi, Q. Zhao, and X. Wang, "Point in, box out: Beyond counting persons in crowds," in *CVPR*, 06 2019.

[22] A. Newell, K. Yang, and J. Deng, "Stacked Hourglass Networks for Human Pose Estimation," in *ECCV*, pp. 483–499.

[23] M. Lin, Q. Chen, and S. Yan, "Network In Network," in *ICLR*, 2014.

[24] R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. López-Sastre, S. Maldonado Bascón, and D. Oñoro-Rubio, "Extremely Overlapping Vehicle Counting," in *IbPRIA*, 2015.

[25] P. R. L. de Almeida, L. S. Oliveira, A. S. Britto, E. J. Silva, and A. L. Koerich, "PKLot – A robust dataset for parking lot classification," *Expert Systems with Applications*, vol. 42, no. 11, pp. 4937–4949, Jul. 2015.

[26] Z. Zou, X. Su, X. Qu, and P. Zhou, "DA-Net: Learning the fine-grained density distribution with deformation aggregation network," *IEEE Access*, vol. 6, pp. 60 745–60 756, 2018.

[27] Z. Zou, H. Shao, X. Qu, W. Wei, and P. Zhou, "Enhanced 3D convolutional networks for crowd counting," in *BMVC*, 08 2019.

[28] D. Guo, K. Li, Z.-J. Zha, and M. Wang, "DADNet: Dilated-attention-deformable convnet for crowd counting," in *ACM Multimedia Conference*, 10 2019, pp. 1823–1832.

[29] D. Babu Sam, S. V. Peri, M. Narayanan Sundararaman, A. Kamath, and V. B. Radhakrishnan, "Locate, size and count: Accurately resolving people in dense crowds via detection," *TPAMI*, 2020.