Class-Agnostic Counting

Erika Lu, Weidi Xie, and Andrew Zisserman

Visual Geometry Group, University of Oxford
{erika,weidi,az}@robots.ox.ac.uk

Abstract. Nearly all existing counting methods are designed for a specific object class. Our work, however, aims to create a counting model able to count any class of object. To achieve this goal, we formulate counting as a matching problem, enabling us to exploit the image self-similarity property that naturally exists in object counting problems.
We make the following three contributions: *first*, a Generic Matching Network (GMN) architecture that can potentially count any object in a class-agnostic manner; *second*, by reformulating the counting problem as one of matching objects, we can take advantage of the abundance of video data labeled for tracking, which contains natural repetitions suitable for training a counting model. Such data enables us to train the GMN. *Third*, to customize the GMN to different user requirements, an adapter module is used to specialize the model with minimal effort, i.e. using a few labeled examples, and adapting only a small fraction of the trained parameters. This is a form of few-shot learning, which is practical for domains where labels are limited due to requiring expert knowledge (e.g. microbiology).
We demonstrate the flexibility of our method on a diverse set of existing counting benchmarks: specifically cells, cars, and human crowds. The model achieves competitive performance on cell and crowd counting datasets, and surpasses the state-of-the-art on the car dataset using only three training images. When training on the entire dataset, the proposed method outperforms all previous methods by a large margin.

Keywords: Category-agnostic Object Counting · Convolutional Neural Networks· Deep Learning.

1 Introduction

The objective of this paper is to count objects of interest in an image. In the literature, object counting methods are generally cast into two categories: detection-based counting [5,10,16] or regression-based counting [2,4,8,19,21,24,34]. The former relies on a visual object detector that can localize object instances in an image; this method, however, requires training individual detectors for different objects, and the detection problem remains challenging if only a small number of annotations are given. The latter avoids solving the hard detection problem – instead, methods are designed to learn either a mapping from global image features to a scalar (number of objects), or a mapping from dense image

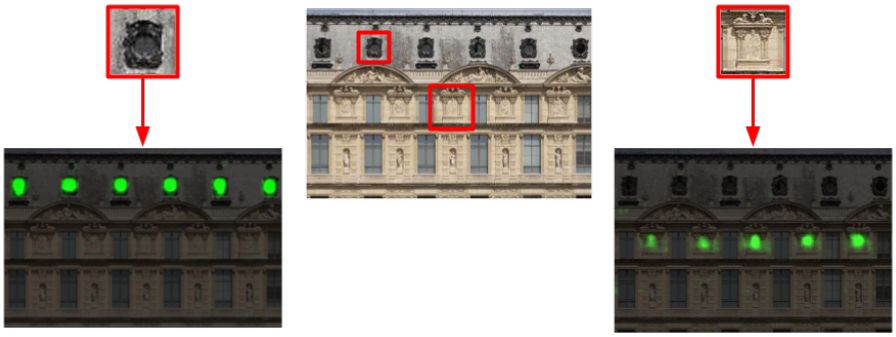


Fig. 1: The model (trained on tracking data) can count an object, e.g. windows or columns, specified as an exemplar patch (in red), without additional training. The heat maps indicate the localizations of the repeated objects. This image is unseen during training.

features to a density map, achieving better results on counting overlapping instances. However, previous methods for both categories of method (detection, regression) have only developed algorithms that can count a particular class of objects (e.g. cars, cells, penguins, people).

The objective of this paper is a class-agnostic counting network – one that is able to flexibly count object instances in an image by, for example, simply specifying an exemplar patch of interest as illustrated in Figure 1. To achieve this, we build on a property of images that has been largely ignored explicitly in previous counting approaches – that of *image self-similarity*. At a simplistic level, an image is deemed self-similar if patches repeat to some approximation – for example if patches can be represented by other patches in the same image. Self-similarity has underpinned applications for many vision tasks, ranging from texture synthesis [11], to image denoising [7], to super-resolution [13].

Giving the observation of self-similarity, image counting can be recast as an image *matching* problem – counting instances is performed by matching (self-similar patches) within the same image. To this end we develop a *Generic Matching Network* (GMN) that learns a discriminative classifier to match instances of the exemplar. Furthermore, since matching variations of an object instance within an image is similar to matching variations of an object instance between images, we can take advantage of the abundance of video data labeled for tracking which contains natural repetitions, to train the GMN. This observation, that matching within an image can be thought of as tracking within an image, was previously made by Leung and Malik [22] for the case of repeated elements in an image.

Beyond generic counting, there is often a need to *specialize* matching to more restrictive or general requirements. For example, to count only red cars (rather than all cars) or to count cars at all orientations (which goes beyond simple similarity measures such as squared sum of differences), extending the

intra-class variation for the object category of interest [14,18]. To this end, we include an adaptor module that enables fast domain adaptation [28] and few-shot learning [32,33], through the training of a small number of tunable parameters, using very few annotated data.

In the following sections, we begin by detailing the design and training procedure of the GMN in § 2, and demonstrate its capabilities on a set of example counting tasks. In § 3, we adapt the GMN to specialize on several counting benchmark datasets, including the VGG synthetic cells, HeLa cells, and cars captured by drones. During adaptation, only a small number of parameters (3% of the network size) are added and trained on the target domain. Using a very small number of training samples (as few as 3 images for the car dataset), the results achieved are either comparable to, or surpass the current state-of-the-art methods by a large margin. In § 4, we further extend the counting-by-matching idea to a more challenging scenario: Shanghaitech crowd counting, and demonstrate promising results by matching image statistics on scenes where accurate instance-level localization is unobtainable.

2 Method

In this paper, we consider the problem of instance counting, where the objects to be counted in a single query are from the same category, such as the windows on a building, cars in a parking lot, or cells of a certain type.

To exploit the *self-similarity* property, the counting problem is reformalized as localizing and counting "repeated" instances by *matching*. We propose a novel architecture – GMN, and a counting approach which requires learning a comparison scheme for two given objects (patches) in a metric space. The structure of the model naturally accommodates class-agnostic counting, as it learns to search for repetitions of an exemplar patch containing the desired instance. Note that, the concept of *repetition* is defined in a very broad sense; in the following experiments, we show that objects with various shapes, overlaps, and complicated appearance changes can still be treated as "repeated" instances.

The entire GMN consists of three modules, namely, *embedding*, *matching*, and *adapting*, as illustrated in Figure 2. In the *embedding* module, a two-stream network is used to encode the exemplar image patch and the full-resolution image into a feature vector and dense feature map, respectively. In the *matching* module, we learn a discriminative classifier to densely match the exemplar patch to instances in the image. Such learning overcomes within image variations such as illumination changes, small rotations, etc. The object locations and final count can then be acquired by simply taking the *local maximums* or *integration* over the output similarity maps, respectively. Empirically, integral-based counting shows better performance in scenarios where instances have significant overlap, while local max counting is preferred where objects are well-separated, and the positional information is of interest for further processing (e.g. seeds for segmentation).

To avoid the time-consuming collection of annotations for counting data, we use the observation that repetitions occur naturally in videos, as objects are seen under varying viewing conditions from frame to frame. Consequently, we can train the generic matching network with the extensive training data available for tracking (specifically the ILSVRC video dataset for object detection [31]). In total, the dataset contains nearly 4500 videos and over $1M$ annotated frames.

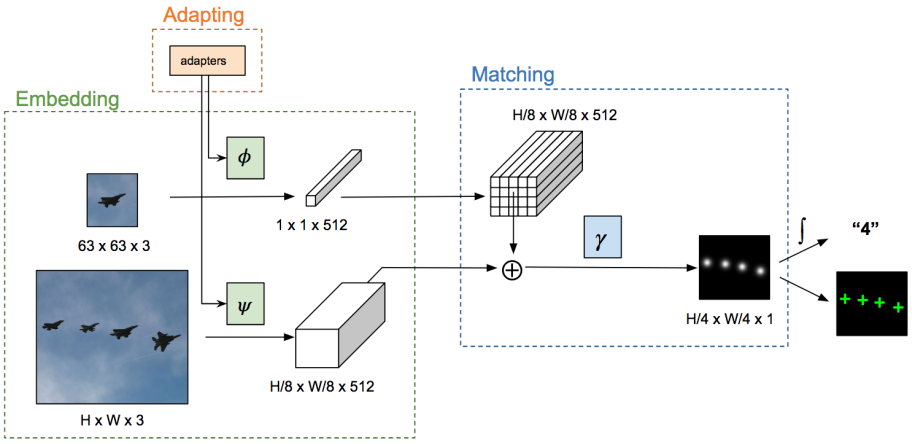


Fig. 2: The GMN architecture consists of three modules: embedding, matching, and adapting. The final count and detections are obtained by taking the integral and local maximums, respectively, over the output heatmap. The integral-based method is used for counting instances with significant overlap, while the local max is used where objects are well-separated and the positional information is of interest for further processing. The $\bigoplus$ represents channel-wise concatenation.

Given a trained matching model, several factors can prevent it from generalizing perfectly onto the target domain: for instance, the image statistics can be very different from the training set (e.g. natural images vs. microscopy images), or the user requirements can be different (e.g. counting cars vs. counting only red cars). Efficient domain adaptation requires a module that can change the network activations with minimal effort (that is, minimal number of trainable parameters and very few training data). Thus, for the *adapting* stage, we incorporate residual adapter modules [28] to specialize the GMN to such needs. Adapting to the new counting task then merely involves freezing the majority of parameters in the generic matching network, and training the adapters (a small number of extra parameters) on the target domain with only a few labeled examples.

2.1 Embedding

In this module, a two-stream network is defined for transforming raw RGB images into high-level feature encodings. The two streams are parametrized by separate functions for higher representation capacity:

$$v = \phi(z; \theta_1) \qquad f = \psi(x; \theta_2)$$

In detail, the function ϕ transforms an exemplar image patch $z \in R^{63\times63\times3}$ to a feature vector $v \in R^{1\times1\times512}$, and ψ maps the full image $x \in R^{H\times W\times3}$ to a feature map $f \in R^{H/8\times W/8\times512}$. Both the vector v and feature maps f are L2 normalized along the feature dimensions. In practice, our choices for $\phi(\cdot;\theta_1)$ and $\psi(\cdot;\theta_2)$ are ResNet-50 networks [15] truncated after the final conv3_x layer. The resulting feature map from the image patch is globally max-pooled into the feature vector v.

2.2 Matching

The relations between the resulting feature vector and maps are modeled by a trainable function $\gamma(\cdot;\theta_3)$ that takes the concatenation of v and f as input, and outputs a similarity heat map, as shown in Figure 2. Before concatenation, v is broadcast to match the size of the feature maps to accommodate the fully convolutional feature, which allows for efficient modeling of the relations between the exemplar object and all other objects in the image. The similarity Sim is given by

$$Sim = \gamma([broadcast(v) : f]; \theta_3)$$

where ":" refers to concatenation, and $\gamma(\cdot;\theta_3)$ is parametrized by one 3×3 convolutional layer and one 3×3 convolutional transpose layer with stride 2 (for upsampling).

2.3 Training Generic Matching Networks

The generic matching network (consisting of embedding and matching modules) is trained on the ILSVRC video dataset. The ground truth label is a Gaussian placed at each instance location, multiplied by a scaling factor of 100, and a weighted MSE (Mean Squared Error) loss is used. Regressing a Gaussian allows the final count to be obtained by simply summing over the output similarity map, which in this sense doubles as a density map.

During training, the exemplar images are re-scaled to size 63×63 pixels, with the object of interest centered to fit the patch, and the larger 255×255 search image is taken as a crop centered around the scaled object (architecture details can be found in Table 1. More precisely, we always scale the search image according to the bounding box (w,h) of the exemplar objects, where the scale factor is obtained by solving $s \times h \times w = 63^2$. The input data is augmented with horizontal flips and small ($< 25°$) rotations and zooms, and we sample both

positive and negative pairs. In all subsequent experiments, the network has been pre-trained as described here.

Module	Exemplar Patch ($N \times 63 \times 63 \times 3$)	Image to Count ($N \times 255 \times 255 \times 3$)	Output Size
Embedding	conv, 7×7, 64, stride 2	conv, 7×7, 64, stride 2	$N \times 32 \times 32 \times 64$ $N \times 128 \times 128 \times 64$
	max pool, 3×3, stride 2 $\begin{bmatrix} \text{conv}, 1 \times 1, 64 \\ \text{conv}, 3 \times 3, 64 \\ \text{conv}, 1 \times 1, 256 \end{bmatrix} \times 3$	max pool, 3×3, stride 2 $\begin{bmatrix} \text{conv}, 1 \times 1, 64 \\ \text{conv}, 3 \times 3, 64 \\ \text{conv}, 1 \times 1, 256 \end{bmatrix} \times 3$	$N \times 16 \times 16 \times 256$ $N \times 64 \times 64 \times 256$
	$\begin{bmatrix} \text{conv}, 1 \times 1, 128 \\ \text{conv}, 3 \times 3, 128 \\ \text{conv}, 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} \text{conv}, 1 \times 1, 128 \\ \text{conv}, 3 \times 3, 128 \\ \text{conv}, 1 \times 1, 512 \end{bmatrix} \times 4$	$N \times 8 \times 8 \times 512$ $N \times 32 \times 32 \times 512$
	Global Maxpool	No Operation	$N \times 1 \times 1 \times 512$ $N \times 32 \times 32 \times 512$
Matching	Vector Broadcasting (32×32)	No Operation	$N \times 32 \times 32 \times 512$ $N \times 32 \times 32 \times 512$
	Feature Map Concatenation		$N \times 32 \times 32 \times 1024$
	Relation Module $\begin{bmatrix} \text{conv}, 3 \times 3, 256 \\ \text{convt}, 3 \times 3, 256 \end{bmatrix}$		$N \times 64 \times 64 \times 256$
	Prediction $\begin{bmatrix} \text{conv}, 3 \times 3, 1 \end{bmatrix}$		$N \times 64 \times 64 \times 1$

Table 1: Architecture of the generic matching networks. "convt" refers to convolutional transpose with stride 2.

Once trained on the tracking data, the model can be directly applied for detecting repetitions within an image. We show a number of example predictions in Figure 3. Note here, several interesting phenomena can be seen: *first*, as expected, the generic matching network has learned to match instances beyond a simplistic level; for instance, the animals are of different viewpoints, the bird in the fourth row is partially occluded, and the persons are not only partially occluded, but also in different shirts with substantial appearance variations; *second*, object overlaps can also be handled, as shown in the airplane cases; *third*, although the ImageNet training set is only composed of natural images, and none of the categories has a similar appearance or distribution to the HeLa cells, the generic matching network succeeds despite large appearance and shape variations which exist for cells. These results validate our idea of building a class-agnostic counting network. However, it is crucial to be able to easily *adapt* the pre-trained model to further specialize to new domains.

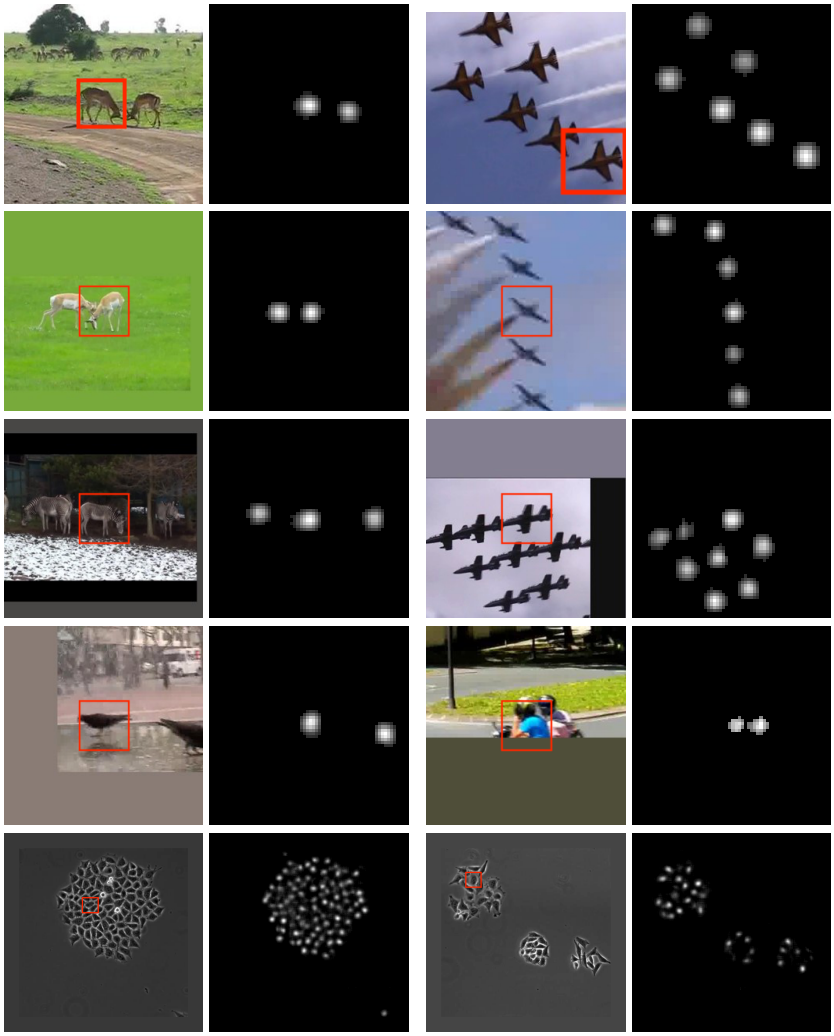


Fig. 3: Similarity predictions of the generic matching network on the video validation set, and on an *unseen* dataset of HeLa cells. The exemplar patch is marked with a red square. Images are padded with the mean value and the resolution has been changed for visualization purposes. As expected, the generic matching network has learned to match instances beyond a simplistic level; for instance, the animals are of different viewpoints, the matched bird in the fourth row is partially occluded, and the people are in different colored shirts. More interestingly, it acts as an excellent initialization for objects from unseen domains, even in the presence of large appearance and shape variation in the case of HeLa cells.

2.4 Adapting

The next objective is to specialize the network to new domains or new user requirements. We add *residual adapter modules* [28] implemented as 1×1 convolutions in parallel with the existing 3×3 convolutions in the embedding module of the network. During adaptation, we freeze all of the parameters in the pre-trained generic matching network, and train only the adapters and batch normalization layers. This results in 178K trainable parameters out of a total network size of 6.0M parameters, only 3% of the total.

2.5 Discussion and relation to prior work

Object counting poses certain additional challenges that are less prominent or non-existent in tracking. First, rather than requiring a single maximum in a candidate window (that localizes the object), counting requires a clean output map to distinguish multiple matches from noise and false positives. Second, unlike the continuous variation of object shape and appearance in the tracking problem, object counting can have more challenging appearance changes, e.g. large degrees of rotation, and intra-class variation (in the case of cars, both color and shape). Thus, we find the approaches used in template matching (SSD or cross-correlation [6,9,22]) to be insufficient for our purposes (as will be shown in Table 4). To address these challenges, we learn a discriminative classifier $\gamma(\cdot;\theta_3)$ between the exemplar patch and search image, an idea that dates back to [23].

The residual adapters [28] are added only to the embedding module, but we train the batch normalization layers throughout the entire network. Marsden et al. [25] also use residual adapters to adapt a network for counting different objects. However, they place the modules in the final fully connected layers in order to regress a count, whereas we add them to the convolutional layers in the residual blocks of the embedding module, such that they are able to change the filter responses at every stage of the base model, providing more capacity for adaptation.

3 Counting Benchmark Experiments

As a proof of concept, the generic matching network is visually validated as a strong initialization for counting objects from unseen domains (Figure 3). To further demonstrate the effectiveness of the general-purpose GMN, we adapt the network to three different datasets: VGG synthetic cells [20,21], HeLa phase-contrast cells [3], and a large-scale drone-collected car dataset [16].

Each of these datasets poses unique challenges. The synthetic cells contain many overlapping instances, a condition where density estimation methods have shown strong performance. The HeLa cells exhibit significantly more variation in size and appearance than the synthetic cells, and the number of training images is extremely limited (only 11 images); thus, detection-based methods with handcrafted features have shown good results. In the car dataset, cars appear in

various orientations, often within the same image, and can be partially occluded by trees and bridges; there is also clutter from motorbikes, buildings, and other distractors (Figure 6). As shown in Hsieh et al. [16], state-of-the-art models for object detection produce a very high error rate.

3.1 Evaluation Metrics

The metrics we use for evaluation throughout this paper are the mean absolute counting error (MAE), precision, recall, and F_1 score. To determine successful detections, we first take the local maximums (above a threshold T) of the predicted similarity map as the detections. T is usually set as the value that maximizes the F_1 score on a validation set. Note that, since multiple combinations of recall and precision can give the same F_1 score, we prioritize the recall score. Following [3], we then match these predicted detections with the ground truth locations using the Hungarian algorithm, with the constraint that a successful detection must lie no further than a tolerance R from the ground truth location, where R is set as the average radius of each object.

3.2 Synthetic fluorescence microscopy

The synthetic VGG cell dataset contains 200 fluorescence microscopy cell images, evenly split between training and testing sets. We follow the procedure proposed by Lempitsky and Zisserman [21] of sampling 5 random splits of the training set with N training images and N validation images. Results in Table 2 and Figure 4 show that our method is not restricted to detection-based counting, but also performs well on density estimation-type problems in a setting with high instance overlap. Note that, we compare with methods that are highly engineered for this dataset.

Method	MAE	Precision	Recall	F_1-score
Xie et al.[34]	2.9 ± 0.2	-	-	-
Fiaschi et al.[12]	3.2 ± 0.1	-	-	-
Lempitsky and Zisserman [21]	3.5 ± 0.2	-	-	-
Barinova et al.[5]	6.0 ± 0.5	-	-	-
Singletons [3]	51.2 ± 0.8	98.87 ± 1.52	72.07 ± 0.85	83.37 ± 1.20
Full system w/o surface [3]	5.06 ± 0.2	95.00 ± 0.75	91.97 ± 0.43	93.46 ± 0.15
Ours	3.56 ± 0.27	99.43 ± 0.05	82.50 ± 0.15	90.18 ± 0.07

Table 2: Results for the synthetic cell dataset. All methods are trained on the $N = 32$ split. Standard deviations are calculated using 5 random splits of training and validation sets and 5 randomly sampled exemplar patches per image. Note here, the exemplar patches are sampled from images in the training set, and different exemplar patches have negligible effect on performance.

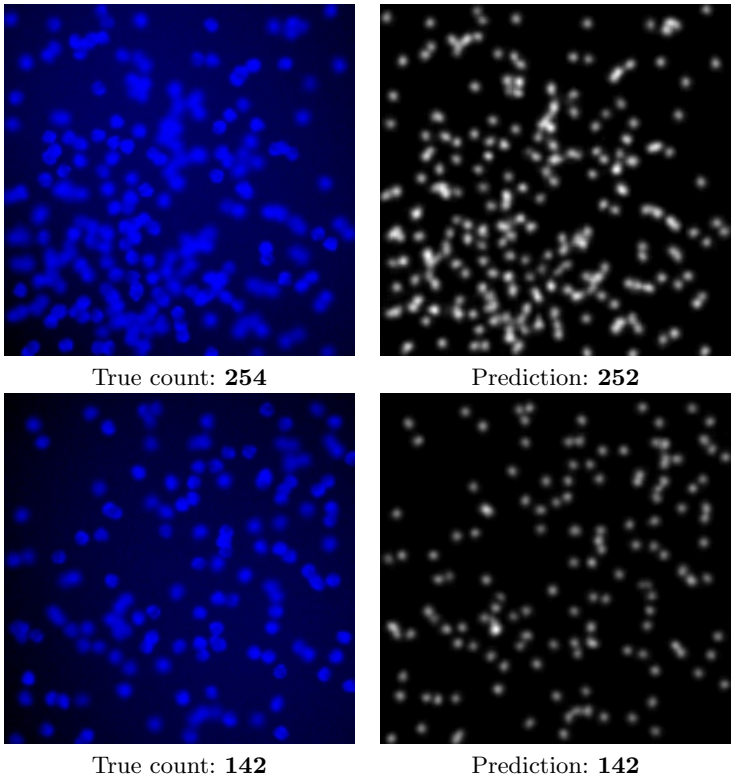


Fig. 4: Example of counting results on synthetic cell images. For each pair of images, left: original image, and right: the network's predicted heat map, which is *summed* to give the estimated count.

3.3 HeLa cells on phase contrast microscopy

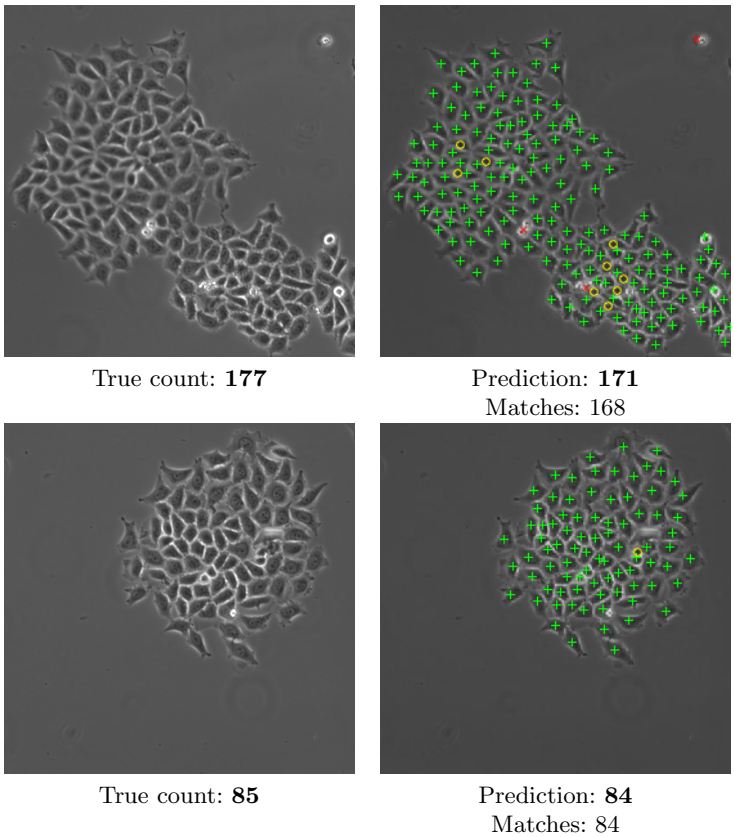
The dataset contains 11 training and 11 testing images. We follow the training procedure of [3] and train in a leave-one-out fashion for selecting hyperparameters, e.g. detection threshold T. Results are shown in Table 3. As shown in Figure 5, our method performs well in scenarios of large intra-class variations in shape and size, where SSD and cross-correlation would suffer. Overall, our GMN achieves comparable results to the conventional methods with handcrafted features, despite the training dataset being extremely small for current deep learning standards.

3.4 Cars

We next demonstrate the GMN's performance on counting cars in aerial images. This drone-collected dataset (CARPK) consists of 989 training images and 459

Method	MAE	Precision	Recall	F_1-score
Correlation clustering [36]	-	-	-	95
Singletons [3]	2.36 ± 0.67	93.70 ± 0.20	91.94 ± 0.72	92.81 ± 0.35
Full system w/o surface [3]	3.84 ± 1.44	98.51 ± 1.16	95.76 ± 0.27	97.10 ± 0.27
Ours	3.53 ± 0.18	96.05 ± 0.04	94.22 ± 0.06	95.12 ± 0.05

Table 3: Results for the HeLa cell dataset. We calculate MAE using the detection counts, since the instances are well-separated. Our standard deviations are calculated using 5 randomly sampled exemplar patches per image. Note here, the 5 exemplar patches are sampled from images in the training set; different exemplar patches have negligible effect on performance.



True count: **177**

Prediction: **171**
Matches: 168

True count: **85**

Prediction: **84**
Matches: 84

Fig. 5: Example detection results on the HeLa cell test set. Correct detections (based on Hungarian matching) are marked with a green '+', false positives with a red 'x', and missed detections with a yellow '○'.

testing images (nearly 90,000 instances of cars), where the images are taken from overhead shots of car parking lots. The training images are taken from three different parking lot scenes, and the test set is taken from a fourth scene. We compare our network to the region proposal and classification methods in Table 4.

In the experiments, we train two GMN models with augmentation: one on just three images (99 total cars) randomly sampled from the training scenes, which achieves state-of-the-art results, and one on the full CARPK training set, which further boosts the performance by a large margin.

Method	T	MAE	RMSE	Recall	Precision
*YOLO [16,29]	-	48.89	57.55	-	-
*Faster R-CNN [16,30]	-	47.45	57.39	-	-
*Faster R-CNN (RPN-small) [16,30]	-	24.32	37.62	-	-
†One-Look Regression [16,26]	-	59.46	66.84	-	-
*Spatially Regularized RPN [16]	-	23.80	36.79	57.5%	-
Template matching (Sum of Squared Distances)	-	49.8	59.7	20.0%	29.1%
Ours (3 images, 99 cars)	2.5	36.71	44.16	60.65%	93.91%
Ours (3 images, 99 cars)	2	22.32	28.72	71.32%	90.23%
Ours (3 images, 99 cars)	1.75	17.32	22.81	74.16%	87.87%
Ours (3 images, 99 cars)	1.5	13.38	18.03	76.1%	85.1%
Ours (full dataset)	2.75	19.66	25.12	78.61%	97.0%
Ours (full dataset)	2.5	14.36	19.01	83.2%	96.0%
Ours (full dataset)	2	8.38	11.55	87.46%	93.4%
Ours (full dataset)	1.75	**7.48**	**9.9**	88.4%	91.8%

Table 4: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Recall and Precision comparisons on the CARPK dataset. The "*" indicates that the method has been fine tuned on the full dataset, and the "†" indicates that the method has been revised to fit the dataset, as described in [16]. We show our method trained on 3 images and on the full dataset, with varying thresholds T. We calculate MAE using 5 randomly sampled exemplar patches per image, and the final counts are obtained from local maximums (counting by detection). Note here, the exemplar patches are sampled from images in the training set, and different exemplar patches have negligible effect on performance. Standard deviation is not reported in this table, but can be easily computed following the previously reported manner.

When determining counts based on local maximums, we note it is possible that our model outperforms the previous detection-based methods due to false positives and false negatives "canceling" each other, making the counting error very low. Thus, we investigate effects of the threshold T (as defined in § 3.1) on selecting detections from candidate local maximums, and report results for several values of T. Note that, by varying this hyperparameter, we are able to

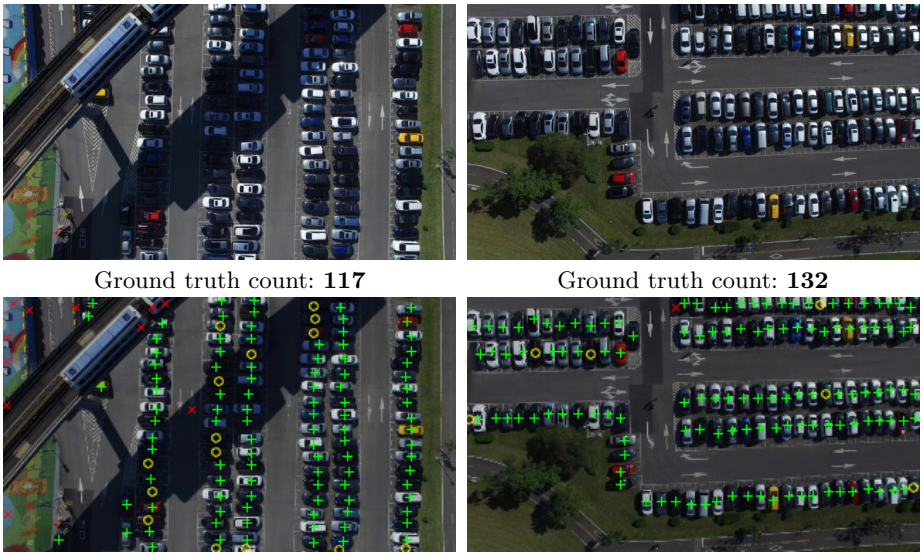


Fig. 6: Sample results on the CARPK dataset. Top row: original images. Bottom row: predicted detections. Correct detections are marked with a green '+', false positives with a red 'x', and missed detections with a yellow '○'. Many of the missed detections are dark cars in shadow, which upon inspection are difficult for even a human eye to discern.

explicitly control the precision-recall of our model. While calculating recall and precision, we consider a detection to be successful if it lies within 20 pixels (determined based on the mean car size) of the ground truth location. The recall reported for the region proposal methods in Table 4 is calculated by averaging across scores from using various IoU thresholds, as described in [16].

As shown in Table 4, the MAE is calculated with 5 randomly sampled exemplar patches per image, and the final counts are obtained by counting local maximums (detection-based counting). Note here, the exemplar patches are sampled from images in the training set, and different exemplar patches have negligible effect on performance. We can see that even with a very high precision (model trained on the full dataset, with $T = 2.75$), our model can still outperform the previous state-of-the-art by a substantial margin (counting error: MAE=23.8 vs MAE=19.7). Further decreasing the threshold yields higher recall at the expense of precision, with our best model achieving a counting error of MAE=7.5.

3.5 Discussion

From our experiments, the following phenomena can be observed:

First, in contrast to previous work, where different architectures are designed for density estimation in scenarios with significant instance overlap (e.g. VGG synthetic cells) and for detection-based counting in scenarios with well-separated objects (e.g. HeLa cells and cars), the GMN has the flexibility to handle both scenarios. Based on the amount of instance overlap, the object counts can simply be obtained by taking either the integral in the former case, or the local maximum in the latter, or possibly even an ensemble of them [17].

Second, by training in a discriminative manner, the GMN is able to match instances beyond the simplistic level, making it more robust to large degrees of rotation and appearance variation than the baseline SSD-based template matching (as shown in Table 4).

Third, in the cases where training data is limited (11 images for HeLa cells, 3 for cars), the proposed model has consistently shown comparable or superior performance to the state-of-the-art methods, indicating the model's ease of adaptation, as well as verifying our observation that videos can be a natural data source for learning *self-similarity*.

4 Shanghaitech Crowd Counting

To further demonstrate the power and flexibility of counting-by-matching, we extend it to the Shanghaitech crowd counting dataset, which contains images of very large crowds of people from arbitrary camera perspectives, with individuals appearing at extremely varied scales due to perspective.

We carry out a preliminary implementation of our method on the Shanghaitech Part A crowd dataset. Inspired by the idea of crowd detection as repetitive textures [1], we conjecture that it is possible to ignore individual instances and match the statistics of patches instead; e.g. the statistics of patches with 10 people should be different from those with 20 people.

We take the following steps: (1) Using the ground truth dot annotations, we quantize 64×64 pixel patches into 10 different classes based on number of people, e.g. one class will be 0 people, another 5 people, etc. (See Figure 7 for an example.) (2) Following the idea of counting-by-matching, we train the self-similarity architecture to embed the patches based on the number of people, i.e. if patches are sampled from the same class, the model must predict 1, otherwise 0. (3) We run the model on the test set using a sample of each class from the training set as the exemplar patch, with the final classification made by the maximum response.

Compared to other models that are specifically designed to count human crowds (e.g. CNNs with multiple branches), we aim for a method with the potential for low-shot category-agnostic counting. Our preliminary experiments show the possibility of scaling the counting-by-matching idea to human crowd datasets.

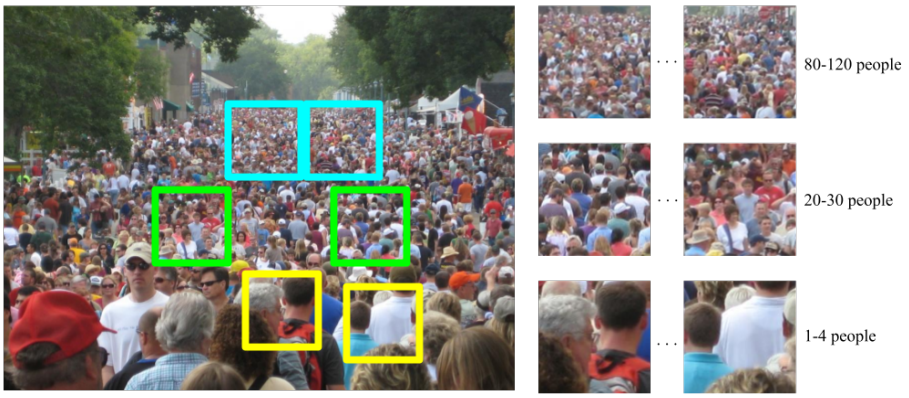


Fig. 7: Example of how different patches are classified. Patches marked by the same color square belong to the same density class. As can be seen, the significant textural differences between the classes enables a network to learn to classify different densities.

Method	MAE	RMSE
†Zhang et al. [35]	181.8	277.7
MCNN-CCR [37]	245.0	336.1
MCNN [37]	110.2	173.2
ic-CNN [27]	**69.8**	**117.3**
Ours	95.8	133.3

Table 5: Preliminary results on Shanghaitech Part A (lower is better). Patches chosen based on validation set performance. The "†" result is from paper [37].

5 Conclusion

In this work, we recast counting as a matching problem, which offers several advantages over traditional counting methods. Namely, we make use of object detection video data that has not yet been utilized by the counting community, and we create a model that can flexibly adapt to various domains, which is a form of few-shot learning. We hope this unconventional structuring of the counting problem encourages further work towards an all-purpose counting model.

Several extensions are possible for future works: *first*, it would be interesting to consider counting in video sequences, rather than individual images or frames. Here the tracking analogue takes on an even greater significance as a counting model can take advantage of both within-frame and between-frame similarities, *second*, a carefully engineered scale-invariant network with more sophisticated feature fusion than the GMN could potentially improve the current results.

Acknowledgements

Funding for this research is provided by the Oxford-Google DeepMind Graduate Scholarship, and by the EPSRC Programme Grant Seebibyte EP/M013774/1.

References

1. Arandjelovic, O.: Crowd detection from still images. In: Proc. BMVC. (2008)
2. Arteta, C., Lempitsky, V., Noble, J.A., Zisserman, A.: Interactive object counting. In: Proc. ECCV (2014)
3. Arteta, C., Lempitsky, V., Noble, J.A., Zisserman, A.: Detecting overlapping instances in microscopy images using extremal region trees. Medical Image Analysis (2015)
4. Arteta, C., Lempitsky, V., Zisserman, A.: Counting in the wild. In: Proc. ECCV (2016)
5. Barinova, O., Lempitsky, V., Kohli, P.: On the detection of multiple object instances using Hough transforms. In: Proc. CVPR (2010)
6. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.S.: Fully-convolutional siamese networks for object tracking. In: Workshop on Visual Object Tracking, ECCV (2016)
7. Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. In: Proc. CVPR. pp. 60–65 (2005)
8. Cho, S., Chow, T., Leung, C.: A neural-based crowd estimation by hybrid global learning algorithm. In: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) (2009)
9. Dekel, T., Oron, S., Rubinstein, M., Avidan, S., Freeman, W.: Best-buddies similarity for robust template matching. In: Proc. CVPR (2015)
10. Desai, C., Ramanan, D., Fowlkes, C.: Discriminative models for multi-class object layout. In: Proc. ICCV (2009)
11. Efros, A., Leung, T.: Texture synthesis by non-parametric sampling. In: Proc. ICCV. pp. 1039–1046 (Sep 1999)
12. Fiaschi, L., Nair, R., Köethe, U., Hamprecht, F.: Learning to count with regression forest and structured labels. In: Proc. ICPR (2012)
13. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: Proc. ICCV (2009)
14. Han, X., Leung, T., Jia, Y., Sukthankar, R., Berg, A.: Matchnet: Unifying feature and metric learning for patch-based matching. In: Proc. CVPR (2015)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. CVPR (2016)
16. Hsieh, M., Lin, Y., Hsu, W.: Drone-based object counting by spatially regularized regional proposal networks. In: Proc. ICCV (2017)
17. Idrees, H., Tayyab, M., Athrey, K., Zhang, D., Al-Máadeed, S., Rajpoot, N., Shah, M.: Composition loss for counting, density map estimation and localization in dense crowds. In: Proc. ECCV (2018)
18. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: ICML 2015 Deep Learning Workshop (2015) (2015)
19. Kong, D., Gray, D., Tao, H.: A viewpoint invariant approach for crowd counting. In: Proc. ICPR. vol. 3, pp. 1187–1190. IEEE (2006)

20. Lehmussola, A., Ruusuvuori, P., Selinummi, J., Huttunen, H., Yli-Harja, O.: Computational framework for simulating fluorescence microscope images with cell populations. IEEE Transactions on Medical Imaging (2007)
21. Lempitsky, V., Zisserman, A.: Learning to count objects in images. In: NIPS (2010)
22. Leung, T., Malik, J.: Detecting, localizing and grouping repeated scene elements from an image. In: Proc. ECCV (1996)
23. Malisiewicz, T., Gupta, A., Efros, A.A.: Ensemble of exemplar-SVMs for object detection and beyond. In: Proc. ICCV (2011)
24. Marana, A., Velastin, S., Costa, L., Lotufo, R.: Estimation of crowd density using image processing. In: Image Processing for Security Applications. pp. 11–1 (1997)
25. Marsden, M., McGuinness, K., S., L., Keogh, C.E., O'Connor, N.E.: People, penguins and petri dishes: Adapting object counting models to new visual domains and object types without forgetting. In: Proc. CVPR (2018)
26. Mundhenk, T.N., Konjevod, G., Sakla, W.A., Boakye, K.: A large contextual dataset for classification, detection and counting of cars with deep learning. In: Proc. ECCV (2014)
27. Ranjan, V., Le, H., Hoai, M.: Iterative crowd counting. In: Proc. ECCV (2018)
28. Rebuffi, S.A., Bilen, H., Vedaldi, A.: Efficient parametrization of multi-domain deep neural networks. In: Proc. CVPR (2018)
29. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proc. CVPR (2016)
30. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS (2016)
31. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, S., Karpathy, A., Khosla, A., Bernstein, M., Berg, A., Li, F.: Imagenet large scale visual recognition challenge. IJCV (2015)
32. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H.S., Hospedales, T.: Learning to compare: Relation network for few-shot learning. In: Proc. CVPR (2018)
33. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: NIPS (2016)
34. Xie, W., Noble, J.A., Zisserman, A.: Microscopy cell counting with fully convolutional regression networks. In: MICCAI 1st Workshop on Deep Learning in Medical Image Analysis (2015)
35. Zhang, C., Li, X., Wang, X., Yang, X.: Cross-scene crowd counting via deep convolutional neural networks. In: Proc. CVPR (2015)
36. Zhang, C., Yarkony, J., Hamprecht, F.A.: Cell detection and segmentation using correlation clustering. In: MICCAI (2014)
37. Zhang, Y., Zhou, D., Chen, S., Gao, S., Ma, Y.: Single-image crowd counting via multi-column convolutional neural network. In: Proc. CVPR (2016)