Counting People In The Street From Video

Emil Korzeń
Department of Data Science and Knowledge Engineering
Maastricht University
Maastricht, The Netherlands

***Abstract*—The thesis aims to determine and compare the performance of different crowd counting methods, both shallow and deep on different kinds of crowds. In order to do that, the thesis investigates two approaches, a shallow approach based on predicting human count using foreground pixel counting, and a deep learning approach that utilizes EfficientDet object detection model to detect and count humans. The pixel counting approach shows promising results on PETS2009 crowd dataset achieving mean average error between predicted and real human count of 3.35 on a video of a dense crowd featuring up to 35 people. We also discover that while fine-tuning a pre-trained object detection model can improve its detection ability, human detection based on detecting whole body performs poorly on dense crowds featuring occlusion.**

***Index Terms*—computer vision, object detection**

I. Introduction

The prevalence of surveillance cameras has resulted in a great amount of videos of crowds, which can be analyzed to extract motion patterns, behaviors, events and to count the number of people in them. Counting people can be used in multiple scenarios, including: tracking amount of customers in a business, video surveillance, counting event attendance.

Recent Survey on Crowd Density Estimation and Counting for Visual Surveillance [1] categorizes two approaches to counting people: a direct approach (detection based) and an indirect approach (feature based). Direct approaches are further classified into model-based analysis and trajectory-clustering based analysis, while indirect approaches are classified into pixel-based analysis, texture-based analysis and corner point based analysis. In addition to these, recent state-of-the-art approaches such as Adaptive Dilated Network With Self-Correction Supervision for Counting [2] and Perspective-Guided Convolution Networks for Crowd Counting [3] achieve the best counting results on various crowd datasets using neural networks to generate density maps and count people based on them.

This thesis was prepared in partial fulfilment of the requirements for the Degree of Bachelor of Science in Data Science and Knowledge Engineering, Maastricht University. Supervisor(s): Alexia Briassouli

The thesis aims to determine and compare the performance of different crowd counting methods, both shallow and deep on different kinds of crowds. We investigate two approaches that offer a possibility of counting people in close to real time:

- the model-based analysis, the purpose of which is to detect every person in the crowd, using techniques such as monolithic detection, where the classifier is trained using images featuring a whole person, or part-based detection, where the classifier is trained to recognize a body part or a combination of body parts (i.e. head and shoulders);
- the pixel-based analysis approaches, which are based on counting foreground pixels of an image and using them to obtain people counts.

With the help of these methods, we will answer following research questions:

- How to apply object detection to detect and count humans in a crowd?
- Which deep learning classification methods are best suited for counting humans in a crowd?
- How can crowd density be approximated, to approximate the number of people in it?
- What are the advantages of traditional vs deep learning based methods for crowd counting?

The rest of the paper is divided into five sections. In the method section, the data and methodology used to count the people are explained. Experiments section is dedicated to explaining setup of the experiments and how different methods are going to be evaluated. In the following section, the results of the experiments are presented. The discussion section focuses on interpretation of the results and compares them with results from other approaches. In the conclusion section, inferences are drawn from the research and the research questions are answered.

II. METHODS

A. Data

The data used for this thesis are from PETS2009 crowd dataset [4]. The data are a collection of video frames recorded using eight static camera views of a path near University of Reading campus. The videos are recorded at 7 frames per second. The dataset features Training Data, containing sets:

- background, featuring background model with occasional moving people or objects;
- city center, featuring sparse crowd with irregular flow;
- regular flow, featuring crowds of medium and high density with regular flows.

The dataset also features test data, of which we used following sets:

- S1L1-13.57, featuring a dense crowd moving away from the camera view;
- S1L1-13.59, featuring a dense crowd moving towards the camera view;
- S2L1-14.06, featuring a sparse crowd moving irregularly.

For the purposes of this thesis, only data from camera 1- "View_001" were used. Sample image from View_001 can be seen in Fig. 1.

Fig. 1. Sample image of View_001 from PETS2009 dataset

Ground truth annotations used for the test data were provided by Milan et al. in [5]. These annotations describe bounding boxes around all pedestrians in the scene, even the fully obstructed people. For the training data, we manually placed bounding boxes around fully visible pedestrians in a selection of frames from all training sets.

B. Density estimation

To count people using density estimation, we propose an algorithm based on pixel-counting. It operates by performing following steps on an input video:

1) Motion detection: First, background subtraction is performed on the input frame using a Gaussian Mixture-based Background/Foreground Segmentation Algorithm [6] [7] available in OpenCV library [8] as *BackgroundSubtractorMOG2*. Thresholding is then performed to get rid of pixels detected as shadows by the aforementioned algorithm. After foreground is extracted, edges are detected using Canny Edge Detection Algorithm [9]. Resulting image containing detected edges is then flood filled using OpenCV *floodFill* algorithm. The image is then inverted, with the final result being a binary image containing pixels the density algorithm considers to be in motion. Representation of image at selected steps of motion detection can be seen in Fig. 2.

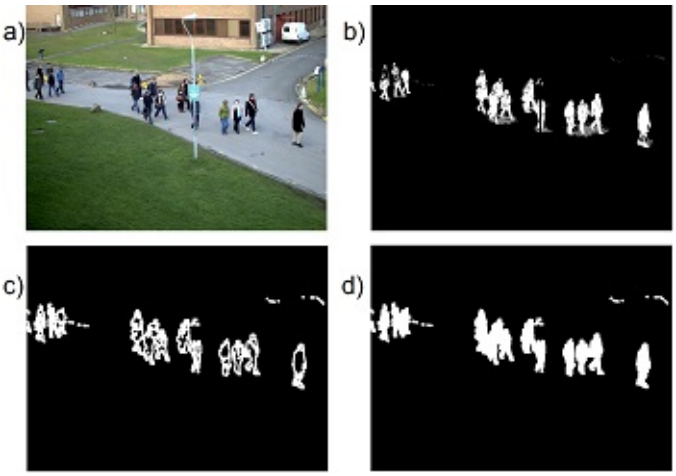

Fig. 2. Input frame at selected steps of motion detection, a) original frame; b) frame after background subtraction; c) frame after edge detection; d) frame after filling in the edges.

2) Perspective normalization: Because of objects closer to camera appearing larger than these further away, perspective normalization needs to be performed. We use an approach inspired by Chan et al. [10]. The pixels of the image are weighted according to a perspective normalization map. The weight is based on expected depth of the location of each pixel. We approximate the perspective map by comparing height of a reference person at a few extreme points of interest in the image (as seen in Fig. 3). The heights $h_1, h_2 ... h_n$ are measured in pixels, then pixels at an area where the person has the smallest height are given weight 1. The weights of pixels in areas at other reference points are calculated

using $\frac{h_1}{h_n}$, where h_1 is the height of a reference person with smallest counted height and h_n is the height of a reference person at the point of interest the weight is calculated for. The remaining weights are calculated using cubic interpolation between the reference points. Pixels not in the cubic interpolation area are assigned the average of the weights of the column they are in.

The resulting perspective normalization map of View_001 of PETS2009 dataset can be seen in Fig. 4. The binary image containing foreground pixels from previous step of algorithm is multiplied by pixel weight values in the perspective normalization map. The resulting values are then summed together to obtain total pixel value for the input frame.

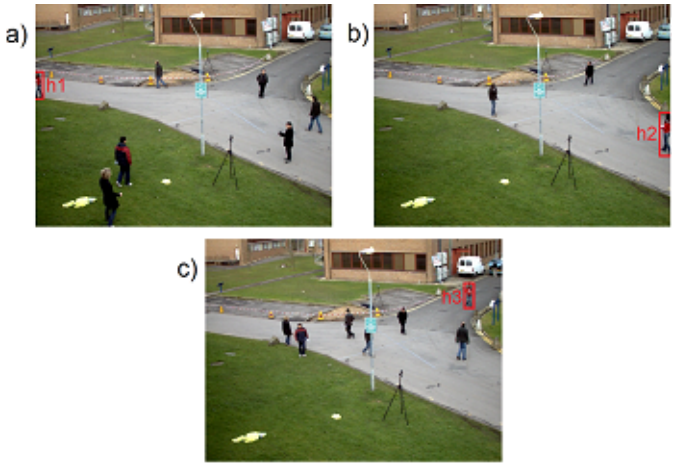

Fig. 3. View_001 of PETS2009, a) reference person at the left of the view; b) reference person at the right of the view; c)reference person at the top right of the view.

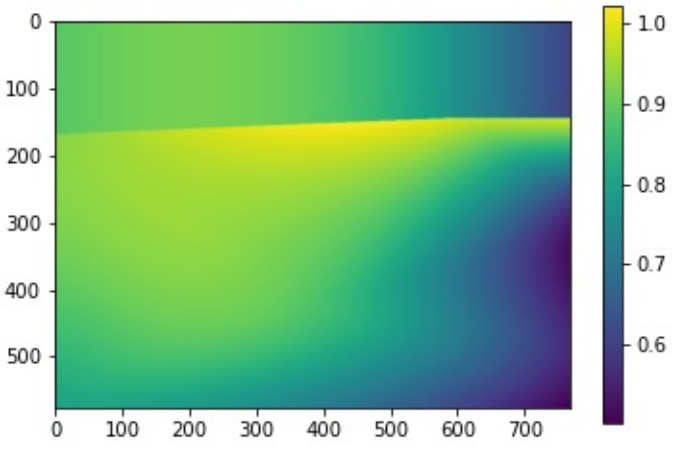

Fig. 4. Perspective normalization map of View_001 of PETS2009 dataset.

3) Linear regression: To obtain the final count of people using total pixel value, supervised learning is performed. In case of View_001, a sample video from the training dataset was given to algorithm to obtain total pixel value for each frame of the video. The amount of people in each frame was counted manually, then linear regression model was trained using these values. For all subsequent runs of the algorithm, the model is used to predict amount of people in each frame based on the total pixel value.

The complete flow of the people count using density estimation algorithm can be seen visualized in Fig. 8

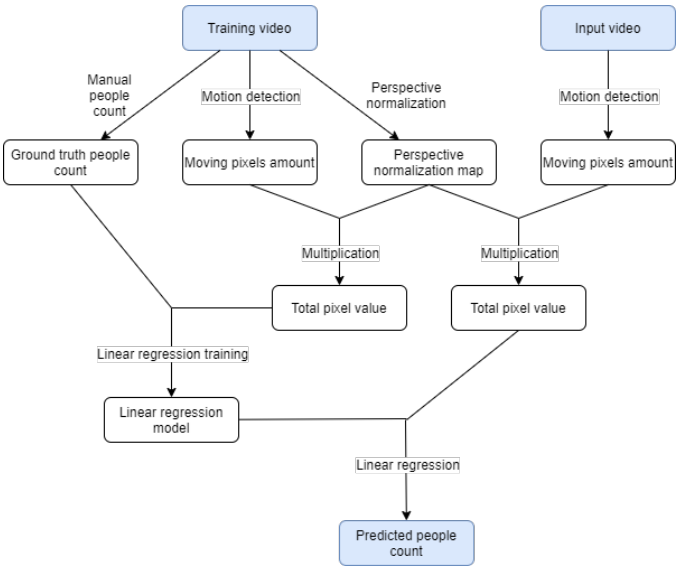

Fig. 5. Flowchart of the people count using density estimation algorithm.

C. Object detection

We use an EfficientDet [11] object detector model from TensorFlow [12] library in order to count people using object detection. The model family features 8 models, starting at 512x512 input image size with EfficientDet D0, and ending with 1536x1536 input image size with EfficientDet D7. The EfficientDet D7 model is a state of the art model achieving 55.1 average precision on COCO [13] *test-dev* dataset. While the D7 model is the most accurate out of the family, we use D0 model since it is the fastest and least computation-heavy variant that still achieves a respectable 34.6 average precision on the same dataset.

1) Backbone: The model uses EfficientNet [14] pre-trained on the ImageNet [15] as the backbone. The EfficientNet backbone is based on scaling of the convolutional neural networks using compound scaling, which

scales depth (adding/removing network layers), width (increasing/decreasing amount of neurons in a layer) and resolution (changing the size of input image) uniformly at the same time with a fixed ratio. The baseline network EfficientNet-B0 is found using multi-objective neural architecture search, then scaled up to obtain a family of EfficientNet models. The EfficientDet D0 model uses EfficientNet-B0 as the backbone network.

2) BiFPN: EfficientDet improves the conventional Feature Pyramid Network approach to feature extraction by adding an extra bottom-up path aggregation network, removing nodes that have only one input edge, adding extra edge from original input nodes to output nodes on the same level, and repeating these layer blocks multiple times. Feature fusion is improved by not treating all input features equally, and instead adding additional weight for each input. Additionally, the network is allowed to learn the importance of each feature. Further explanation and figures explaining structure of efficient bidirectional cross-scale connections and weighted feature fusion (BiFPN) can be found in EfficientDet: Scalable and Efficient Object Detection [11].

3) Training: The default EfficientDet D0 found in the TensorFlow2 Detection Model Zoo is pre-trained on the Microsoft COCO dataset and detects 90 categories of objects. Since person is one of categories it is trained to recognize, it is fine as a baseline model for the purposes of detecting people in a crowd.

The detector is monolithic and trained to detect humans as a whole, separate entities, so some problems might arise with the detection accuracy when the persons to be detected are gathered close to each other or are partially occluded. The speed at which input images are processed might also be slower than intended, given that the model still detects objects of 89 other classes it is trained to detect.

In order to work around these flaws, we have fine-tuned the EfficientDet D0 from the provided checkpoint. Since no ground truth data is provided with the training set of PETS2009 data, we manually annotated a selection of images from the training data to use for fine-tuning the model. The annotated ground truth includes bounding boxes only for fully and almost-fully visible humans. The selected images were mostly chosen from regular flow and city center sets, with only a few images from the background set. Only images from View001 of the data were used. There was no specific heuristic to choosing the images, roughly every 20th frame of regular flow set and every 40th frame of city center set were chosen, resulting in final count of 69 training images. In order to not overfit the object detector model by using such a small amount of pictures, training was performed for 3000 steps with a batch size of 4. The model was trained only to detect instances of persons. At the end of training, total loss value of 0.2557 was achieved.

Sample detections using fine-tuned and default EfficientDet D0 can be seen in Fig. 6 and Fig. 7 respectively. It can be seen that the default model detects a truck in addition to detecting humans.

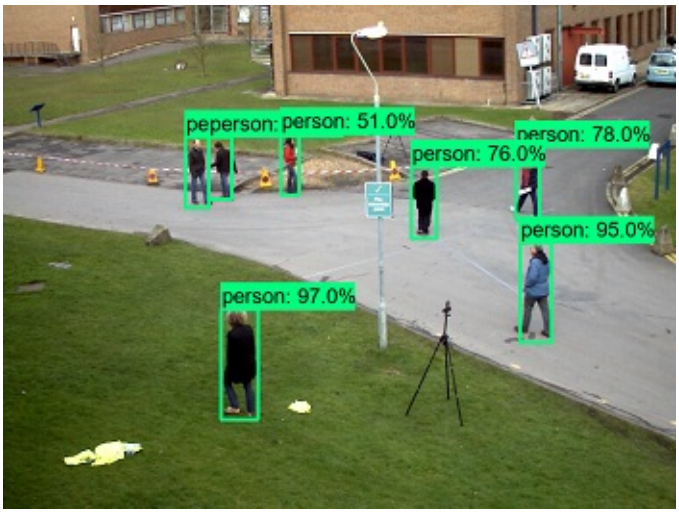

Fig. 6. Sample detection using fine-tuned EfficientDet D0

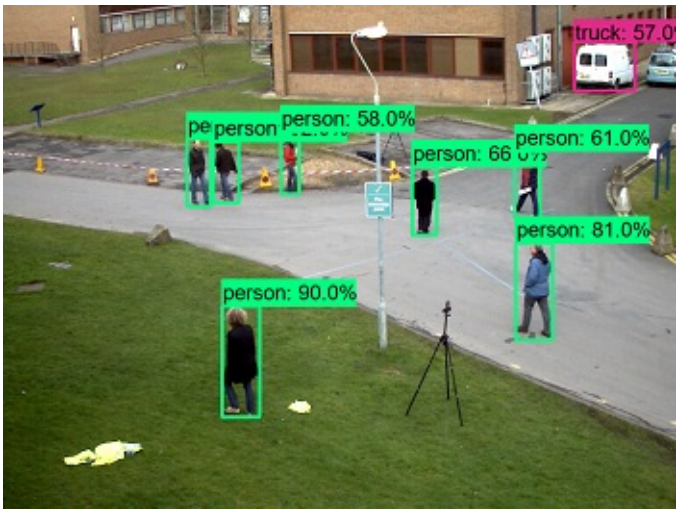

Fig. 7. Sample detection using default EfficientDet D0

III. Experiments

To evaluate the performance of chosen methods, a comparison between real and predicted people counts need to be made on videos of different kinds of crowds.

In addition to that, average precision of object detection models can be calculated by comparing the bounding boxes around persons the models detect to the ground truth boxes. In order to measure measure efficiency of the counting methods, the time taken to process a frame needs to be taken into account too.

A. Setup

In order to measure aforementioned parameters, the following inputs have been used:

1) S1L1-13.57: this video contains 201 frames from S1L1-13.57 set of test data from PETS2009. In this video, a dense crowd reaching up to 35 people moves along the street towards the left side of the view. Most of the people in this video are not facing the camera. People in this video obstruct each other in the camera view, a portion of people is not fully visible.

2) S1L1-13.59: this video contains 201 frames from S1L1-13.59 set of test data from PETS2009. In this video, a crowd of medium density reaching up to 23 people moves along the street towards the right side of the view. Majority of the people are facing towards the camera. Some of the people in this video are obstructed, but to a lesser extent compared to S1L1-13.57 crowd.

3) S2L1-12.34: this video contains 201 frames from S2L1-12.34 set of test data from PETS2009. In this video, a sparse crowd of up to 8 people moves irregularly in the camera view. Only minor obstructions occur and most of the people are fully visible.

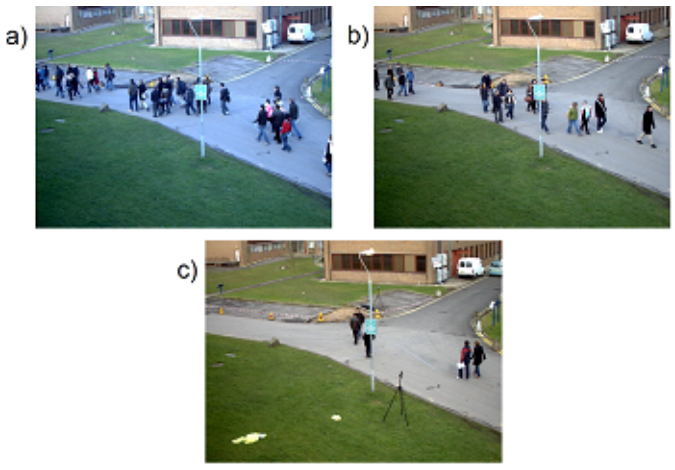

Fig. 8. Sample frame from a) S1L1-13.57 video; b) S1L1-13.59 video; c) S2L1-12.34 video.

Each of these videos has been given as an input to previously mentioned people counting methods:

- people count using density estimation algorithm, further referred to as 'density estimation';
- default EfficientDet D0 object detection model, further referred to as 'EfficientDet';
- fine-tuned EfficientDet D0 model, further referred to as 'FT-EfficientDet'.

For each of these methods, for each frame of the input videos, detected people count and time taken to process the frame in seconds have been measured. In case of object detectors, only people detected with above 50% confidence have been counted. In addition to these, the bounding boxes detected by object detection models have been compared to ground truth values using average precision evaluation metric, further explained in the next subsection. The experiments have been performed on a PC with Intel Core i5-6600K CPU.

B. Evaluation methods

To statistically compare predicted counts with real counts, mean absolute error (MAE) and rooted mean square error (MSE) have been calculated using respectively (1) and (2), where n is the total frame count, r_i is the real people count at ith frame, and p_i is the predicted people count at ith frame.

$$MAE = \frac{\sum_{i=1}^{n} |r_i - p_i|}{n} \tag{1}$$

$$MSE = \sqrt{\frac{\sum_{i=1}^{n} (r_i - p_i)^2}{n}} \tag{2}$$

To calculate average processing time per frame (ATPF), (3) has been used, where n is the total frame count and t_i is the time (in seconds) it took the method to process ith frame.

$$ATPF = \frac{\sum_{i=1}^{i} t_i}{n} \tag{3}$$

In order to evaluate performance of object detection models, Pascal VOC challenge [16] evaluation metrics have been used to compare ground truth bounding boxes (B_{GT}) to predicted bounding boxes (B_P). The detection metrics algorithm iterates through detections and tries to match B_P to B_{GT} with the highest Intersection over Union (IoU), which is the area of intersection between B_P and B_{GT} divided by the area of union of B_P and B_{GT}. If no ground truth can be matched to the detection with the IoU above 50%, the detection is classified as a false positive (FP). Otherwise, the detection is a true positive (TP). Using TP, FP, and total number of detections and ground truths, precision/recall curve is calculated. Using area under the curve, final measure of average precision for IoU above 50% (AP_{50}) is obtained.

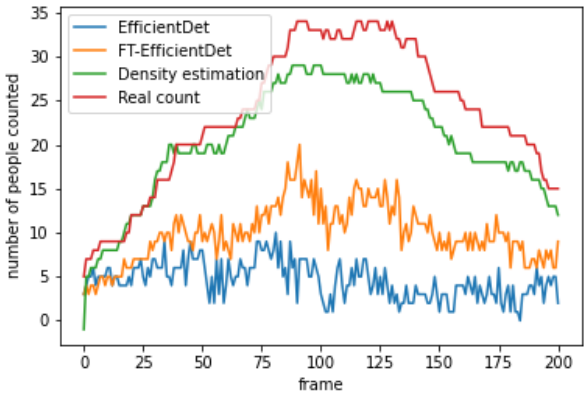

Fig. 9. Predicted people counts compared to real counts on S1L1-13.57 video.

IV. Results

A. People count

1) S1L1-13.57: Fig. 9 presents the results of Density estimation, EfficientDet and FT-Efficientdet predicting people counts compared to the real count on S1L1-13.57 video. Density estimation achieves count closest to the real one, but the predicted count is lower than the real count by a few people in most of the frames. EfficientDet achieves the lowest count, miscounting the number of persons in the frame up to difference of 32. FT-EfficientDet shows an improvement over EfficientDet, but still under-counting significantly.

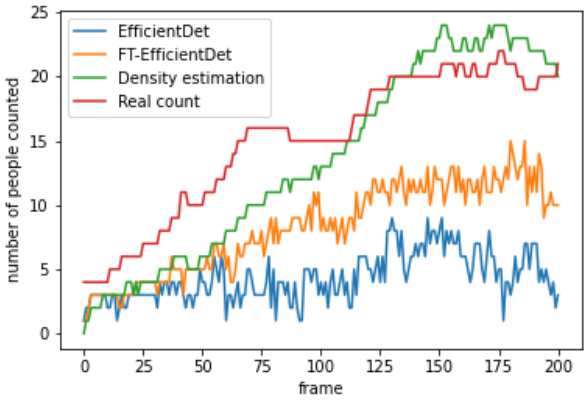

Fig. 10. Predicted people counts compared to real counts on S1L1-13.59 video.

2) S1L1-13.59: Fig. 10 presents the resulting counts of tested methods on the S1L1-13.59 video. In this case, density estimation still achieves a count that is closest to the real one. EfficientDet fails to count majority of people in the video, and FT-EfficientDet performs slightly better than it.

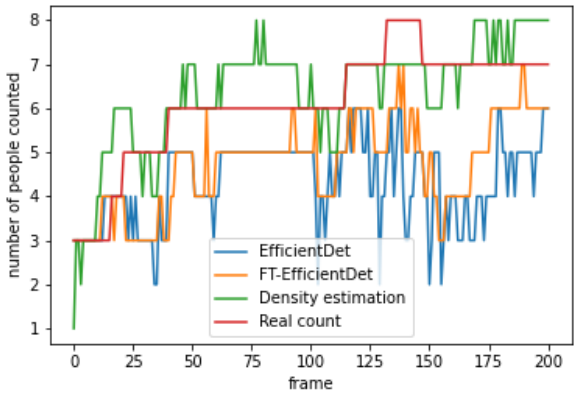

Fig. 11. Predicted people counts compared to real counts on S2L1-12.34 video.

3) S1L2-12.34: Fig. 11 presents the resulting counts of tested methods on the S1L2-12.34 video. Since there is not a lot of people in the video, the amount of detections in the video fluctuates rapidly between frames for each method, the count is not stable. Again, it is visible that FT-EfficientDet achieves a count closer to the real one compared to EfficientDet. FT-EfficientDet and Density estimation both achieve counts that are close to the real one.

B. Statistical analysis

TABLE I
MEAN ABSOLUTE ERROR AND MEAN SQUARE ERROR ON PEOPLE COUNTS ACHIEVED USING TESTED METHODS

Method	**S1L1-13.57**		**S1L1-13.59**		**S2L1-12.34**	
	MAE	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**
Density estimation	3.35	3.95	2.75	3.18	0.63	0.86
EfficientDet	18.89	20.65	10.61	11.55	1.84	2.18
FT-EfficientDet	13.50	14.59	6.69	7.16	1.44	1.69

Table I presents the MAE and MSE for each method and video. Looking at these error values, one can see that density estimation achieves the lowest mean error values for each video. EfficientDet achieves high error values for the dense and medium density crowd, but performs better with the sparse crowd. The same can be said about FT-EfficientDet, although it achieves lower error values across all videos compared the EfficientDet.

C. Average precision

Table II highlights AP_{50} values for each video for the object detection methods. The default EfficientDet

TABLE II
AVERAGE PRECISION OF TESTED OBJECT DETECTION MODELS

Method	**S1L1-13.57**	**S1L1-13.59**	**S2L1-12.34**
	AP_{50}	**AP_{50}**	**AP_{50}**
EfficientDet	0.746	0.798	0.924
FT-EfficientDet	0.735	0.744	0.913

achieves higher accuracy across all videos compared to FT-EfficientDet. Both models achieve relative worst accuracy with the densest video and best accuracy with the video featuring the sparse crowd.

D. Average time per frame

TABLE III
AVERAGE TIME TO PROCESS A VIDEO FRAME USING TESTED METHODS

Method	**S1L1-13.57**	**S1L1-13.59**	**S2L1-12.34**
	ATPF	**ATPF**	**ATPF**
Density estimation	0.03s	0.03s	0.03s
EfficientDet	0.52s	0.77s	0.72s
FT-EfficientDet	0.23s	0.47s	0.44s

Table III shows average time to process a video frame using each of the methods. Density estimation is the fastest method, taking 0.03 seconds on average to process a frame of each of input videos. EfficientDet takes on average time between 0.52s and 0.77s to process a video frame. FT-EfficientDet performs slightly faster, achieving on average between 0.23s and 0.47s time to process a video frame. For object detectors, average time per frame fluctuates between input videos.

V. DISCUSSION

A. Density estimation

The results obtained by people count using density estimation algorithm show that it is an algorithm that is sufficient for counting people in scenarios that do not require absolute precision, such as counting pedestrians on a street. Mean average error is under 1 for tested sparse crowd and around 3 for tested denser crowds, which can be considered an acceptable error for such a task.

The algorithm takes 0.03s to process a video frame, so it can be used for counting people from video feeds of up to 30 frames per second as is. Since count for each frame is not needed for tasks such as counting people in the street, the algorithm could be given e.g. only one frame per second of a video to accommodate for different hardware setup and videos with higher frame per second count.

Considering that the algorithm requires creation of perspective normalization map and manual people counts for supervised learning, it is not easily usable for new camera locations. The camera view needs to remain static, which might not be desirable in some pedestrian surveillance cases.

The algorithm detects all motion, so it is suited for areas with only pedestrian traffic. Additional modifications such as not counting motion in part of the frame where other moving objects (e.g. cars) appear would be required to accurately count people in such areas. The algorithm is based on motion detection and while the used background subtraction method detects shadows, sudden changes in brightness of the whole scene might cause erroneous results until background model gets updated by the algorithm. Accuracy of the count can also decrease if people in the video are static for a long duration and become considered a part of the background by the algorithm.

PETS2009 and Winter-PETS 2009 Results: A Combined Evaluation [17] compares people counts achieved on the dataset using various methods. Although methods presented in our paper were not prepared for the PETS2009 challenge and slightly different evaluation metrics were used, a comparison can still be presented. Average frame error between real and predicted count for a region that encompasses most of the frame of View001 has been presented in the PETS2009 results for different methods. This metric is close to MAE we have used. Ellis et al. [17] acknowledge algorithm proposed by Albiol et al. [18] as top performer with average frame error of under 2 for S1L1-13.57 and S1L1-13.59 sets. Accurate value is not provided, as the results are only presented in a graph form. Our density estimation algorithm therefore performs slightly worse than the top performing algorithm, but stronger conclusions cannot be made given the aforementioned circumstances. PETS2009 evaluation does not evaluate counting people in sparse crowd from S2L1-12.34 set, so no comparison can be made there.

B. Object detection

The results delivered by experiments using object detection models show that while accurate people counts can be achieved for sparse crowd, accuracy of predictions fall off when denser crowds are involved. This happens because monolithic detection is performed, persons are recognized only when they are fully visible. This method

is therefore not suitable for the task of counting people in the street, but might be suitable for different scenarios involving counting people in only sparse crowds. Fine-tuning a model with training data involving people in crowds can improve the ability to count people in a crowd, but accurate counts in crowds cannot be achieved with monolithic detection. Other approaches, such as detecting humans based on parts of them (e.g. head) should be investigated instead to count people in dense crowds.

The models achieve good accuracy when comparing bounding boxes of detections to ground truth bounding boxes, but the accuracy fades with the increase of crowd density. While fine-tuning a model improves its ability to count people, bounding box accuracy decreases slightly in the process in our case.

An advantage to using object detection when counting people is that the models can be used on different kinds of input images with varying angles and locations of the cameras without additional training. While the accuracy might be worse in some cases depending on training of the models, this is still a considerable boon compared to our density estimation method.

On average, it takes on average between 0.52 and 0.77 seconds for EfficientDet D0 to process a frame of a video. Fine-tuned EfficientDet D0 takes on average between 0.23 and 0.47 seconds. The increase in speed of FT-EfficientDet might be coming from the fact that it only detects objects of 1 class as opposed to 90 classes of the default EfficientDet. The people count from video feed can be obtained in real time using these models by counting people using a single frame for every second of the video.

Comparing MAE of our object detection models with PETS2009 results [17] on S1L1-13.57 and S1L1-13.59 shows that the people counts are considerably worse than those of other methods. Since object detection accuracy was evaluated using different metrics and a different set of ground truths, no comparison can be made in regards to it.

VI. Conclusions

Based on our experiments, it can be seen that the proposed people count using density estimation method shows promising results for counting people in sparse and dense crowds.

The ability to count humans in a crowd of a pre-trained object detection model can be improved by fine-tuning, but the models based on fully body detection are not suited for counting humans in dense crowds, failing to achieve accurate results with occlusion present.

Crowd counting algorithms can help with public safety by assisting with identification of too dense crowds, where an issue of stampeding or disease spread might arise. Achieving accurate people counts can be also helpful for commercial purposes, where an assessment of number of people in an event such as a festival might be needed.

Object detection is applied to detect humans in a crowd by training deep learning object detection models such as EfficientDet using training data containing annotated human images. Count of people in a crowd can be then obtained using detections made by these models.

While it is possible to count humans using monolithic detection where humans are classified based on their whole body, a better count can most likely be achieved by classifying them using specific body parts, such as head or head and shoulders that might remain visible even when human is partially obscured in camera view.

Crowd density can be approximated to count number of people in it using both deep and shallow methods. Deep learning methods count people based on predicted density maps, while shallow methods count people based on low-level features. In case of our density estimation method, the number of humans is approximated using foreground pixels and edges.

Shallow crowd counting methods are faster in both training and processing input compared to deep methods. In case of methods investigated in this thesis, shallow methods handle partially obscured humans better. Deep learning methods are more flexible and it is easier to apply them when considering different camera angles and locations. They are also better at handling non-human objects which might appear in crowd counting scenarios.

Proposed methods could be tested on a wider variety of datasets in order to obtain better comparisons to different crowd counting methods. Further research is also needed in order to investigate part-based human detection.

References

[1] S. A. M. Saleh, S. A. Suandi, and H. Ibrahim, "Recent survey on crowd density estimation and counting for visual surveillance," *Engineering Applications of Artificial Intelligence*, vol. 41, 05 2015.

[2] S. Bai, Z. He, Y. Qiao, H. Hu, W. Wu, and J. Yan, "Adaptive dilated network with self-correction supervision for counting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[3] Z. Yan, Y. Yuan, W. Zuo, X. Tan, Y. Wang, S. Wen, and E. Ding, "Perspective-guided convolution networks for crowd counting," 10 2019, pp. 952–961.

[4] J. Ferryman and A. Shahrokni, "Pets2009: Dataset and challenge," in *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009, pp. 1–6.

[5] A. Milan and S. Roth, "An analytical formulation of global occlusion reasoning for multi-target tracking," 11 2011, pp. 1839–1846.

[6] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 2, 2004, pp. 28–31 Vol.2.

[7] Z. Zivkovic and F. Van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognition Letters*, vol. 27, pp. 773–780, 05 2006.

[8] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[9] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[10] A. Chan, J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," 06 2008.

[11] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," *arXiv e-prints*, p. arXiv:1911.09070, Nov. 2019.

[12] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[13] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common Objects in Context," *arXiv e-prints*, p. arXiv:1405.0312, May 2014.

[14] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *arXiv e-prints*, p. arXiv:1905.11946, May 2019.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[16] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[17] A. Ellis, A. Shahrokni, and J. M. Ferryman, "Pets2009 and winter-pets 2009 results: A combined evaluation," in *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009, pp. 1–8.

[18] A. A. J. M. M. Antonio Albiol, Maria Julia Silla, "Video analysis using corner motion statistics," *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, pp. 31–38, Jun. 2009.