

博士論文

CNN Based Robust Pedestrian Counting for
Helicopter Footage

(ヘリコプター画像を用いた深層学習ベースの
ロバストな歩行者カウント)

チョンデ ゲルゲイ
(Gergely Csoende)

CNN Based Robust Pedestrian Counting for Helicopter Footage

by

Gergely CSOENDE

Submitted to the Department of Civil Engineering
at The University of Tokyo in fulfillment of the requirements for
PhD in Civil Engineering

January 2021

Supervisor: Dr. Yoshihide SEKIMOTO

CNN Based Robust Pedestrian Counting for Helicopter Footage

by

Gergely CSOENDE

Abstract

The task of pedestrian counting has various useful applications. Owing to the rapid progression of computer vision technologies the achievable accuracy increased significantly in the past decade. Various convolutional neural network (CNN) based methods were developed, which use large amount of training data to learn the appropriate algorithm to solve the task. In my thesis I show that helicopter footage is useful for pedestrian counting. While one important goal in the field is to develop CNNs which can generalize well, still the resulting models are biased towards the training dataset. Typical datasets are from street level footage such as closed-circuit television (CCTV) cameras, or handheld devices, or from drones flying at low altitude. These provide higher resolution images of people; however, they have numerous problems. I show that using helicopter footage instead can help to alleviate these issues. Furthermore, I present experimental results of state-of-the-art CCN based pedestrian counting methods, both static and dynamic, to show that helicopter footage can be used to solve the task with comparable accuracy. Specifically, we have investigated object-detection, density map estimation (DME), and multi-object tracking (MOT) methods. For these experiments we introduced a new helicopter based urban dataset. In addition to its advantages helicopter footage carries some weaknesses as well. These are the issues of lack of deeper semantic information in DME methods and the strong sudden camera motions in MOT methods. I also present solution for these problems.

Our contribution is threefold. First, we have developed a large-scale helicopter based pedestrian dataset called Tokyo Hawkeye 2020 (TH2020). It comprises 6237 static images and six 20 seconds long video sequences. The dataset is publicly accessible at <https://github.com/sekilab/TokyoHawkeye2020>. Second, we have developed a static crowd counting method that combines density map estimation with semantic segmentation, which can reduce the error of static counting in helicopter footage. We have managed to achieve 0.2 relative MAE on the TH2020 dataset. Incidentally, we have achieved 0.79 AP@0.5IoU for object detection with an unaltered CenterNet architecture. Finally, we have developed a simple CMC functionality through 2D partial affine transformation for dynamic methods. It can be applied to multiple state-of-the-art MOT algorithms to increase accuracy on helicopter footage, especially in online use-cases. Furthermore, we introduced a concurrent pipelined implementation for the algorithms that can reduce execution time by up to 50%. The best MOT accuracy we have managed to achieve on TH2020

video sequences is 59.30 MOTA.

Acknowledgments

Throughout the writing of this dissertation I have received a great deal of support and assistance.

I would first like to thank my supervisor, Professor Yoshihide Sekimoto, whose expertise was invaluable in formulating the research questions and methodology.

I would also like to thank Assistant Professor Takehiro Kashiya for his continued professional support throughout the research.

I would also like to acknowledge the committee members, Professor Takashi Fuse, Associate Professor Muneyoshi Numada, Associate Professor Takeshi Oishi, and Professor Wataru Takeuchi, for their invaluable feedback on the research.

In addition, I would like to thank our laboratory secretaries, Ryoko Danjo, and Rieko Homma, without whose dedicated annotation work the research would not have been possible.

Contents

1	Introduction	13
2	Technological overview	19
2.1	Static pedestrian counting	19
2.1.1	Object-detection-based methods	19
2.1.2	Density-map-based methods	23
2.2	Dynamic pedestrian counting (Tracking)	25
2.3	Aerial semantic segmentation	29
2.4	Camera motion cancellation	29
2.5	Existing datasets	31
3	Introduced helicopter-based datasets	33
3.1	Datasets	33
3.2	Crowdsourcing	39
4	Assessment of existing methods on helicopter-based datasets	41
4.1	Pretrained models	41
4.2	Training on helicopter-based dataset	43
4.2.1	Object-detection based methods	43
4.2.2	Density-map based methods	45
4.2.3	Multi-object tracking methods	46
4.3	Weaknesses	47
4.3.1	False detections due to lack of deeper semantic information . .	47

4.3.2	Camera motion in tracking	48
5	Density map estimation with semantic segmentation	54
5.1	Generic Methodology	54
5.2	Simple masking with separate CNNs	54
5.3	DME segmentation multitask training	56
6	Multi-object tracking on helicopter footage with camera motion cancellation	59
6.1	Overview	59
6.2	Camera motion cancellation with 2D partial affine transformation . .	62
6.3	Efficiency in various online use-cases	63
6.4	Concurrent pipelined execution	64
7	Results	67
7.1	Object detection results	67
7.2	Density map estimation with semantic segmentation	68
7.2.1	Simple DME and segmentation methods	68
7.2.2	Simple masking with separate CNNs	69
7.2.3	DME segmentation multitask training	69
7.2.4	Masking efficiency	70
7.2.5	Training details	73
7.3	Multi-object tracking on helicopter footage with camera motion can- cellation	74
7.3.1	Used multi-object tracking methods	74
7.3.2	Efficiency of camera motion cancellation	75
7.3.3	Speed improvement with concurrent pipelined execution . . .	77
7.3.4	Implementation details	78
7.4	Result summary	79
8	Discussion	81
8.1	Correlation between density map quality and masking efficiency . . .	81

8.2	MTSM model parameters	82
8.3	Considerations about the dataset	83
8.4	Transfer capabilities	84
8.5	Assumptions regarding our CMC method	85
8.6	Relevance of the accuracy metrics	86
8.7	Unanswered questions	87
9	Conclusions	88

List of Figures

2-1	Illustration for the differences between street level footage and aerial footage. $(d + \Delta d)/d$ is much farther from 1.0 than $(d' + \Delta d')/d'$. Also, the dark gray area is obstructed from both cameras while the light gray area is only obstructed from the street level camera. Note that the image is not to scale.	32
3-1	Sample images from the TH2020 dataset.	35
3-2	Semantic segmentation of the background for helicopter footage. Humans are ignored.	37
3-3	Github page for the TH2020 dataset.	38
4-1	Sample image exhibiting the problem of "fake human" detection . It is clear that the billboard with humans on top of the building is producing false positive detections.	48

4-2	The image illustrates the difference between tracking close and far away targets with rotating camera. The green figure is the current position, and the dashed figure is the position resulting from camera rotation. The blue circle represents the gating distance, which is proportional to the target size. The left side shows a close target, while the right side show a further one. The target is standing at the same (x, y) image coordinates on both sides. The rotation angle and therefore the displacement is the same, however while on the left side the center point of the new position is inside the gating range, on the right side it is outside therefore it will get discarded, which makes the tracking less accurate.	52
5-1	Flow of region-based masking procedure.	55
5-2	Visual representation of difference between regular DME network and our proposed multitask segmentation method.	58
6-1	Simplified sequence diagram of our concurrently executed tracking algorithm. The exact queueing and blocking mechanisms are omitted for transparency.	66
7-1	Sample results for comparison across methods. The segmentation for the regular DM methods was generated by MTSM-CAN.	71
7-2	Correlation between DME quality and masking efficiency in case of high-quality segmentation mask.	72

7-3	The upper left and the upper right images are frames of the same video sequence from different times and camera viewpoints. The lower image depicts the result of our CMC method transforming the right image to the left one’s coordinate system and merging them together. The quality can be assessed from the overlapping region by checking the misalignment of stationary objects. The images are aligned over points that are few meters above the ground level therefore the taller something is the more misaligned its two views become.	75
7-4	MOTA results for the original FairMOT and our modified version with CMC evaluated on 30 FPS helicopter videos with regards to the number of skipped frames. Our method is superior in every case. . . .	76
7-5	MOTA results for the original JDE and our modified version with CMC evaluated on 30 FPS helicopter videos with regards to the number of skipped frames. Our method is superior in every case.	76
7-6	MOTA results for the original DeepSORT and our modified version with CMC evaluated on 30 FPS helicopter videos with regards to the number of skipped frames. Our method is superior in every case. . . .	77
7-7	Execution speed of the FairMOT algorithm on our target configuration with the original implementation and our concurrent one for MOT17 video sequences.	78
7-8	Execution speed of the FairMOT algorithm on our target configuration with the original implementation and our concurrent one for helicopter video sequences.	78

List of Tables

4.1	Comparison between models trained on other dataset and models trained on TH2019 when evaluated on TH2019 (we used only one class).	43
7.1	Comparison between object detection results for state-of-the-art detectors when trained and evaluated on TH2020 and MSCOCO. AP values are the main metric for comparison. Precision and recall values are at 0.25 confidence and 0.5 IoU and they are only for notification. Also note that MSCOCO results are only reported for 80 classes.	67
7.2	Summary of crowd density estimation accuracies for models trained on TH2020. The mIoUcolumn is only included to indicate the accuracy of the segmentation mask used.	69
7.3	Per-category IoUs and their mean for DeepLabv3 and our MTSM models trained on TH2020.	69
7.4	Valid-invalid IoUs and their means for DeepLabv3 and our MTSM models trained on TH2020.	70
7.5	L2/L1 norm ratio compared to change in accuracy after masking for investigated models.	72
7.6	The final best accuracies of pedestrian counting methods on helicopter footage compared to relevant reference values on benchmark datasets.	
	*Reference AP is reported for 80 classes, whereas our result is for 1.	79

Chapter 1

Introduction

In the past decade, the field of computer vision and image processing went under a rapid development thanks to the resurfacing of convolutional neural networks (CNNs). By applying these, many previously infeasible tasks suddenly became quite possible to solve. One such task is the counting of humans, however in this work we are interested in the counting of humans on the streets of an urban environment, therefore we further narrow this down to pedestrian counting. Following a broad perspective, we consider everything that yields the count or the spatial and temporal distribution of pedestrians in some visual material. These can be divided in to static and dynamic counting tasks depending on whether they are based on static images or a series of static images in temporal ordering (a video). In the former the task is to count or determine the spatial distribution in a single stationary image, while in the latter the goal is to reconstruct the trajectories of the individuals or the people flow over a given frame sequence.

Solving this task has many utilizations. It can be used in security monitoring. By monitoring regular daily traffic one can learn to detect disturbances. For example, if a large mass is moving fast in the same direction, that is not a normal everyday phenomenon, and it may indicate some danger. Furthermore, it gives some indication about the whereabouts of the danger because people are usually trying to get away from such situations.

Another application is traffic routing. While pedestrian traffic problems are much

less likely than vehicle related ones, in highly congested situations it can be beneficial to direct people to some other ways. Infrastructural objects are designed to bear much more load than what can be expected during normal use. But in extreme situations, for example evacuations, the increased load may put these objects to their limits. Such situations can be prevented if part of an undesirably large crowd is instructed to take a different evacuation route.

Disaster mitigation is also an excellent example. Sadly, earthquakes, tsunamis and other natural disasters usually come with a life toll. And only part of it originates from the event itself. Many deaths occur because of the aftermath of the disasters, therefore it is of the utmost importance to look for survivors and stranded people as fast and as efficiently as possible, because time is of the essence. Additionally, the number of stranded people also impacts the necessary rescue actions.

Infrastructural planning is a higher accuracy requirement application. When the parameters of an infrastructural object, such as a pedestrian bridge, are decided the potential traffic it will handle is considered. The size of this traffic can be estimated by counting the pedestrians in the vicinity, but to conform with civil engineering design standards this count must be within a certain error range with a preselected high level of confidence.

Various research fields such as human mobility also have use for such data. It can provide ground truth information to check the quality of traffic simulations.

By further pondering on this matter one may come up with additional utilizations. However not necessarily with honest intentions. It is already clear that big companies are collecting all manner of personal information purely for their own business gain, and people are giving those away willingly for some very insignificant discounts or just to get access to services. Geospatial location is also such sensitive information, and it is also an excellent source for spying, therefore researchers must consider protecting it from being exploited or used with malicious intentions.

I have listed several utilizations for pedestrian counting, however whether the counting accuracy is enough for a certain application is a tough question to answer since there are no clear requirements. Many deep-learning-based business endeavors

operate on a best effort basis and their success is evaluated simply by the subjective judgement of the customer without the need of satisfying any specific numerical constraints. One thing can be said for sure though, if a given application is making people’s life easier or it helps saving lives any accuracy is better than nothing. For instance, if the task is to look for survivors after a disaster and applying such methods can increase the chance of survival of the injured people then even very low accuracy can be useful. As for the rest, without specific requirements only rough estimates can be given.

A decade ago, most algorithms for solving this problem were handcrafted. Existing footage was only used for evaluating their accuracies. On the other hand, modern CNN based methods require training data to learn the proper algorithm to solve the problem. We see a great surge in the number of such datasets. The ones we can consider in pedestrian counting can be categorized by the altitude at which the footage was taken. Street level such as CCVT camera, low altitude from drones, and high altitude from airplanes. In case of the former two in order cover a meaningful area, the camera angle must be low. This causes strong perspective distortions and occlusions; most novel counting methods are trying to address these issues.

Another big problem with these is the privacy issues they present. In these types of footage people’s face is clearly recognizable. In the case of airplane footage, the described problems do not exist, but the price of this is the very low spatial resolution which makes the pedestrian counting very difficult if not impossible. Altitude-wise there is a gap between drones and airplanes at several hundred meters that can be filled with helicopters. Although technologically there are already unmanned aerial vehicles (UAVs) which can also do so, and in the future these might be widely used, but at the moment in many countries around the globe there are strict regulations regarding the use of drones making helicopters the only viable option for creating footage at the altitude in question. The reason for the lack of such footage is mostly financial. Creating street level footage is very cheap, while airplane footage has other uses which are deemed worthy of the high costs of flying. However currently

there are no worthy applications for helicopter footage that cannot use other cheaper means as a data source.

The purpose of my thesis is to show that helicopter footage and datasets created from it are in fact useful for the task of pedestrian counting. To do this first I must specify what useful means. For a dataset to be useful it must satisfy two conditions. Firstly, it must bring some novelty to the task at hand, or it must have some advantage compared to other datasets, that is there must be some reason to use it instead of other already existing data source. For instance, if there is a crowd counting CNN trained on a closed-circuit television (CCTV) footage, and there is another CCTV footage on which the CNN performs well, there is no need for the second CCTV footage. On the other hand, if the first footage is in bright sunny weather and the second is in dark rainy the accuracy might be worse therefore the second footage brings novel data to the task.

Secondly, the task must be feasible using the novel dataset or its feasibility must be like when using other already existing data source, that is it must yield comparable accuracy for the task. For instance, one may argue that satellite imagery is good for pedestrian counting because it covers large areas, and it is possible to count the population of a whole city with just one image. However, because of the spatial resolution the task is infeasible. If the execution of the task meets obstacles with the novel dataset it must be shown that those obstacles can be passed.

To answer the first part, I present a comparison between already existing footage and helicopter videos. I point out that due to the high altitude and steep camera angle perspective distortions on the footage are negligible, and occlusions pose much less of a problem than in street level footage or drone footage. Additionally, helicopter footage can cover much larger area in a single image, and most importantly it does not raise privacy issues because people cannot be recognized. The advantage compared to airplane footage is that from such altitude features are still recognizable, therefore the task can be done with higher accuracy although some degradation can be expected when compared to low altitude footage. I also explain what difference the use of this footage type means for the individual applications. And lastly, I

present a new helicopter-based image dataset called Tokyo Hawkeye 2020 (TH2020) we introduced in [18].

I discuss the second part of usefulness by investigating methods that can be used for counting. While the output of these is not a count, their results can be directly used to calculate count, and the counting accuracy depends on the accuracy of these methods, therefore we investigate these. I discuss static and dynamic methods separately. The static methods can be further separated into direct and indirect methods. Direct methods are object detection solutions, meaning that bounding boxes are predicted for the targets, and the total count can be calculated by counting the boxes. Indirect methods predict a crowd density map, and the total count can be calculated by integrating over the whole map. I present experimental results for the accuracy on helicopter footage with both pretrained models and models trained on TH2020. The result from pretrained models show that when trained on other dataset, static counting methods are not useful on helicopter dataset further confirming the need for such data. As for methods trained on TH2020 the resulting accuracy is comparable to that of benchmark datasets.

During these experiments we have identified one issue with static methods, that is the lack of deeper semantic understanding. This means that everything that looks like a human is identified as a human including inanimate objects like statues, dummies, or billboards picturing humans. In addition to this there are people present in the urban scenery who are not participants of the traffic (people on rooftops, balconies, etc.) who should also be differentiated from pedestrians. In the case of direct methods this can be countered by multiclass classification, however it is not so simple in the case of indirect methods. In the thesis I present a solution where we used semantic segmentation to mask out entities that are falsely detected as pedestrians.

Regarding the dynamic methods, I present experiments with state-of-the art multi-object tracking (MOT) algorithms. Since these methods are based on static object detectors their accuracy can only be as good as the underlying detector CNN. This means that pretrained models are not useful on helicopter footage in this case

either. Just like in the case of static counting the accuracy of tracking methods is comparable to when trained and tested on benchmark datasets.

We have identified a weakness with CNN based tracking methods also, specifically because of the nature of helicopters the video exhibits strong camera movement, much stronger than for example CCTV cameras mounted on building walls. The tracking algorithms can handle these motions very well, especially when the frame rate of the input is high. Unfortunately, whether these algorithms can run real-time depends on many things, including the resolution of the footage, the number of targets on screen, and of course the parameters of the computer running the algorithm. Because of this to achieve online execution, frames must be skipped often, and that is when strong camera motion starts to become a problem. I present a simple method to cancel this motion and significantly increase the accuracy of tracking in a scenario where multiple frames are skipped from the data source. This way the accuracy can be kept on a comparable level. Additionally, our algorithm improves execution speed by up to 100% in realistic situations by better utilization of computational resources through concurrent pipelined execution.

To sum it up in my thesis I show that helicopter footage is useful in the task pedestrian counting by comparing it to other types of data sources. In addition to this I present training results from static and dynamic methods on a new helicopter dataset called TH2020 with comparable accuracy which shows that this type of footage is adequate for pedestrian counting. Moreover, I present modified, and improved algorithms which make the methods more resilient to problems identified during the experiments.

In chapter 2 I overview other researcher's work on which I based my thesis. Chapter 3 introduces our dataset. In chapter 4 I go over our preliminary experiments with existing pedestrian counting methods and the issues we have identified related to medium altitude helicopter footage. Chapters 5 and 6 are about how we have solved these problems by improving the existing algorithms. I present the results of our solutions in chapter 7, as well as summarized results for pedestrian counting methods on our helicopter dataset.

Chapter 2

Technological overview

2.1 Static pedestrian counting

As already mentioned, I investigate both static and dynamic counting methods. In this section I review the work related to static counting. This is fundamentally divided into two types, direct and indirect methods. The former is about detecting each individual target. The result can be obtained by counting the people one-by-one. In this case geometric information such as central coordinates or bounding boxes are also acquired for every detected person. In the case of the indirect methods instead of the count or the individual geometric data, first some kind of more abstract data is produced. Then the count can be derived from this data. There is a third approach, simple regression, where there is only one output, the estimated number, but this method did not prove to be competitive compared to the others, so it has disappeared from the field.

2.1.1 Object-detection-based methods

The direct methods are originating from the quite simple task of image classification. In this situation the input of the network is a single image, and the output is a vector of confidence scores for several predefined categories, therefore they are not strictly for human recognition but for general purpose. There are already countless

solutions available for the task; one prominent example is the VGG network [86] where the authors were investigating the potential effects of using very deep CNNs. The VGG-16 variant is used as a starting point for a great many CNNs.

Other later developed popular networks are ResNet [37] and Inception [92] with several versions. The former enhances training speed with its residual or skip connections while the latter consists of smaller building blocks. Every smaller block reduces dimensionality with 1×1 convolution layers and this allows the creation of very deep networks while keeping the parameter count on a manageable level. This "network of networks" structure resembles the concept of "dream within a dream" from the infamous movie, Inception, hence the naming.

DenseNet [39] takes these ideas one step further with an incrementally growing network structure divided into blocks. Inside these blocks every layer starting from the first one is directly connected to the block output as well as the consequent layer and every block is directly connected to next block as well as the final output.

Following and parallel to this was the development of object detection networks which aimed to solve both localization and classification for multiple targets simultaneously on the same image by outputting a list of bounding boxes and corresponding categories. Before CNNs got widespread in object detection tasks researchers were using classic image processing methods with handcrafted algorithms to crack this very tough nut. Typical solutions combine image segmentation and feature extraction methods to produce a feature vector which is then used to make predictions with a classifier such as support vector machine (SVM) or decision tree.

One of the first methods which utilized CNNs for object detection was the region-based-CNN (R-CNN) [31]. It is a three-stage detector as explained before. The novelty of the method is that it replaces feature extraction step with a CNN. In the first stage the method performs a selective search which produces many regions of interest (RoI). These RoIs are then propagated through the CNN to create feature vectors. In the final step a SVM classifies the RoIs using the feature vectors. This method proved to be a lot more efficient and accurate than the state-of-the-art at that time.

The next step in the evolution of this network was to substitute the classifier with neural networks. Doing so simplifies the training. Around that time a big problem with image classifier networks was that in the final layers they used fully connected layers which have the undesirable property of requiring a fixed size input. So even though a regular CNN can be used on arbitrary image sizes (with certain practical limitations), the moment a fully connected layer is attached to it, it loses this rather desirable property. One smart solution for this is the spatial pyramid pooling (SPP) architecture [35]. It connects the convolutional layers to the fully connected ones with a pooling layer which down-samples the arbitrary sized feature maps to several fixed sized feature vectors. Fast R-CNN [30] uses this technique to replace the SVM with a neural network, but instead of spatial pyramid pooling it is called RoI pooling layer. And it is not a pyramid since it has only one level, thus it is just a special case of SPP. The resulting network reduced the number of stages from three to two, and the neural network can be trained end-to-end which is elegant and easier to optimize.

The final step in the improvement was the replacement of the selective search. The idea is that the feature extractor CNN can also be used to produce region proposals, and this saves time. What Faster R-CNN [79] does is that it first calculates the feature maps using a CNN. The feature maps go through a region proposal network (RPN) which produces a set number of bounding box proposals parameterized relative to predefined anchor boxes for every feature pixel. The problem of different scales is handled by anchor boxes of different size. Preferably a non-maximum suppression (NMS) is performed (not mandatory) and then these proposals are extracted from the feature maps. These extracted proposals then are fed to the RoI pooling layer same as in Fast R-CNN. The underlying CNN of Faster R-CNN can be arbitrary thus there are several different versions with different accuracy.

What is important to note is that even though the whole method uses neural networks it is still considered as a two-stage approach. The main reason for this is that after the feature maps are created they are first being read through by the RPN and then once more by the RoI pooling layer. To be more precise the RoI pooling

layer reads many small areas of the feature maps which can add up to another one or even more than one passthroughs. Because of this Faster R-CNN is slow compared to single stage methods. There is another reason though for it being considered as a two-stage approach. Faster R-CNN is not trained end-to-end. The reason for this according to the developers is that the backpropagation between the RoI pooling layer and the RPN is “nontrivial”. One solution for this is the RoI warping layer [20] which is technically a bilinear interpolation.

As semantic segmentation improved the focus shifted from bounding box estimation to instance segmentation that is instead of using bounding boxes targets are identified by their region mask which covers the exact area occupied by the target. I do not go into details regarding this because in the case of helicopter footage the creation of such annotation is extremely difficult and inefficient for experimental purposes. Several methods for this approach are [19, 21, 36].

The other important attribute next to the accuracy is the computational performance. Training and using neural networks are computationally very expensive, so any improvement which can reduce resource consumption is worth examining. Single stage methods, which only pass through the image once, outshine Faster R-CNN in this. One of the first representatives for this the You-only-look-once (YOLO) architecture [76]. This network is not so deep, and it ends in a fully connected part (thus it requires fixed image size). The genuineness of the method lies in how it grasps the task. Similarly to Faster R-CNN it predicts a set number of bounding boxes for every cell in the deepest feature map, but it does not use anchor boxes, predictions are attached to the grid cell which contains the center of their bounding box. But while Faster R-CNN uses a different network for classification, YOLO uses the same, and outputs box and class predictions simultaneously. The weakness of the method is the fixed resolution and the low number of detectable objects. But it is good example for that defining the problem (that is the loss function) in the right way can make even a simple network perform well.

Perhaps as a response to YOLO, the Single Shot Multistage Detector (SSD) [60] tries to overcome its shortcomings. The problem of different scales is tackled

by making predictions from several different scale feature maps. The network is also fully convolutional (similarly to the RPN in Faster-RCNN) therefore the input size is not fixed. It utilizes anchor boxes but also adds class predictions to them, so the anchor box becomes the so called “default box”. With this and the several different scales SSD yields a lot more predictions than YOLO. Both architectures have several different versions employing methods such as batch normalization (BN) [42] or feature pyramid network (FPN) [52].

One big advantage of two-stage methods is that they only classify a relatively small number of image regions, whereas single-stage methods are classifying the whole image, therefore they have a lot of easy-to-classify and irrelevant boxes in the output which suppress the really relevant hard, misclassifies examples during training. RetinaNet [53] uses focal loss to overcome this problem.

The Scale-transferrable detection network (STDN) [106] builds upon the DenseNet architecture.

M2Det [104] introduces a rather unique one-stage architecture with a sort of “slice and dice” strategy. It reduces resolution and then increases it back several times and combines the features with the same resolution.

The Deep Layer Aggregator (DLA) [99] tries to combine existing layer aggregation methods into one architecture.

CenterNet [107] follows a pixelwise strategy where it tries to detect objects from an estimated heatmaps peak points. This is a sort of combination of the object detection and density-map estimation task. It is based on DLA, although other backbones are used as well.

2.1.2 Density-map-based methods

The idea with this approach is that the output of the network is an image or pixel map rather than a bounding box list. Each output pixel has a single value indicating the average number of people in that pixel. Naturally, this is most likely a fraction number. The count can be achieved by integrating over any given area. The resolution of the output image is often a fraction of the original image, however recently

solutions aim to estimate a map with the same resolution as the input image.

The ground truth for this method is created from head annotations. The original ground truth map contains ones at the head locations, and this map is cross-correlated with a two dimensional zero mean gaussian filter with a preselected, standard deviation. This essentially means that the head annotations are “smeared” over the map. As if one would put small butter globules on a board and then press is against another board, and as a result the globules get thin and spread out. The reason to do this is because these continuous patches are easier to learn than discrete 0-1 maps. Some methods try to apply adaptive filter size, so the generated patches better fit the size of the annotated heads, and while this is an interesting idea, the distance between the heads are used to calculate the size, and this approach does not work when people are standing at a distance from each other.

The majority of recent works focused on solving two problems that arise in state-of-the-art crowd counting methods: varying scales and perspective distortions. Several researchers have addressed the varying target sizes directly [103, 87, 82, 69, 56]. Others have experimented with increasing or changing the receptive field of the output pixels [71, 49, 14, 110]. Certain scholars have attempted to tackle perspective distortions by using focused attention mechanisms [88, 100, 97, 32, 84]. Learning residual errors and correcting density estimations with these errors has also proven to be a viable approach [90, 83, 57].

All of these methods serve the purpose of improving the accuracy of generic human detection. However, if we wish to differentiate between people on roofs, balconies, or posters and those on the street, we need to include the surroundings of the people in the calculations. Although certain methods may do so unintentionally and implicitly to an extent, our aim is to address this explicitly by using semantic segmentation.

The use of semantic segmentation in crowd counting is not new [85, 43, 93, 89, 29]. All of these works aimed to create segmentation to separate human regions from non-human regions. In many situations, this is a simple binary classification, but even when multiclass segmentation is used, the problem is eventually reduced to human

versus non-human separation.

In our work, we ignore humans as a segmentation class and focus on the segmentation of the background. We classify the background from the functional perspective of how likely it is that pedestrians exist in that region. This approach can be used with steep-angle, high-altitude footage, but is unfeasible with conventional crowd counting benchmark datasets.

Besides density map methods there is another closely related paradigm called localized regression. The key difference lies with the target map. In this approach, every map pixel aims to represent the total count in a small, localized region of the input. One typical way to create such a target map is to apply a uniform square-shaped filter to the head annotations [73]. The result will be a map of redundant localized counts. [96] also changes the counting task to a classification task by predicting count intervals instead of specific counts. These methods achieve competitive accuracy.

While most research in crowd counting is based on image processing techniques, it is worth mentioning that there are efforts underway to use the rest of the electromagnetic spectrum as well for this purpose. Some methods rely on on-person devices such as smart phones or radio frequency identification cards. The problem with these is that they are working with the assumption that people would always carry the device, not to mention the serious privacy issues raised by use of such devices. For indoor counting in buildings, where there are plenty of Wi-Fi enabled devices installed, personal-device-free Wi-Fi based methods yield high count accuracy [109, 59] while preserving privacy. However, in open outdoor spaces these methods are less viable due to the significantly lower number of installed devices and the lack of refractive surfaces.

2.2 Dynamic pedestrian counting (Tracking)

This task is also commonly known as multi-object tracking (MOT). It dates back well into the 20th century, and it does not necessarily require visual input. Given a

number of spatial observations of objects through a timespan the goal is to assign IDs to the observations so that the observations for the same object have the same IDs while observations for different objects have different IDs. To solve this task, generally a probabilistic model is set up for the ID distribution and then through some sort of optimization they try to find the distribution with the highest probability, that is the globally optimal solution.

The success of this approach depends on how well the probabilistic model is representing reality and how close the optimization can get to the optimum defined through this model. The main problem is that setting up a realistic model and solving it for the optimal solution takes an enormous time. Realistic models are usually very complex and the more complex it is the more difficult it is to find the optimum. At high level of complexity the only way is a greedy method to extensively search the solution space, which is exponentially large. Therefore, the real juggling here is to find a rather simple probabilistic model for which there are efficient optimization algorithms which can find an optimal or a close to the optimal solution.

The first algorithms developed were targeting radar or sonar-based applications, because around that time these were the only viable options for making spatial observations. In multiple hypothesis tracking (MHT) [78] they try to solve the problem of high complexity by pruning the search space through heuristics like distance gating. Recently as computer vision developed MHT was revisited for computer vision based application [45].

At the same time many other algorithms were developed along the “tracking-by-detection” paradigm. In these methods the probabilistic model often makes strong assumptions such as the trajectories are independent, or the observations cannot be collocated, which are obviously not true. In a big crowd the flow of people resembles the laminar flow in liquids, therefore there is some correlation between the trajectories. As for colocation, in 3D tracking it is a valid assumption, but in 2D people are often occluding each other. Nevertheless, the quality of these assumptions is never evaluated, only the tracking accuracy they present. As a

result, the probabilistic models are greatly simplified and finding the globally optimal solution can be modeled as a graph optimization problem.

Some methods are modelling the probabilities of object transitions in a flow graph and find the globally optimal solution by max-flow optimization [75, 50]. [6] models the problem as K-shortest path and solves a linear program to find the optimal solution. Another conceptualization of the problems is to think of the tracks as subsets of the set of all detections. Then the task is to find a maximum weight independent set (MWIS) [10].

As the reliability of visual object detection increased simpler probabilistic models were gaining ground. Incidentally partially optimal solutions were also on the rise which opened the way towards online and realtime tracking. In [15] optimization was done on a given time frame. The Simple Online and Realtime Tracking (SORT) method [9] was the first one to truly scratch the surface of high-quality online MOT. With the surge in CNN-based object detection accuracy the probabilistic model could be simplified as far as a bipartite association problem. That is only the last and current frames are used in finding the optimal pairing between previous and current detections. The Hungarian algorithm is used for solving this. The weights of the graph edges are calculated from intersection-over-union (IoU) and distance between the detections which are refined by a Kalman-filter.

[98] points out that existing methods associate between detections either by ignoring visual features or using hand-crafted feature detectors and promotes the use of deep learning based features. Deep SORT [95] uses the same strategy to improve the SORT algorithm. It introduced a deep association metric, which is supposed to capture the features of the targets. SORT solely based the linear association on geometric information, whereas Deep SORT used the association features to establish visual similarity metric between targets. These features were calculated by a separate CNN specifically trained for this task. This approach divided the MOT algorithm into two basic steps, detection and reidentification.

Afterwards various approaches were presented on how to solve the reidentification task better. Most of them were long short-term memory (LSTM) based solutions

[63, 81] which were using timeseries information in the training data to learn a feature metric. These approaches have one downside, specifically they require ID based annotation in the training data, which is scarcely available compared to ID-less training data.

The authors of the Joint Detection and Embedding (JDE) method [94] realized that the detection and reidentification parts do not have to be separated. Detection models are already producing object features. The only thing that was required is an extra prediction head producing features with the required dimensionality. Their other great contribution is that they posed the feature learning task as classification instead of feature estimation, so there was no need for handmade feature ground-truth. The number of output classes are equal to the unique entities in the training data, while the detection head is a regular object detector. This allows the use of training data without ID information because all entities can have a unique ID. They use uncertainty loss [44] to auto-balance the partial losses. This training method works surprisingly well. JDE also changes to solving algorithm of the linear association to the Jonker-Volgenant algorithm which is much faster than the Hungarian method.

In Fair MOT [102] they argue that while the task of detection and reidentification are solved jointly there is an imbalance between the two from several aspects. In some detectors the features are generated from the region inside the bounding box, which includes non-object features too. Moreover, FPN style detectors produce detections using several different grades of features, while the reidentification benefits mostly from deep abstract features. And finally, the dimensionality of the feature vector is too large, which decreases detection accuracy. By addressing these Fair MOT achieves state-of-the-art accuracy on multiple benchmark datasets.

The common benchmarks for MOT evaluation are the MOT challenges (MOT15, MOT16, MOT17, MOT20) [47, 70, 2]

Theoretically offline globally optimal solutions should be able to achieve better accuracy given that they have a larger amount of information at their disposal, but the focus is on fast online methods. The accuracy of MOT algorithms can only be

as good as the underlying object detector, and we can see that online methods are rapidly closing the gap. If this keeps going on even the theoretical advantage of globally optimal solutions will disappear completely.

2.3 Aerial semantic segmentation

The first significant breakthrough in CNN-based semantic segmentation was the realization that the task is essentially no different from classification, and any arbitrary classification network can easily be converted into a segmentation network [62]. Subsequently, various network architectures were rapidly developed in an attempt to improve the segmentation accuracy. For example, FuseNet [34] incorporates depth information into the training to improve the segmentation accuracy.

An important issue in the segmentation task is that the network is usually required to provide a larger receptive field compared to object detection. Dilated convolutions can solve this problem. DeepLabv3 [13] is a model based on ResNet, which uses dilated convolutions in its atrous spatial pyramid pooling layer.

Several architectures have been developed specifically for aerial semantic segmentation [72, 68, 28, 67, 3]. A common feature of the above methods is that they are multimodal or multispectral. The ISPRS aerial datasets [66, 1] are typical benchmark datasets for aerial segmentation. These methods are extremely accurate mainly because of the multimodality. Depth information helps in not just finding the correct object boundaries but also in classification. Unfortunately, the footage I present does not contain distance measurements, therefore we had to use methods which simply rely on RGB information.

2.4 Camera motion cancellation

Camera motion cancellation is seldom discussed in the MOT literature. The reason for this is that the benchmark video sequences do not exhibit strong camera motion, and what they do is handled well by the Kalman-filter in state-of-the-art algorithms.

The goal is to find a transformation between image pairs so that it satisfies some minimal difference requirement.

There are two main approaches for this, feature-based and direct methods. In the former there are few well distinguishable control-points, which are used for the calculation of the transformation and the rest of the image is transformed based on these regardless of how well the result is aligned. As for direct methods the whole image is used for computing the transformation.

In feature-based methods one approach is to estimate the camera motion. By knowing the difference between the two viewpoints (x , y , depth) coordinates can be transformed from one image to the other with three-dimensional linear transformations. Camera motion estimation is well studied in the field [74, 38, 33, 27]. Unfortunately, our medium altitude footage only has (x , y) coordinates.

In the lack of depth information one can try to find a two-dimensional linear transformation, however this only works if the difference between the images is small, otherwise strong distortions occur unless the image pixels all lie on the same three-dimensional plane.

The enhanced correlation coefficient (ECC) method [25] follows this path but in a direct manner. It optimizes over the pixel intensity difference of the whole image. Its accuracy is good; however, it is an iterative method, and it can take a while until a good result is reached. [46] follows a similar approach, however optimization is only done over the edges of the images. These are also known as global parametric models.

Other direct methods include dense optical flow such as [26]. A much simpler feature-based alternative for this is the Lucas-Kanade method [64] where the flow is only determined for edge or corner points. Furthermore, recently CNNs for this task have been developed as well [24, 41, 91, 58], however these come with the computational requirement of neural networks.

In mesh-based warping [48, 55, 101] there are certain key-point pairs for which the displacement can be measured and from these a dense displacement is estimated for example with linear interpolation. [51] introduces a mesh-based photometric

alignment (MPA) method where the mesh points are aligned based on photometric similarity.

2.5 Existing datasets

Numerous CNN-based human detection or counting methods have been developed and tested on standard datasets, such as the UCSD [12], INRIA [22], Caltech Benchmark [23], UCF_CC_50 [40], and ShanghaiTech [103] datasets. Although these image sets can be distinguished from one another by the average person count, they were all obtained from near-street-level cameras and from low angles. Therefore, these images contain many obstructions and perspective distortions. These issues pose substantially less difficulty in aerial footage.

Obstructions are caused by people walking alongside one another. In images taken from a steep angle, the obstruction is not very large because people do not walk over one another. Perspective distortion is caused by the relative difference between the distances of objects from the camera. The size of an object in an image is inversely proportional to its distance from the camera. Therefore, if the ratio of the distance of two objects is much larger (or smaller) than 1.0, the difference in their sizes in the image will also be very large. This ratio is closer to 1.0 in photographs taken from a high altitude. Figure 2-1 depicts these issues.

Of course, other aerial datasets are also available, which have been created either with airplanes or, more recently, using drones. Airplane footage is typically obtained from a high altitude, which is not suitable for the detection of small targets. The authors of [4] introduced a high-altitude, vertical-angle airplane dataset with acceptable resolution. The elevations are lower in the case of drone datasets, so human detection can be achieved. The angle is often vertical, which is advantageous for considering occlusion, but the human features are less recognizable. If the angle is low, perspective distortions occur, but a better viewpoint is provided for human features. Examples of drone datasets include the SDD [80], VisDrone2019[108], and Okutama-Action [5] datasets.

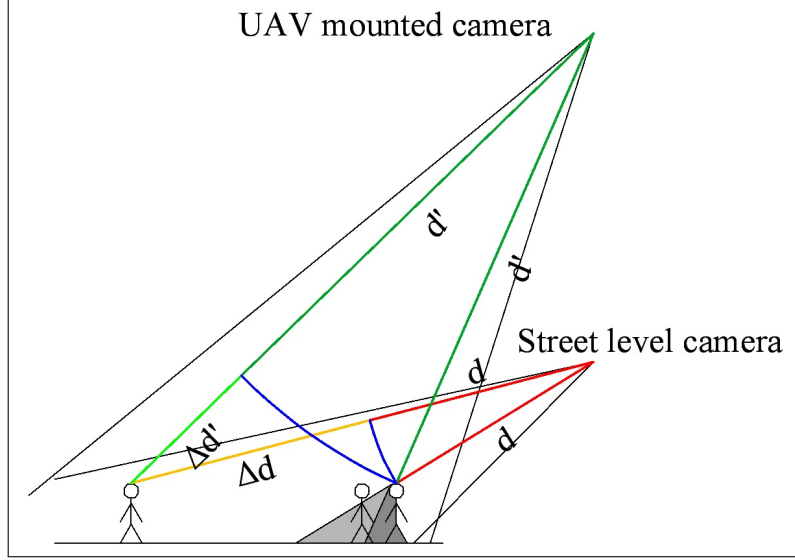


Figure 2-1: Illustration for the differences between street level footage and aerial footage. $(d + \Delta d)/d$ is much farther from 1.0 than $(d' + \Delta d')/d'$. Also, the dark gray area is obstructed from both cameras while the light gray area is only obstructed from the street level camera. Note that the image is not to scale.

Some of the abovementioned datasets are image based, whereas others are obtained from videos. The image-based datasets were annotated manually. The common annotation strategy for video-based datasets is to annotate certain key frames, such as every tenth one, and then interpolate the annotation between frames. This vastly increases the dataset size, but not the unique object instance count. Certain video-based datasets such as SDD and Okutama-Action use unique ID-based annotation so that it can also be considered for tracking.

Chapter 3

Introduced helicopter-based datasets

3.1 Datasets

Considering the applications mentioned in the introduction helicopter footage have numerous advantages. A single CCTV camera can cover a rather small area, because it is very close to the ground, whereas helicopter footage can cover entire blocks in one image, which means that a lot more CCTV camera is need to cover the same area that can be covered with a single aerial camera. Moreover, this is not just a logistic challenge, because the CCTV cameras have partially overlapping fields of vision which need to be synchronized in some way to avoid redundant counting. Furthermore, to cover a meaningful area, CCTV cameras have low angles. As a result, such footage exhibits strong perspective distortions and occlusions, whereas in helicopter footage these effects are negligible. In these regards low altitude drone footage is somewhat better than CCTV but still worse than helicopter footage.

Disaster mitigation highlights an additional logistic advantage. In disaster-stricken regions the availability of CCTV cameras is highly questionable, whereas aerial reconnaissance is always an option. Privacy issues are another important aspect of comparison. While there are certain countries where these are nonexistent, and the word “privacy” does not mean much, most countries in the world protect

the right to personal privacy at some level. The problem with CCTV and low flying drones in this regard is that a person’s face is clearly recognizable, whereas on helicopter footage it is not, therefore it is anonymous. This is helpful to avoid any legal issues with such footage.

In human mobility research there are various ways to achieve ground-truth data, such as surveys, global navigation satellite systems, call detail records. The first one has an inherent inaccuracy because only very small percentage of the people participate. As for the rest, they operate on the assumption that people are carrying appropriate “tracking” devices, such as smartphones. And while at the moment this is a good assumption, as conscious smartphone use grows this might change. Furthermore, this information is the property of the telecommunication service provider company or someone else, from whom the researchers must purchase it. Helicopter footage offers a viable alternative for this.

While these reasons are enough to promote the helicopter based pedestrian counting, it is not enough to justify such datasets. To truly establish the need for helicopter dataset we had to evaluate pretrained models on such footage. For this purpose, we created a smaller dataset to execute preliminary experiments (I explain these in details in section 4.1). We named this "Tokyo Hawkeye 2019" or "TH2019" for short [17]. This first dataset contains roughly 300 images with a resolution of 960×540 from helicopter above densely congested urban environment in Shibuya, Tokyo, Japan. From these results we concluded that indeed there is a need for such data therefore we decided to create a larger dataset from the same footage.

In [18] we introduced and published this new helicopter-based pedestrian dataset. As this dataset is an extended version of the TH2019 dataset [17], we named it TH2020. It has the same attributes as TH2019, except for the dataset size; there are roughly 20 times more images and 15 times more pedestrian annotations. Specifically, TH2020 comprises 6237 static images with a resolution of 960×540 , containing 120772 pedestrian annotations from 10 different locations and 22 different sessions. The images were obtained from helicopter footage from a high altitude at a steep but not vertical angle. Owing to the high altitude, the perspective distortions and



Figure 3-1: Sample images from the TH2020 dataset.

scale differences are negligible, although the average target size is very small. It is important to note that, because the camera angle is not vertical, pedestrians are partially visible from the sides, thereby increasing the number of recognizable features compared to those of a vertical camera angle, which is the most typical in airplane footage. Figure 3-1 presents some sample images.

The dataset includes two types of ground-truth annotations: a pedestrian annotation and a semantic segmentation annotation. The former contains only pedestrian head coordinates, including bikers, as there is no reason to differentiate between these from a practical viewpoint. Moreover, distinguishing bikers from pedestrians is very challenging; for example, if there are 10 people surrounding a bicycle, any one of these may be the rider. People on rooftops or on construction sites are excluded from the ground-truth. We generated a ground-truth density map from the head coordinate annotation using the conventional method. We created a matrix with the same dimensions as the input image. The matrix contained 1s at the head coordinates, and 0s otherwise. Thereafter, we convolved this matrix with a Gaussian kernel. We used a fixed kernel with a standard deviation of 15 instead of the adaptive method.

For the segmentation, owing to financial and time limitations, only 5040 images were annotated. The annotation assigned one category to every pixel of the image. When we created the segmentation ground-truth we were not aiming for an extensive annotation. Rather, we attempted to achieve simplicity and focused on a functional perspective for pedestrian detection. We wanted to ignore objects that were too small to occlude humans completely, so we considered those objects as part of their surrounding entities. Moreover, as we did not want the annotation to be too detailed, we required high-level classes. Therefore, we decided to annotate the following categories:

- Road: areas designated primarily for vehicle traffic; humans are often present.
- Sidewalk: areas designated for pedestrian traffic; humans are often present.
- Building: areas covered by construction that pedestrians can enter (houses, offices, etc.); humans may appear in windows, on balconies, and on rooftops.
- Structure: areas covered by construction, often with girder-like features, where entry is not possible (walls, columns, railings, etc.); humans are unlikely.
- Plant: areas covered by vegetation that can block visibility, such as trees with leaves and large bushes; humans are unlikely.
- Vehicle: areas covered by cars, buses, trucks, etc.; depending on the vehicle and lighting conditions, human heads may be present, but are unlikely.
- Background: anything else; typically grass, rubble, railway tracks, etc.; humans may be present, but are unlikely.

Figure 3-2 depicts some of these.

Regarding privacy matters, it should be mentioned that the resolution of the footage is not enough to identify any individual; only coarse features such as hair, clothing, or luggage can be identified. Therefore, there are no privacy issues with the dataset as opposed to conventional benchmark datasets where there are clearly recognizable people.

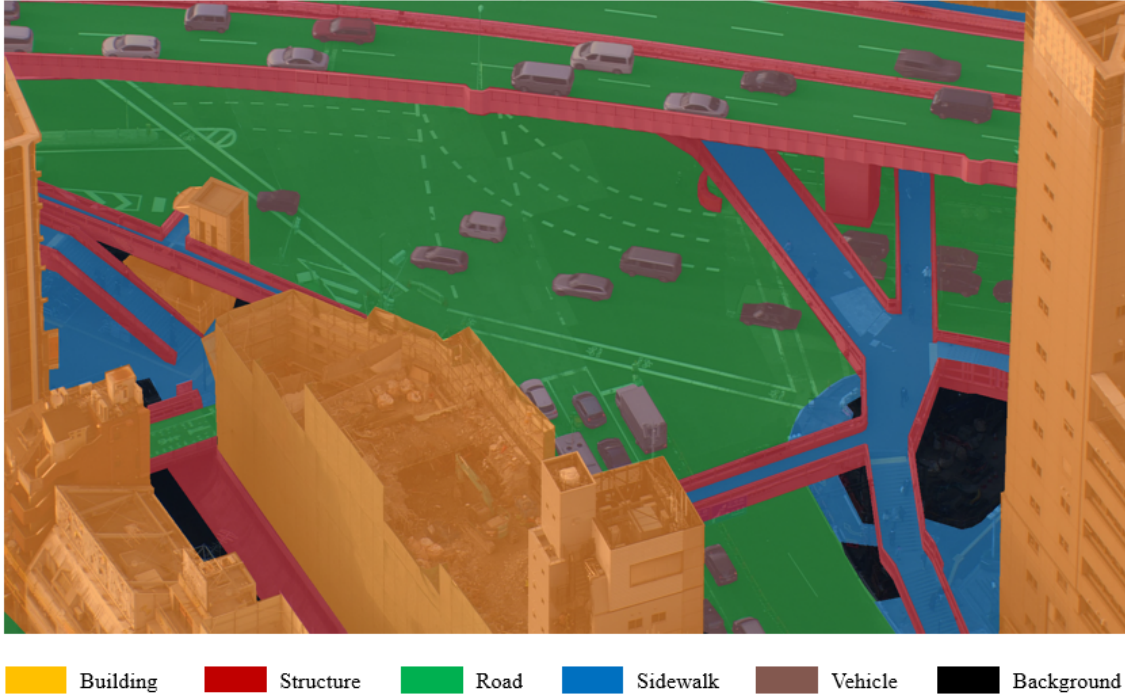


Figure 3-2: Semantic segmentation of the background for helicopter footage. Humans are ignored.

Our first experiments were executed with dividing the dataset into training and evaluation parts randomly, but these showed surprisingly high accuracy which suggested some level of overfitting. This kind of trickery is quite common in the field especially in the case of benchmark evaluations. We have noticed on more than one occasion that architectures reporting state-of-the art accuracy on more than one benchmark dataset have several separately trained models severely overfitted for each dataset. These overfitted models sometimes produce 2-3 times larger error on the other datasets than their overfitted counterparts. To avoid overfitting, we divided the dataset into training and evaluation parts scene-wise instead of applying a random split. Nevertheless, it is important to note that the reported errors can be reduced significantly, simply by introducing scenes from the evaluation set into the training set, which is a completely viable approach in practical applications.

One additional point regarding the dataset split is that the training set was the same for the segmentation and density map estimation, whereas the evaluation set for the segmentation was a subset of the density map estimation evaluation set, as

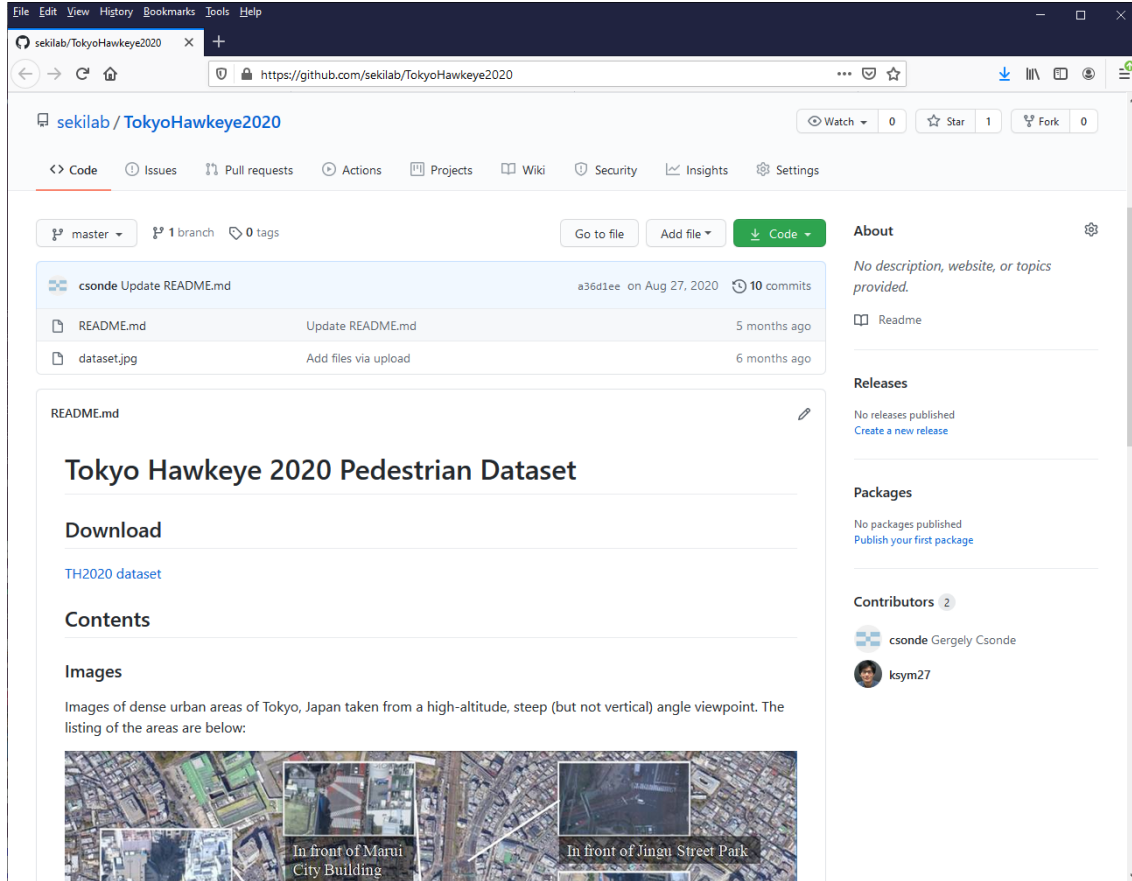


Figure 3-3: Github page for the TH2020 dataset.

not all images had segmentation ground-truth. Specifically, there were 4114 training images, 926 segmentation evaluation images, and 2123 density map estimation evaluation images.

In addition to the pedestrian head coordinate annotation we also have bounding box annotation, for object detection-based methods. In the case of MOT methods high frequency ID-based annotation is required for evaluation but most methods also require for training as well. Unfortunately involving more people in the annotation task takes its toll on the accuracy and the step of assigning IDs adds an extra chance for mistake, therefore we had to work with limited manpower. Because of this we did not create ID-based training data only evaluation.

We created ID-based annotation for two full HD 20 seconds long video sequences. We hand-annotated every 10th frame of these and for the rest of the frames the

bounding boxes are linearly interpolated, but only when the given ID exists in both enclosing 10th frames. From these sequences arbitrary 960×540 cropped videos can be extracted. For our evaluation set we extracted a total of six sequences which cover all the existing targets and have minimal overlap with each other.

The TH20202 dataset is publicly available at <https://github.com/sekilab/TokyoHawkeye2020>. See figure 3-3.

3.2 Crowdsourcing

The object detection annotation for TH2020 was created by external workers through an online service and the head annotation was generated from this. However, we have observed several problems with crowdsourcing. While according to the terms of use, the workers must do the job manually, it is obvious that many of them were using machines. Unfortunately, this is nearly impossible to prove, we could only speculate based on the annotation results.

Another problem is that while we require good quality annotations, the workers aim for a cost-efficient solution. Even in the simplest scenes with only 20-30 people we often found trivial mistakes. If the ratio of these mistakes was low, we corrected them ourselves, otherwise we asked the worker to do the job again. In extremely crowded scenes though checking the quality is just as taxing as annotating. However, in many cases the accuracy was so bad that we could tell it just by looking at it. Because of this we came up with a filtering strategy.

For crowded scenes we created high quality annotations for a few reference images by ourselves and then assigned these images as jobs to the workers. We filtered the workers based on what accuracy they achieved and selected only the ones who got above 90% recall. After this we assigned crowded images only to these workers. As I mentioned, thoroughly checking these images takes just as much time as annotating them, which would have beaten the purpose of crowdsourcing, therefore we only did a superficial check for obvious mistakes and assumed that by correcting those we are safely above 90% recall. However, this estimation is purely based on our previous

filtering.

This 90% can be both enough and insufficient depending on the use. In deep learning a seldom mentioned quality metric is how well a model can be trained with noisy data. From this point of view in our experience 90% is good. However, when it comes to evaluation, we are checking the results against the ground-truth, and as its name suggests it should be the truth, reality, the way things actually are. If not, then a good detector may score better than a perfect detector. If the ground-truth is only 90% accurate then if two models produce accuracy within 10% of each other, there is no telling which one is better. Fortunately, our goal is not to hold a challenge and rank methods, but rather to show comparable accuracy for which purpose 90% recall is enough.

We created the ID based MOT annotation by ourselves. There were two main reasons for this. First, the selected methods do not need ID information for training, therefore we only had to create an evaluation set which means significantly less images. The second is again the quality of annotation. The task itself is more complicated which requires more vigilance from the worker’s side, and we can hardly expect more conscientiousness from them than what was observed before. Furthermore, the annotation errors in this case have more severe consequences. If an ID is missing from one anchor frame, then because of our interpolation method, it would be missing from 19 frames in total. If this happens in a way that in a consequent series of anchor frames every second anchor frame is missing the ID, the total error rate may be almost twice as much as the error rate on anchor frames. And the last issue is the matter of ID switches which is another source of error that can further decrease the credibility of the ground-truth.

Chapter 4

Assessment of existing methods on helicopter-based datasets

In this section I explain the experiments we have executed to evaluate the accuracy that can be achieved with CNN-based pedestrian counting from helicopter footage. If existing pretrained methods were to perform well on helicopter footage, there would be nothing more to discuss. I would conclude that there is no need for creating such training datasets and existing pretrained models should be used for helicopter-based pedestrian counting, which has numerous advantage as I already established in chapter 1. However, this is not the case. To show this, first I present preliminary results from pretrained models which justify the further experiments. Afterwards I detail those and explain the weaknesses we identified regarding helicopter footage.

4.1 Pretrained models

The primary goal in CNN-based computer vision tasks is to achieve human level accuracy or even surpass it. While the phrase “human level” sounds good it is not a scientifically rigorous expression. The way we can measure this is to compare the computer accuracy to the accuracy of a well sampled control group of people. While this is scientifically sounder it is not a reliable evaluation method, because it assumes that the people are trying to solve the task to the best of their abilities. From

experience we know that this is not true. For example, if we look at the VisDrone 2018 dataset it is packed with rather trivial false annotations. In the case of our dataset we also almost always had to correct the work of external annotators, because they preferred speed above accuracy. Because of such factors it is not possible to tell when exactly the human level accuracy is surpassed. However unfortunately in some tasks we are so far from it that showing the opposite is trivial.

One measure for the human level accuracy in object detection and crowd density estimation is the generalization capability. The ability to recognize objects and people in various positions, from various points of view and under various lighting conditions. The earlier approaches were very bad at this and the only way to solve this problem was to feed all kinds of training data to the developed models. Even then, the models based on their generalization capabilities showed mediocre or overfitted results. Recently, better architectures were developed however they still require various learning data to generalize. Ideally, if a method could learn the detailed geometric and color attributes of humans it would be able to calculate the visuals of humans under any circumstance and be able to detect them. But this is not the case.

In table 4.1 I present our experimental results from state-of-the-art object detectors and crowd density estimators pretrained on benchmark datasets and evaluated on the TH2019 dataset. Clearly, their accuracy is far below human level. Only the model trained on the VisDrone dataset shows some promise, but it is still very far from practical applications. The reason for this is that the generalization capabilities of these methods are still in an infant state and they cannot handle images from a viewpoint that they have never seen during training. Our dataset offers a unique viewpoint and only the VisDrone dataset can be compared to it at some level. This means that if we want to do CNN-based pedestrian counting from helicopter footage we have to train our CNNs specifically with a helicopter dataset.

Method and dataset	mAP @IoU=0.5	relative MAE
Faster RCNN Inception Resnet v2 Atrous + MSCOCO	20.08%	—
Faster RCNN Inception Resnet v2 Atrous + TH2019	67.00%	—
YOLOv3 + MSCOCO	11.80%	—
YOLOv3 + VisDrone2019	36.25%	—
YOLOv3 + TH2019	81.47%	—
YOLOv3 1 detector + TH2019	82.91%	—
MCNN + ShanghaiTech B	—	1.09
MCNN + TH2019	—	0.44
CSRNet + ShanghaiTech B	—	1.08
CSRNet + TH2019	—	0.13
CAN + ShanghaiTech A	—	2.35
CAN + TH2019	—	0.1

Table 4.1: Comparison between models trained on other dataset and models trained on TH2019 when evaluated on TH2019 (we used only one class).

4.2 Training on helicopter-based dataset

4.2.1 Object-detection based methods

In object detection we inspected three bounding-box-based state-of-the-art models for training purposes. Specifically, Faster-RCNN [79], YOLOv3 [77] and Center-Net [107]. We trained all of these networks using the TH2020 training dataset to avoid overfitting. Our goal was to determine whether the helicopter footage based pedestrian counting can be solved with viable and comparable accuracy. To this end we evaluated the trained models on the evaluation part of the TH2020 dataset and compared it to the accuracy of the same models trained on other benchmark datasets.

Simple evaluation metrics in this task are precision and recall. These are defined through true or false negative and positive detections. The specific definition of positive and negative vary among methods but essentially it works as follows. Every model generates a finite list of predictions. These predictions have an estimated bounding box and a class confidence. For evaluation there is a certain

class confidence threshold and a bounding box intersection-over-union (IoU) threshold. A prediction is considered true positive (TP) if its box overlaps more than the IoU threshold with a non-paired ground-truth object and its class confidence in the ground-truth object’s class is above the confidence threshold. The ground-truth object is considered paired after this. If only the second condition is met than the prediction is a false positive (FP), if neither conditions are true, than the prediction is a true negative (TN), and finally, every ground-truth object still non-paired at the end are considered as false negatives (FN).

In our case we chose the IoU threshold to be 50% and the confidence threshold to be 25%. After this explanation precision is calculated as:

$$Precision = \frac{TP}{TP + FP}, \quad (4.1)$$

and Recall is calculated as:

$$Recall = \frac{TP}{TP + FN}. \quad (4.2)$$

The baseline metric in object detection currently is average precision (AP). This is a triple average precision over all confidence thresholds (or all recall levels which is equivalent), over 10 equidistant IoU levels ranging from 0.50 to 0.95, and over all classes. The reason for this metric’s existence is obviously to have one single number by which detectors can be ranked. Some researchers argue that it unnecessarily promotes bounding box regression.

In applications precision and recall are the metrics upon which clear requirements can be defined. In an application the decision of which prediction will become a detection is based on the confidence and IoU thresholds, therefore the accuracy of a network is somewhat irrelevant at higher thresholds.

Furthermore, in the case of small objects the IoU metric is unreliable. Just by simple annotator errors a perfectly fine detector can score worse than a mediocre one. For example, in our dataset a typical bounding box size is 20×40. Assuming a 2 pixels large mouse positioning error resulting in a box of 24x44 resolution the

IoU between the annotation and the actual box cannot be more than 0.76. This means that a perfect detector with perfect bounding box regression cannot have an AP higher than 0.60, whereas a perfect detector that regresses the annotation boxes perfectly instead of the real ones would have an AP of 1.00. And this error margin is only further increased if there is some strong annotator bias between training and evaluation sets, for example one annotator systematically draws larger boxes, while the other draws smaller. There is no telling which detector is better with such data. In MSCOCO [54] benchmarks AP values are also reported in three separate target size categories and indeed the smallest is always significantly worse than the larger ones. It is true that such errors diminish with relatively large targets, unfortunately our dataset comprises completely of small targets. Regardless, since precision and recall values are generally not reported, for comparison AP will have to suffice.

4.2.2 Density-map based methods

In density map estimation we investigated three more CNNs. These are CSRNet [49], CAN [61], and SPNet [35]. The first two estimates a density map with a downscaled resolution, while the last one scales the output back to close to the original size. We trained these on the TH2020 training dataset. Our main metrics for evaluation were mean absolute error (MAE) primarily and root mean squared error (RMSE) secondarily.

$$MAE = \frac{1}{n} \sum |y_j - \hat{y}_j|, \quad (4.3)$$

$$RMSE = \sqrt{\frac{1}{n} \sum (y_j - \hat{y}_j)^2}, \quad (4.4)$$

Where n is the number of images in the evaluation set y_j is the predicted count and \hat{y}_j is the ground-truth count for the j th image.

4.2.3 Multi-object tracking methods

In dynamic counting we had a very strong limitation to deal with, specifically the lack of ID-based training data annotation. This means that we were limited to methods which do not require ID information for training. Fortunately, there are a handful of such methods including some of the state-of-the-art. These methods are Deep SORT [95], JDE [94], and FairMOT [102]. The authors of these shared link to their source code in the paper so those can be considered official implementations. This is especially important because the papers lack detailed explanation in certain parts of the method, so the only way to fully understand it is to check the source code. Additionally, the source code has minor differences compared to what is written in the paper, one should be aware of these.

In case of JDE and FairMOT we used the official implementations in our experiments, but in the case of Deep SORT we had to use another one. The reason for this is that the deep association metric predictor CNN is trained over the Market1501 dataset [105], which is very different from our dataset so it is not useful. We do not have feature annotation for the TH2020 dataset so instead we used the classification-based training.

In Deep SORT and JDE we used YOLOv3 variants as detector CNN while in FairMOT we used CenterNet.

There are various metrics for evaluation MOT performance, however the Clear MOT metrics [7] are the most commonly used, especially multi-object tracking accuracy (MOTA).

$$MOTA = 1 - \sum \frac{m_t + fp_t + mme_t}{\sum gt_t}, \quad (4.5)$$

Where m_t , fp_t , and mme_t are the number of misses, of false positives, and of mismatches, respectively, while gt_t is the ground truth for time t . The thing that should be noted about this metric is that it has no lower bound. Depending on the number of false positives it can be infinitely small. Of course, real implementations generate limited number of predictions that limits how small this metric can get, but what is important is that this number is not a percentage, even though it is

reported as such. However, it can easily be negative in case of a bad object detector.

4.3 Weaknesses

The experiments showed comparable results for the TH2020 dataset, however they also revealed two weaknesses related to helicopter footage which can reduce the accuracy. It is important to investigate these, because if they make the counting methods less resilient than that is a strong argument against the use of helicopter footage. In the following sections I present the issues.

4.3.1 False detections due to lack of deeper semantic information

While generic human detection has its uses, it also exhibits limitations from certain practical viewpoints. If the task is to identify a subset of humans, identifying all humans will produce faulty results. The simplest example is the differentiation between actual living humans and fake ones such as dummies, statues, or posters, which inevitably appear in urban footage whether or not they are exhibited in conventional datasets.

Depending on the task, the set of features used for identifying humans can also be used for identifying human subcategories; for example, people with a certain hair color, wearing a certain type of clothing, or carrying a bag. However, more abstract subcategories exist that are difficult or impossible to identify based on humanoid features alone. One such subcategory is pedestrians.

For many pedestrian counting applications, people inside buildings, on balconies, or even on rooftops are irrelevant, as are people tending to their gardens or working at construction sites. Moreover, as mentioned previously, inanimate human-like objects may also exist, such as billboards and other forms of street advertisements (Figure 4-1). From the perspective of human detection, there is no real difference between an “image of a human” or an “image of an image of a human.” We must

recognize the billboard itself to differentiate between the two. The latter is not specific to pedestrians but is a generic issue. Furthermore, seemingly random false positive detections may occur in the regions of an image where there are not supposed to be people. Finally, the density map may contain background noise, which distorts the accuracy.

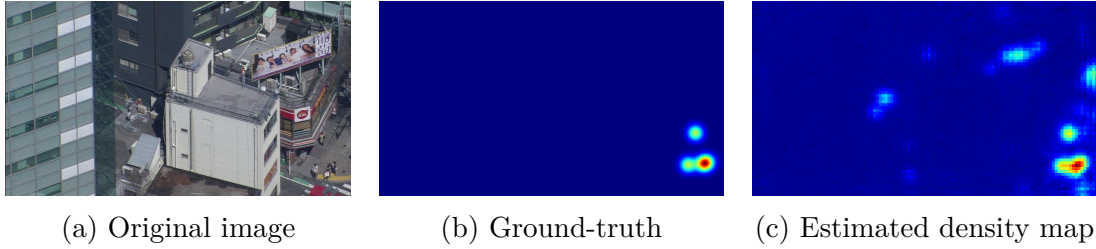


Figure 4-1: Sample image exhibiting the problem of "fake human" detection . It is clear that the billboard with humans on top of the building is producing false positive detections.

While these phenomena affect both object detection and density map estimation, in object detection it can easily be countered by introducing custom classes. For example, “statue”, “billboard”, or “worker”. On the other hand, in density map estimation there is only one class, so this cannot be addressed without modifying the architectures.

4.3.2 Camera motion in tracking

When it comes to tracking methods there are two ways in which these are affected by camera motion. Currently the main principle in tracking is to use visual similarity and the geometric distance to match entities to each other on consecutive frames of a video. When a camera is moving the visual similarity and the distance both can become more apart as opposed to a stationary camera. Of course, the opposite is also true, that is camera movement may cancel individual target movement, however when one plans for real life applications one must plan for the worst.

The other way in which camera motion is causing problems is that tracking targets on a video by itself is useless. Saying something like “target 364 left the

screen to the left” has no meaning by itself. To make this information useful the camera must be aligned to the real world in some way. This can be something coarse or some high precision real-world coordinate system, it really depends on the application. In the case of stationary cameras like CCTV, their real-world position is known and fixed, therefore it is easy to convert image coordinates to real-world coordinates. In the case of a moving camera one must also be constantly aware of the camera position to be able to convert image coordinates precisely to real-world coordinates. Solving this coordinate alignment problem is not the topic of my thesis, I simply note that camera motion is affecting tracking in this way too.

These things affect all moving cameras not just the ones that fly several 100 meters above the ground. One could say naively that in this case there is nothing more to do, helicopter footage shows the same accuracy for tracking as other footage. However, this is not true. The level at which camera movement is affecting street level footage and helicopter footage is not the same. To understand the difference, one must investigate the inner workings of the tracking algorithm. It is not my intention to fully explain it just to supplement what is written in the papers with the necessary information required for proper understanding.

By looking at the source code it is obvious that the authors of JDE and FairMOT copied the Kalman-filter implementation from Deep SORT without putting many considerations into it. Therefore, there is no need to differentiate between them so from now on when I refer to the Kalman-filter I mean this implementation common among the three approach.

For all methods, the weights in the linear association matrix are calculated as some combination of a visual similarity metric and a geometric distance metric (for both position and box shape). In addition to this there is a gating mechanism which sets the weights between any existing track and the new filtered positions estimated by the Kalman-filter to an infinitely large value if the distance metric is larger than a gating threshold. The purpose of this is to reduce the valid elements in the association matrix because the less connected it is the faster a minimal weight pairing can be found.

Unfortunately, this gating mechanism can also disable valid connections if the target moves too far from its previous position in one frame. The Kalman-filter handles this problem to some level. Both the object detection and the prediction have some uncertainty modelled by its own covariance matrix and based on these covariance matrices the Kalman-filter calculates new filtered predictions in the update step which are closer to the object detection result. This means that even if the existing tracks and the current object detections are too far away from each other, after the update step the filtered position may still be within the gating range.

The distance metric is calculated based on Mahalanobis-distance and the gating range is set to the 0.95 quantile of the chi-squared distribution.

The covariance values are proportional to the target size, which means that the Mahalanobis-distance is inversely proportional to the target size. This also means that the same Mahalanobis-distance for a larger target corresponds to a larger geometric (Euclidean) distance, which holds true even when the Mahalanobis-distance equals to the gating threshold, which finally brings us to the conclusion that the geometric (Euclidean) gating distance is also proportional to the target size. This is quite logical if we think about it for a second. Humans movement speed is always within a small range in the real world. However, the way this appears on a video depends on how far they are from the camera. A closer person will have much faster on-screen speed than a further one even though they move at the same pace. Therefore, the on-screen speed is proportional to the target size and so is its uncertainty, that is the covariance.

And finally, the Kalman-filter operates with a constant speed assumption. Specifically, it calculates an average speed for all people from a given number of recent frames and in the prediction step it predicts new positions based on that speed.

After this investigation let us check what happens when the camera is moving. In the following parts when I use “camera speed” or “camera acceleration” it means the negative of the on-screen speed or on-screen acceleration of objects owed to the camera movement, may it be from translation, rotation, or zoom. Of course, in this sense the camera speed or acceleration is different for many objects in the same

video, depending on their position relative to the camera and its motion, but under certain conditions this difference can be ignored.

If the camera moves at a constant speed, for example it is on an airplane facing down, there is not much change, up to the point where the speed becomes so large that people on consecutive frames are moving out of the gating range. After that point tracking becomes impossible. If the change in the movement speed is smooth, then it is a similar situation because the speed is calculated from the last several frames.

The interesting things happen when there are sudden movements. This case can be further divided into two scenarios. In the first the camera acceleration is close to the natural accelerations of the people. Sometimes it will enhance a person's acceleration, sometimes it will completely cancel it, but most of the time it will not leave the allowed range. Some degradation could be observed in the accuracy. The real problem occurs when the camera speed changes drastically compared to the change in people's natural speed. Which is proportional to the target size. Which is proportional to the geometric gating distance.

To sum it up camera movement causes problem when the camera speed is much larger than the normal speed of people or when the camera acceleration is much larger than the normal acceleration of people, which in return results in much larger speed.

This is the point where the problem with helicopter footage lies. On-screen speed that originates from camera rotation is independent of the target distance, therefore a CCTV camera and a high-altitude camera with the same rotation speed yields the same camera speed. However, on a high-altitude camera the average target size is much smaller, therefore the geometric gating range is much smaller, so the same camera speed much more likely yields disabled connections. This is depicted in figure 4-2. The naïve solution to this is to increase the geometric gating range, however this would make the prediction even worse, and much slower. In a crowded scene if the geometric gating range is large than there are many connections in the association matrix, and this allows for a lot of mistakes, moreover the linear

association solver needs to deal with many large cliques, which takes a long time.

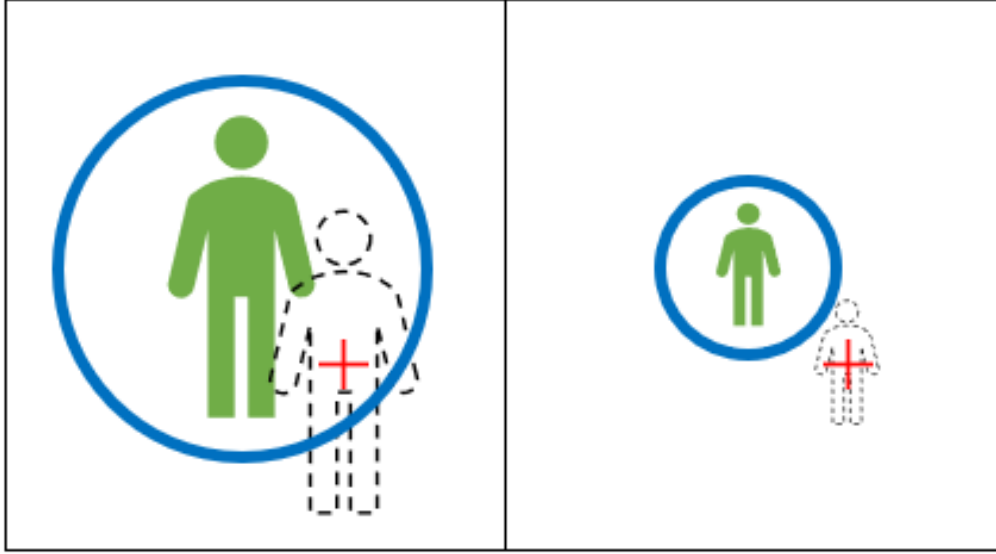


Figure 4-2: The image illustrates the difference between tracking close and far away targets with rotating camera. The green figure is the current position, and the dashed figure is the position resulting from camera rotation. The blue circle represents the gating distance, which is proportional to the target size. The left side shows a close target, while the right side shows a further one. The target is standing at the same (x, y) image coordinates on both sides. The rotation angle and therefore the displacement is the same, however while on the left side the center point of the new position is inside the gating range, on the right side it is outside therefore it will get discarded, which makes the tracking less accurate.

This problem is further enhanced by online tracking. The investigated methods are promoted as online trackers. But this is only true with the assumption that they can run in real-time. Real-time capabilities depend on implementation, computer hardware, and the number of detected targets and their density in a single frame, and in the TH2020 dataset there are some highly congested scenes with a target count above 500. We found that the algorithms have a hard time with real-time execution even on strong computers. Therefore, online execution can only be achieved by skipping frames, which increases the latent speed of the objects between frames and reduces the tracking accuracy.

Since the online accuracy depends on the execution speed, we have also investigated how we could improve it. We have noticed that the existing state-of-the-art

algorithms are realized in a completely sequential manner. This is close to optimal only if we assume that there is a huge computational bottleneck in the workflow. However, this is seldom the case on a typical computer with both CPU and GPU. There are two main steps in all algorithms, detection and pairwise association (reidentification). The former is typically executed on the GPU while the latter is executed on the CPU. The detection time has a very small variation while the reidentification depends on the target count and the crowd density. Our experiments show that the two runtimes are on comparable levels in realistic cases such as the MOT benchmarks or medium altitude aerial footage. This means that in sequential execution one of the key resources, either the GPU or the CPU is idle for a significant time. This problem can be alleviated with concurrent pipelined execution.

Chapter 5

Density map estimation with semantic segmentation

5.1 Generic Methodology

The issue described in section 4.3.1 can be solved by identifying the regions in which pedestrians cannot be present. We refer to these as invalid regions. Consequently, we refer to regions in which pedestrians may be present as valid regions. The invalid region information can be used to modify a base crowd density map. We do so by integrating the density map estimation and semantic segmentation.

In the following sections, we explain two methods for region-based masking.

5.2 Simple masking with separate CNNs

In this approach, we used an arbitrary DME CNN and an arbitrary semantic segmentation CNN, and trained both on the TH2020 dataset. For inference, we used the input image and generated an ordinary crowd density map with the unaltered DME CNN. Thereafter, we used the semantic segmentation CNN to generate a segmentation map for the same image. This segmentation map could have arbitrary classes. In our experiments, we used those explained in chapter 3. Subsequently, we took the segmentation map and simplified it as a valid-invalid region map based

on a predetermined table. Valid pixels had a value of 1, whereas invalid ones were 0. As the two CNNs were independent from one another, their output resolutions could differ. In such a case, we resampled the valid–invalid map to the DME output resolution using nearest-neighbor interpolation. As the final step, we masked the original crowd density map by multiplying it element-wise with the valid–invalid map. Alternatively, it would be possible to assign real numbers between 0 and 1 to every segmentation class and to use that as a mask, but because this was our first attempt with masking, we aimed for simplicity. Therefore, we used the zero–one map. The flow is illustrated in Figure 5-1.

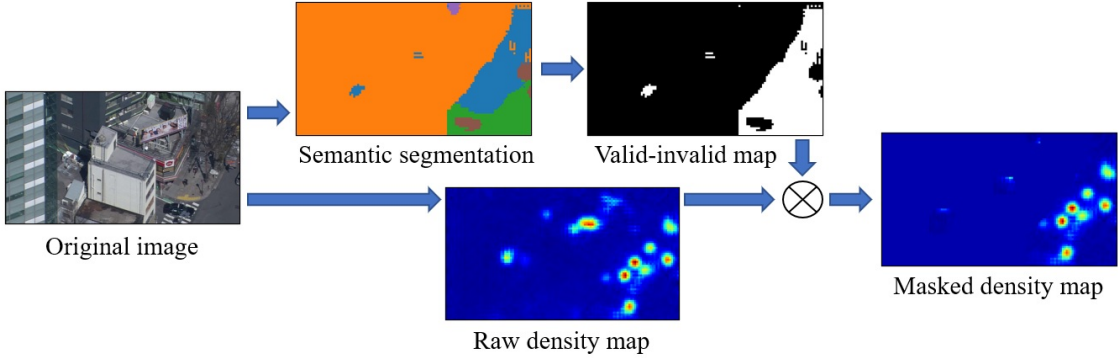


Figure 5-1: Flow of region-based masking procedure.

There is more than one means of creating a valid–invalid map from segmentation. The most obvious method is that whereby for every pixel, the value for that segmentation class from the predetermined table is inserted. Although this is certainly a viable approach, it has two drawbacks. Firstly, the segmentation is not perfect, so it is possible that some estimated invalid regions will overlap with some pedestrians. Secondly, it is very common for pedestrians to appear at the edges of invalid regions because the camera angle is not vertical. For example, when someone walks in front of a building and very close to it, most of the body will overlap with the building.

To solve the above problems, we proposed max-pool masking. We applied a max-pool layer to the valid–invalid map with a carefully selected kernel size (we used a kernel size of seven) and a single pixel stride. That is, an invalid pixel would only remain invalid if it did not have any valid pixels in a certain vicinity. In this manner,

the sizes of the invalid regions were shrunk to counter the effects of segmentation errors and natural overlaps.

5.3 DME segmentation multitask training

As demonstrated in [62], any arbitrary classification network can easily be converted into a segmentation network. However, we decided not to limit our approach to classification, as many image processing CNNs are based on classification networks in any case. Furthermore, technically, any arbitrary fully CNN can be changed into a semantic segmentation network by replacing the final output layers and loss function, which is the approach we used.

We used crowd counting CNNs, increased the number of channels in the final output layer to C , and changed the loss function to a conventional segmentation loss, specifically, the softmax cross-entropy loss. However, it occurred to us that the output did not contain any parameters, which meant that the model was exactly the same as that used for crowd counting, so we could attempt to perform the two tasks simultaneously. Moreover, we realized that because we were generating a crowd density map and segmentation map at the same time, we could use the segmentation map directly for masking the density map. Thus, our final architecture contained $C+2$ output channels: one for crowd density map estimation, C for the segmentation categories, and one for the masked crowd density map. Figure 5-2 presents a graphical layout of our proposed method.

For the loss function, we calculated the Euclidean norm over the density map per-pixel errors and softmax cross-entropy for the segmentation channels, and combined these two losses with a balancing factor α .

$$L_c = \frac{1}{N} \sum_{i=0}^N (d_i - \hat{d}_i)^2, \quad (5.1)$$

$$L_s = \frac{1}{N} \sum_{i=0}^N -\log \left(\frac{e^{-p_i^g}}{\sum_{c=0}^C e^{-p_i^c}} \right), \quad (5.2)$$

$$L = L_c + \alpha L_s, \quad (5.3)$$

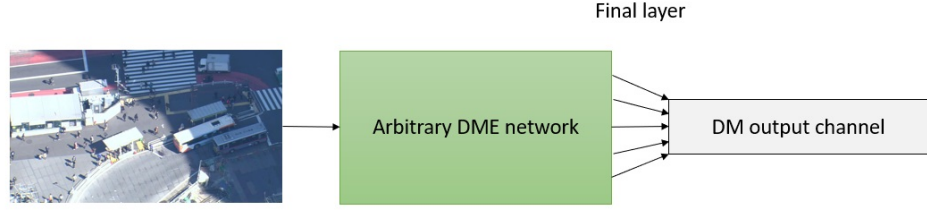
where N is the number of pixels in the mini-batch, d_i and \hat{d}_i are the ground-truth and estimated per-pixel crowd density values, respectively, C is the number of categories, p_i^g is the predicted confidence score for the ground-truth category at pixel i , and p_i^c is the predicted confidence score for category c at pixel i .

The Euclidean norm could also be calculated over the masked crowd density map instead of the raw one, but in this case, finding the correct alpha was very difficult, because the valid–invalid map could easily become stuck in the complete 0 or complete 1 state.

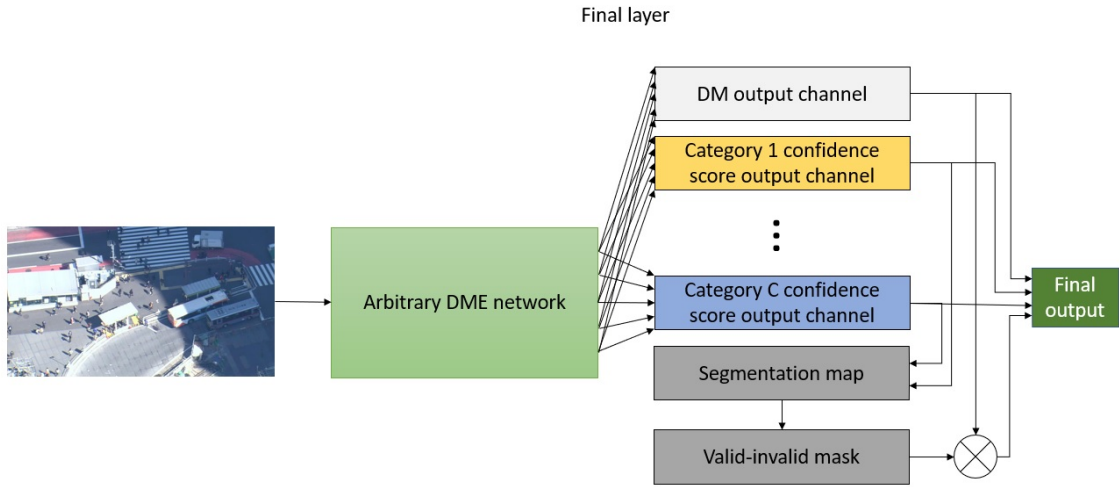
In this approach, we also evaluated the semantic segmentation quality, for which we used the mean intersection-over-union (mIoU).

To distinguish between the original backbone crowd counting network and the multitask version, we added the MTSM prefix (for MultiTask Segmentation Method). For example, if the backbone was CSRNet architecture, we referred to our modified version as MTSM-CSRNet.

Finally, it should be noted that the conversion method also works backwards; that is, a semantic segmentation network can be converted into a DME network, but we have not conducted such experiments yet, and thus, the accuracy of the backwards direction remains to be determined.



(a) Regular DME with one output channel



(b) Proposed multitask architecture with $C+2$ output channels, where C is the number of segmentation categories

Figure 5-2: Visual representation of difference between regular DME network and our proposed multitask segmentation method.

Chapter 6

Multi-object tracking on helicopter footage with camera motion cancellation

6.1 Overview

To correct the tracking errors originating from camera movement the latent on-screen movement of objects must be mitigated. There are various ways to do this, and I explain how we choose from these options.

The first question is whether to use absolute camera motion cancellation (CMC) or relative. Absolute here means that there is a dedicated base frame, and all other frames are transformed in a way so that they fit this base frame. In the relative case the goal is to transform frames in a way so that it is aligned with the preceding frame. Naturally, this also means that with the repeated application of relative CMC absolute CMC can be achieved, however with this approach errors would ramp up.

If the camera motion shows some bias, for example constant translation in one direction, eventually everything will shift out of the field of vision. In this case absolute CMC can only be used until some point. This can be countered by occasionally choosing a new base frame for the CMC. A special case of this is when the CMC

is rebased on every frame which is equivalent to the repeated application of relative CMC.

If there is no bias, then the movement can be modelled as Gaussian noise. This is the case when relatively stationary cameras are trying to aim at a fix point. For example, a camera on a floating helicopter or drone is aiming at an intersection in a city. In this case a significant area of the base frame is likely going to be visible in the consequent frames too. Of course, the specific details of this depend on the parameters of the distribution.

Our footage exhibits the latter behavior, therefore we can use absolute CMC, although theoretically the relative approach would yield the same accuracy, however it further complicates the CMC algorithm and introduces overhead.

The next thing that needs to be decided is how to fit the images to each other. The perfect solution would be to find the 3D homogenous transformation between the reference system of the two camera setups. The problem is that to be able apply this transformation one must have 3D coordinates. Unfortunately we do not have depth information, only (x,y) image coordinates, so this is not an option for us.

After the 3D homogenous transformation, the next best thing is the 2D projective transformation. With this approach the frames can be perfectly aligned to each other along any given plane. Of course, as a result objects which do not lie on the plane will be misaligned, however we are only interested in the locations where traffic takes place. This may not necessarily be on one single plane, especially when there are more than one traffic levels, for example pedestrian bridges. Fortunately, the further away the camera is, the better this can be approximated with one plane, and as luck would have it our helicopter footage is at several hundred meters away from the traffic level. Our experiments confirm that this is indeed enough for such approximation. Although in certain sci-fi movies multilevel traffic is distributed along a wide elevation range, which renders the single plane approximation useless. In this sense this solution is not futureproof, if indeed the future will look like this and not like an apocalyptic wasteland.

In case of the 2D projective transformation at least 4 key-point pairs are required

to calculate the transformation matrix. However, this can be further simplified in the case of helicopter footage. Because of the long camera-target distance, the images are close to an isometric view. If we assume that this is true then there are only four possible transformations, translation in two directions, one rotation and one scaling. Solving this partial affine transformation matrix requires only 2 key-point pairs, therefore this is the method we choose. I address the question of how to obtain key-point pairs in the next subsection.

The transformation matrix could also be calculated through other means. The effective correlation coefficient (ECC) method produces a transformation that is minimizing the pixel intensity difference between the two images. There are two reasons why we ignored this. The method optimizes over whole images; therefore, the result will not be aligned to the ground plane but rather some average plane that optimizes the difference. Moreover, the method is iterative, and it takes many iterations to reach a good solution, therefore it is slow. The only upside to this method is that it does not require key-points.

For completeness, optical flow or mesh based warping algorithms can also be used to cancel out the motion, however if only a few points are involved in the calculation we must have some means to determine which of those points are background and which are moving objects, since everything is moving because of the camera. Unfortunately, we do not have a trivial method for this at this moment. On the other hand, if the whole image is involved in the calculation it becomes slow.

Regarding regular street level footage, it is important to note that the 2D partial affine transformation yields worse accuracy because the single plane approximation has bigger error. In such a case, different regions of the images must be warped by a different transformation. Optical flow methods and mesh based warps would prevail in this situation. However, the higher computational requirement would be still an issue.

6.2 Camera motion cancellation with 2D partial affine transformation

In this section I describe our CMC algorithm in detail. As I mentioned earlier, we choose the absolute CMC using 2D partial affine transformation. For this method to work we needed key-point pairs. We solved this question with a rather simple approach, visual object tracking (VOT). The current state-of-the-art in VOT is achieved by Siamese CNN networks [8], which may be highly accurate but require a lot of computation which takes resources away from the detector CNN. Because of this we turned towards hand-crafted tracking algorithms. Unfortunately, these are not as robust, however we found that the “discriminative correlation filter with channel and spatial reliability” (DCF-CSR tracking or CSRT) algorithm [65] is suitable for our needs and can track one point real-time on an average CPU. Of course, we needed to track more than one point, but modern CPUs have more than one core therefore every tracked point can have its own thread. The real-time execution is important because otherwise we would introduce overhead into the tracking algorithm and reduce its online capabilities.

One thing to note is that the VOT algorithm requires a bounding box as input in the first frame. The trivial solution to this is the human input. At the start someone marks the key-points in the video, and after that tracking is set. This is what we used in the experiments. Automated solutions can also be designed, for example highly overfitted object detectors can be used to identify distinguishable key points on the ground level.

The relevant steps in the baseline MOT tracking algorithm are the following:

1. Fetching the next frame.
2. Predicting detections.
3. Establishing optimal association between existing tracks and new detections.
4. Updating tracks.

5. Storing tracking result for latest frame.
6. Repeat from 1.

The way we modified this is as follows:

- (0. Specify key-points.)
1. Fetching the next frame.
2. Predicting detections.
3. Updating key-points with VOT.
4. Calculating transformation between initial key points and updated ones.
5. Transforming detections to the base frame coordinates.
6. Establishing optimal association between existing tracks and new detections.
7. Updating tracks.
8. Transforming tracking results back to current frame coordinates.
9. Storing tracking result for latest frame.
10. Repeat from 1.

For optimal execution step 3) and 4) could be executed parallel to step 2)

6.3 Efficiency in various online use-cases

Currently researchers are promoting online execution, however at the same time model accuracies are reported by processing all frames. Processing all frames online is only possible if the tracker algorithm runs real-time. Whether this is the case depends on many factors, however our experiments showed that there are plenty of realistic situations when the tracking does not run real-time. In this case online

execution is only possible with skipping certain frames. This fact has an interesting implication on the efficiency of CMC.

Skipping frames can be thought of as processing a video with a lower frame rate. If we consider stationary cameras, the frame rate of the videos strongly impacts the achievable MOT accuracy. If the frame rate is high enough there is minimal change between two frames. As the frame rate decreases the difference starts to increase, assuming that there is movement. After a certain point, the difference grows large enough so that valid associations between consecutive frames are disabled, and the accuracy starts to drop. Finally, at a too low frame rate the tracking of moving targets becomes impossible.

If we add camera movement on top of this there is still a high enough frame rate where the change is negligible, therefore CMC does not change anything. If the frame rate is too low, again, the difference grows too large, so large that even CMC cannot improve MOT accuracy. However, between these two extremities there must be a section where the difference between frames with CMC is small enough whereas without CMC it is too large.

This means that the accuracy of online MOT depends on the execution speed, which for example depends on the computer which runs the algorithm. Furthermore, it means that the efficiency of CMC depends on the computer, therefore it is worth investigating and so we did. However, the execution speed depends on many factors, and we did not have a plethora of computers at our disposal to run tests. Instead, we emulated different execution speeds by running tracking with different frame skip rates and evaluated their MOT accuracy. I present these experiments in the results section.

6.4 Concurrent pipelined execution

A common attribute of the state-of-the-art MOT algorithms is that while they contain steps that can be done concurrently their implementation was presented in a sequential manner. Of course, when there is one step that is much longer than

the rest of the algorithm there is little to gain from concurrent execution. Such is the case if the algorithms are executed purely on CPU, where the detection step takes hundreds of times longer than the rest. However, the whole idea of using CNNs for object detection is based on the use of GPUs. Even around just 10-20 simultaneously tracked people the runtime of the reidentification section on CPU is at the same magnitude as the runtime of detection on GPU. Moreover, as the simultaneously tracked target count rises the runtime of the reidentification also rises meanwhile the runtime of the detection stays the same. On our target configuration we estimate that they are roughly equal around 80-90 simultaneously tracked people, and from that point on the reidentification takes over and it will become the bottleneck. This means that sequential execution is wasting a lot of computational resource.

To counter this, we introduce a concurrent pipelined implementation of the algorithm described in section 6.2. In the version without CMC, we use three threads (the CMC version has more for tracking key-points). These threads communicate through queues. The detection thread fetches frames in sequential order, predicts detections, and puts them in the detection queue. The update thread fetches the detections from the detection queue, calculates associations and puts the updated tracks in the results queue. The main thread is fetching results and store them for whatever purpose the tracking is being executed. In our case it is MOT evaluation. Figure 6-1 depicts this multithreaded execution.

The runtime of these loops is almost certainly different. Because of this it is possible that the queues become full. In such a case the queue consumer thread will block the queue feeder thread. This could be delayed by increasing the queue size; however, the queue size cannot be increased indefinitely, therefore the queue will eventually become full. This means that it is pointless to increase the queue size above 2. If one loop is much longer than the other then this approach will not change the execution speed of the algorithm much, however if the detection thread and the update thread runs for about the same time the execution time can be reduced by 50%. We compare the execution speed of the original sequential implementation and our concurrent one for one state-of-the-art MOT algorithm. Our approach can

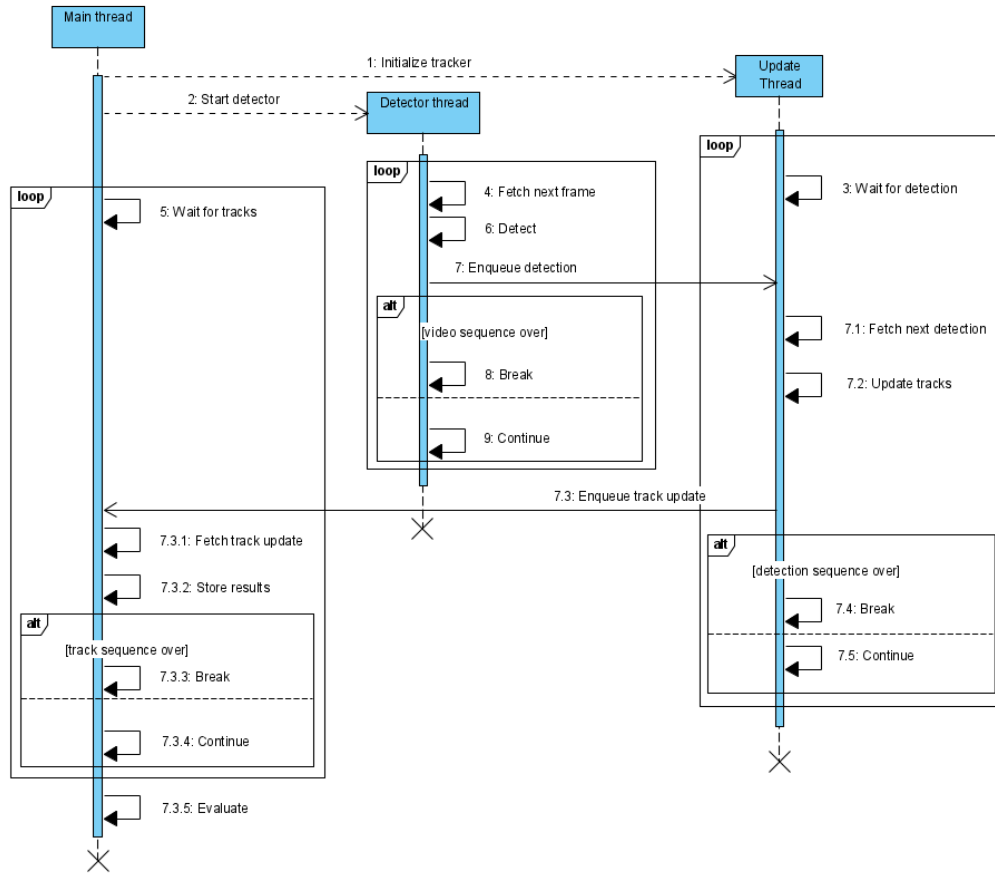


Figure 6-1: Simplified sequence diagram of our concurrently executed tracking algorithm. The exact queueing and blocking mechanisms are omitted for transparency.

be similarly applied to other algorithms.

Chapter 7

Results

7.1 Object detection results

We could not identify any significant problems with object detection networks regarding helicopter footage, therefore I report accuracy metrics for unaltered object detection networks trained on TH2020. I put these results against official benchmark scores for the same networks on the MSCOCO dataset. For both sides I report AP at 0.5 IoU threshold. For the models trained by us I also add precision and recall at 0.5 IoU and 0.25 confidence thresholds. This is simply for informative purposes. The results are shown in table 7.1

Dataset	TH2020			MSCOCO
Metric	Precision	Recall	AP@0.5IoU	AP@0.5IoU
Faster-RCNN Inception resnet	0.68	0.64	0.56	0.56
YOLOv3	0.77	0.64	0.67	0.58
CenterNet-DLA34	0.76	0.81	0.79	0.60

Table 7.1: Comparison between object detection results for state-of-the-art detectors when trained and evaluated on TH2020 and MSCOCO. AP values are the main metric for comparison. Precision and recall values are at 0.25 confidence and 0.5 IoU and they are only for notification. Also note that MSCOCO results are only reported for 80 classes.

While the results are somewhat different, we can conclude that the models trained by us have reasonable accuracy and we can expect that as more accurate

models are being developed the pedestrian counting accuracy using these new models also improves.

7.2 Density map estimation with semantic segmentation

7.2.1 Simple DME and segmentation methods

In our experiments, we used CSRNet [49], CAN [61], and SPNet [14] as the DME networks and DeepLabv3 [13] as the semantic segmentation network, although as we will demonstrate, there is no real difference. We selected a general-purpose segmentation network because methods for aerial footage also require height information, which we did not have for our footage.

As we were using a custom dataset instead of a conventional benchmark dataset, we first had to establish a baseline accuracy value for all of the networks.

The DME results can be viewed in the first line of Table 7.2, whereas Tables 7.3 and 7.4 display the DeepLabv3 results. In the segmentation task, we assigned valid values to the road, sidewalk, and background classes. The average pedestrian count in the evaluation set was approximately 30, which means that the relative MAE was slightly over 0.2. As these models produced similar relative errors on conventional datasets, we were confident that the models were well trained. In the case of DeepLabv3, the evaluation was not so simple. We could not compare the results to those of aerial segmentation datasets, as those also contain height information, which aids in achieving very high mIoU values. Our best option was a comparison with the Cityscapes dataset [16]. Deeplabv3 achieved an mIoU of 81.3 on Cityscapes. The accuracy achieved on our dataset was slightly worse than this, but the annotation was very different, so this difference did not mean that the trained model was not as effective as it could possibly be.

Moreover, we made several attempts with SANet [11], because the authors claimed to achieve high accuracy on conventional benchmark datasets, but we could

Method		CSRNet			CAN			SPNet		
		MAE	RMSE	mIoU	MAE	RMSE	mIoU	MAE	RMSE	mIoU
Simple DME		7.35	19.63		6.5	13.54		6.29	16.97	
Masked	Simple DME	7.26	19.9	0.7562	6.03	13.95	0.7562	6.14	17.15	0.7562
	MTSM	6.84	20.94	0.7332	6.9	22.3	0.7562	6.58	19.56	0.5821
Raw		6.84	20.85		6.82	22.2		6.56	19.56	

Table 7.2: Summary of crowd density estimation accuracies for models trained on TH2020. The mIoU column is only included to indicate the accuracy of the segmentation mask used.

not train the model. Even the lowest error we could achieve was almost three times higher than that of the other networks. Therefore, we excluded it from our investigation.

7.2.2 Simple masking with separate CNNs

We took our most accurate segmentation model, which happened to be MTSM-CAN, and used it to mask the density maps for all three DME networks. The MAE improved in all scenarios. The results are displayed in the second line of Table 7.2. Figure 7-1 presents a comparison of the different models.

Network	Building	Structure	Plant	Sidewalk	Vehicle	Road	Background	Mean	Upsampled mean
DeepLabV3	0.8494	0.3431	0.6552	0.7905	0.732	0.8632	0.6765	0.7014	0.7014
MTSM-CSRNet	0.9207	0.6263	0.6115	0.7571	0.7004	0.8584	0.6583	0.7332	0.7131
MTSM-CAN	0.9428	0.6693	0.642	0.7632	0.7072	0.8834	0.6858	0.7562	0.7358
MTSM-SPNet	0.9043	0.4162	0.4541	0.673	0.4674	0.8308	0.3288	0.5821	0.5778

Table 7.3: Per-category IoUs and their mean for DeepLabv3 and our MTSM models trained on TH2020.

7.2.3 DME segmentation multitask training

For this method, we modified CSRNet, CAN, and SPNet according to section 5.3. The DME results are displayed in the third and fourth lines of Table 7.2.

We reported the results for the models with the lowest MAE and reasonable

Network	Invalid (Building-Structure-Plant-Vehicle)	Valid (Road-Sidewalk-Background)	Mean
DeepLabV3	0.8916	0.8846	0.8881
MTSM-CSRNet	0.8999	0.8979	0.8989
MTSM-CAN	0.9112	0.9115	0.9113
MTSM-SPNet	0.8533	0.8508	0.852

Table 7.4: Valid–invalid IoUs and their means for DeepLabv3 and our MTSM models trained on TH2020.

segmentation quality. The latter means that the softmax cross-entropy loss was lower than 0.2.

Tables 7.3 and 7.4 present the per-class and valid–invalid IoUs compared to the same metrics of DeepLabv3. Figure 7-1 displays a comparison among the models.

The output of DeepLabv3 had the same resolution as the original image, whereas our backbone networks applied downsampling. To enable a fair comparison, we reported the mIoU for both the downsampled segmentation map and the downsampled segmentation map upsampled back to the original resolution.

CSRNet and CAN were demonstrated to be strongly adaptable to our MTSM model, with comparable and even improved accuracy. In the case of SPNet, both the counting and segmentation metrics decreased. The most important difference between SPNet and the other architectures is that SPNet is shallow compared to the others.

It can be observed from Table 7.2 that the masking actually decreased the counting accuracy. In the following section, we investigate the reason for the varying masking efficiency.

7.2.4 Masking efficiency

The level of improvement depends on the quality of both the density map estimation and semantic segmentation. The dependence on the segmentation quality is straightforward. If the invalid areas are larger in the estimation than in the ground-truth map, true positive detections can be masked out. However, if they are smaller

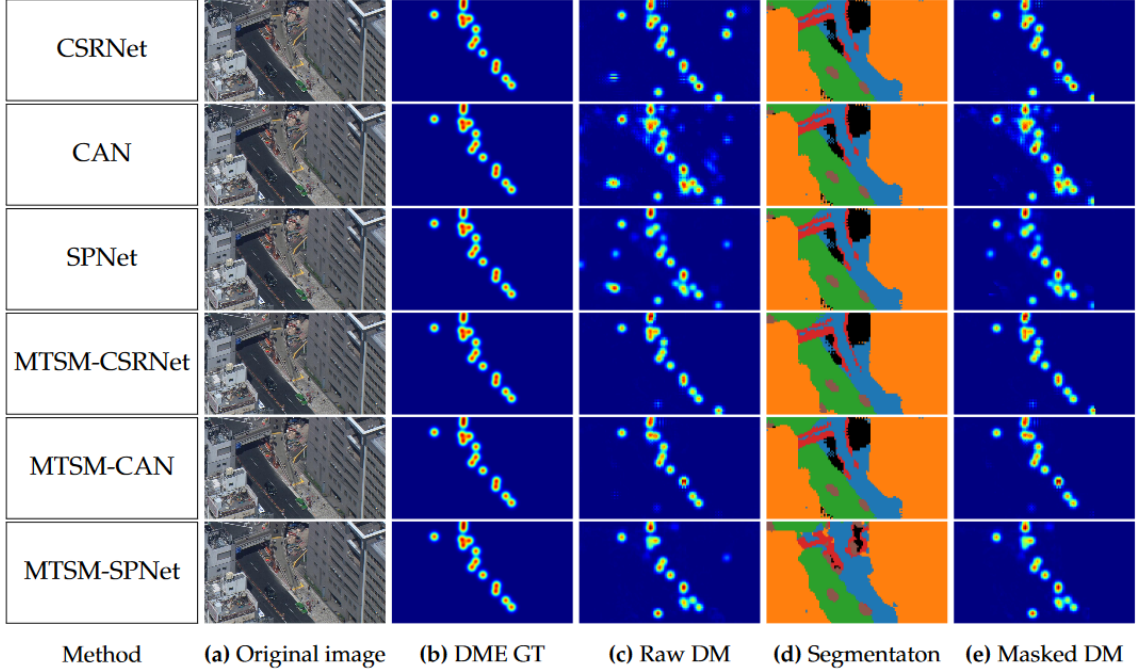


Figure 7-1: Sample results for comparison across methods. The segmentation for the regular DM methods was generated by MTSM-CAN.

than the ground-truth, nothing may change.

The dependence on the crowd density estimation quality is slightly more complicated. For argument’s sake, let us assume that the segmentation map is correct. This is a safe assumption because we achieved very high invalid region IoUs (Table 7.4), and the possibility of errors is further reduced by the max-pool masking. Moreover, let us assume that the density map does not contain negative elements. Although many architectures do not enforce this, in our experiments the robust networks eliminated negative values (or reduced them to very close to 0).

If the total count is overestimated, there must be several false positive detections. As we assumed perfect segmentation, the masking can only remove false positives. If there is a removed false positive count that is more than twice the original error, the MAE will increase. However, this also means that the estimation has numerous false negatives and the network attempts to compensate for the error with false positives.

If the total count is underestimated and something is still masked out, thereby

increasing the MAE, the model produces false negatives and false positives simultaneously, but the false negatives outweigh the false positives. In summary, the masking efficiency depends on whether or not the estimated density map is littered with both false positives and false negatives.

Method	Average L2/L1 norm ratio	Change in MAE	Invalid mask mIoU
CSRNet	0.0457	-0.09	0.9112
CAN	0.0315	-0.47	0.9112
SPNet	0.0457	-0.15	0.9112
MTSM-CSRNet	0.0509	0	0.8999
MTSM-CAN	0.0529	0.08	0.9112
MTSM-SPNet	0.0444	0.02	0.8533

Table 7.5: L2/L1 norm ratio compared to change in accuracy after masking for investigated models.

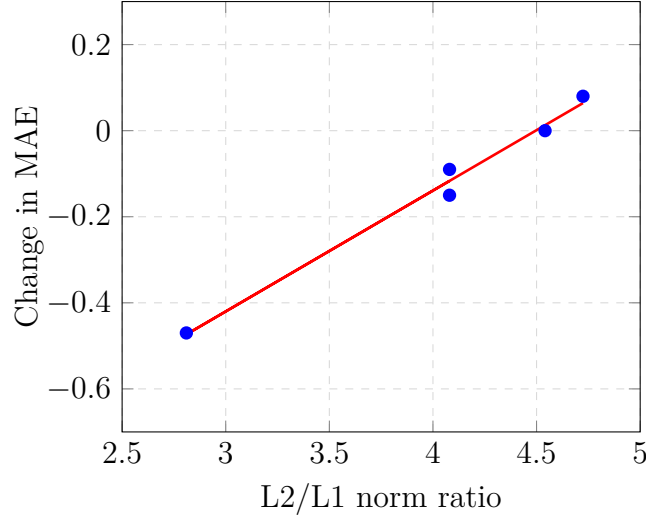


Figure 7-2: Correlation between DME quality and masking efficiency in case of high-quality segmentation mask.

If there are many false positives or false negatives, they will cause spike-like errors in the density map. These spikes significantly increase the L2 norm of the pixel errors (that is, the Euclidean loss). In contrast, if there is mostly only background noise, the L2 norm will be much smaller than the L1 norm. Therefore, the ratio of the two norms provides information regarding the amount of spike-like errors; that is, the

number of false positives and false negatives in the estimation. We collected this information and plotted it against the MAE improvement, which can be observed from Table 7.5 and Figure 7-2.

7.2.5 Training details

We downloaded open-source implementations for all models except for SPNet, which we implemented ourselves (excluding the final ReLU layer).

As DME networks are downscaling and sometimes upscaling, the output resolution differs from the original resolution. Thus, because the ground-truth has the same resolution as the original image, it must be resampled. We used bicubic interpolation for the density map and nearest-neighbor interpolation for the segmentation map. For the density map, we also had to multiply the pixel magnitudes by the resampling factor to preserve the total count. We applied a random horizontal flip as augmentation. As the footage had a well-defined upwards vertical direction, it was pointless to apply rotation or a vertical flip, and we did not want to complicate the training further. As all three DME networks have a VGG-16 [86] backbone, we initialized them using pre-trained VGG-16 parameter weights. There were no suitable pre-trained weights available for DeepLabv3; therefore, we had to train it from scratch. Parameters without pre-trained values were randomly initialized with a normal distribution ($\mu = 1, \sigma = 0.02$).

We trained the models on full images, because DeepLabv3 struggled with patch-wise training, and it was important for all models to be trained on the same dataset for comparison. We used one or four GPUs for the DME methods. In the case of four GPUs, the batch size was increased, but we did not observe any change in the accuracy; only in the execution speed. We used four GPUs for DeepLabv3 owing to the large memory requirement. Each GPU had 16 GB memory in which a batch of two images could fit, yielding a total batch size of 8. For the DME, we set the batch size to range from 12 to 16 per GPU (the maximum number that could fit). We used the Adam optimizer with an initial learning rate of 10^{-4} and a momentum of 0.95. The learning rate was directly halved every 20 epochs, starting from the

40th epoch.

The details of the MTSM training were mostly the same as those in the case of the baseline models. The baseline model parameters were used as initial values for MTSM-CSRNet and MTSM-CAN. As MTSM-SPNet did not perform effectively when initialized from the baseline model, we launched several training sessions from scratch to achieve the highest accuracy.

However, the most important aspect was to determine the appropriate value for α . The Euclidean loss and softmax cross-entropy are very different functions. Therefore, using a simple constant scaling factor would result in the gradient components having different weightings in different sections of the training. During our training attempts on MTSM-CSRNet and MTSM-CAN, we often observed that with a constant α , the training shifted towards either the crowd density or segmentation very rapidly. We used an initial α of 10^{-6} and whenever the training became too one-sided, we shifted it towards the weaker loss, which usually meant increasing α . In the case of MTSM-SPNet, the training could be performed by calculating the loss function over the masked density map using a constant α of 10^{-5} . In practice, we used two balancing factors: one for each loss component, so the final loss would be closer to 1, because small fractions are difficult to read.

7.3 Multi-object tracking on helicopter footage with camera motion cancellation

7.3.1 Used multi-object tracking methods

In our experiments we used Deep SORT [95], JDE [94], and FairMOT [102]. All of these methods conform to the baseline steps explained in section 6.2. These methods are building upon each other, and they are improving the accuracy of the previous method. Deep SORT has significantly worse accuracy, while JDE is close to FairMOT. We modified all of these algorithms by introducing camera motion cancellation as explained in section 6.2. Figure 7-3 illustrates how well our

algorithms work with separate frames being aligned to each other.

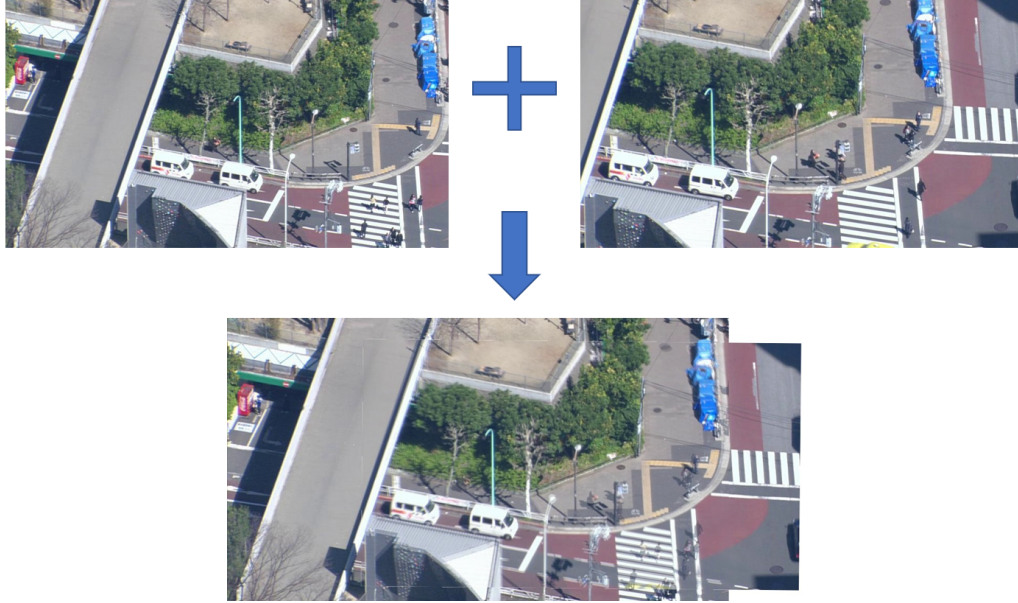


Figure 7-3: The upper left and the upper right images are frames of the same video sequence from different times and camera viewpoints. The lower image depicts the result of our CMC method transforming the right image to the left one's coordinate system and merging them together. The quality can be assessed from the overlapping region by checking the misalignment of stationary objects. The images are aligned over points that are few meters above the ground level therefore the taller something is the more misaligned its two views become.

7.3.2 Efficiency of camera motion cancellation

We evaluated the MOT accuracy of both the original algorithms and our modified versions with various frame skip rates on the evaluation videos we created from helicopter footage. The frame rate of the test videos is 30 frames per second (FPS). We started with this, which corresponds to no frame skip. After this we increased the number of skipped frames by 1 until 10 skipped frames. At this point it was already obvious that the accuracy is plummeting, so we only evaluated at 15 and 20 skipped frames after this only to show a trend. In the frame skip cases the results were only evaluated over the processed frames, otherwise the accuracies would naturally decrease due to the growing number of missed detections. The results can be seen in figures 7-4, 7-5, and 7-6

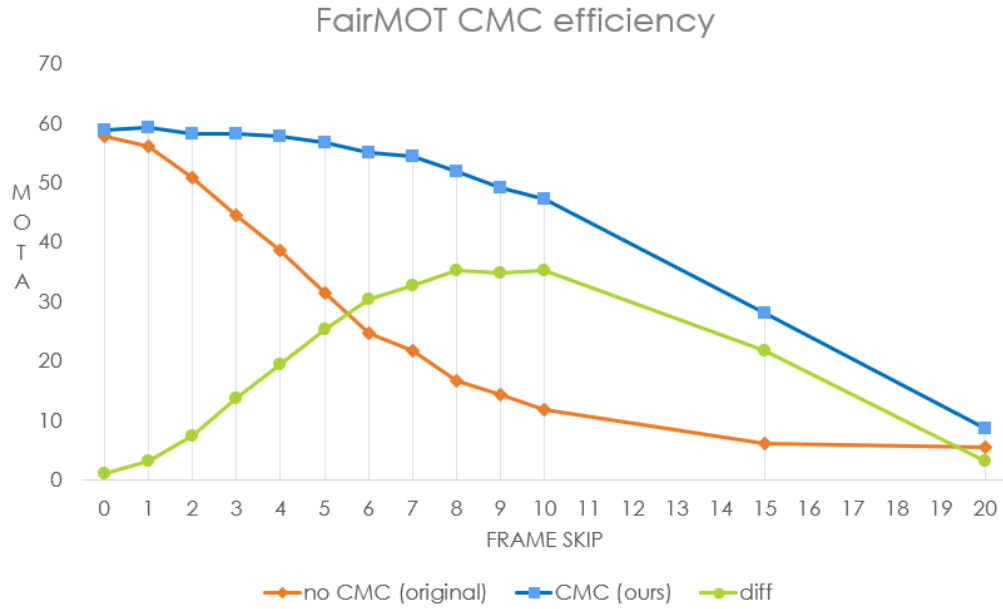


Figure 7-4: MOTA results for the original FairMOT and our modified version with CMC evaluated on 30 FPS helicopter videos with regards to the number of skipped frames. Our method is superior in every case.

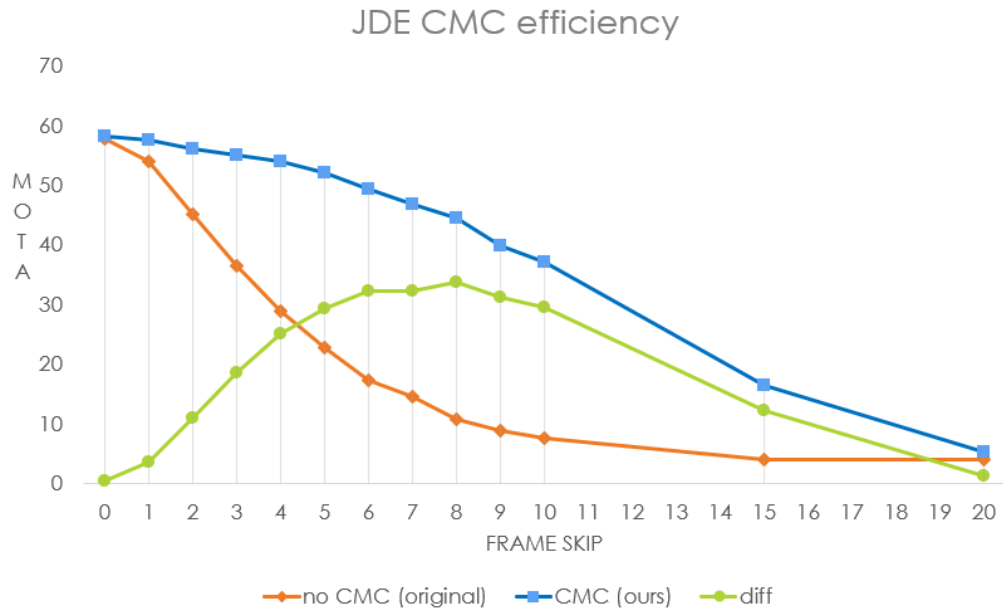


Figure 7-5: MOTA results for the original JDE and our modified version with CMC evaluated on 30 FPS helicopter videos with regards to the number of skipped frames. Our method is superior in every case.

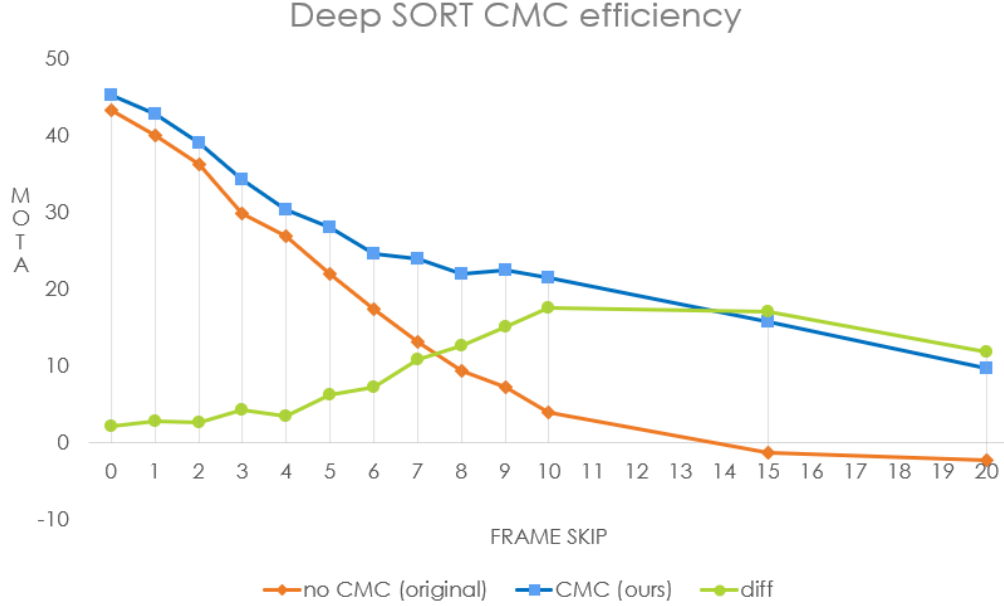


Figure 7-6: MOTA results for the original DeepSORT and our modified version with CMC evaluated on 30 FPS helicopter videos with regards to the number of skipped frames. Our method is superior in every case.

Three lines are plotted on each diagram, the original algorithm, our modified version with CMC, and the difference between the two. At 0 frame skip there is very small improvement, but improvement, nonetheless. As the frame skip increases though, the two lines grow separate rapidly, showing that our method has significantly better accuracy when frames must be skipped for online execution. At higher frame skip counts both methods decrease down to or below zero (the MOTA metric has no theoretical lower limit). These results fit the expectations; therefore we can conclude that our modified algorithms are superior compared to the original ones on helicopter footage.

7.3.3 Speed improvement with concurrent pipelined execution

Our CMC extension of FairMOT was implemented in a concurrent manner. We have evaluated its execution speed improvement with CMC disabled on MOT17 video sequences as well as our medium altitude video sequences. This can be seen in

tables 7-7, and 7-8. We also show the average ground truth target count per frame. In all scenarios there is significant improvement and it truly depends on the average ground-truth. To be more precise it depends on the simultaneously tracked targets, which is correlated to the average ground-truth.

Video sequence	02-SDP	04-SDP	05-SDP	09-SDP	10-SDP	11-SDP	13-SDP	Average
Average GT per frame	50.00	102.86	9.57	19.83	26.68	11.79	26.93	
Original FPS	22.94	19.15	26.6	26.32	24.38	25.89	24.87	23.71
Concurrent FPS (Ours)	34.65	33.48	27.69	33.5	33.5	32.61	33.84	32.3

Figure 7-7: Execution speed of the FairMOT algorithm on our target configuration with the original implementation and our concurrent one for MOT17 video sequences.

Video sequence	01B-1	01B-2	01B-3	01B-4	02D-1	02D-2	Average
Average GT per frame	10.14	14.93	15.12	15.30	22.31	26.40	
Original FPS	25.14	23.97	24.35	23.99	21.75	20.68	23.22
Concurrent FPS (Ours)	27.97	27.3	27.75	26.86	26.79	26.02	27.11

Figure 7-8: Execution speed of the FairMOT algorithm on our target configuration with the original implementation and our concurrent one for helicopter video sequences.

7.3.4 Implementation details

We implemented our algorithms in Python using the PyTorch and OpenCV libraries. As for concurrent execution, the global interpreter lock (GIL) prevents Python programs from true parallelization. Even though it creates threads, these are never executed concurrently on different cores. However, the mentioned libraries are implemented so that when they leave the Python environment, they release the GIL, and this is exactly what we need.

Our target configuration is an Amazon Web Services (AWS) “p3.2xlarge” instance with an Nvidia V100 GPU (16 GiB memory) and an Intel Xeon E5-2686 v4 processor (4 cores, 2 threads per core) and 61 GiB system memory.

7.4 Result summary

In table 7.6 I present summarized result from all of our pedestrian counting experiments either with unmodified or our improved methods. Only the best achieved accuracies are listed and compared to the relevant results of the same method on a conventional benchmark dataset. For these results it can be said that our achieved accuracy on helicopter datasets is on a comparable level to that of benchmark datasets.

Task	Reference Method	Metric	TH2020 results	Reference results	Reference Dataset
Object detection	CenterNet (unmodified)	AP@0.5IoU*	0.79	0.60	MSCOCO
Density map estimation	CAN (simple masking)	Relative MAE	0.20	0.12	ShanghaiTechA
Multi-object tracking	FairMOT (with CMC)	MOTA	59.30	60.60	MOT15

Table 7.6: The final best accuracies of pedestrian counting methods on helicopter footage compared to relevant reference values on benchmark datasets. *Reference AP is reported for 80 classes, whereas our result is for 1.

It is worth to note that these benchmark scores are typically the result of lengthy finetuning procedures, and customarily overfitted on the evaluation dataset. Because of the sheer number of experiments, we had to execute, we could not afford to spend time with too much finetuning, and we were also trying to avoid overfitting. One crucial step in the latter was that we split the image dataset into training and evaluation parts so that they contain different video sequences. Initially we used a random split method for crowd counting and as a result we achieved a relative MAE below 0.1 which is better than the reference accuracy. However, we did not want to resort to such tricks.

Unfortunately, we cannot compare our object detection results with official benchmark scores in a completely fair way, because MSCOCO AP@0.5 values are averaged over 80 classes, whereas we only have 1 class, hence our significantly better score.

From an application point of view these accuracies are not good enough for high precision requirements such as infrastructural planning, but neither are the reference values presented. But for other applications it is already good enough. During evacuation extra eyes in the air are helpful even if they are just 50% percent accurate. In safety monitoring a human observer can be assisted by such systems. As for human mobility research this can be used with the limitation that the evaluation result can only be considered as reliable as the accuracy of the pedestrian counting method. For example, if the pedestrian counting has a 20% percent error rate, then the evaluation result of a human mobility research also has 20% percent uncertainty. This may seem a bit much, however if we consider the alternatives, for example call detail record data, there is no telling how many people are actually carrying cell phones, or how many they are carrying, so there is no information about the accuracy of such data at all.

To sum it up, our experiments show that pedestrian counting can be achieved with comparable accuracy on helicopter footage.

Chapter 8

Discussion

8.1 Correlation between density map quality and masking efficiency

Our data exhibited a linear correlation between the average L2/L1 norm ratio of the images and the change in the MAE caused by masking. This means that a model with a smaller average L2/L1 norm ratio can benefit more from the masking method. A factor that could not be observed from our experimental results is that models with very different MAEs may have the same average L2/L1 norm ratio. For example, if there are two models and one produces proportionally larger per-pixel errors than the other, it will also have proportionally larger MAEs, while the average L2/L1 norm ratio will be the same. However, if the per-pixel errors are all proportionally larger, the per-pixel errors that are masked will also be proportionally larger, which is exactly the change in the MAE. In summary, based on the MAE value, the same average L2/L1 norm ratio may be correlated to different changes in the MAE. This suggests that the average L2/L1 norm ratio is probably linearly correlated to the change in the MAE relative to the MAE, but as our experimental results had MAEs that were very close to one another, there was also a correlation with the absolute change.

Referring back to the results in Table 7.2, it can be observed that although the

MAE values improved, the RMSE increased slightly. This may occur if there are moderate improvements in images with small errors along with small degradations in images with large errors. If the dataset has a small variance in the per-image ground-truth count, a high RMSE indicates that images exist in the dataset that are greatly over- or underestimated. However, if the dataset has a large variance in the per-image ground-truth count, a high RMSE is inevitable, because every method exhibits a certain proportionality between the ground-truth count and absolute error; that is, if the ground-truth has high variance, the absolute error will also have high variance. This means that the RMSE is not a very useful metric for datasets with high ground-truth variation, such as ours, unless a method is developed that produces an error that is independent from the ground-truth count.

Moreover, it should be noted that the proposed masking method is not feasible with conventional crowd counting benchmark datasets for two reasons. Firstly, in conventional datasets, the ratio of the area covered by humans is very large, and often almost the entire image is covered with humans. Therefore, they cannot be ignored in the segmentation. Secondly, even if segmentation could be achieved by ignoring the humans, owing to the low camera angle and close distance, the heads would overlap excessively with invalid regions, and therefore, they would be masked out.

8.2 MTSM model parameters

Our aim was to delve deeper into the inner workings of the MTSM models. We were interested in understanding how these are realized, and how they differ from training two independent networks for crowd counting and segmentation apart from the obvious memory and/or computational requirements. We were dealing with two different tasks that could either be intertwined or not. The latter case means that inside the network, two parallel subnetworks are realized and the channels from the one do not affect the output of the other. However, if the two tasks are intertwined, there cannot be such separation and the parameters affect both outputs.

The existence of two separable subnetworks would mean that the current architectures for crowd counting are excessive and the same accuracy can be achieved with a substantially smaller model. However, our experiments demonstrated that this is not the case. If there were two subnetworks, the parameters in the final layer would have to be separable into two groups by affecting either one output or the other, which we investigated. For every output channel in the final layer, we calculated the absolute mean of its parameters over the input channels. Thereafter, we verified whether there were any input channels with parameters that were thousands of times smaller than the mean. If so, the given output channel would barely be affected by those input channels.

We found that only a handful of input channels exhibit such behavior, and most of these do not affect any output channels, regardless of the type. Therefore, we can confirm that the two tasks are strongly intertwined. In fact, there is no difference between them in the sense that both tasks are performed using the same feature set, and they are only divided by the form of the output. Our point is that well-designed architecture performs effectively in more than one image processing task, and not just in a very particular narrow field. Therefore, combining the task of crowd counting with other computer vision tasks to save on computation time offers development potential and future research possibilities.

Furthermore, whereas region-based masking is unfeasible for different types of datasets, the multitask segmentation method can be applied to any human dataset. Unfortunately, to the best of our knowledge, the TH2020 dataset is the only one that includes both human head annotation ground-truth and semantic segmentation ground-truth; therefore, we cannot conduct experiments on any other datasets.

8.3 Considerations about the dataset

We designate the dataset as helicopter footage; however, the attributes of the dataset are what more significant. The most important of these is the high altitude. High enough so that perspective distortions and size differences can be markedly reduced

with a steep camera angle while still covering large areas, but not so high that the targets become too small for detection. While technologically it is feasible to create such footage with drones, there are a few reasons why the use of helicopters is the only option currently.

In most countries, the use of airspace is very strictly regulated. Moreover, in many large metropolitan regions, commercial or public use of drones is prohibited. In regions with more flexible regulations, the use of drones is tied to special permissions but only allowed up to a very limited altitude. Certainly, if authorities are considering putting aerial surveillance systems in place, they may change these regulations for cost efficiency. However, it is still important to keep in mind technological requirements. Due to the high altitude, the camera equipment must have high zooming capabilities to achieve sufficient resolution. Moreover, precise aiming equipment is required to find and keep the target region in focus. These devices greatly increase the payload weight, thereby requiring larger drones with larger energy consumption and shorter flight times.

We also want to mention that there are practical applications where the use of drones is not possible at the current technological level. One such example is a search and rescue operation, where simultaneous crowd counting and semantic segmentation is a helpful tool to identify people in immediate danger. For the rescue, helicopter is the only option and splitting the operation into search and rescue parts to save money may cost time and, consequently, human lives.

8.4 Transfer capabilities

The dataset I have presented is from a well specified relatively small geographic location. A question arises: would the methods also work in other geographical locations as well? As I mentioned earlier images used for training are from different video sequences than the images used for evaluation. Therefore, we are confident that the accuracy would not drop in other districts of Tokyo, or many other Japanese cities. However, in our first attempts we did a random split to create training

and evaluation sets, that is both sets contained images from all video sequences, and this way we achieved significantly lower error rates. This suggests that the models develop some level of context dependency during training. Consequently, it is possible that the accuracy gets lower in cities with different architecture styles. For example, our model trained on the TH2020 dataset may struggle in European historical cities with medieval structures. There is no telling without additional footage from cities with various architecture styles. However, if the accuracy drops significantly in other locations the model can be adapted to the new surroundings with fine-tuning. Of course, this would require annotated footage from the new location, which is an additional cost that people try to avoid.

8.5 Assumptions regarding our CMC method

When we decided how to solve CMC in our algorithms, we made two assumptions which may seem to be strong limitations at first look. Firstly, a predefined bounding box is required for the key-points in the first frame, and secondly the key-points must be always visible in the video. However, if one ponders more on this one realizes that these are not limitations but requirements for the aerial pedestrian counting.

To have any real-life use for pedestrian counting other than providing entertainment to children at the fair, images must be aligned in some way to the real world. In case of CCTV cameras this is a simpler problem, they are installed on buildings or streetlights, their position is known and fixed. In case of moving cameras this alignment must be established in some way. Well known surveying methods and simultaneous localization and mapping (SLAM) can be used to solve this. For instance, camera localization can be solved with GPS coordinates, and for orientation the camera can use certain landmark points which can be detected by well-trained object detectors. After this the same landmark points can be used as key-points for CMC.

As for the other assumption, if one wishes to protect personal privacy using MOT, trajectory information must be discarded because starting and destination

points can be used for identification. Only line-of-interest counting data can be used. However, to achieve counting over a line its endpoints must be kept in the field of vision always.

8.6 Relevance of the accuracy metrics

In my thesis I use the metrics that are common for evaluating pedestrian counting to establish that helicopter footage is useful for this task. However, from an engineering perspective their relevance is strongly questionable. In engineering design, the common approach for setting reference values is to execute many experiments for measuring said value, and then select some very high (or low) percentage quantile. The reason for this is to make sure that whenever there is a mistake it is on the safe side. As opposed to this, in machine learning everything is evaluated by average values, which does not say anything about the reliability of the method. I can hardly imagine that in autonomous driving there are requirements like for instance “the vehicle has to detect any object in its way 3 seconds prior to collision on average”, because this requirement does not ensure that the vehicle does not hit any pedestrian.

For some reason in public research these average metrics became widespread. One possible reason for this is that the main driving engine for the field of machine learning are the various challenges, where victors must be declared and for that purpose one single metric is required. The more complicated a single metric is the more difficult it is to make it fair. Nevertheless, average metrics without deviation cannot guarantee failsafe operation, which means that rankings on deep learning benchmark leaderboards are inadequate to determine which method is good for an engineering application and which is not. Therefore, if anyone hopes to use deep learning in a safety critical engineering applications, the evaluation standards need to change.

8.7 Unanswered questions

In my thesis I have presented some methods that we have developed specifically for helicopter dataset. I have shown that these methods allow us to use helicopter footage in pedestrian counting with similar accuracy to that of low altitude footage. I did this by showing experimental results on a very specific dataset, and while the different video sequences show variation in camera parameters and lighting conditions, we do not know what exactly those are, and what the extent of them, in which our methods can be still helpful, is. We also do not know what kind of stabilizer, if any, was used to dampen random camera motion. However, this information is essential for planning flight paths. To answer these questions more well documented footage is required with a wide variety of parameters.

Additionally, the footage only presents pedestrians in an ordinary environment. However, in rescue operations the monitored area may be disaster stricken, which results in a very different scenery with collapsed buildings and damaged structures, and therefore may result in reduced accuracy. In such a situation it is also likely that survivors are stuck under rubble and only parts of their body are visible, or nothing at all. This obviously makes the task more difficult, if not impossible, therefore further studies are required in this type of application.

Another open question is the georeferencing of the footage. The footage is not very useful unless it is aligned with a real-world coordinate system somehow. I personally could not identify half of the scenes until someone showed them to me on a map. This task already has very strong technological support, therefore it is more of an engineering problem, however new cost-efficient methods can be researched alongside the existing ones. Nevertheless, it must be solved if one hopes to use this technology in real applications.

Finally, I have presented comparable accuracies, however we are confident that these are not the best possible results that can be achieved, because we did not spend much time with fine-tuning and parameter optimization. For such a purpose a challenge may be held for the dataset to find the most accurate and fastest methods.

Chapter 9

Conclusions

I have shown that helicopter footage is useful in the task of CNN based pedestrian counting, by pointing out its numerous advantages compared to typical pedestrian counting datasets and showing that state-of-the-art methods can achieve comparable accuracy on this type of data.

I have presented a large-scale human dataset obtained from a helicopter. Based on the experimental results, the dataset itself is not particularly challenging owing to the lack of scale differences and perspective distortions. However, it offers a viewpoint that is not covered by other conventional crowd counting benchmark datasets. Moreover, the area covered by one image is much larger than that of conventional crowd counting datasets, meaning that the same region requires less computational resources to process. It also protects personal privacy because people are not recognizable from such high altitude.

I have overviewed existing static and dynamic pedestrian counting methods and presented our work about the evaluation of their performance on helicopter footage in both pretrained scenario and when trained on helicopter footage. In the pretrained case we have found that the accuracy is not good for real-life applications. From this I have concluded that a helicopter-based training dataset is necessary for good performance on helicopter footage.

After training on helicopter-based dataset, the methods showed comparable accuracy, however we have identified issues which are not present or much less dominant

in ordinary footage. These are the issues of lack of deeper semantic information in DME methods and the strong sudden camera motions in MOT methods. I have presented solution for these problems.

Our contribution is threefold. First, we have developed a large-scale helicopter based pedestrian dataset called Tokyo Hawkeye 2020 (TH2020). It comprises 6237 static images and six 20 seconds long video sequences. The dataset is publicly accessible at <https://github.com/sekilab/TokyoHawkeye2020>.

Second, we have developed a static crowd counting method that combines density map estimation with semantic segmentation, which can reduce the error of static counting in helicopter footage. We have managed to achieve 0.2 relative MAE on the TH2020 dataset. Incidentally, we have achieved 0.79 AP@0.5IoU for object detection with an unaltered CenterNet architecture.

Finally, we have developed a simple CMC functionality through 2D partial affine transformation for dynamic methods. It can be applied to multiple state-of-the-art MOT algorithms to increase accuracy on helicopter footage, especially in online use-cases. Furthermore, we introduced a concurrent pipelined implementation for the algorithms that can reduce execution time by up to 50%. The best MOT accuracy we have managed to achieve on TH2020 video sequences is 59.30 MOTA.

For future study, an analysis of camera parameters and environmental conditions is necessary to determine the extents in which our methods can be used. Furthermore, georeferencing must be solved. Finally, an investigation of the accuracy in disaster-stricken areas is required.

Bibliography

- [1] “Isprs aerial dataset with 2d semantic labeling,” <http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html>, accessed on June 26, 2020.
- [2] “Mot benchmark,” <https://motchallenge.net/>, accessed on Nov 30, 2020.
- [3] N. Audebert, B. L. Saux, and S. Lefèvre, “Beyond rgb: Very high resolution urban remote sensing with multimodal deep networks,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 140, pp. 20 – 32, 2018, geospatial Computer Vision. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0924271617301818>
- [4] R. Bahmanyar, E. Vig, and P. Reinartz, “Mrcnet: Crowd counting and density map estimation in aerial and ground imagery,” 2019.
- [5] M. Barekatain, M. Marti, H.-F. Shih, S. Murray, K. Nakayama, Y. Matsuo, and H. Prendinger, “Okutama-action: An aerial view video dataset for concurrent human action detection,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.
- [6] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, “Multiple object tracking using k-shortest paths optimization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1806–1819, 2011.
- [7] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: The clear mot metrics,” *EURASIP Journal on Image and Video Processing*, vol. 2008, no. 1, pp. 1–10, 2007.
- [8] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, “Fully-convolutional siamese networks for object tracking,” in *Computer Vision – ECCV 2016 Workshops*, G. Hua and H. Jégou, Eds. Cham: Springer International Publishing, 2016, pp. 850–865.
- [9] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and real-time tracking,” in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3464–3468.
- [10] W. Brendel, M. Amer, and S. Todorovic, “Multiobject tracking as maximum weight independent set,” in *CVPR 2011*, 2011, pp. 1273–1280.

- [11] X. Cao, Z. Wang, Y. Zhao, and F. Su, “Scale aggregation network for accurate and efficient crowd counting,” in *The European Conference on Computer Vision (ECCV)*, September 2018.
- [12] A. Chan, Z.-S. Liang, and N. Vasconcelos, “Privacy preserving crowd monitoring: Counting people without people models or tracking,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–7, 01 2008.
- [13] L. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *CoRR*, vol. abs/1706.05587, 2017. [Online]. Available: <http://arxiv.org/abs/1706.05587>
- [14] X. Chen, Y. Bin, N. Sang, and C. Gao, “Scale pyramid network for crowd counting,” in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019, pp. 1941–1950.
- [15] W. Choi, “Near-online multi-target tracking with aggregated local flow descriptor,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [17] G. Csönde, Y. Sekimoto, and T. Kashiwayama, “Congestion analysis of convolutional neural network-based pedestrian counting methods on helicopter footage,” 2019.
- [18] —, “Crowd counting with semantic scene segmentation in helicopter footage,” *Sensors*, vol. 20, no. 17, p. 4855, Aug 2020. [Online]. Available: <http://dx.doi.org/10.3390/s20174855>
- [19] J. Dai, K. He, Y. Li, S. Ren, and J. Sun, “Instance-sensitive fully convolutional networks,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 534–549.
- [20] J. Dai, K. He, and J. Sun, “Instance-aware semantic segmentation via multi-task network cascades,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [21] J. Dai, Y. Li, K. He, and J. Sun, “R-fcn: Object detection via region-based fully convolutional networks,” in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016, pp. 379–387. [Online]. Available: <https://proceedings.neurips.cc/paper/2016/file/577ef1154f3240ad5b9b413aa7346a1e-Paper.pdf>

- [22] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05) - Volume 1 - Volume 01*, ser. CVPR ’05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893. [Online]. Available: <http://dx.doi.org/10.1109/CVPR.2005.177>
- [23] P. Dollár, C. Wojek, B. Schiele, P. Perona, and T. Darmstadt, “Pedestrian detection: A benchmark,” in *In CVPR*, 2009.
- [24] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, “Flownet: Learning optical flow with convolutional networks,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [25] G. D. Evangelidis and E. Z. Psarakis, “Parametric image alignment using enhanced correlation coefficient maximization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1858–1865, 2008.
- [26] G. Farnebäck, “Two-frame motion estimation based on polynomial expansion,” in *Image Analysis*, J. Bigun and T. Gustavsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 363–370.
- [27] K. Fathian, J. P. Ramirez-Paredes, E. A. Doucette, J. W. Curtis, and N. R. Gans, “Quest: A quaternion-based approach for camera motion estimation from minimal feature points,” *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 857–864, 2018.
- [28] G. Fu, C. Liu, R. Zhou, T. Sun, and Q. Zhang, “Classification for high resolution remote sensing imagery using a fully convolutional network,” *Remote Sensing*, vol. 9, no. 5, p. 498, May 2017. [Online]. Available: <http://dx.doi.org/10.3390/rs9050498>
- [29] J. Gao, Q. Wang, and X. Li, “PCC net: Perspective crowd counting via spatial convolutional network,” *CoRR*, vol. abs/1905.10085, 2019. [Online]. Available: <http://arxiv.org/abs/1905.10085>
- [30] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [31] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.
- [32] D. Guo, K. Li, Z.-J. Zha, and M. Wang, “Dadnet: Dilated-attention-deformable convnet for crowd counting,” in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM ’19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1823–1832. [Online]. Available: <https://doi.org/10.1145/3343031.3350881>

- [33] R. Hartley and H. Li, “An efficient hidden variable approach to minimal-case camera motion estimation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, pp. 2303–2314, 2012.
- [34] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, “Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture,” in *Computer Vision – ACCV 2016*, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds. Cham: Springer International Publishing, 2017, pp. 213–228.
- [35] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.
- [36] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask r-cnn,” in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [37] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.
- [38] Hongdong Li and R. Hartley, “Five-point motion estimation made easy,” in *18th International Conference on Pattern Recognition (ICPR’06)*, vol. 1, 2006, pp. 630–633.
- [39] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [40] H. Idrees, I. Saleemi, C. Seibert, and M. Shah, “Multi-source multi-scale counting in extremely dense crowd images,” in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR ’13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 2547–2554. [Online]. Available: <https://doi.org/10.1109/CVPR.2013.329>
- [41] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “Flownet 2.0: Evolution of optical flow estimation with deep networks,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [Online]. Available: <http://lmb.informatik.uni-freiburg.de/Publications/2017/IMSKDB17>
- [42] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *CoRR*, vol. abs/1502.03167, 2015. [Online]. Available: <http://arxiv.org/abs/1502.03167>
- [43] S. Jiang, X. Lu, Y. Lei, and L. Liu, “Mask-aware networks for crowd counting,” *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2019.

- [44] A. Kendall, Y. Gal, and R. Cipolla, “Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [45] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, “Multiple hypothesis tracking revisited,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [46] M. Kuse and Shaojie Shen, “Robust camera motion estimation using direct edge alignment and sub-gradient method,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 573–579.
- [47] L. Leal-Taixé, A. Milan, I. D. Reid, S. Roth, and K. Schindler, “Motchallenge 2015: Towards a benchmark for multi-target tracking,” *CoRR*, vol. abs/1504.01942, 2015. [Online]. Available: <http://arxiv.org/abs/1504.01942>
- [48] S. Li, L. Yuan, J. Sun, and L. Quan, “Dual-feature warping-based motion model estimation,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 4283–4291.
- [49] Y. Li, X. Zhang, and D. Chen, “Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [50] Li Zhang, Yuan Li, and R. Nevatia, “Global data association for multi-object tracking using network flows,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [51] K. Lin, N. Jiang, S. Liu, L.-F. Cheong, M. Do, and J. Lu, “Direct photometric alignment by mesh deformation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [52] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [53] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, “Focal loss for dense object detection,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [54] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.
- [55] F. Liu, M. Gleicher, H. Jin, and A. Agarwala, “Content-preserving warps for 3d video stabilization,” *ACM Trans. Graph.*, vol. 28, no. 3, jul 2009. [Online]. Available: <https://doi.org/10.1145/1531326.1531350>

- [56] L. Liu, Z. Qiu, G. Li, S. Liu, W. Ouyang, and L. Lin, “Crowd counting with deep structured scale integration network,” in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- [57] L. Liu, H. Wang, G. Li, W. Ouyang, and L. Lin, “Crowd counting using deep recurrent spatial-aware network,” *CoRR*, vol. abs/1807.00601, 2018. [Online]. Available: <http://arxiv.org/abs/1807.00601>
- [58] P. Liu, M. Lyu, I. King, and J. Xu, “Selfflow: Self-supervised learning of optical flow,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [59] S. Liu, Y. Zhao, F. Xue, B. Chen, and X. Chen, “Deepcount: Crowd counting with wifi via deep learning,” *CoRR*, vol. abs/1903.05316, 2019. [Online]. Available: <http://arxiv.org/abs/1903.05316>
- [60] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37.
- [61] W. Liu, M. Salzmann, and P. Fua, “Context-aware crowd counting,” *CoRR*, vol. abs/1811.10452, 2018. [Online]. Available: <http://arxiv.org/abs/1811.10452>
- [62] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [63] Y. Lu, C. Lu, and C.-K. Tang, “Online video object detection using association lstm,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [64] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *IJCAI*, 1981.
- [65] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan, “Discriminative correlation filter with channel and spatial reliability,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [66] Y. Lyu, G. Vosselman, G.-S. Xia, A. Yilmaz, and M. Y. Yang, “Uavid: A semantic segmentation dataset for uav imagery,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 165, pp. 108 – 119, 2020. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0924271620301295>

- [67] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, “High-resolution aerial image labeling with convolutional neural networks,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 12, pp. 7092–7103, 2017.
- [68] D. Marmanis, J. D. Wegner, S. Galliani, K. Schindler, M. Datcu, and U. Stilla, “Semantic segmentation of aerial images with an ensemble of cnss,” in *ISPRS Congress*, L. Halounova, K. Schindler, A. Limpouch, V. Šafář, T. Pajdla, H. Mayer, S. O. Elberink, C. Mallet, F. Rottensteiner, J. Skaloud, U. Stilla, and M. Brédif, Eds., vol. III-3. Copernicus Publications, 2016, pp. 473–480. [Online]. Available: <https://elib.dlr.de/108960/>
- [69] M. Marsden, K. McGuinness, S. Little, and N. E. O’Connor, “Resnetcrowd: A residual deep learning architecture for crowd counting, violent behaviour detection and crowd density level classification,” in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2017, pp. 1–7.
- [70] A. Milan, L. Leal-Taixé, I. D. Reid, S. Roth, and K. Schindler, “MOT16: A benchmark for multi-object tracking,” *CoRR*, vol. abs/1603.00831, 2016. [Online]. Available: <http://arxiv.org/abs/1603.00831>
- [71] Ming Liu, Jue Jiang, Zhenqi Guo, Zenan Wang, and Yang Liu, “Crowd counting with fully convolutional neural network,” in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 953–957.
- [72] S. Paisitkriangkrai, J. Sherrah, P. Janney, and A. Van-Den Hengel, “Effective semantic pixel labelling with convolutional networks and conditional random fields,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2015.
- [73] J. Paul Cohen, G. Boucher, C. A. Glastonbury, H. Z. Lo, and Y. Bengio, “Count-ception: Counting by fully convolutional redundant counting,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.
- [74] J. Philip, “A non-iterative algorithm for determining all essential matrices corresponding to five point pairs,” *The Photogrammetric Record*, vol. 15, no. 88, pp. 589–599, 1996. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1111/0031-868X.00066>
- [75] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, “Globally-optimal greedy algorithms for tracking a variable number of objects,” in *CVPR 2011*, 2011, pp. 1201–1208.
- [76] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

- [77] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv*, 2018.
- [78] D. Reid, “An algorithm for tracking multiple targets,” *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.
- [79] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015, pp. 91–99. [Online]. Available: <https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf>
- [80] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, “Learning social etiquette: Human trajectory understanding in crowded scenes,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 549–565.
- [81] A. Sadeghian, A. Alahi, and S. Savarese, “Tracking the untrackable: Learning to track multiple cues with long-term dependencies,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [82] D. B. Sam, S. Surya, and R. V. Babu, “Switching convolutional neural network for crowd counting,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4031–4039.
- [83] D. B. Sam and R. V. Babu, “Top-down feedback for crowd counting convolutional neural network,” 2018. [Online]. Available: <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16726/16328>
- [84] M. Shi, Z. Yang, C. Xu, and Q. Chen, “Revisiting perspective information for efficient crowd counting,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [85] Z. Shi, P. Mettes, and C. G. M. Snoek, “Counting with focus for free,” in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- [86] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *International Conference on Learning Representations*, 2015.
- [87] V. A. Sindagi and V. M. Patel, “Cnn-based cascaded multi-task learning of high-level prior and density estimation for crowd counting,” in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Aug 2017, pp. 1–6.
- [88] V. A. Sindagi and V. M. Patel, “Generating high-quality crowd density maps using contextual pyramid cnns,” in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

- [89] —, “Inverse attention guided deep crowd counting network,” *CoRR*, vol. abs/1907.01193, 2019. [Online]. Available: <http://arxiv.org/abs/1907.01193>
- [90] V. A. Sindagi, R. Yasarla, and V. M. Patel, “Pushing the frontiers of unconstrained crowd counting: New dataset and benchmark method,” in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- [91] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, “Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [92] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [93] J. Wan, W. Luo, B. Wu, A. B. Chan, and W. Liu, “Residual regression with semantic prior for crowd counting,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [94] Z. Wang, L. Zheng, Y. Liu, and S. Wang, “Towards real-time multi-object tracking,” *CoRR*, vol. abs/1909.12605, 2019. [Online]. Available: <http://arxiv.org/abs/1909.12605>
- [95] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 3645–3649.
- [96] H. Xiong, H. Lu, C. Liu, L. Liu, Z. Cao, and C. Shen, “From open set to closed set: Counting objects by spatial divide-and-conquer,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.
- [97] Z. Yan, Y. Yuan, W. Zuo, X. Tan, Y. Wang, S. Wen, and E. Ding, “Perspective-guided convolution networks for crowd counting,” in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- [98] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan, “Poi: Multiple object tracking with high performance detection and appearance feature,” in *Computer Vision – ECCV 2016 Workshops*, G. Hua and H. Jégou, Eds. Cham: Springer International Publishing, 2016, pp. 36–42.
- [99] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, “Deep layer aggregation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [100] A. Zhang, J. Shen, Z. Xiao, F. Zhu, X. Zhen, X. Cao, and L. Shao, “Relational attention network for crowd counting,” in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

- [101] F. Zhang and F. Liu, “Parallax-tolerant image stitching,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.
- [102] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, “Fairmot: On the fairness of detection and re-identification in multiple object tracking,” 2020.
- [103] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, “Single-image crowd counting via multi-column convolutional neural network,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [104] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, “M2det: A single-shot object detector based on multi-level feature pyramid network,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 9259–9266, Jul. 2019. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/4962>
- [105] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, “Scalable person re-identification: A benchmark,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [106] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu, “Scale-transferrable object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [107] X. Zhou, D. Wang, and P. Krähenbühl, “Objects as points,” *CoRR*, vol. abs/1904.07850, 2019. [Online]. Available: <http://arxiv.org/abs/1904.07850>
- [108] P. Zhu, L. Wen, X. Bian, L. Haibin, and Q. Hu, “Vision meets drones: A challenge,” *arXiv preprint arXiv:1804.07437*, 2018.
- [109] H. Zou, Y. Zhou, J. Yang, and C. J. Spanos, “Device-free occupancy detection and crowd counting in smart buildings with wifi-enabled iot,” *Energy and Buildings*, vol. 174, pp. 309 – 322, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0378778817339336>
- [110] 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.