Multi-modal line-crossing using a multi-headed model for flow-enhanced density maps and a pixel-wise accumulator

Jan Erik van Woerden

30 October 2020

Contents

Chapter 1

Introduction

In recent years the amount of surveillance camera's has increased immensely to a point that it is hard to supervise them all manually by humans. Since the upcoming of surveillance camera's a lot of research is done to automate the information extraction from those camera's [Sreenu and Saleem Durai, 2019].

A widely researched area for extracting information from camera's is Crowd Counting [Chan and Vasconcelos, 2008, Wang et al., 2020, Li et al., 2018, Fang et al., 2019, Liu et al., 2019b]. Where the amount of pedestrians present in the frame is counted. Where low density pedestrians can be easily counted by general object recognition, higher density area's need specialized methods to accurately count the amount of pedestrians [Zhang et al., 2016].

While Crowd Counting only focusses on counting the exact amount of pedestrians in a frame, it doesn't take into account the amount of pedestrians walked by over time. This is no issue when camera's are present in the whole area of interest, however with large areas this becomes a much bigger issue. In for example a shop it would be much more convenient to count the amount of customers inside the shop by only tracking the customers walking inside and outside the shop, instead of having cameras in the whole shop.

This research area of Crowd Crossing Counting is much less researched [Zhao et al., 2016, Zheng et al., 2019]. By adding Flow Estimation to the Crowd Density (Crowd Counting) the flow of pedestrians can be obtained and the amount of people going inside and outside can be measured.

In early papers on Crowd Crossing Counting prediction was done using keypoint extraction and feeding the keypoints in a regression model [Ma and Chan, , Ma and Chan, 2013]. More recently the introduction of Convolutional Neural Networks was made into the field of both Crowd Counting [Zhang et al., 2016, Liu et al., 2019b, Li et al., 2018, Wang et al., 2020] and Flow Estimation [Sun et al., , Dosovitskiy et al., 2015]. Which sparked the research in those fields.

In Crowd Crossing Counting the amount of research done using Neural Networks is limited [Zhao et al., 2016, Cao et al.,]. New research in both Crowd Counting and Flow estimation provides lot's of new opportunities to improve the State-of-the-Art of Line Crossing. New research also adds some new challenges, which we try to solve as well in this thesis.

Current benchmarks for Crowd Crossing Counting still make use of low resolution datasets [Ma and Chan, 2013, Ma and Chan,] (256x156). In Crowd Counting higher resolution datasets are already available. In this thesis Crowd Crossing

Counting labeling is provided and published for the Fudan-Shanghai dataset (Full HD, 25FPS) [Fang et al., 2019]. Additionally to show the generality and ability to use the proposed method cross-domain the AI City challenge dataset is used, on which crossings of cars is counted.

Secondly this thesis presents two novel models. The first model is a multi-headed network which which improves on Crowd Crossing Counting papers and the research done in the field of Crowd Counting and Flow Estimation. The second model is an extension on the first model which introduces a novel method to enhance the crowd density prediction by using the flow estimation as extra information.

Thirdly this thesis presents a method to further enhance the combining of both the Density Map and Flow Estimation to a more reliable method to predict the Crowd Crossing Counting.

Lastly thorough research is done on the usability of the presented system in real world scenario's.

1.1 Thesis outline

The rest of this thesis is divided into the next chapters:

- **Background**, explains several fields to understand the starting point for this thesis.
- **Related work**, which explains more in depth related work which is used in this thesis.
- **Method**, presents the method of the proposed solution.
- **Implementation**, presents the hyperparameters and evaluation methods.
- **Datasets**, presents the used datasets and used approach to label the proposed new datasets.
- **Results**, discusses the results of the experiments.
- **Conclusion**, wraps it up and summarizes what we can conclude.

Chapter 2

Background

In this chapter a selection of terms is explained which gives a basis to understand the rest of this thesis. This background is created with the assumption that the reader has a basic background in Machine Learning and (Convolutional) Neural Networks.

2.1 Region of Interest

The Region of Interest problem is a widely studied problem in which the goal is to estimate the amount of pedestrians given a single image. Directly predicting the count given a Neural Network is a hard task, because of the lack of supervision, this would require a large amount of samples to accurately solve this task. All recent State-of-the-Art methods therefore use an intermediate representation to give the model enough supervision to perform Crowd Counting with a low amount of training samples.

In the early days of Crowd Counting several methods have been proposed which use *detection-based* methods to estimate the amount of pedestrians [Dalal and Triggs, 2005, Dollár et al., 2012] . Several papers were introduced which tried to detect only the head [Subburaman et al., 2012] and others tried to focus on general part detection [Wu and Nevatia, 2007, Lin and Davis, 2010]. These methods rely on individually detecting the pedestrians. This becomes much harder when occlusion of the pedestrians start to happen. This is why the performance of these methods start to degrade when the density of the pedestrians in an image start to increase.

Later papers introduced a *regression-based* solution, which tries to predict the amount of pedestrians in crowd blobs [Chan and Vasconcelos, 2009, Idrees et al., 2013, Zheng et al., 2019]. Using SVM or other regressor methods and several features such as the amount of foregrounds pixels of the blob and detected key points the count inside crowd blobs were predicted. Regression based solutions were an improvement over the detection methods, but still lack the capabilities to estimate pedestrians counts in highly occluded areas.

2.1.1 Density Map

Replace to Fudan dataset

With the introduction of Convolutional Neural Networks in the field of Crowd Counting density maps were proposed as well to count pedestrians [Zhang et al., 2016,

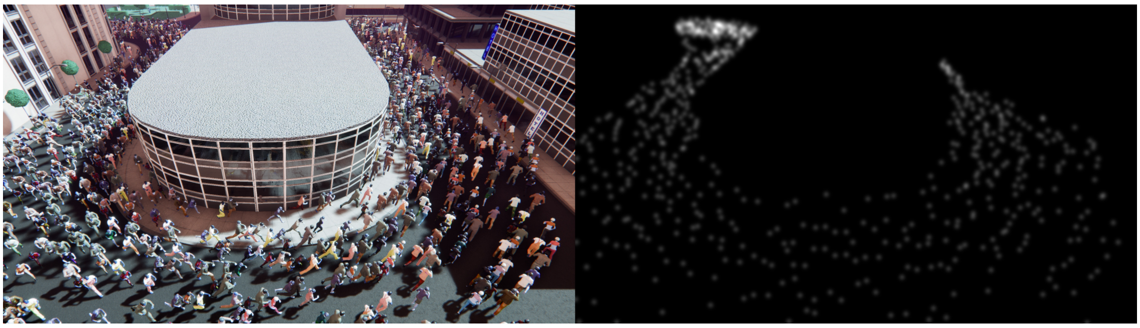


Figure 2.1: Example of generated density map on the right side, for the left image

Liu et al., 2019b, Li et al., 2018]. A *density map* (Figure 2.1) used for Region of Interest is a map which represents the density of pedestrians of each pixel. The density map is generated by taking the locations of each pedestrian ($p = \begin{bmatrix} x_p \\ y_p \end{bmatrix}$ in equation 2.1) and place those locations on the the density map.

Individual dots are very hard for a Neural Network to detect correctly and are proned to errors. To circumvent this a Gaussian shaped circle is created around this location, still with with a sum of 1. The amount of pedestrians in the frame can be extracted from the density map by taking the sum over all the pixels of the density map (Equation 2.2, where $D_t(p)$ is the density for location $p = \begin{bmatrix} x_p \\ y_p \end{bmatrix}$ for trainings frame t).

$$D_t(p) = \frac{1}{2\pi\sigma_p^2} \sum_{p \in P} e^{\frac{(x_p - x)^2 + (y_p - x)^2}{-2\sigma_p^2}} \tag{2.1}$$

$$C_t = \sum_{p \in P} D_t(p) \tag{2.2}$$

Several methods have been presented to optimize the generation of density maps [Zhang et al., 2016, Li et al., 2018, Wan and Chan, 2019]. For most medium dense frames the difference in methods is minimal. Often in benchmarks with medium dense frames a fixed sigma is used ($\sigma_p = \sigma_i$ in equation 2.1). For highly dense frames the use of different methods can have a difference, especially when the difference in size between close pedestrians and pedestrians in the background is large [Li et al., 2018].

A bit more papers of recent improvements

2.2 Flow Estimation

The research which is done on the Flow Estimation problem is widely used. Approaches on this topic can be used in a wide range of applications which makes it very interesting. Already in the early 1980's Horn and Schunck [Horn and Schunck, 1981] published the first paper which tried to predict flow. Since then lot's of different approaches have been published [Mémin and Pérez, 1998, Bruhn et al., 2005, Brox et al., 2014]. Long conventional mathematical approaches have ruled the flow estimation field. Later also learnable models were introduced [Pock et al., 2008, Wedel et al., 2009].

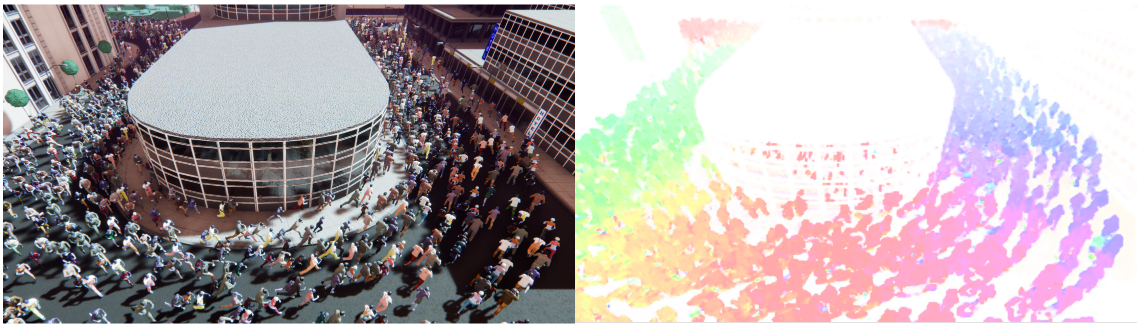


Figure 2.2: Example of generated velocity map on the right side, for the left image

Replace to Fudan Dataset

Recent papers however make use of Convolutional Neural Network based models [Dosovitskiy et al., 2015, Ilg et al., 2016, Sun et al., , Ranjan and Black, 2017, Hui et al., 2018]. These model predict pixel-precise velocity maps. The *velocity map* (Figure 2.2) is a map which predict per pixel of the frame the amount of movement to another location. In equation 2.3, $V_t(p)$ shows the velocity map as a difference between the location of the pixel in the current frame (p) and the location of this pixel in the next frame ($N_t(p)$).

$$V_t(p) = N_t(p) - \begin{bmatrix} x_p \\ y_p \end{bmatrix} \tag{2.3}$$

Creating a real world dataset that utilizes the power of pixel-wise flow estimation is very hard [Dosovitskiy et al., 2015]. There are no real world devices which could capture both video and create pixel perfect ground-truths to train the flow estimation models on. Most of the flow estimation benchmarks are therefore generated videos. Computer 3D-engines make it possible to generate pixel-perfect flow estimation based on the generated videos in the engine.

However the large gap in domain and scene between the generated datasets and real world applications [Liu et al., 2008]. Recent supervised papers [Dosovitskiy et al., 2015, Sun et al.,] tend to overfit on the datasets. Therefore perform rather poor on real world applications. One solution and promising direction is unsupervised learning [Yu et al., 2016, Janai et al., 2018, Liu et al., 2019a, Liu et al.,]. Early papers only predicted non-occluded pixels [Yu et al., 2016, Janai et al., 2018], but recent papers use methods to estimate occluded pixels as well [Liu et al., 2019a, Liu et al.,]. Further details about these methods in related work.

2.3 Line of Interest

Line of Interest is very similar to Region of Interest. Where Region of Interest is the interest of the amount of people inside the ROI, the Line of Interest is the focus on the amount of pedestrians that cross the specified line during a certain timeframe. This LOI is defined as a single line between two points p_1 and p_2.

Add an image with a line drawn inside the TUB dataset

With the Line of Interest problem the goal is to give the amount of pedestrians crossing of each side given a set of frames (a pre-captured video or video stream). The output of the prediction should give two numbers c_1 and c_2 which are the amount of pedestrians crossing from each side.

Only a handful of papers are published about Line of Interest. In the earlier papers [Ma and Chan, , Cao et al.,], slicing was a widely used approach to estimate the Line of Interest. With slicing a small area, called the LOI area, is taken around the LOI. Over a set of consecutive frames each slice of the frame was taken and stitched together into a single image. On the images slow walking pedestrians appear rather wide and fast walking pedestrians shallow. By counting the amount of pedestrians present on the stitched image, the total amount of pedestrians crossing the line can be counted.

The area is defined by all the pixels that have a maximum distance to the LOI of d and can be projected on the LOI. When projected, the pixels fall between p_1 and p_2.

Recent papers discard this method [Zhao et al., 2016, Zheng et al., 2019], because it makes it hard to track pedestrian with different speeds and walking in different directions give artifacts which make it hard to track those pedestrians [Zhao et al., 2016]. The slicing method is replaced with an actual frame by frame prediction method. Using two consecutive frames the amount of pedestrians crossing the line is measured. These newer methods predict both location and direction of the pedestrian.

Based on these new papers, the problem of Line of Interest is divided into three separate problems. Locating the pedestrians (Region of Interest), estimate the direction (Flow Estimation) of the pedestrians and combining these two streams of information into the count for Line of Interest. Further details of this approach will be provided in related work.

Chapter 3

Related Work

In this chapter we try to explain a couple of key papers on which the proposed methods are build.

3.1 Crowd Counting

3.1.1 CSRNet

Since Zhang et al. [Zhang et al., 2016] lot's of research was done on better predicting the density maps for Crowd Counting. Zhang et al. and following proposed several multi-column models to predict pedestrians of different sizes. However Li et al. [Li et al., 2018] changed this with the proposal of CSRNet. Instead of using multi-column models it uses dilated kernels. This method beat the multi-column models on several benchmarks.

The advantage of dilated kernels is the increase of the reception field without increasing the amount of parameters and layers in the network. As show in figure 3.1 the dilation rate is the distance between each filter input of the kernel. By increasing the dilation rate the size of area the kernel covers without increasing the amount of parameters of a small-sized kernel.

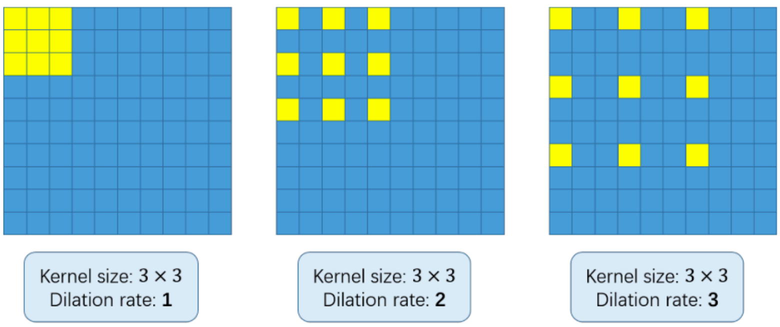


Figure 3.1: Dilation rates on a 3x3 kernel

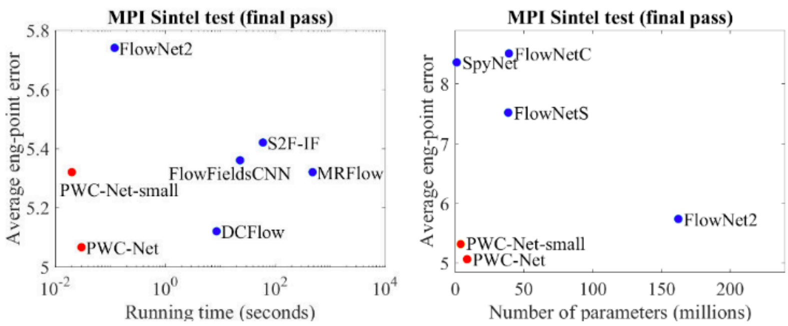


Figure 3.2: PWCNet compared to other existing flow estimation architectures

3.2 Flow Estimation

3.2.1 PWCNet

A popular Flow Estimation network is PWCNet [Sun et al.,]. It uses the original ideas of FlowNet, but it improves FlowNet in a lot of ways. FlowNet traditionally is used to fully predict the flow with a neural network. PWCNet massively reduces the number of weights (See figure 3.2), which results in faster training and much quicker prediction. Additionally the network shows a higher accuracy on several benchmarks.

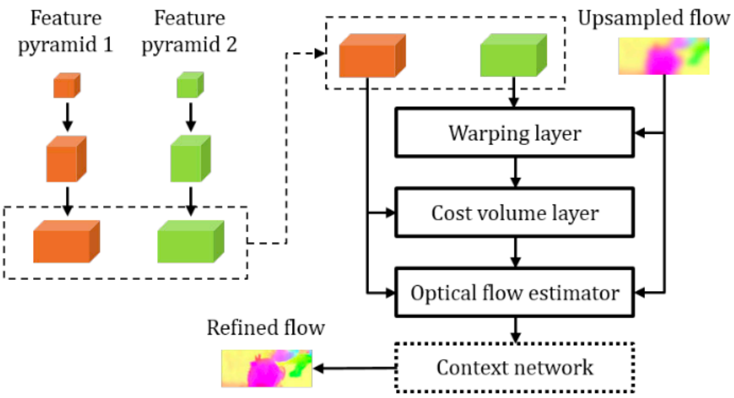


Figure 3.3: PWCNet architecture

PWCNet uses a pyramid shape architecture to predict the velocity map (See figure 3.3). The encoder encodes both input frames seperately into two feature volumes. During decoding several decoding steps are taken. After the initial velocity map estimation, each decoding step upscales the so-far estimated map and further refines the map. At the start of each decoding step the estimated velocity map is used to warp the feature volume of the second frame and correlate this warped volume with the volume of the first image to focus on area of refinement in the velocity map.

3.2.2 DDFlow

In DDFlow [Liu et al., 2019a] a general method is proposed to estimate a velocity map in an unsupervised matter. Earlier methods already proposed several methods to optimize using a photometric loss [Yu et al., 2016] and ignore occluded-pixels during loss calculation [Janai et al., 2018]. However all these earlier methods used handcrafted methods to estimate the occluded-pixels.

The paper [Liu et al., 2019a] proposes a method which learns occluded-pixels using a distillation from unlabeled data without supervision of humans to label the ground truth. Because of the generality of the method, the method can be wrapped around all existing supervised flow estimation architectures.

The method uses a teacher and a student approach to train the occluded pixels. The teacher network is trained on all the non-occluded pixels, with exclusion of the occluded pixels using only the photometric loss with occlusion-awareness. After training the teacher network, the velocity maps of the teacher network are used as ground-truth for the student network.

The student network is then trained on patches of the original predicted velocity map. On the borders of the patches, occluded pixels will appear, because moving pixels from inside the patched frame will move to outside the patched frame. These border pixels will be marked as occluded pixel by the student network, but will be non-occluded pixels according to the ground-truth of the teacher network.

$$\begin{aligned} L_p = &\sum \psi\left(I_1 - I_2^w\right) \odot \left(1 - O_f\right) / \sum \left(1 - O_f\right) \\ &+ \sum \psi\left(I_2 - I_1^w\right) \odot \left(1 - O_b\right) / \sum \left(1 - O_b\right) \end{aligned} \tag{3.1}$$

The photometric loss [Yu et al., 2016] with occlusion-awareness [Janai et al., 2018] proposed in the earlier papers is defined in equation 3.1. Where I_1 and I_2 are the full input frames and I_i^w the backwarped image based on the predicted velocity map. Additionally O_b and O_f are masks for the occluded pixels. These maps are calculated by checking if the mismatch between forward flow and backward flow is too large.

Maybe to background? Additionally add unflow to the related work, Meister2018 for the optimized census loss

For the student model the photometric loss is used together with the loss for occlusion (equation 3.2). The loss of the student model then just $L_p + L_o$. In equation 3.2, $\widetilde{\mathrm{w}}_f$ and $\widetilde{\mathrm{w}}_b$ are the predicted velocity map forward and backward using the student model. w_b^p and w_f^p are the cropped patches from the ground-truth velocity map predicted by the teacher model. M_f and M_b are just defined as the difference of occluded pixels between the ground-truth of the parent and the teacher ($M_f = \mathrm{clip}\left(\widetilde{O}_f - O_f^p, 0, 1\right)$. These pixels are not occluded in the full frame and occluded in the patch. This means that the ground-truth using distillation is available.

$$\begin{aligned} L_o = &\sum \psi\left(\mathrm{w}_f^p - \widetilde{\mathrm{w}}_f\right) \odot M_f / \sum M_f \\ &+ \sum \psi\left(\mathrm{w}_b^p - \widetilde{\mathrm{w}}_b\right) \odot M_b / \sum M_b \end{aligned} \tag{3.2}$$

3.3 Line of Interest

3.3.1 Instant LOI counting

Zhao et al. [Zhao et al., 2016] introduces a new approach to calculate the amount of pedestrians crossing the Line of Interest. Instead of slicing a range of consecutive frames around the Line of Interest and stitching them together, they present a method that directly predicts the amount crossing pedestrians using two consecutive frames.

They use both a density map and velocity map and finally merge them together to obtain the LOI counts. Both the maps are trained fully supervised. Because annotating raw velocity maps is a very hard task, they simplify the velocity map in disk-shaped area's around the pedestrian locations. Annotating each frame is still very demanding, but lowers the amount of labeling significantly in comparing to full velocity maps.

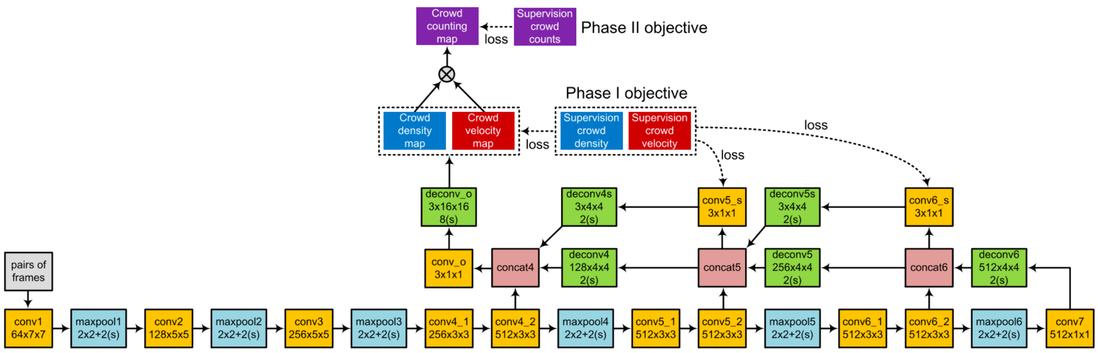


Figure 3.4: Pipeline of Zhao et al. with FlownetSimple as architecture, where in the last layer both the density map as the velocity map are predicted

Tell about re-aligning!

They are as well the first who predict both the density map and the velocity map in a single Convolutional Neural Network. A network they use a simplified model of FlowNet. Because the velocity map and density map are so similar in their location, they predict both the velocity map and the density map in the final layer (See figure 3.4).

To finally merge both the density map and velocity map together they use a pixel-wise approach. Where they turn te density map and velocity map in a directional counting map $C_t = D_t \otimes V_t$. According to the paper the directional counting map gives the amount of pedestrians crossing that pixel between the time of the two frames.

$$
\begin{aligned}
c_{1,t} &= \sum_{\{p|\cos(\theta_p)\geq 0\}} \sqrt{C_{t,x}(p)^2 + C_{t,y}(p)^2} \cdot \cos(\theta_p) \\
c_{2,t} &= \sum_{\{p|\cos(\theta_p)< 0\}} \sqrt{C_{t,x}(p)^2 + C_{t,y}(p)^2} \cdot (-\cos(\theta_p)),
\end{aligned} \tag{3.3}
$$

$$
c_1 = \sum_{\{t|t\in T\}} c_{1,t}, \quad c_2 = \sum_{\{t|t\in T\}} c_{2,t} \tag{3.4}
$$

To calculate per frame pair the amount of pedestrians crossed the line, a set of locations p around the LOI is taken. Then the directional counting map is normalized and summed together in equation 3.3, where θ_p is the angle between $V_t(p)$ and the LOI.

To calculate the total line crosses over a certain timeframe, the results of equation 3.3 can be summed as in equation 3.4.

3.3.2 Region-level LOI counting

Zheng et al. [Zheng et al., 2019] provides a fast method of predicting the Line of Interest. The paper outperforms [Zhao et al., 2016] in the UCSD benchmark. Zheng et al. discusses the problems with Neural Networks and the complexity the models, which makes it hard to run the models real time. It therefore introduces a non-CNN based method based on a SVM, linear regression and the Lucas-Kanade optical flow tracker.

It uses the idea of [Zhao et al., 2016] to discard the slicing method and uses pair-wise prediction and uses the same method to merge density and velocity. Because the methods used in [Zheng et al., 2019] are not on a pixel-level, a method is proposed to bin on a region-level. In equation 3.5 a single velocity $v_{t,r}$ is calculated with a weighted average over all moving keypoints inside the region.

$$v_{t,r} = \frac{\sum_p w_r(p) \cdot v^{(t)}_{towards}(p)}{\sum_p w_r(p)} \tag{3.5}$$

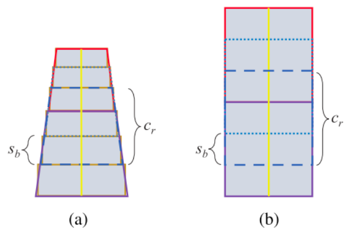


Figure 3.5: Regions around the LOI with skewness and normalized to a straight LOI

Additional to the region-level LOI counting, they introduce the method to skew the region to take into account the perspective of the camera view (See figure 3.5). This helps the SVM and linear regressor to more accurately predict the amount of pedestrians inside the region.

Chapter 4

Method

4.1 Instant LOI counting

The proposed method by [Zhao et al., 2016] in related work section 3.3.1 uses some simplifications. By reframing the pixel-level counting in the following way. The approach is much more theoretical correct.

We define v_{perp} as the normalized directional vector perpendicular to the LOI (Two solutions are perpendicular on the LOI and this defines sides 1 and 2 of the LOI counting). Then we define the collection of the pixels on the left side of the LOI and inside the LOI area as M_1 (side 1) and the pixels on the right side (side 1 and inside the LOI area) as M_2.

Draw picture with LOI area, sets of pixels and vperp

The velocity towards the LOI is then defined as the dot-product of V_t and v_{perp} (Equation 4.1).

$$Q_t(p) = V_t(p) \cdot v_{perp} \tag{4.1}$$

$$\begin{aligned} c_{1,t} &= \sum_{\{p \in M_1 | Q_t(p) > 0\}} C_t(p) \cdot \frac{Q_t(p)}{d} \\ c_{2,t} &= \sum_{\{p \in M_2 | Q_t(p) < 0\}} C_t(p) \cdot \frac{-Q_t(p)}{d} \end{aligned} \tag{4.2}$$

Then then the LOI count on timestep t is then defined in equation 4.2. Where $\frac{Q_t(p)}{d}$ defines the percentage that the density on the specific pixel, has crossed the LOI area. Lastly we can sum the count over a timespan into a single count for each side as in equation 3.4.

4.2 Realigning

Both density map and flow map are trained on separate targets. Whereas the flow map is done in an unsupervised way, it is hard to make sure the flow map and density map are perfectly aligned. Using objects with the density map gaussian in the middle of the object this would not be an issue. However with pedestrians the density tagging is on the top of the head, which could lead to a gaussian which spreads outside the flow map contours of the pedestrian.

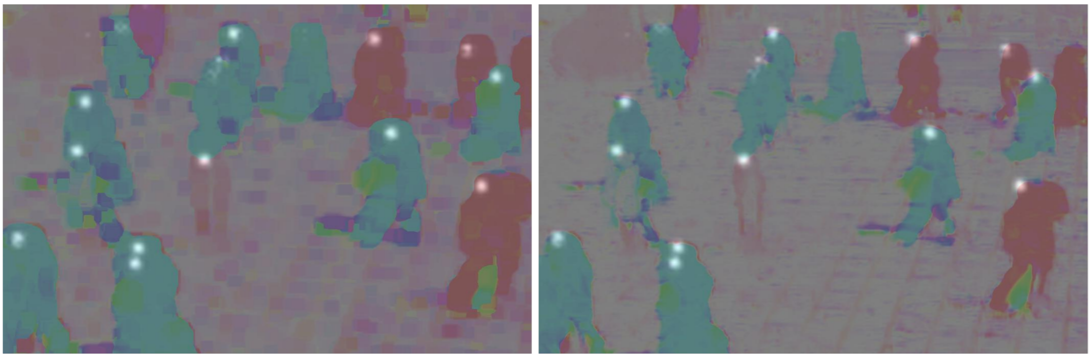


Figure 4.1: On the right a non-maxed flow estimation and on the right the flow estimation with maxing filter applied

To fix this problem we propose an expanding method by applying a maxing filter on the flow estimation. This maxing filter takes the local maximum value in a surrounding of each pixel. Looking at figure 4.1 this helps to cover a lot of misaligned density maps and flow maps. To optimize for heads on the top side of the pedestrian, the maxing filter is focused on the bottom side of the selected pixel. Maximum values above the pixel are ignored.

4.3 Models

4.3.1 Unified model

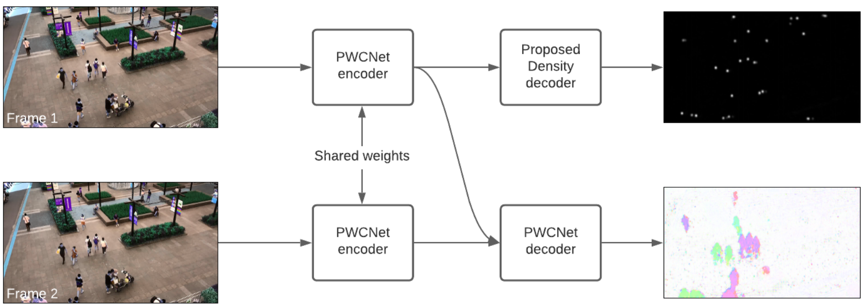


Figure 4.2: Unified model

The first proposed model is a unified model (Fig 5.1). By unifying both flow map and density map predictions the increase in speed is substantial. Additionally both predictions can learn from eachother to further increase their performance.

The model uses the original PWCNet network [Sun et al.,]. The proposed model shares the complete encoder and decoder of the PWCNet, but adds a second decoder to predict a density map as well.

This decoder uses a decoder structure with feeding features in several stages of the decoding stage. Additionally the proposed dilation kernels in CSRNet[Li et al., 2018] are used for a larger reception field which boosts the performance of the density map prediction.

4.3.2 Flow informed density map

The unified model both predicts both the density map and the flow map. In the unified model the multi-headed model only shares the encoder, but doesn't use the final outcomes of either of de decoders to enhance the other decoder. It would be especially beneficial when the model can be optimized for moving pedestrians, because these are the ones counted during Line Crossing.

Therefore we propose a novel method to enhance the density map predictor which makes use of both of these available information streams. This model starts with the unified model (Figure 5.1) as base. By adding the output flow map as input for the density map decoder the decoder could use this information to better detect moving pedestrians.

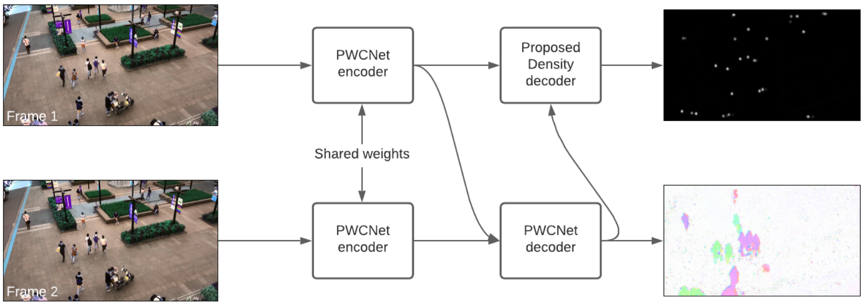


Figure 4.3: Flow enhanced model

Chapter 5

Implementation

In this chapter details about the actual implementation are explained in more depth.

5.1 Models

5.1.1 Baseline 1

As baseline the model of [Zhang et al., 2016] is used. Because of the difference in task, parts of the proposed method can't used. Only the proposed model therefore is used as baseline, where the exact network is shown in figure 3.4.

Due to full shared nature of the network the assumption is that the network will perform very poor when predicting both the density map and the unsupervised flow map. Therefore we propose a second baseline as well.

5.1.2 Baseline 2

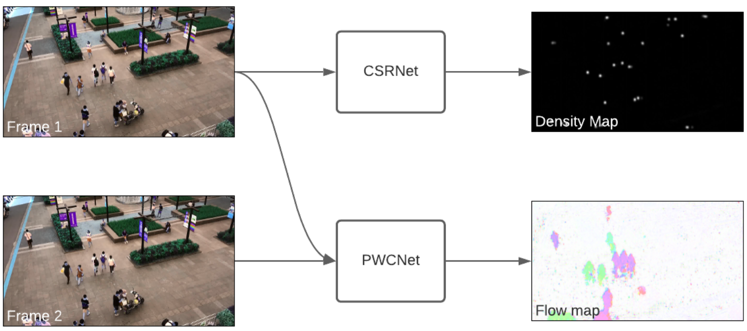


Figure 5.1: CSRNet+PWCNet baseline

This stronger baseline is a combination of two independent models. CSRNet [Li et al., 2018] for predicting the density map and PWCNet [Sun et al.,] for flow estimation. Both models perform very well in comparing to papers in their respective field. Therefore will this baseline be a very powerful baseline to compare to. The models are trained independently of each other with no shared loss function to further optimize their performance.

5.1.3 Shared encoder

The first proposed model shares the decoder of the PWCNet model (Full network in appendix, figure A.1). The PWCNet makes use of a pyramide shaped architecture and the encoder provides the decoder with 6 feature maps ranging from 1/2 the size to 1/64 the size of the original image.

The decoder contains two essential processing blocks. The dilation block, which is a block with 4 conv-layers with a dilation of 2. Additionally an upscaling block is used, which first upscales the input by two and then refines by 2 conv-layers. All conv-layers use a kernel of 3x3 and a stride of 1.

Each of the four tiniest feature map are processed by a dilation block. The smallest feature map is processed first and individually upscaled using the upscaling block. The second and third feature maps are first concatenated with the earlier feature maps and then upscaled.

After processing the fourth feature map and concatenation the upscaled feature maps, the features are processed by two dilation blocks which then predicts the density map using a single output layer.

5.1.4 Flow enhancing

The second model proposed enhances the feature maps with the output of the flow map. The flow enhanced model uses the first proposed model as base decoder. Instead of only decoding the feature maps on each level, the output flow map is reshaped and concatenated to each feature map. Which results the information of the flow on each level available.

5.2 Environment

5.2.1 Maxing filter

During experiments the maxing filter is optimized per dataset. For the Fudan-ShanghaiTech dataset two times a maxing filter is applied width a distance of 6px. For the UCSD dataset two times a maxing filter of 4px is applied.

The maxing filter is applied multiple times to avoid a huge maxing filter which increases the processing time exponentially after a certain distance.

5.2.2 Line Crossing

In the equation 4.2 all the parameters for the Line Crossed are shown. The width of the line (parameter d) is the last parameter which is not defined. In preliminary results the difference is width is minimal, during all the experiments a width of 20px is used therefore.

5.2.3 Loss function

$$L_t = L_v + \lambda \cdot L_c \tag{5.1}$$

The loss function for the final training is given in equation 5.1. Where L_v is the loss function for the velocity map. We finally applied the occlussion photometric loss on all results (Equation 3.1).

L_c is used for the density map. Where the L2 loss (Mean squared error) is applied for comparing the ground truth density map with the predicted map.

For λ a default of 5 is applied when not mentioned else.

5.2.4 Optimizer

For all the experiments the Adam optimizer is used with a learning-rate of $5e^{-5}$. No regularization is applied.

5.2.5 Augmentation

To augment the dataset several augmentations will be applied on the trainingssamples. First a crop of the image is made. Ranging from 1/3 and 1/6 of the total image size. Then the cropped image is resized to a size of 1/4 of the original image (So 1/2 the width and 1/2 the height). Lastly the cropped image is half of the time flipped horizontally.

5.2.6 Metrics

For both the ROI and the LOI the Mean Average Error and the Mean Squared Error are used. Additionally the LOI uses the Relative Mean Average Error.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |C_i - P_c^{(i)}| \tag{5.2}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (C_i - P_c^{(i)})^2 \tag{5.3}$$

For the ROI the MAE and the MSE are defined as equation 5.2 and 5.3. Where $P_c^{(i)}$ is the predicted density map for the given frame.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |G_l^{(i)} - P_l^{(i)}| + |G_r^{(i)} - P_r^{(i)}| \tag{5.4}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (G_l^{(i)} - P_l^{(i)})^2 + (G_r^{(i)} - P_r^{(i)})^2 \tag{5.5}$$

$$RMAE = \frac{1}{n}\sum_{i=1}^{n} \frac{|G_l^{(i)} - P_l^{(i)}| + |G_r^{(i)} - P_r^{(i)}|}{G_l^{(i)} + G_r^{(i)}} \tag{5.6}$$

For the LOI the MAE and MSE are defined as quation 5.4 and 5.5. The RMAE is simply defined as in equation 5.6. Where $G_l^{(i)}$ is the ground truth for sample i for side left-to-right. And $P_r^{(i)}$ is the predicted value for right-to-left.

Chapter 6

Datasets

In this chapter we explain the datasets in more depth. First we explain the requirements of the dataset to correctly train and evaluate each dataset. To compensate for lack of some required labeling a tool is written and explained. Lastly all the datasets are explained in more depth.

6.1 Requirements

For training and evaluation we need two different methods of labeling. For the density map generation the position of each pedestrian is required for the frames which are used for training. For evaluation the line crossing it is required to label the amount of pedestrians crossing the LOI from each side. Ideally the training set is purely labeled with head-tags and the evaluation set only with line crossing labeling.

6.2 Labeling

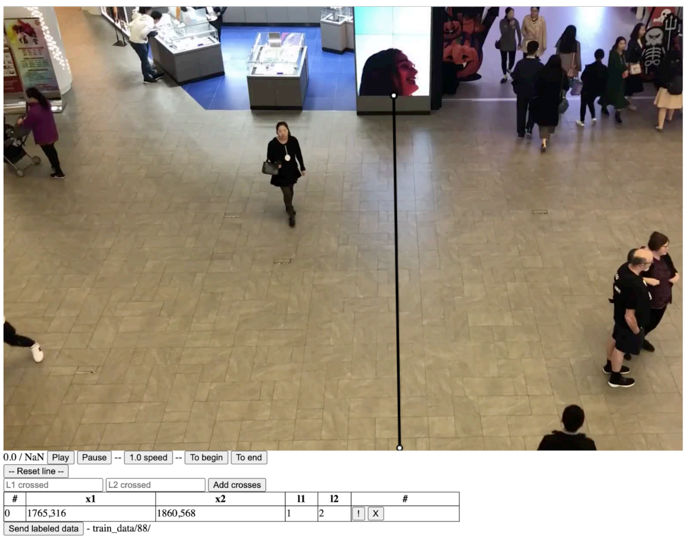
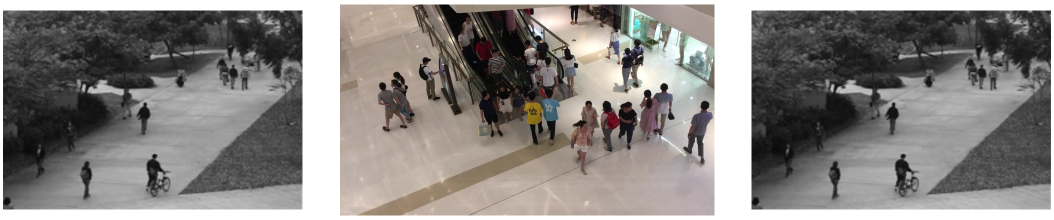


Figure 6.1: User interface of the labeler

Several Crowd Counting datasets provide sequences of frames with corresponding pedestrian labeling. However most of those datasets don't provide line crossing labeling. Therefore I build a tool to label videos for line crossing (Figure 6.1). The labeler loads a video from the unlabeled videos. In this video the user can label multiple lines by first clicking on the video to draw a line and afterwards fill in the amount of pedestrians crossing the line during the video. Additionally the user can scroll and view through the video using multiple manipulations.

6.3 Datasets



(a) UCSD **(b)** Fudan-ShanghaiTech **(c)** AI City Challenge

Figure 6.2: Samples of each dataset

6.3.1 UCSD

The UCSD Pedestrian dataset [Chan and Vasconcelos, 2008] is a public dataset created in 2008. The dataset contains a video of a path where people cross each other (See figure 6.2a). The dataset is 720x480 in resolution. The original videos are in 30fps, but are downscaled to 10fps. A total of 5680 frames are captured. The first 2000 frames contain line of interest labeling [Ma and Chan, 2013]. Frames 0-599 and 1400-1999 are used for testing. The rest of the frames are used as training samples.

For the training samples all the locations of each pedestrian is provided, so no extra labeling required for density generation [Chan and Vasconcelos, 2012].

6.3.2 Fudan-ShanghaiTech

The Fudan ShanghaiTech dataset [Fang et al., 2019] is a public dataset with 100 videos of 13 different scenes. Each video contains 6 seconds of footage at 25 fps and have a resolution of 1920x1080 or 1280x720 (Sample of scene in figure 6.2b). The scenes have between 20-100 pedestrians per frame. In each frame the pedestrians in the frame are labeled with a bounding-box and a center-point of the bounding-box. The dataset contains 60 training videos and 40 testing videos.

The lack of trajectories and custom line crossing labeling requires the use of the custom build labeler (Figure 6.1). This is done on the 40 videos of the test set.

6.3.3 AI City Challenge

Chapter 7

Results

7.1 Fudan-ShanghaiTech

Method (LOI)	MAE	RMAE
Baseline 1	5.762	0.940
Baseline 2	3.273	0.531
Baseline 2+m	1.490	0.291
Proposed+m	1.539	0.281
Proposed+f+m	1.368	0.266

Table 7.1: The LOI results for Fudan

Method (ROI)	MAE	MSE
Baseline 1	16.402	340.435
Baseline 2	7.569	80.51
Proposed	5.066	50.37
Proposed+f	9.516	132.920

Table 7.2: The ROI results for Fudan

For te Fudan-ShanghaiTech dataset both the LOI performance and the ROI performance is displayed in table 7.1 and table 7.2.

Looking at table 7.1. The original baseline 1 doesn't perform very well on the new task, especially looking at the RMAE. Which is as expected, because the buildup of the model assumes that the Flow Map is trained supervised. Baseline 2 is performing much better and therefore a much stronger baseline.

By aligning the flow map and density map with the maxing filter a huge increase in performance is measured as well. With decreasing both the MAE and RMAE by a factor of 2. Which suggests that the aligning using the maxing filter has some serious benefits for Crowd Crossing Counting.

The proposed model performs slightly worse than our baseline, which is probably due to the multi modal performance of the encoder. However when the Flow map is fed to the density map decoder, the model significantly outperforms the baseline.

Looking at table 7.2, the results show that the proposed model outperforms CSRNet [Li et al., 2018] on this dataset. Adding Flow significantly decreases the ROI performance, which could mean that the model is focussed on the flow features and therefore ignores some of the non-moving pedestrians.

7.2 UCSD

Stated in table 7.4 that the proposed method doesn't produce a state-of-the-art results on the existing UCSD. But more info about UCSD soon in table 7.4 and table 7.3

Method (ROI)	MAE	MSE
Paper 1	0.0	0.0
Paper 2	0.0	0.0
Paper 3	0.0	0.0
Baseline 1	5.762	0.940
Baseline 2	3.273	0.531
Proposed	1.539	0.281
Proposed+f	1.368	0.266

Table 7.3: The ROI results for UCSD

Method (LOI)	MAE	MSE
Paper 1	0.0	0.0
Paper 2	0.0	0.0
Paper 3	0.0	0.0
Baseline 1	5.762	0.0
Baseline 2	3.273	0.531
Baseline 2+m	1.490	0.291
Proposed+m	1.539	0.281
Proposed+f+m	1.368	0.266

Table 7.4: The LOI results for UCSD

7.3 AI City Challenge

Method (ROI)	MAE	MSE
Baseline 1	5.762	0.0
Baseline 2	3.273	0.531
Proposed	1.539	0.281
Proposed+f	1.368	0.266

Table 7.5: The ROI results for AI City Challenge dataset

Method (LOI)	MAE	MSE
Baseline 1	5.762	0.0
Baseline 2	3.273	0.531
Baseline 2+m	1.490	0.291
Proposed+m	1.539	0.281
Proposed+f+m	1.368	0.266

Table 7.6: The LOI results for AI City Challenge dataset

To be done about table 7.5 and table 7.6

7.4 Real world performance

To compare real world performance the models are compared on processing speed. Additionally the optical FPS is calculated. This is both done on the Fudan-ShanghaiTech dataset.

Method	FPS	ms
Baseline 2	2.7f	376ms
Baseline 2+m	2.3	440ms
Proposed+m	4.0	248ms
Proposed+f+m	0.0	
Proposed+f+m+o	0.0	

Table 7.7: Processing time

FPS	MAE	MSE
25	0.0	0.0
12.5	0.0	0.0
8.3	0.0	0.0
5	0.0	0.0
2.5	0.0	0.0

Table 7.8: Optimal FPS

7.5 Flow estimation impact

Qualitative comparison

Chapter 8

Conclusion

In this thesis we have shown several points. Crowd Crossing Counting is possible without the supervised knowledge of the Flow Estimation. Two new models are proposed with which the flow enhanced model outperforms the strong baseline which suggests that knowledge about the flow improves the density map prediction for moving objects.

Additionally we have shown both quantitive and qualitative that the maxing filter is an effective method to realign the flow map and the density map, which drastically increases the performance.

Lastly we did a thorough analysis on real world performance. According to the results it would be possible to run the model real time on a fast GPU without optimization. Further optimization could lead to an even quicker model.

8.1 Further research

Due to the lack of high density videos, the full potential of this network can't be shown to its full extend. For further research it would be ideal to propose such a high density map.

Bibliography

[Brox et al., 2014] Brox, T., Papenberg, N., and Weickert, J. (2014). High Accuracy Optical Flow Estimation based on warping - presentation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3024(May):25–36.

[Bruhn et al., 2005] Bruhn, A., Weickert, J., and Schnörr, C. (2005). Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):1–21.

[Cao et al.,] Cao, L., Zhang, X., Ren, W., and Huang, K. Large scale crowd analysis based on convolutional neural network. 48(10):3016–3024.

[Chan and Vasconcelos, 2009] Chan, A. and Vasconcelos, N. (2009). Bayesian Poisson regression for crowd counting Cited by me. *Computer Vision, 2009 IEEE 12th International ....*

[Chan and Vasconcelos, 2008] Chan, A. B. and Vasconcelos, N. (2008). Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):909–926.

[Chan and Vasconcelos, 2012] Chan, A. B. and Vasconcelos, N. (2012). Counting people with low-level features and bayesian regression. *IEEE Transactions on Image Processing*, 21(4):2160–2177.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, I:886–893.

[Dollár et al., 2012] Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2012). Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761.

[Dosovitskiy et al., 2015] Dosovitskiy, A., Fischery, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Smagt, P. V. D., Cremers, D., and Brox, T. (2015). FlowNet: Learning optical flow with convolutional networks. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter:2758–2766.

[Fang et al., 2019] Fang, Y., Zhan, B., Cai, W., Gao, S., and Hu, B. (2019). Locality-constrained spatial transformer network for video crowd counting. *Proceedings - IEEE International Conference on Multimedia and Expo*, 2019-July:814–819.

[Horn and Schunck, 1981] Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203.

[Hui et al., 2018] Hui, T. W., Tang, X., and Loy, C. C. (2018). LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 8981–8989.

[Idrees et al., 2013] Idrees, H., Saleemi, I., Seibert, C., and Shah, M. (2013). Multi-source multi-scale counting in extremely dense crowd images. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2547–2554.

[Ilg et al., 2016] Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2016). FlowNet 2.0: Evolution of optical flow estimation with deep networks.

[Janai et al., 2018] Janai, J., Güney, F., Ranjan, A., Black, M., and Geiger, A. (2018). Unsupervised Learning of Multi-Frame Optical Flow with Occlusions. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11220 LNCS:713–731.

[Li et al., 2018] Li, Y., Zhang, X., and Chen, D. (2018). Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1091–1100.

[Lin and Davis, 2010] Lin, Z. and Davis, L. S. (2010). Shape-based human detection and segmentation via hierarchical part-template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):604–618.

[Liu et al., 2008] Liu, C., Freeman, W. T., Adelson, E. H., and Weiss, Y. (2008). Human-assisted motion annotation. *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*.

[Liu et al., 2019a] Liu, P., King, I., Lyu, M. R., and Xu, J. (2019a). DDFlow: Learning optical flow with unlabeled data distillation.

[Liu et al.,] Liu, P., Lyu, M., King, I., and Xu, J. SelFlow: Self-supervised learning of optical flow.

[Liu et al., 2019b] Liu, W., Salzmann, M., and Fua, P. (2019b). Context-aware crowd counting. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:5094–5103.

[Ma and Chan,] Ma, Z. and Chan, A. B. Counting people crossing a line using integer programming and local features. 26(10):1955–1969.

[Ma and Chan, 2013] Ma, Z. and Chan, A. B. (2013). Crossing the line: Crowd counting by integer programming with local features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2539–2546.

[Mémin and Pérez, 1998] Mémin, E. and Pérez, P. (1998). Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719.

[Pock et al., 2008] Pock, T., Schoenemann, T., Graber, G., and Bischof, H. (2008). Learning optical flow. 5304(May 2014).

[Ranjan and Black, 2017] Ranjan, A. and Black, M. J. (2017). Optical flow estimation using a spatial pyramid network. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-January:2720–2729.

[Sreenu and Saleem Durai, 2019] Sreenu, G. and Saleem Durai, M. A. (2019). Intelligent video surveillance: a review through deep learning techniques for crowd analysis. *Journal of Big Data*, 6(1):1–28.

[Subburaman et al., 2012] Subburaman, V. B., Descamps, A., and Carincotte, C. (2012). Counting people in the crowd using a generic head detector. *Proceedings - 2012 IEEE 9th International Conference on Advanced Video and Signal-Based Surveillance, AVSS 2012*, pages 470–475.

[Sun et al.,] Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. PWC-net: CNNs for optical flow using pyramid, warping, and cost volume.

[Wan and Chan, 2019] Wan, J. and Chan, A. (2019). Adaptive density map generation for crowd counting. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-Octob:1130–1139.

[Wang et al., 2020] Wang, Q., Gao, J., Lin, W., and Li, X. (2020). Nwpu-crowd: A large-scale benchmark for crowd counting. *arXiv preprint arXiv:2001.03360*.

[Wedel et al., 2009] Wedel, A., Cremers, D., Pock, T., and Bischof, H. (2009). Structure- and motion-adaptive regularization for high accuracy optic flow. *Proceedings of the IEEE International Conference on Computer Vision*, pages 1663–1668.

[Wu and Nevatia, 2007] Wu, B. and Nevatia, R. (2007). Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. *International Journal of Computer Vision*, 75(2):247–266.

[Yu et al., 2016] Yu, J. J., Harley, A. W., and Derpanis, K. G. (2016). Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9915 LNCS:3–10.

[Zhang et al., 2016] Zhang, Y., Zhou, D., Chen, S., Gao, S., and Ma, Y. (2016). Single-image crowd counting via multi-column convolutional neural network. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:589–597.

[Zhao et al., 2016] Zhao, Z., Li, H., Zhao, R., and Wang, X. (2016). Crossing-line crowd counting with two-phase deep neural networks. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, volume 9912, pages 712–726. Springer International Publishing.

[Zheng et al., 2019] Zheng, H., Lin, Z., Cen, J., Wu, Z., and Zhao, Y. (2019). Cross-line pedestrian counting based on spatially-consistent two-stage local crowd density estimation and accumulation. 29(3):787–799.

Appendix A

Appendix

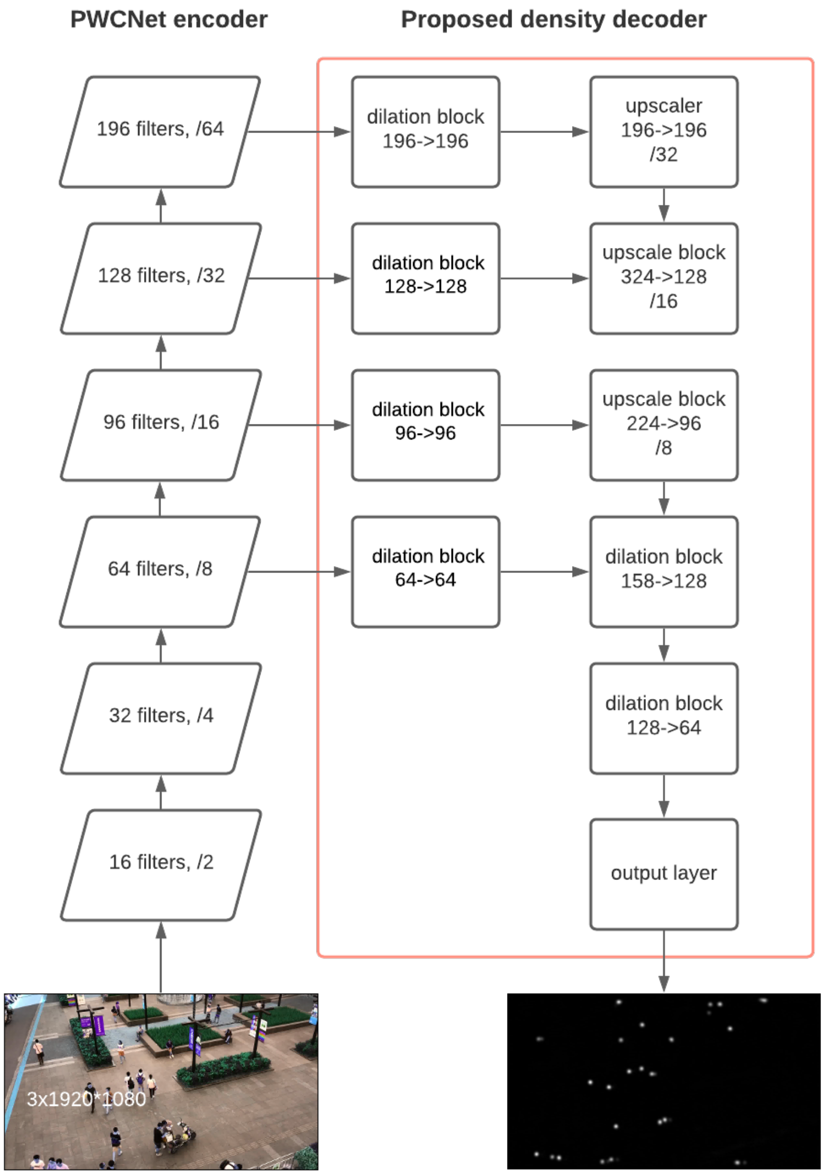


Figure A.1: Full decoder for density prediction